

Integration of a Wearable IoT-Based Smart Healthcare System with Predictive Modelling

by

Pragesh Ashley Govender

Dissertation presented in partial fulfillment of the requirements for
the Degree of

MASTER OF SCIENCE IN ENGINEERING
(MECHATRONICS ENGINEERING)

In the faculty of Engineering
at the University of KwaZulu-Natal

Supervised by

Prof. R Stopforth

Prof. S Sivarasu

Prof. E Markus

Mr. Clive Hands

December 2021

Declaration – Plagiarism

I **Pragesh Ashley Govender** declare that:

1. The research reported in this dissertation, except where otherwise indicated, is my original research.
2. This dissertation has not been submitted for any degree or examination at any other university.
3. This dissertation does not contain other persons' data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.
4. This dissertation does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
 - a. Their words have been re-written but the general information attributed to them has been referenced.
 - b. Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.
5. This dissertation does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the dissertation and in the References sections.

Signed:

.....

As the candidate's Supervisor I agree to the submission of this dissertation

Signed:

.....

Declaration – Publications & Courses

Publication:

Submission of the following chapters in book, to the *Mediventors* initiative:

P Govender, R Stopforth, “*An IoT based approach to healthcare monitoring*”, *MediVentors, 2021*

Signed:

.....

Medical Related Courses Completed:

Participant in the *Mediventors* consortium (a joint initiative between merSETA, WITS University, UKZN, CPUT and UCT) aimed at improving medical device development in the local market. The following medical device courses were completed with distinctions: 1) Medical device sector essentials, 2) Systems Engineering and product lifecycle management and 3) Tooling, manufacturing and industrialization. The coursework was composed of 93 lectures, 71 Quizzes, 14 assignments and 2 exams. A detailed project proposal was also completed based on this dissertation topic to participate in the *Mediventors Lions Den* initiative to be held in February 2022

Abstract

The usage of the Internet of things (IoT) equipped with intelligent machine learning (ML) or fuzzy logic (FL) systems in the medical sector is becoming a widespread normality. This following the implications of the 4th and 5th industrial revolutions (5IRs) aiming to integrate technology with the way we live and work. Many research areas have focused on incorporating individual elements of an intelligent IoT based system. However, none have holistically looked at encompassing a multi-faceted sensory device with an IoT framework backed by both ML and FL capabilities for unique monitoring and diagnostics. This implies that there is an opportunity to create such a product targeted for the medical sector.

The developed system of systems (SoS) discussed therefore aims at eliminating the barrier to patient care outside of a hospital setting through the use of real time patient diagnostics.

The study was approached by first establishing a streamlined IoT infrastructure. A multi-faceted Arduino based sensory device was then developed to collect user parameters. The electronics were housed in a common of the shelf (COTS) casing satisfying the criteria of wearability. Bluetooth connectivity then allowed for the transmission of sensory data from the WBAN to the IoT smartphone gateway. Sensory parameters being measured were electrocardiography (ECG), electromyography (EMG), body temperature (BT), infrared (IR) output, pulse rate (PR) as well as environmental humidity readings. Blood sugar (BS) levels were obtained non-invasively by calibrating the IR output voltage with that of a calibrated glucometer to obtain a straight line equation representing the relationship between the sensor ADC output and glucose in the blood.

A Java based mobile application (MA) was then developed, which allowed for the processing of the sensory data before storage in a local SQLite database. A two interface MA for use by registered doctors and patients was necessary to allow for data sharing and security. Registration and login for the MA was done through a NoSQL Firebase database. A JDBC was established to enable the transmission of data to a MS Azure Structured Query Language (SQL) database. This served as the cloud interface within the IoT network layer. The MA is able to integrate with a phone's Global Positioning System (GPS) to allow for simultaneous tracking of patients. A return application programming interface (API) connection between MS Azure and the developed application layers was then created. Here the mobile doctor interface as well as a developed node.js based web application (WA) served as the mediums through which health practitioners could access patient data.

ML models were developed using the MS Azure ML Classic Studio suite which allowed for a real time analysis of received data through deployment and subsequent consumption of data using POST Requests via the MA and a Java software development kit (SDK). An accuracy of 92% was achieved for the stroke prediction model based on the boosted decision tree algorithm. Various ML models were analyzed to ensure that a high precision was obtained while preventing overfitting. Additional FL models were also developed, which took into consideration unique sets of vitals combinations to create a rule base depicting the patient health status and health risk.

This data was compared to the intuition of a doctor and received an 87% accuracy for model 1 which took into consideration a patient's PR, BS, BT and age to predict the health status of a patient. The model was then compared to the Modified Early Warning Score (MEWS) rating, a popular measure of health risk utilized in the medical field. The results of comparison also yielded an accuracy of 87%.

A second FL model was then developed looking at the effect of environmental conditions on the risk rating of patients. Here input variables of BT, age and humidity readings were used to determine their effect on the risk rating of a patient. This model scored an 80% accuracy when compared to the expertise of a physician. Both models were programmed onto the MA to predict the patient's health status. Both models were also re-developed on MATLAB software to simulate the effect of various input variables on the response variable.

Overall the designed system was able to possess around 15 of the typical features found in smart wearable systems which far exceeded the features of those devices it was compared to. The designed system satisfied the requirement of a feature rich experience while also satisfying the criteria of cost effectiveness.

Acknowledgement

I would like to thank my family for their support, motivation and for perpetually encouraging me to achieve more.

I would also like to thank Prof. R Stopforth for giving me the opportunity to pursue this dissertation and for the wisdom, guidance and mentorship he has provided me throughout the process.

Appreciation also goes out to my co-supervisors, Prof. S Sivarasu, Prof. E Markus and Mr. Clive Hands for their guidance and words of encouragement.

Lastly I would like to extend a great deal of gratitude to the two medical doctors (Dr. Pillay and Dr. Pawel Wisniewski) for assisting in the model benchmarking process and for also providing guidance and advice on building a fit for purpose system aligned to the medical sector.

Dedicated to the healthcare practitioners

Who work tirelessly to save lives

Table of Contents

List of Figures	xiii
List of Tables.....	xvii
Glossary & Nomenclature	xvii
Chapter 1: Introduction	1
1.1. Problem Statement.....	3
1.2. Research Questions.....	4
1.3. Literature Review.....	5
1.3.1. The Origin of Telemonitoring Systems	5
1.3.2. Non-LoT Telemonitoring Systems.....	6
1.3.3. Non-Invasive Glucose Monitoring Systems.....	7
1.3.4. GPS Enabled Telemonitoring Systems.....	9
1.3.5. IoT Enabled Telemonitoring Systems.....	9
1.3.6. AI and Fuzzy Logic in Healthcare.....	13
1.3.7. IoT & AI based Telemonitoring Systems.....	14
1.3.8. IoT & FL Based Telemonitoring Systems	15
1.3.9. IoT Fuzzy Logic and AI Hybrid Telemonitoring Systems	16
1.4. Research Contributions.....	17
1.5. Hypothesis Statement	17
1.6. Objectives and Outline.....	18
1.7. Chapter Summary	20
Chapter 2: Requirements Analysis, Design Specifications & Concept Development	21
2.1. Stakeholder Expectations	21
2.1.1. Patient	21
2.1.2. Physician/Doctor	22
2.1.3. Other Healthcare Personnel.....	22
2.2. Design Specifications and Challenges.....	22
2.2.1. Telehealth, Telemedicine and Telemonitoring	23
2.2.2. The IoT Infrastructure.....	24
2.2.3. Software Specifications and Challenges	26
2.2.4. Electronics Specifications and Challenges	26
2.2.5. Wearability Specifications and Challenges.....	29
2.3. Proposed System Description.....	29

2.4. Concept Development and Selection	31
2.4.1. The Pugh Matrix.....	31
2.4.2. Proposed IoT Infrastructure Designs	31
2.4.3. Proposed Wearable Designs	33
2.5. Chapter Summary	35
Chapter 3: Sensing Layer - WBAN Device	36
3.1. Sensing Layer Characteristics.....	36
3.1.1. ECG	36
3.1.2. Electromyography (EMG).....	37
3.1.3. Oxygen Saturation & Pulse Rate.....	38
3.1.4. Humidity	38
3.1.5. Body Temperature	39
3.1.6. IR Sensor & Receiver.....	39
3.2. High Level Specifications	40
3.3. Electronic Design	40
3.3.1. MCU.....	40
3.3.2. Bluetooth	42
3.3.3. Body Temperature Sensor	43
3.3.4. Humidity Sensor.....	44
3.3.5. ECG Sensor.....	46
3.3.6. EMG Sensor	47
3.3.7. Pulse Rate Sensor.....	48
3.3.8. IR Transmitter and Receiver	49
3.3.9. Lithium Battery and Balanced Charger	50
3.3.10. Veroboard Design.....	50
3.4. MCU Software Design	51
3.4.1. Program Flow Initiation	51
3.4.2. Program Flow Operation.....	52
3.4.3. ADC	53
3.4.4. Timers and Interrupts for Sending Data Packages.....	54
3.5. WBAN Strap Integration.....	54
3.5.1. Enclosure Selection and Sensor Positioning.....	54
3.5.2. Integration of WBAN Strap and Electronics.....	56
3.6. Chapter Summary	57

Chapter 4: IoT Gateway and GUI's	58
4.1. MA and IoT Gateway Development	58
4.1.1. Patient Registration	59
4.1.2. Patient Tracking	59
4.1.3. Receiving Information via the RFCOMM Protocol	59
4.1.4. Storage of Data in the SQLite Database	60
4.1.5. Viewing of Vitals on the Patient Interface	61
4.1.6. Cloud Upload	61
4.1.7. Granting Firebase Permissions	62
4.1.8. Physician Registration Process	63
4.1.9. Displaying of Information on the Physician Interface	63
4.1.10. Accessing Patient Timestamp Readings	64
4.1.11. Integration with Models (FES, ML and MEWS)	64
4.1.12. MA Front End Design	64
4.2. WA Development	65
4.2.1. Back- End Design	66
4.2.2. Integration with Databases	67
4.3. Chapter Summary	67
Chapter 5: Cloud Infrastructure	68
5.1. Firebase Integration	68
5.1.1. Setting up A Firebase Project	69
5.1.2. MA Integration	69
5.1.3. Web Integration	70
5.1.4. JSON Rules	71
5.2. MS Azure Integration	73
5.2.1. Setting-up Microsoft Azure	73
5.2.2. Mobile Integration	74
5.2.3. Web Integration	75
5.3. Chapter Summary	76
Chapter 6: Modelling	77
6.1. ML using MS Azure Toolbox	77
6.1.1. Background to ML Applications	77
6.1.2. Dataset for Developed Model	81
6.1.3. Design Approach	82

6.1.4. Exploratory Research.....	84
6.1.5. Data Pre-processing.....	90
6.1.6. Model Evaluation	91
6.1.7. Model Selection.....	97
6.1.8. Cross validation.....	98
6.1.9. Test for Overfitting and under fitting.....	99
6.1.10. Web deployment and Model Consumption.....	100
6.2. FL Model.....	100
6.2.1. Fuzzy Logic Explained	100
6.2.2. Fuzzy Logic Architecture	101
6.2.3. Types of fuzzy inference systems	101
6.2.4. Approach.....	101
6.2.5. Input, Output and Linguistic Variables.....	104
6.2.6. Fuzzification with MFs.....	105
6.2.7. Developing a Rule Base	111
6.2.8. Rule Evaluation	113
6.2.9. Defuzzification.....	113
6.3. Chapter Summary	116
Chapter 7: Use Case Scenarios	117
7.1. Patient Use Case Scenario.....	117
7.1.1. Overview	118
7.1.2. Sequence of Steps	118
7.2. Doctor Use Case Scenario.....	125
7.2.1. Overview	125
7.2.2. Sequence of Steps	125
7.3. Other Health Practitioner Use Case Scenario.....	133
7.4. Chapter Summary	133
Chapter 8: Results	135
8.1. Sensor Testing.....	135
8.1.1. Temperature Sensor	135
8.1.2. Humidity Sensor.....	142
8.1.3. Glucose Sensor.....	145
8.1.4. Pulse Rate Sensor.....	150
8.2. Model Testing	152

8.2.1. MATLAB Simulations	152
8.2.2. FL Model Testing	162
8.2.3. ML Model	168
8.3. MA Testing	169
8.3.1. Battery Performance Testing	169
8.3.2. RAM Usage Testing	170
8.4. Overall System Performance.....	170
8.4.1. System Integration Testing.....	171
8.4.2. System Comparison	173
8.5. Chapter Summary	175
Chapter 9: Conclusion and Recommendations	176
9.1. Dissertation Summary	176
9.2. Conclusion.....	177
9.3. Recommendations	179
References	180
Appendices	190
Appendix A: Bill of Materials (BOM).....	190
Appendix B: Project Planning.....	195
Appendix C: Mobile App UML diagram	196
Appendix D: Stripboard layout	197
Appendix E: Detailed Process Flow	198
Appendix F: Systems flow diagram for physiological parameters	199
Appendix G: Circuit Diagram.....	200
Appendix H1: SQL Queries	201
Appendix H2: SQLite Android Query code	204
Appendix I: Battery Sizing.....	206
Appendix J: High Level Design approach	207
Appendix K: Sensor Positioning Matrix and Sketches	208
Appendix L: Rules Bases for FL Models	209
Appendix M: MATLAB Code	212

List of Figures

Figure 1-1: U.S. IoT healthcare market size	2
Figure 1-2: Block Diagram of Bluetooth Based Telemonitoring System.....	6
Figure 1-3: Temperature Monitoring on WA Using Bluetooth	6
Figure 1-4: Zigbee Technology Being Used For Data Transmission.....	7
Figure 1-5: Relationship between Output Voltage and Glucose Levels	8
Figure 1-6: Set-up for Diabetes IR Detection System.....	9
Figure 1-7: Data Transfer from Smartwatch to Smartphone IoT Gateway	10
Figure 1-8: “Smartstone” IoT Architecture	11
Figure 1-9: ESP32 IoT Based System for Pacemaker Monitoring	12
Figure 1-10: System Architecture with Raspberry PI IoT Gateway	12
Figure 1-11: IoT System for Atrial Fibrillation using classification model & LoRa Gateway.....	14
Figure 1-12: IoT HealthCare System with FES.....	15
Figure 1-13: Hybrid Fuzzy Neural Network System.....	16
Figure 1-14: Dissertation Objectives & Outline	19
Figure 2-1: Telehealth Taxonomy (Elena Muller, 2021).....	23
Figure 2-2: Device to cloud architecture.....	25
Figure 2-3: Gateway IoT Architecture.....	25
Figure 2-4: Mobile Gateway as an IoT Architecture.....	26
Figure 2-5: Proposed Smart IoT Telemonitoring System.....	30
Figure 2-6: UML Diagram for Proposed Telemonitoring System.....	30
Figure 2-7: Pugh Matrix for IoT Gateway Selection	32
Figure 2-8: Pugh Matrix for WBAN Selection.....	34
Figure 3-1: Normal ECG Wave	37
Figure 3-2: Normal EMG Wave	38
Figure 3-3: HC-06 Connection with Arduino Nano MCU.....	42
Figure 3-4: LM35 to MCU connection.....	43
Figure 3-5: HIH-4000 Connection with MCU	45
Figure 3-6: Effect of Different Temperatures on Relative Humidity.....	46
Figure 3-7: AD8232 Connection to MCU.....	47
Figure 3-8: Myoware EMG Connection with MCU.....	48
Figure 3-9: Pulse Rate Sensor	48
Figure 3-10: Pulse Rate Sensor Connection with MCU	49
Figure 3-11: IR Sensor Connection with MCU.....	50
Figure 3-12: Components connected to Veroboard.....	51
Figure 3-13: CodeVision AVR Wizard Set-up.....	52
Figure 3-14: Single Loop Operational Flow MCU Code.....	53
Figure 3-15: Small Enclosure (black) and Large Enclosure (grey) with Holes for Electrodes.....	55
Figure 3-16: Positioning of EMG Sensor of Forearm Using Velcro Strap.....	55
Figure 3-17: Electronics and Battery Pack within Smaller Enclosure	56
Figure 3-18: Electronics within Larger Enclosure	56
Figure 3-19: Fitting of WBAN on Patient’s Arm.....	57
Figure 4-1: Operational Flow of MA.....	58
Figure 4-2: Schema for SQLite Database.....	60
Figure 4-3: Schema for SQL Database Table on Azure Showing Uploaded Data	62
Figure 4-4: Sample Patient Node Showing Doctor Permissions	62
Figure 4-5: (a): Main Page Wireframe, (b) Patient Interface Wireframe, (c) Doctor interface Wireframe	65

Figure 4-6: Flow of Information through WA.....	66
Figure 4-7: Node.js Server set-up	66
Figure 5-1: Traditional Database Set-up.....	68
Figure 5-2: Firebase Database Set-Up.....	68
Figure 5-3: Firebase Console Set-Up.....	69
Figure 5-4: Public Patient Firebase JSON rules	71
Figure 5-5: WA Doctor Firebase JSON Rules	72
Figure 5-6: Private Patient Firebase JSON Rules	72
Figure 5-7: Setting-Up MS Azure Services.....	74
Figure 5-8: Creating a Connection between MS Azure and MA Using JDBC SDK.....	75
Figure 6-1: PRC Curve.....	79
Figure 6-2: ROC Curve	80
Figure 6-3: Various types of ML techniques	80
Figure 6-4: Kaggle Dataset Statistics	81
Figure 6-5: High-Level Overview of Model Development within MS Azure ML Classic Studio...	82
Figure 6-6: Work Flow of Stroke Prediction Model	82
Figure 6-7: Histogram Showing Unbalanced Dataset.....	84
Figure 6-8: Extract from MS Azure Showing missing BMI Values.....	84
Figure 6-9: histogram of Gender Variable	86
Figure 6-10: Histogram of BMI Status Variable	86
Figure 6-11: Histogram of Residence Type Variable	86
Figure 6-12: Histogram of Hypertension Status.....	87
Figure 6-13: Histogram of Work Type Status.....	87
Figure 6-14: Histogram of Glucose Status Variable.....	87
Figure 6-15: Histogram of Heart Disease Status	88
Figure 6-16: Histogram of Smoking Status Variable.....	88
Figure 6-17: Histogram of Marital Status Variable	88
Figure 6-18: Histogram of Stroke Cases	89
Figure 6-19: (a) Age of Stroke Patients Box Plot, (b) Age of Non- Stroke Patients Box Plot.....	89
Figure 6-20: (a) Rectified BMI Box Plot, (b) Box Plot Glucose levels of Non-Stroke Patients, (c) Box plot of Glucose Levels of Patients with Stroke.....	90
Figure 6-21: (a) Sample Size Split before SMOTE, (b) Sample Size Split after SMOTE.....	91
Figure 6-22: Obesity Ranges (Wright, 2021).....	92
Figure 6-23: (a) ROC Curve Bayes Point Machine, (b) Precision/Recall Curve Bayes Point Machine	92
Figure 6-24: Bayes Point Machine Statistics	92
Figure 6-25: (a) Boosted Decision Tree, (b) Precision/Recall Curve Boosted Decision Tree	93
Figure 6-26: Statistics Boosted Decision Tree.....	93
Figure 6-27: (a) ROC Curve Two-Class Decision Forest, (b) Precision/Recall Curve Two-Class Decision Forest.....	94
Figure 6-28: Statistics of the Two-Class Decision Forest	94
Figure 6-29: (a) ROC Curve Logistic Regression, (b) Precision/Recall Curve Logistic Regression	94
Figure 6-30: Statistics Logistic Regression.....	95
Figure 6-31: (a) ROC Curve Two Class Neural Network, (b) Precision/Recall Curve Two Class Neural Network.....	95
Figure 6-32: Statistics Two Class Neural Network.....	96
Figure 6-33: (a) ROC Curve SVM, (b) Precision/Recall Curve SVM.....	96

Figure 6-34: Statistics for the SVM Model	97
Figure 6-35: Graph Showing Different Performance Statistics of 6 Models	97
Figure 6-36: K-Fold Method Statistics	98
Figure 6-37: (a) Workflow of Test Overfitting and Under Fitting, (b) ROC Curve Comparing Training and Test Data with Model.....	99
Figure 6-38: Training Set Model Statistics.....	99
Figure 6-39: Test Set Model Statistics.....	99
Figure 6-40: Fuzzy Logic Block Diagram.....	101
Figure 6-41: Typical FL Model Development Process	102
Figure 6-42: Operational Flow FES.....	103
Figure 6-43: Triangular Membership Function Pulse Rate.....	105
Figure 6-44: Triangular Membership Function Body Temperature.....	106
Figure 6-45: Triangular Membership Function Glucose Levels.....	107
Figure 6-46: Triangular Membership Function Age	107
Figure 6-47: Triangular Membership Function Output.....	108
Figure 6-48: Triangular Membership Function Humidity.....	109
Figure 6-49: Triangular Membership Function Age	109
Figure 6-50: Triangular Membership Function Body Temperature.....	110
Figure 6-51: Output Membership Function Model 2.....	111
Figure 7-1: Use Case Diagram for Patient Accessing IoT system	117
Figure 7-2: (a) Registration Screen, (b) Login Screen	118
Figure 7-3: (a) to (d) – registration pages	119
Figure 7-4: Registration Complete Page	119
Figure 7-5: Bluetooth Prompt	120
Figure 7-6: Bluetooth Discovery	120
Figure 7-7: Screen showing Smartphone receiving data from WBAN.....	121
Figure 7-8: (a) GPS tracking enabled, (b) GPS tracking disabled.....	121
Figure 7-9: Cloud Sharing Transmission Screen	122
Figure 7-10: (a) Searching for Doctor on Database, (b) Sharing info with Doctor Screen	122
Figure 7-11: Patient Monitoring Vitals Screen	123
Figure 7-12: Screen for Patients to Add Their Data to WA Database	123
Figure 7-13: Firebase Node with Doctor ID Who Patient Has Chosen to Share Info With	124
Figure 7-14: Screen on WA for Patients to Add Doctors They Want on Their Network.....	124
Figure 7-15: (a) Main Page, (b) Doctor Login Page.....	125
Figure 7-16: (a) Doctor Registration selection, (b) Doctor details Registration Activity, (c) Registration Complete Activity	126
Figure 7-17: Doctor Interface Activity	126
Figure 7-18: (a) ECG monitoring Activity, (b) Location Tracking and Environmental Monitoring Activity, (c) General Patient Info.....	127
Figure 7-19: Timestamps Activity.....	128
Figure 7-20: Doctor Registration Page WA	128
Figure 7-21: Doctor Registration Details Page	129
Figure 7-22: Doctor Login Page	129
Figure 7-23: Doctor Forgot Password Page	130
Figure 7-24: Screen for Doctor to Capture ID for “Patient Info”.....	130
Figure 7-25: Patient Info Page	131
Figure 7-26: Patient Vitals Screen WA.....	132
Figure 7-27: Example of ECG Graph on WA Page.....	132

Figure 7-28: Patient Location Tracking on WA	132
Figure 8-1: Graph of Temperature over time (seconds) for the LM35 Temperature Sensor.....	137
Figure 8-2: 3D Scatter Plot of Each Patient's BT Using a Calibrated Temperature Sensor.....	138
Figure 8-3: Calibration Curve for LM35 Body Temperature Sensor	139
Figure 8-4: Temperature Output with Straight Line Equation vs Calibrated Sensor	141
Figure 8-5: Graph Showing Price of Current System vs Commercial Products.....	142
Figure 8-6: Graph of Humidity over Time for the HIH-4000 Sensor	143
Figure 8-7: Humidity readings for the HIH-4000 vs Weather Reports for the Same Area	144
Figure 8-8: Surface Plot of Glucose Readings at Different Intervals	145
Figure 8-9: Blood Glucose Levels Calibration Fit Using Commercial Sensor.....	146
Figure 8-10: IR Sensor Output Relative to Commercial Sensor.....	148
Figure 8-11: Clarke Error Grid Analysis.....	148
Figure 8-12: Cost Comparison of Glucose Detection System with Other Commercial Products... ..	150
Figure 8-13: Pulse Rate Sensor vs Calibrated Sensor	152
Figure 8-14: FL Designer Screen MATLAB (Model 1)	153
Figure 8-15: Membership Function Editor on MATLAB (PR Variable Model 1).....	154
Figure 8-16: MATLAB Rule Generator (Model 1)	155
Figure 8-17: MATLAB Rule Generator (Model 1)	156
Figure 8-18: (a) Surface Plot BT and PR on Health Status, (b) Contour Plot of BT vs PR	157
Figure 8-19: (a) Surface Plot BS and PR on Health Status, (b) Contour Plot of BS vs PR.....	158
Figure 8-20: (a) Surface Plot Age and PR on Health Status, (b) Contour Plot of Age vs PR....	158
Figure 8-21: Surface Plot Age and BT on Health Status.....	159
Figure 8-22: Surface Plot Age and BS on Health Status	159
Figure 8-23: Surface Plot BS and BT on Health Status	160
Figure 8-24: (a) Surface Plot Age and Humidity on Risk Rating, (b) Contour Plot	160
Figure 8-25: Surface Plot Temperature and Humidity on Risk Rating, (b) Contour Plot.....	161
Figure 8-26: Surface Plot Temperature and Age on Risk Rating, (b) Contour Plot.....	161
Figure 8-27: FL Model Comparison with MEWS and Physician.....	164
Figure 8-28: (a) Model 1 Surface Plot Comparison with MEWS and Physician FL Model Comparison with MEWS and Physician (b) Line Graph Comparison of Model 1 with MEWS and Physician Rating	165
Figure 8-29: Line Graph Comparison of Model 2 with Physician Rating.....	167
Figure 8-30: Battery Level % over Time for Different Phone Models Using MA.....	169
Figure 8-31: Memory Usage (megabytes) over Time for Different Phone Models Using MA ...	170
Figure 8-32: Memory Usage (megabytes) over Time for Different Phone Models Using MA ...	173

List of Tables

<i>Table 3-1: Requirements and Specifications of Chosen MCU</i>	41
<i>Table 3-2: Analogue and Digital Pin Connections on MCU</i>	41
<i>Table 3-3: Requirements and Specifications of Chosen Bluetooth Module</i>	42
<i>Table 6-1: Dataset Description for Stroke Prediction Model</i>	82
<i>Table 6-2: Input Field and Corresponding Linguistic</i>	104
<i>Table 6-3: Sample Rule Base Model 1</i>	112
<i>Table 6-4: Sample Rule Base Model 2</i>	113
<i>Table 8-1: Profiles of Participants in Study</i>	135
<i>Table 8-2: LM35 Response Times</i>	136
<i>Table 8-3: Body Temperature Readings of LM35 and Calibrated Sensor</i>	137
<i>Table 8-4: ADC output Values and Thermometer Readings for Calibration of LM35</i>	138
<i>Table 8-5: LM35 Sensor Output Using Straight Line Equation vs Calibrated Sensor</i>	140
<i>Table 8-6: Statistics for Comparison of LM35 with Calibrated Temperature Sensor</i>	140
<i>Table 8-7: Comparison in Price between Designed System and Commercial Products</i>	141
<i>Table 8-8: Table of Response Times for the Humidity Sensor</i>	143
<i>Table 8-9: Humidity Sensor Readings Compared to Weather</i>	144
<i>Table 8-10: Statistics for the Comparison of the Humidity Sensor with Weather Forecasts</i>	144
<i>Table 8-11: Glucose Readings for 3 Patients at Different Times of the Day</i>	145
<i>Table 8-12: Glucose Sensor Calibration Table</i>	146
<i>Table 8-13: IR Glucose Readings Using Straight Line Equation Compared to Calibrated</i>	147
<i>Table 8-14: Statistics for IR Sensor Compared to Commercial Sensor</i>	147
<i>Table 8-15: Regions of Clarke Error Grid</i>	149
<i>Table 8-16: Table Showing Cost of Glucose Monitoring Solution vs Commercial Products</i>	149
<i>Table 8-17: Pulse Rate Sensor and Calibrated Sensor Values for different patients</i>	150
<i>Table 8-18: Pulse Rate Sensor comparison with calibrated sensor</i>	151
<i>Table 8-19: Pulse Rate Sensor vs Calibrated Sensor Statistics</i>	151
<i>Table 8-20: MEWS Score Parameters and Ranges</i>	163
<i>Table 8-21: FL Model 1 Performance Relative to MEWS and Medical Doctor Scoring</i>	163
<i>Table 8-22: Scoring Matrix for FL Model 1</i>	164
<i>Table 8-23: FL Model 2 Performance Relative to Physician Diagnosis</i>	166
<i>Table 8-24: Scoring Matrix for FL Model 2</i>	167
<i>Table 8-25: Comparison of FL Models with Similar Literature Models</i>	168
<i>Table 8-26: Comparison of ML model Performance with Similar Literature Model Performances</i>	168
<i>Table 8-27: System Response Time Testing</i>	171
<i>Table 8-28: System Validation Testing</i>	172
<i>Table 8-29: Price Comparison of Current System with Other Commercial Systems</i>	173
<i>Table 8-30: Feature Comparison of Current System with Other Telemonitoring System</i>	174

Glossary & Nomenclature

IoT	Internet of Things
AI	Artificial Intelligence
ML	Machine Learning
SMOTE	Synthetic Mining Oversampling Technique
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
AUC	Area under the Curve
PRC	Precision – Recall Curve
ROC	Receiver Operating Characteristic
FES	Fuzzy Expert System
WoT	Web of Things
M2M	Machine-to-Machine
M2H	Machine to Human
WBAN	Wireless Body Access Network
CAGR	Compound Annual Growth Rate
WHO	World Health Organization
GPS	Global Positioning System
MA	Mobile Application
UML	Unified Modelling Language
JDBC	Java Database Connection
WA	Web Application
EJS	Embedded Javascript
MoM	Mean of Maxima

IR	Infrared
NIR	near Infrared
AP	Action Potential
GPRS	General Pack Radio Service
IPA	Intelligent Personal Assistant
GSM	Global System for Mobiles
ECG	Electrocardiogram
EMG	Electromyography
MCU	Microcontroller Unit
UART	Universal Asynchronous Receiver and Transmitter
COTS	Common off the Shelf
FHSS	Frequency Hopping Spread Spectrum
RFCOMM	Radio Frequency Communication
SPP	Serial Port Profile
RH	Relative Humidity
BPM	Beats per Minute
BMI	Body Mass Index
ADC	Analogue to Digital Converter
BT	Body Temperature
BS	Blood Sugar
PR	Pulse Rate
SQL	Structure Query Language
SDK	Software Development Kit
GUI	Graphical User Interface
IDE	Integrated Development Environment
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
RAM	Random Access Memory

MEWS	Modified Early Warning Score
API	Application Programming Interface
SoS	System of Systems
PAN	Personal Area Network
BSN	Body Sensor Network
MF	Membership Function
FL	Fuzzy Logic

Chapter 1: Introduction

The healthcare industry faces a massive systems dilemma. Fatalities due to late response rates are attributed to factors such as late diagnosis. This in turn is as a result of at risk patients (e.g. elderly or post op patients) not being monitored consistently, a lack of real time information to make pre-emptive medical decisions and overburden on hospital resources. In addition, according to the World Health Organization (WHO) (World Health Organization, 2021), conditions such as stroke can be avoided by addressing behavioral risk factors such as obesity and tobacco usage which will reduce the strain on the healthcare system. Another area within the healthcare sector with severe shortcomings is knowledge sharing amongst health practitioners and healthcare organizations. According to a WHO report, 46% of the population lives in rural areas, yet only 12% of doctors and 19% of nurses are working there (World Health Organization, 2010) which once again means limited resources and increased fatalities in these areas. Increasing the number of competent health care professionals is therefore only possible through knowledge sharing between experienced and less experienced doctors and nurses.

This research aims to deliver a smart IoT based system which can successfully close the above mentioned gaps. The Internet of Things (IoT) can be defined as an emerging paradigm whereby physical objects embedded with sensors, processors and communication hardware, are able to establish a connection with the internet through either wired or wireless protocols (Kumar et al., 2019), (Oracle, 2021). Through cloud computing and database integration, IoT devices can both store and share information between various devices in its ecosystem with minimal human intervention (Gillis, 2021). This exchange of information makes it possible for better decision making or intelligent actions between integrated devices often through the use of machine learning (ML) algorithms or fuzzy expert systems (FESs). ML is a subsection of Artificial intelligence (AI) that is based on the underlying premise that if a computer is provided with sufficient data – it is capable of developing patterns to predict or make decisions with little or no human intervention (SAS, 2021). A FES which also falls in the category of AI, is a rule base system which is used to classify outcomes which are not clearly defined in terms of Boolean logic. The data obtained from IoT frameworks can be made available to humans via an application layer such as a mobile or web application which falls in the domain of the Web of Things (WoT) (KDnuggets, 2017). The graphical user interface (GUI) in the application layer makes it possible for humans to interact with IoT systems to acquire data in real time.

According to the CISCO annual internet report (Cisco, 2020), nearly two-thirds of the global population (5.3 billion people) will have internet access by the year 2023. The report also suggests that around 70% of the world population will have mobile connectivity by the year 2023 with machine-to-machine (M2M) which also includes IoT applications growing from 33% to 55% in 2023. It is also expected that faster networks with reduced latency like 5G will be utilized in about 10% of all global mobile devices. These statistics are promising and therefore suggests that IoT frameworks integrating mobile smartphones as gateways is a lucrative area of development.

One such industry that stands to gain from the implementation of IoT frameworks is the healthcare industry. According to Grandview research (Grand View Research, 2019), the IoT healthcare market is expected to have a surge in compound annual growth rate (CAGR) of 19.9% for 2019-2025. Figure 1-1 illustrates the progressive growth of the IoT healthcare market in the US. One of the largest revenue contributors to this market share is telemedicine of which smart wearable devices (wireless body network sensors) for the application of remote monitoring is a major subcategory. Coupled with intelligent capabilities like FESs and ML for predictive analytics, these devices serve as a fundamental component of the 5th industrial revolution (5IR) as it aims to assist rather than eliminate health practitioners in their daily duties (Sarfranz et al., 2021).

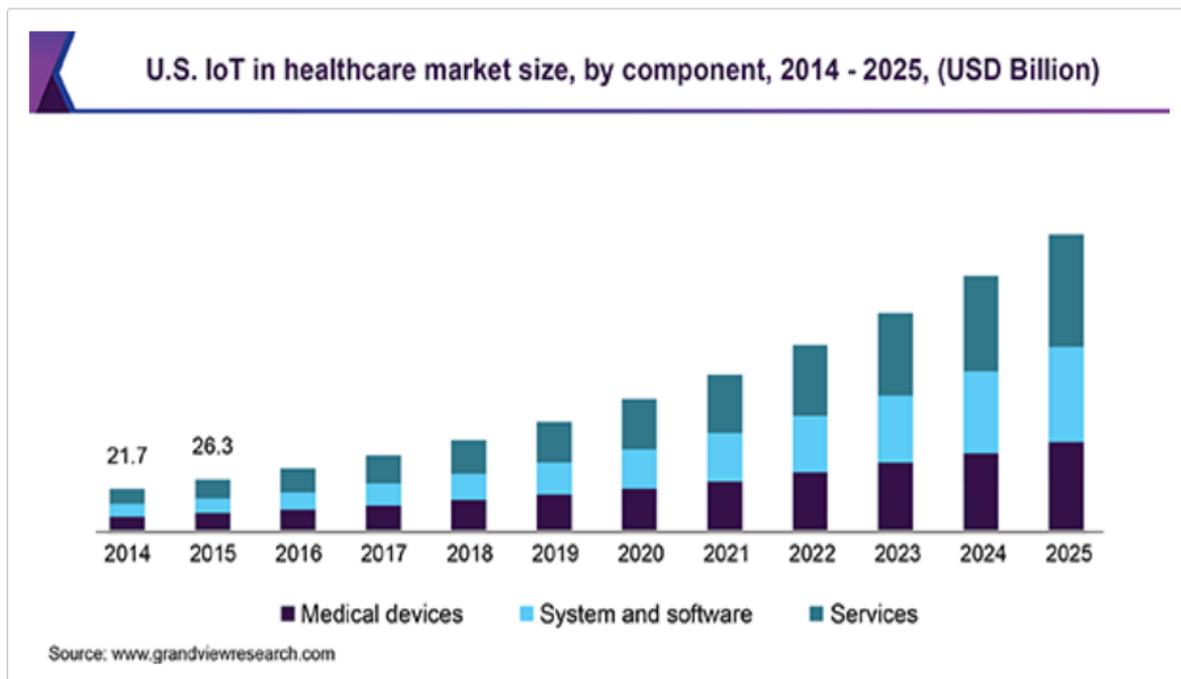


Figure 1-1: U.S. IoT healthcare market size

The proposed IoT system therefore aims to effectively aid health practitioners and patients to preemptively make better healthcare decisions. Thus preventing fatalities and burden on hospital resources through a remote monitoring IoT system. This research also aims to create a prototype system aimed to be used as a complementary service within the healthcare domain to effectively speed up healthcare response time. Such a system will employ an integrated approach of bringing technology closer to the users (patient and health practitioners). The development of a remote monitoring device with IoT, ML and FES capabilities will ensure real time and predictive data is made available to both patient and health practitioners.

The expected result of deploying such a system is therefore to reduce hospital stays, burden on health infrastructure and exacerbated health conditions caused by the late detection of communicable and non-communicable diseases. This while simultaneously promoting early

healthcare awareness with patients. To address proactive monitoring of non-communicable diseases, the proposed ML model in the employed system aims to identify high risk patients as a result of behavioral risk factors. Patients can then proactively monitor their health status or health practitioners can identify which patients require more attention.

The proposed system will utilize the Global Positioning System (GPS) tracking abilities of smartphones which will enable early ambulatory dispatch during emergency cases.

The wearable devices within the IoT system will make it possible to monitor patients remotely and non-invasively placing little or no burden on hospital resources (health practitioners or equipment). Health practitioners can then access this information to prevent emergency room visits through a proactive approach to healthcare, observe patient trends for risks and view historic patient data to identify anomalies. Users of these devices also stand to benefit by becoming more aware of their general health status and can therefore make better lifestyle changes to facilitate a better health outcome.

In addition, employing such as system will break barriers between levels of patient care – allowing urban doctors to offer their services to rural areas without relocating. This while also empowering rural doctors with knowledge in healthcare best practices. In addition, the development of (fuzzy logic) FL systems with a rule base created from expert intuition can improve accuracy and precision for diagnosis without the presence of trained medical doctors.

Lastly the benefit of the proposed system extends to the utilization of data availability. Living in the data age implies knowledge is power provided it is used responsibly. With IoT systems and cloud integration– the system will allow for the accessing of historical information from an array of patients and sensors allowing experts to make connections between anomalies and trends. This while contributing effectively to the future of healthcare education.

1.1. Problem Statement

The healthcare systems of especially developing countries are becoming excessively burdened. This puts a strain on both hospital infrastructure as well as medical personnel. There are also many medical cases which are identified at severe stages often leading to deaths, chronic ailments or prolonged hospitals stays. To break this cycle there is a need for a proactive approach towards remote healthcare monitoring. With the advent of the 4IR and 5IR, technologies such as IoT, predictive modelling, MAs and WAs offer the opportunity to reduce or eliminate hospital stays through remote monitoring and preventative healthcare management. Predictive models offer the medical personnel the foresight into potential medical conditions their patients may be experiencing before conditions escalate. Such systems also allow for bulk diagnosis which reduces response time especially with scarce resources. In addition, real time monitoring and tracking of patients using mobile GPS technology, allows for early deployment of ambulatory services during emergency situations. Through the use of wearable sensors, vital patient

information can be sent through to doctors on a regular basis to enforce healthcare monitoring long after a patient leaves the clinical setting. This means that high risk patients may potentially have the option of early discharges and consequently remote post-operative care becomes a high probability.

1.2. Research Questions

The following research questions will be addressed in this dissertation:

- 1) How can an IoT framework coupled with wearable sensor technology and smartphone capabilities create a reliable channel for the acquisition of a patient's vitals?
- 2) What sensors would be required to create a WBAN to provide enough information to give doctors and patients a wide view of healthcare data?
- 3) How can we use IoT, a WA and MA technology to give doctors foresight into their patients' state of health and help them practice preventative healthcare?
- 4) Which predictive algorithms can help predict potential health conditions of patients?
- 5) Is it possible to provide health practitioners with the patient's location so that they can dispatch ambulatory services during emergency situations?
- 6) Can an IoT- based healthcare monitoring system aid in reducing the chain of survival execution time?

1.3.Literature Review

The previous sections highlighted the forecasted growth within the healthcare telemonitoring sector and identified shortfalls that need to be addressed to improve the healthcare system. This part of the chapter will now explore the variations of healthcare monitoring system designs that have been researched and developed. A critical assessment of the gaps within this research area will then be done with the aim of identifying possible areas for improvement. The chapter starts of by discussing the origins of telemonitoring systems and then progressively discusses researched systems which have been categorized according to the technologies they have utilized and their increasing complexity.

1.3.1. The Origin of Telemonitoring Systems

The origin of telemedicine dates back to the 19th Century, however the first published accounts took place in the 20th Century which describes the case of electrocardiograph transmission over telephone wires (World Health Organization, 2010). The area telemedicine as we know it today, started forming in the 1960's primarily driven by the military and space industries along with early adopters utilizing commercially available equipment (Craig & Patterson, 2005), (Ruiz et al., 2008).

The following sections aim to review the various IoT based telemonitoring device adaptations utilized in the medical field including their links to predictive modelling methods in the ML and FES spectrum. The various technologies employed in this space can be broken down into high level groups that essentially describe the key technologies that the specific system utilizes. Below describes the key high level characteristics that most smart IoT based devices employ:

- IoT based framework (server or mobile gateway)
- Application layer (web or mobile)
- ML capabilities
- FES capabilities
- Location tracking
- WBAN for retrieving vitals
- Non-invasive sensor technology (e.g. IR sensing)
- Wearable/ real time remote monitoring capabilities
- Cloud/database integration

Based on the above characteristics, the following healthcare applications will be discussed 1) IoT based telemonitoring systems, 2) ML models for disease diagnosis, 3) FES based predictions for health diagnostics, 4) IR sensor based devices for non-invasive telemonitoring, 5) IoT telemonitoring systems employing FE systems, 6) IoT telemonitoring systems employing ML and 7) IoT telemonitoring devices utilizing a hybrid of FES and ML capabilities which will be the focus of this research.

1.3.2. Non-IoT Telemonitoring Systems

This section discusses telemonitoring devices that do not possess an IoT based infrastructure. These devices can generally be grouped by their data transmission mechanism which is usually technologies like Bluetooth or Zigbee. These types of telemonitoring systems enable wireless transmissions but lack remote monitoring capabilities. This is due to the absence of long range data transmission technologies like Wi-Fi or Global System for Mobiles (GSM). Zanoguera et al. (2019) developed such as system using an ECG sensor module with Arduino Nano microprocessor, a data logger shield and a Bluetooth module to facilitate the transmission of data from the ECG sensor to the data logging shield. The set-up can be seen in Figure 1-2.

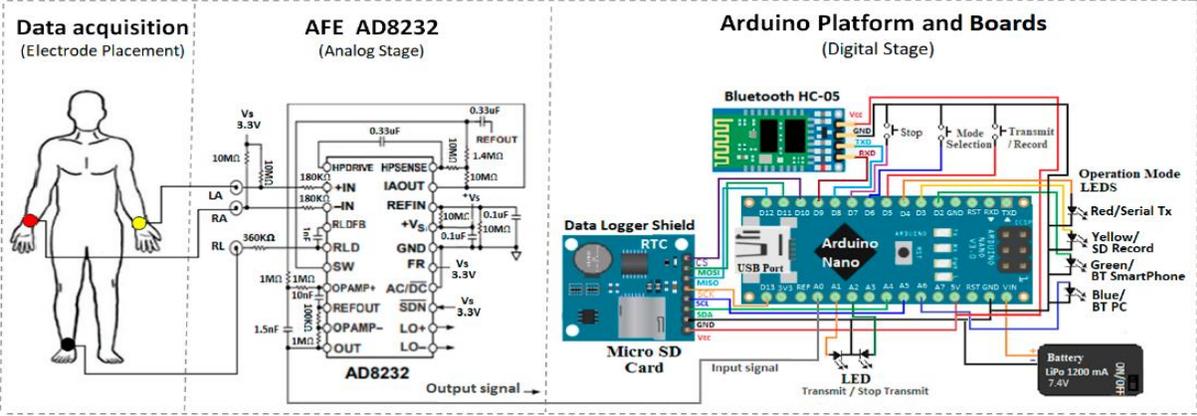


Figure 1-2: Block Diagram of Bluetooth Based Telemonitoring System

A wearable heat monitoring device was developed by Fang et al. (2019) to enable temperature monitoring via heat flux modelling. Temperature sensors were used to obtain temperature readings and filtered via a microcontroller unit (MCU) before finally being transmitted to a WA for monitoring and modelling. The system architecture is shown in Figure 1-3.

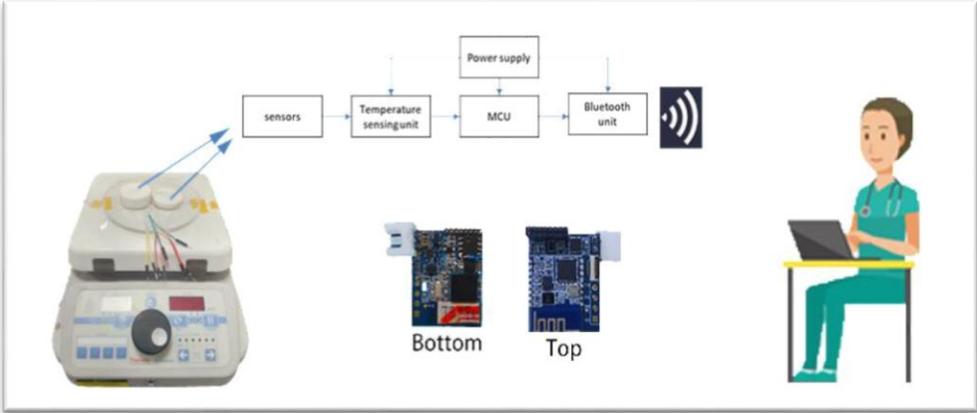


Figure 1-3: Temperature Monitoring on WA Using Bluetooth

An example of ZigBee technology being used as the transmission technology can be illustrated by the prototype put forward by Ehresh et al. (2020). A portable ECG monitoring device is developed using an ECG sensor, filtering modules and an Atmega 328P processor interfaced with a ZigBee module for transmission to a GUI on a computer. Figure 1-4 illustrates the component set-up. A variation of this ECG monitoring system using Bluetooth can be seen in the work of Roihan et al. (2019).

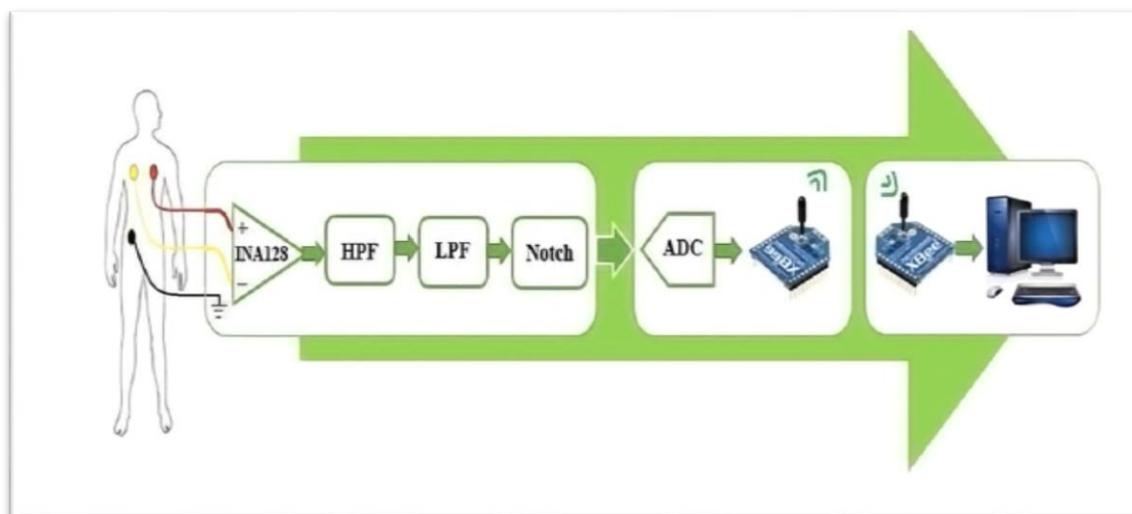


Figure 1-4: Zigbee Technology Being Used For Data Transmission

These telemonitoring devices offer advantages over traditional wired devices. Armstrong (2007) for example highlights the benefits of wireless monitoring of users in applications like sports where unobtrusive methods of data acquisition that doesn't inhibit movement is required. However, this type of technology lacks the remote monitoring capabilities which makes it unsuitable for cases where health practitioners need to monitor patients outside the hospital setting.

1.3.3. Non-Invasive Glucose Monitoring Systems

Types of non-IoT based systems with their sensor integration have been looked at. Although most data acquisition in the form of patient vitals do not require invasive methods, there are cases where diagnosis is usually done via direct blood or tissue sampling (Mete et al., 2012). One such case is the detection of Diabetes. This however poses a risk in the healthcare sector as people are often dissuaded from doing regular glucose level tests due to the invasive nature of the testing, high expense and unnecessary exposure to infectious disease (Daarani & Kavithamani, 2013). Several applications have therefore looked at non-invasively determining glucose levels by means of technologies such as infrared (IR) sensors and ML techniques. This section expands on some of these implementations:

1.3.3.1. Capacitive sensor for non-invasive measurement of blood glucose

Dutta et al. (2019) proposes a non-invasive blood glucose level approximation using capacitive sensors. The technology works on the premise that capacitance varies linearly with blood glucose concentration. The device makes use of an operational amplifier circuit to measure capacitance in terms of output voltage. A ratio of output voltage to input voltage is then used to determine a linear representation of the ratio relative to the concentration. Figure 1-5 illustrates the relationship between the ratio of output voltage/input voltage and blood glucose concentration.

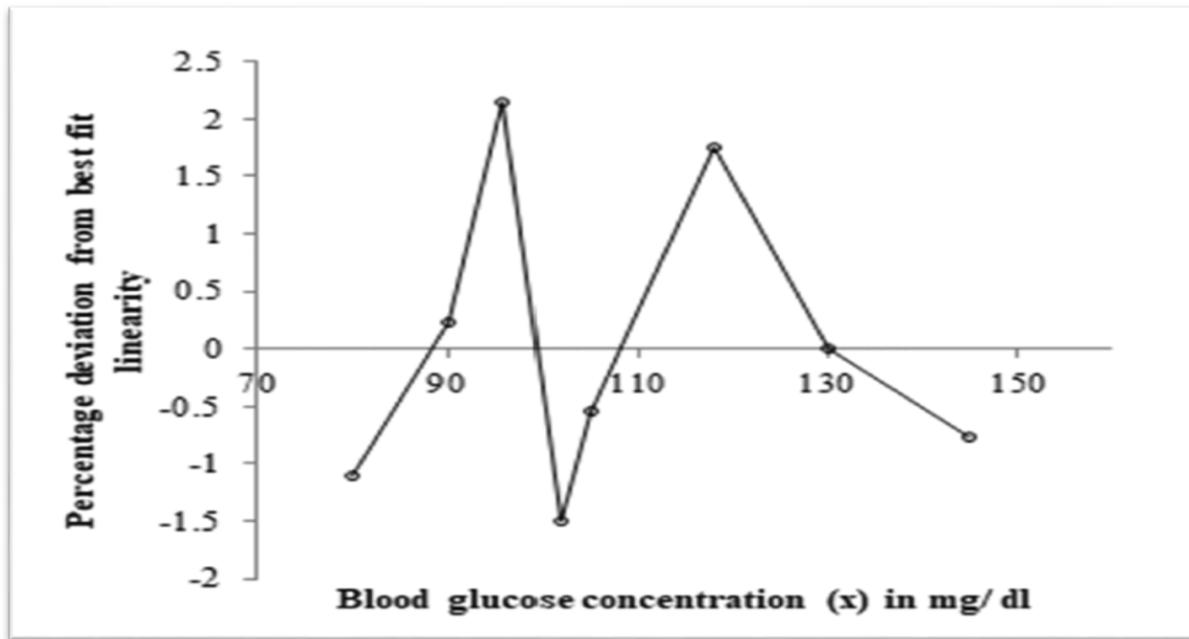


Figure 1-5: Relationship between Output Voltage and Glucose Levels

1.3.3.2. Near Infrared (NIR) Sensors for the Determination of Glucose Levels

The use of an IR sensor and receiver to determine glucose levels is another non-invasive technique that has been greatly explored (Apar & Dr Autee, 2019), (Daarani & Kavithamani, 2013), (Sagar, 2020), (Alarcón-Paredes et al., 2019), (Dr Kavitha et al., 2019), (Sari & Luthfi, 2016), (Haxha & Jhoja, 2016), (Prawiroredjo & Engelen, 2016). The technology works by placing the fingertip between the IR sensor and receiver. When the NIR light passes through the fingertip, a portion of the light gets absorbed by the glucose molecules present in the blood. A greater amount of glucose molecules present in the blood will result in more NIR light being absorbed hence this fundamental relationship can be used to establish the blood glucose concentration levels. Truong et al. (2020), Gusev et al. (2020) and Manurung et al. (2019) utilize the IR method discussed with ML and artificial neural networks to improve their model accuracy. Figure 1-6

shows a diagram illustrating the basic set-up for a diabetes IR detection system (Apar & Dr Autee, 2019).

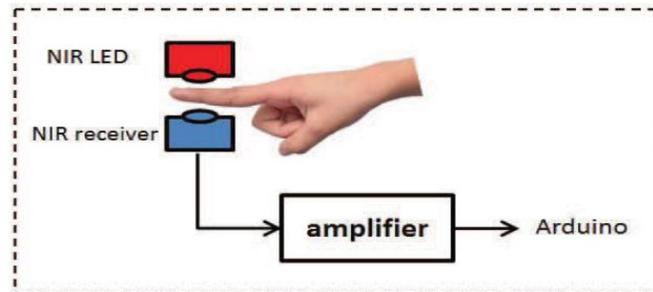


Figure 1-6: Set-up for Diabetes IR Detection System

1.3.4. GPS Enabled Telemonitoring Systems

GPS is a location tracking service that utilizes GPS satellites to send signals to GPS receivers to acquire their speed, location and direction (Maddison & Mchurchu, 2009). According to NASA, the origins of GPS can be traced back to the Sputnik era where scientists used the principle of the “Doppler effect” for satellite tracking using shifts in radio signals. Maddison & Mchurchu (2009) offered a differing viewpoint and suggest that the first GPS was developed by the US Department of Defense for military applications.

Many of the telemonitoring devices researched have the element of GPS locations. Munir et al. (2013) for example talks about the importance of GPS devices for the monitoring of elderly patients health status and to prevent them from getting lost. Satayanarayana et al. (2013), designed a system that uses an ARM/911 microcontroller with a GPS module that focuses on creating a two way communication between hospital personnel and ambulatory services. Zhang & Lu (2009) showed how GPS and general packet radio service (GPRS) can be used for ECG monitoring and response in cases of cardiac failure. Another similar system was developed by (Fang & Kun, 2007) that focuses on monitoring out of hospital cardiac patients.

These systems form an integral part of modern day telemonitoring systems as they allow for emergency reactions to proceed without the intervention of the patient.

1.3.5. IoT Enabled Telemonitoring Systems

A number of IoT based telemonitoring systems have been developed with the aim of making remote monitoring of patients possible (Rghioui et al., 2020), (Pacchierotti, 2018), (Beach et al., 2018), (Nazar et al., 2017) (P.Ortiz et al., 2018). Many of these have employed multiple sensors through a wearable set-up for the data acquisition of fundamental patient physiological data. This info is subsequently passed to a gateway (mobile or server based) where temporary storage is possible before transmission to a cloud interface. The data which is stored in the cloud database

can later be retrieved via a WA or MA for use by health practitioners. The variations of the two IoT gateway configurations will be explored below.

1.3.5.1. Mobile as a Gateway Configuration

Numerous IoT based applications have explored the use of a smartphone as a gateway. According to khaddar & Boulmalf (2017), the smartphone forms the perfect IoT device due to its ability to interact effortlessly with various devices and sensors through protocols such as Bluetooth, near field communications (NFC), Wi-Fi etc. Many researched devices (Rodrigues et al., 2013), (Kominos & Stamou, 2006), (Horta et al., 2013), (Schrader et al., 2010), (Virone et al., 2006), for example have looked at utilizing IoT with intelligent based personal assistants for healthcare monitoring using the smartphone as a gateway for communication. Santo et al. (2016) developed a mobile health solution that utilizes a body sensor network (BSN) to collect the patient's location, fall detection and pulse rate data in real time for transmission to an intelligent personal assistant (IPA) via the smartphone. The IPA is then able to manage the data and activate alarms in the case of emergencies. Figure 1-7 illustrates the proposed Bluetooth connection service to enable data transferal from the smartwatch to the smartphone.

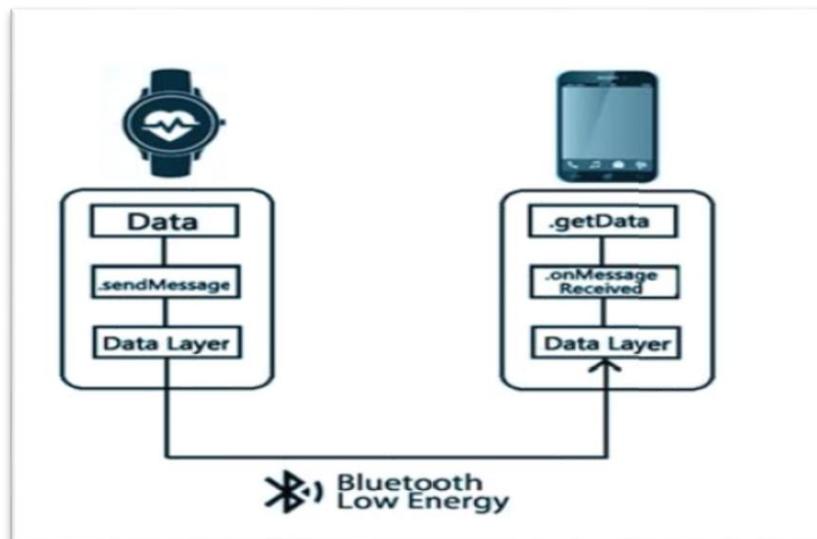


Figure 1-7: Data Transfer from Smartwatch to Smartphone IoT Gateway

Bellagente et al. (2016) introduced the concept of “Smartstone,” a low cost and easy to use smartphone to act as an IoT gateway to create a telemonitoring solution targeted towards addressing the needs of senior citizens. The system acquires data through various sensors located on wearables and personal area networks (PANs). Cloud integration and transmission is then established using a Wi-Fi or GSM signal. A remote server located on the cloud then allows for easy storage and retrieval of information by patient or doctor through Wi-Fi and GSM connections so that data can be viewed or analyzed via an application layer such as a MA or WA. Figure 1-8 illustrates this system.

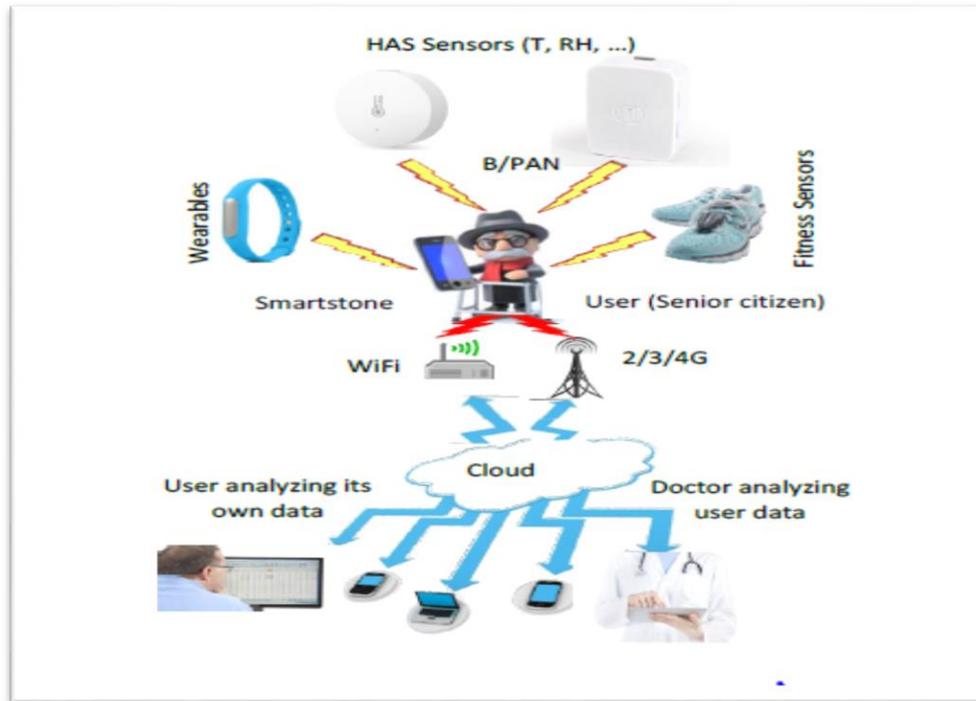


Figure 1-8: "Smartstone" IoT Architecture

1.3.5.2. Other IoT Gateway Configurations

The flexibility in the use of IoT gateways have led to other gateway mechanisms being explored in the area of healthcare telemonitoring (Brezulianu et al., 2019), (Abed, 2020), (Ozkan et al., 2020). Abdul-jabbar & Abed (2020), developed such a system which was used for real time monitoring of patients with pacemakers. The system made use of an ESP Wroom 32 module which served as both the microcontroller and Wi-Fi module enabling IoT capabilities. The system employed various sensors viz. ECG, temperature, heart rate and Hall Effect sensor (magnetic field detection) to capture patient vitals. This data was then filtered and amplified by the ESP Wroom 32 module before being sent directly to a browser site which can be viewed by the patient on their smartphone or computer. The proposed IoT infrastructure is illustrated in Figure 1-9.

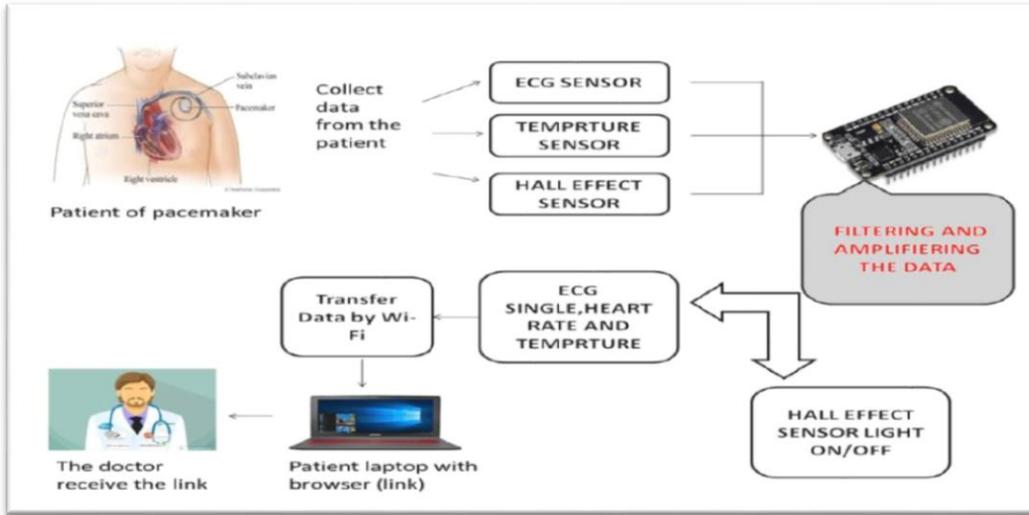


Figure 1-9: ESP32 IoT Based System for Pacemaker Monitoring

Chandini & Kumar (2018) developed a similar set-up for monitoring patient ECG and temperature trends but instead used a Raspberry Pi module to serve as the gateway between the sensors and cloud platform. Also, Sanfilippo & Pettersen (2015) opted to create a set-up to monitor various patient vital signs through a fusion of sensors so that haptic feedback (via LED and vibration motor) can be given when dangerous levels are exceeded. The system utilizes the e-Health sensor shield to connect sensor information and then transmits this information via Wi-Fi to the cloud computing center for data storage. Data can then be accessed and reviewed by health practitioners via mobile or web interfaces. The system architecture is illustrated in Figure 1-10. The system makes use of a three tiered hierarchical server to maximize performance and scalability.

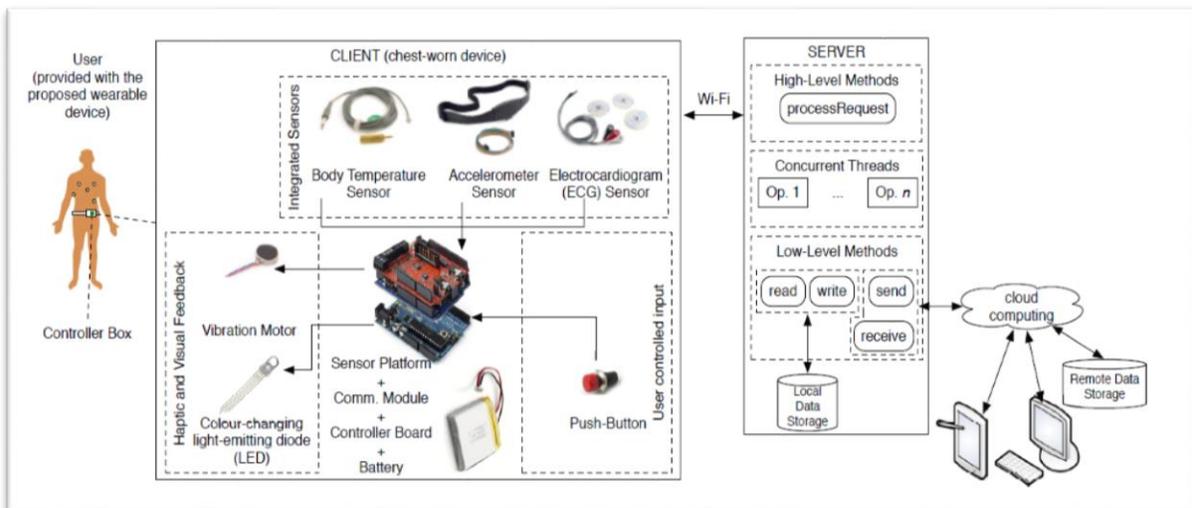


Figure 1-10: System Architecture with Raspberry Pi IoT Gateway

Another variation was by Chamundeeswari et al. (2019) who utilized a system of ECG detection and monitoring using a heart rate sensor, Atiny 85 microcontroller and ESP8266 Wi-Fi module to establish an IoT framework.

These types of gateway devices can be an effective mechanism for the implementation of IoT systems however lack the benefits of smartphones which are widely available and easily accessible by many individuals. Smartphones make it possible to implement health monitoring systems without burdening users with additional technology expenses. In addition, Boulos et al. (2011) suggested that smartphones generate a great amount of engagement from individuals of all ages which makes it a perfect medium for health applications as opposed to other server based gateways.

1.3.6. AI and Fuzzy Logic in Healthcare

A vast number of research topics have focused on AI (ML and deep learning) and FES models applied to disease diagnosis. These systems were either primarily AI focused, primarily FES based or a hybrids of the two. The models were developed with the fundamental aim of creating improved prediction models, however have not been integrated with an IoT platform which enables remote monitoring. For the purpose of highlighting the effectiveness of ML and FES models in the area of healthcare diagnostics, these will be explored further.

1.3.6.1. ML and Deep Learning Algorithms

A number of developments have been made in the ML and deep learning space for predictive diagnostics in healthcare. Many of these developments have focused on analyzing ECG trends to diagnose various cardiovascular conditions. Aspuru et al. (2019) for example carried out ECG waveform segmentation using a linear regression model and thereafter utilized classification algorithms which led to a 95% sensitivity. Ribeiro et al. (2020) developed a deep learning model for a 12 lead ECG and was able to outperform resident doctors with F1 scores above 80% and specificity over 90%. Majumde et al. (2018) also worked on developing a model for cardiac arrest detection using a decision tree algorithm. Another innovative application of ML in healthcare can be attributed to the work of Simjanoska et al. (2018), who utilized ECG data and a combination of a classification and regression models to estimate blood pressure readings.

1.3.6.2. FES Models

FL based applications have also been used in an array of healthcare applications. Duodu et al. (2014) created a FL rule based solution for the detection of Malaria which attained a 15.4% accurate diagnosis which outperformed the diagnostic accuracy of doctors for the same dataset. Oad et al. (2014) employed a fuzzy rule based approach for the prediction of heart disease which

matches similar neural networks and decisions tree algorithms. Other fuzzy logic systems developed by Djatna et al. (2018), Kasbe & Pippal (2017), Desai et al. (2021), Buczak et al. (2015), Jindal, et al. (2020) have been linked to the diagnostics of stroke, heart disease and Renal cancer.

1.3.7. IoT & AI based Telemonitoring Systems

To this point technology implementing Bluetooth and IoT in the areas of telemonitoring systems have been looked at. Another growing area of research in alignment with the 4IR and 5IR is IoT and AI fused systems. Devices that employ this type of infrastructure utilize the powerful area of AI (deep learning and ML) with IoT to bring predictive analytics to the disposal of the healthcare sector. AI inclusive of ML are bringing about massive paradigm shifts in the healthcare sector due to the availability of healthcare data and strides in analytical techniques (Jiang et al., 2017).

Majumder et al. (2019) created an energy efficient predictive cardiac arrest system with IoT and ML functionality. The system utilized a decision tree ML model and utilized a smartphone as the IoT gateway and application layer. XU (2020) utilized an ECG monitoring solution with ML classification models and feature extraction. Shao et al. (2020), utilized a classification model trained on 17 different ECG features for the detection of Atrial Fibrillation using a LoRa IoT gateway and Fog AI interface as shown in Figure 1-11.

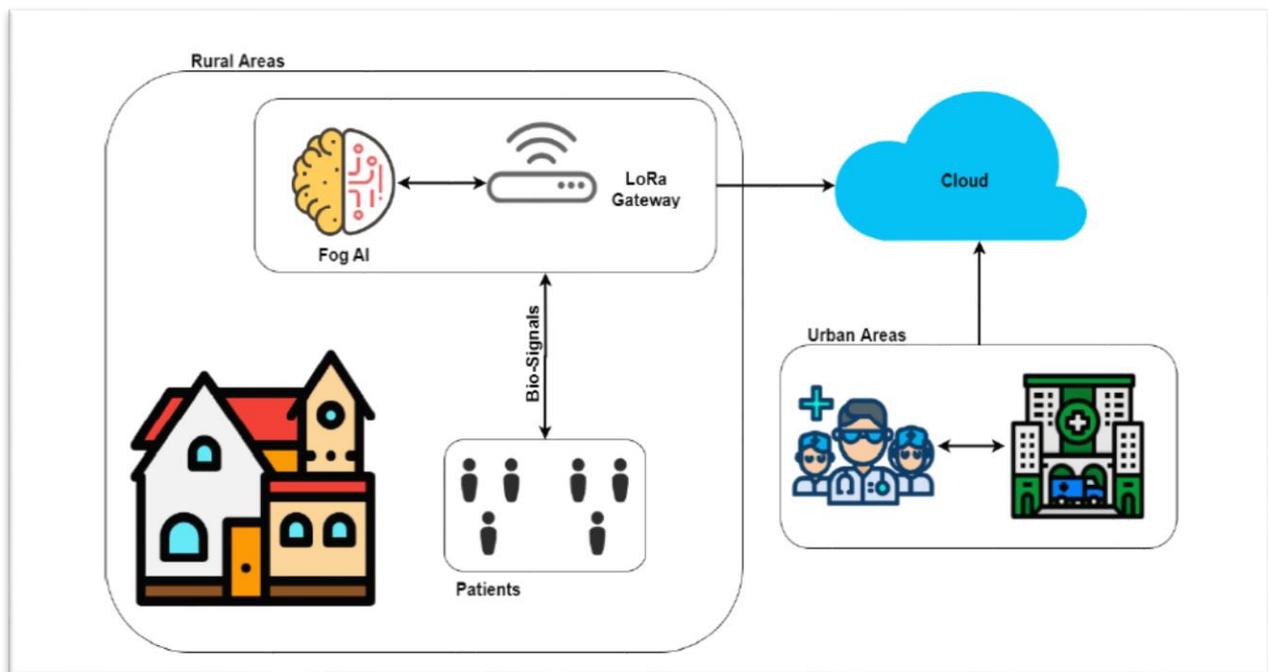


Figure 1-11: IoT System for Atrial Fibrillation using classification model & LoRa Gateway

ML based telemonitoring systems offer great value to the healthcare industry, however they lack the intuitive expertise that fuzzy logic brings to the table. Hence, hybrid models may be needed to overcome its shortcomings.

1.3.8. IoT & FL Based Telemonitoring Systems

The effectiveness of FL systems were seen in section 1.3.6. A further adaptation of such FL based diagnostic systems is their integration with IoT (Santamaria et al., 2016), (Neeralagi, 2017) (Al-Adhab et al., 2016). Hussain et al. (2016) developed a smart home healthcare system that illustrated this concept. The system utilizes a body area network to acquire sensory information and uses a smartphone as a gateway for cloud integration. A FES then analyzes the data based on given rules to determine the patient’s risk for cardiac disease. A drawing of the infrastructure is shown in Figure 1-12.

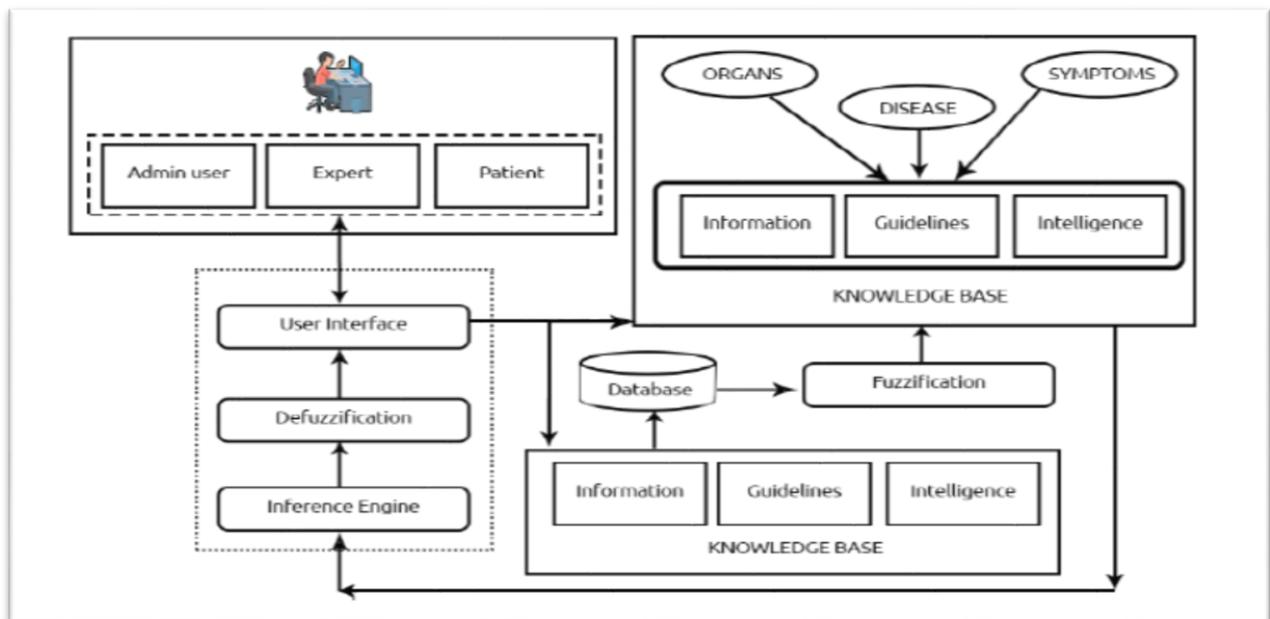


Figure 1-12: IoT HealthCare System with FES

FL systems are an effective means to determine the “how” aspect of predictions through its inference rules, however are often intuitively based so lack the accuracy that ML models may bring. In addition, FL systems lack the ability to generate patterns from large datasets. Hence, these systems may also benefit from hybrid implementations.

1.3.9. IoT Fuzzy Logic and AI Hybrid Telemonitoring Systems

Harvard business review suggests that the future of healthcare will see advancements in complex algorithms that make incredibly accurate predictions about our health status (Burt & Volchenboun, 2018). It is no surprise that predictive models are being combined to form hybrid systems which provide a greater accuracy for disease diagnostics. One such combination which has been explored is the combination of FL and AI determination models. When these hybrid type systems are formed, they produce a more effective solution to the problem (Sattar et al., 2019).

Hameed et al. (2020) illustrates such a concept with an IoT home healthcare system which utilizes a fuzzy neural network to assist doctors by monitoring sensor data and alerting them of potential patient issues. The combination of AI (Deep learning and ML) and FL systems complement each other well. While AI techniques like neural networks and ML are good at perceiving patterns in the data (Linn & Lee, 1991), they don't explain how end decisions are derived. FL can better explain this through inference rules (Nelles, 2001) but these rules are difficult to establish without the help of techniques like neural networks (Jang, 1993). Figure 1-13 shows the integration of the hybrid fuzzy neural networks proposed by Hameed et al. (2020).

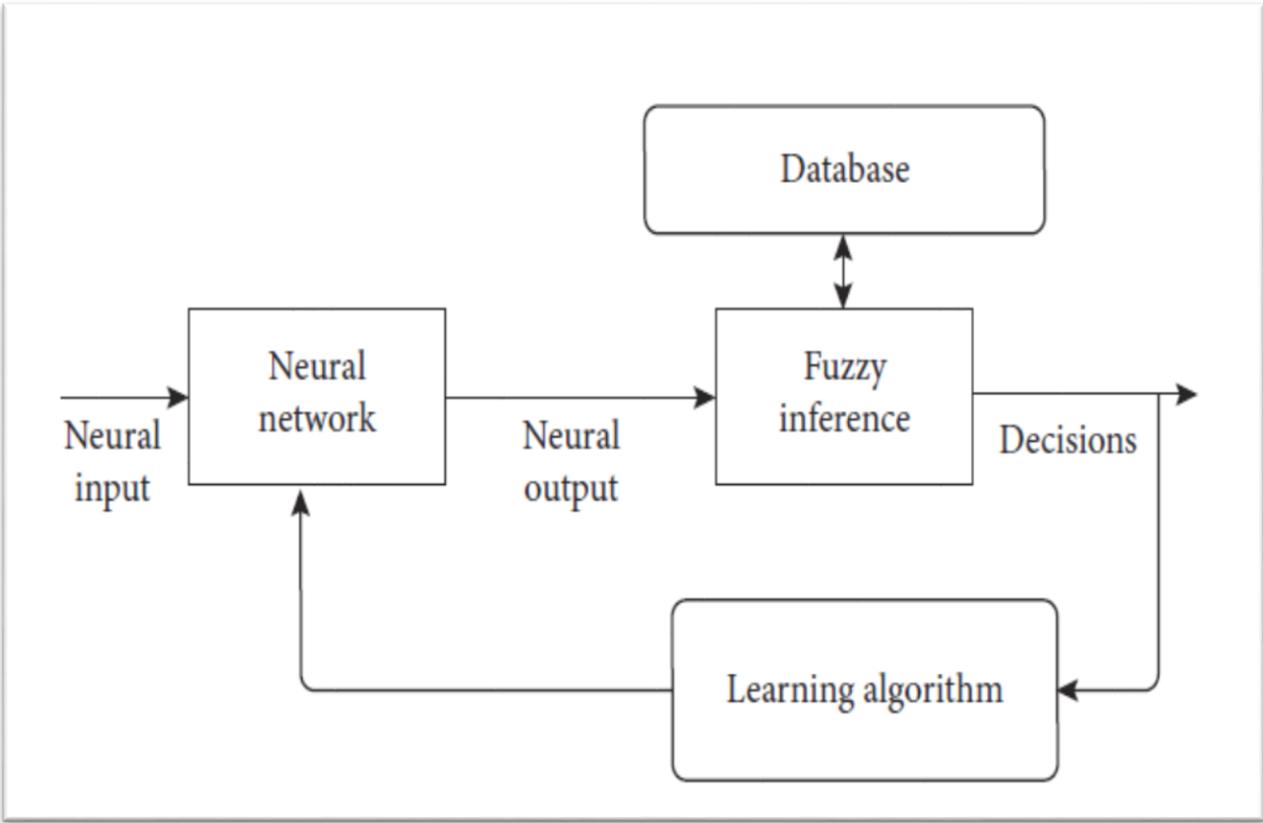


Figure 1-13: Hybrid Fuzzy Neural Network System

Despite the massive progress that has been made in the area of IoT based hybrid ML and FL based systems, there still exists massive room for creating solutions that utilize more sensory inputs combined with healthcare intuition to predict certain conditions. For example, although FL systems with ML capabilities have been integrated with IoT for predictive diagnosis of cardiovascular diseases (Khan, 2020), (Mohammad & Fahad, 2020) little work has been done in creating holistic systems for predicting other non-communicable diseases. One such example is stroke prediction which has been modelled using both ML and Fuzzy logic (Chun et al., 2021), (Islam, 2018) but never integrated with IoT capabilities (Yun-Hsuan Chen, 2021). A remote monitoring infrastructure based on IoT can therefore greatly benefit from stroke prediction model usage. Continuous real time input from a wide sensory network can stand to benefit the accuracy and precision of such models. In addition, the development of multi-sensory input to create a combination of different risk probabilities/ratings using FL can greatly improve the way doctors and patients handle health monitoring. The basis of further exploration in this research will therefore be aimed at creating a holistic IoT system integrated with ML and FES models to create a remote monitoring system with authentic predictive capabilities.

1.4. Hypothesis Statement

The development of an IoT based telemonitoring system with models for predictive capabilities will reduce burden on healthcare resources and assist with reduced fatalities due to late diagnosis.

1.5. Research Contributions

The following research contributions were achieved in this research:

- A multi-sensory WBAN integrated with an IoT infrastructure which allows for multiple combinations of variables to be modelled and monitored to determine risk on patient health.
- An intuitive FL model that utilizes a unique combination of patient vital signs to allow for pre-emptive medical care without a doctors supervision.
- The use of humidity sensor input variable in a FES model used for patient monitoring.
- The development of a second FL model to track the effect of environmental factors on human health.
- Multiple rule base FESs implemented on an Android MA.
- Integration of a ML based stroke prediction model with an IoT framework which allows for remote prediction and model consumption.
- A Java database connection (JDBC) with MS Azure which reduces application programming interface (API) connection costs associated with the typical Microsoft WA services method.
- A feature rich MA with a lower power consumption.
- A feature rich MA with a low memory usage.

- IoT System with a double cloud platform (Firebase and Azure) integration for flexibility and efficiency of data transmission.
- A two interface MA for patient and doctor machine to human (M2H) interaction.
- A Single MA integrated with a primary and secondary Firebase database connection.
- Implementation of both a MA and WA within an IoT infrastructure.
- The use of the MEWS rating within an IoT remote monitoring application.

1.6.Objectives and Outline

The objectives of this research topic is to (1) develop a multi-purpose wearable sensory device to acquire reliable patient vitals under remote operating conditions, (2) develop an IoT based data pipeline for the storage and transmission of patient data during emergency situations, (3) develop user friendly application layers (mobile and web) for accessing and viewing of patient vitals and location and (4) utilize ML and FES models to analyze patient parameters in real time to aid health practitioners in early prediction of patient diagnostics.

The research work was composed of the following tasks:

1. Define the stakeholders and their requirements and subsequently identify the minimal system requirements to satisfy stakeholder needs. Once the system architecture is defined from the stakeholder requirements, an evaluation of the proposed conceptual designs using the Pugh matrix will be evaluated (**chapter 2**).
 - Outcome: An early indication of the proposed high-level system architecture from the perspective of the customer/end-user. Identification of critical characteristics and components that should be prioritized in the subsequent design stages which will reduce potential downstream costs associated with poor preliminary considerations. Lastly the identification of the optimal design that meets all stakeholder requirements in terms of usability, IoT connectivity and physical characteristics.
2. An in depth discussion of the design of the wireless body access network (WBAN) which serves as the sensing layer in the design (**chapter 3**).
 - Outcome: A completed WBAN device capable of reading patient vitals and transmitting data over Bluetooth to the MA.
3. Development of the MA using Java on the Android Studio integrated development environment (IDE) and the WA using node.js, JavaScript, HyperText Markup Language (HTML) and Cascading Style Sheets (CSS). Then the integration of the MA with the WBAN device using broadcast receivers (**chapter 4**).
 - Outcome: A completed MA with a patient and doctor interface able to receive data via Bluetooth from the WBAN and subsequently store this data in a SQLite database for future cloud transactions. Also a completed secondary WA layer ready to receive and display information on a hospital server.
4. Setting up the cloud infrastructure layer and further integration with the rest of the system (MA and WA) (**chapter 5**).

- Outcome: Firebase NoSQL database set-up and integration as well as MS Azure SQL database set-up and integration. The resulting ability of the MA and WA to receive and transmit data to and from the cloud layer.
5. Development of a stroke prediction model using the MS Azure ML classic studio service and an online Kaggle dataset. Thereafter the development of two FE systems for patient monitoring using the Mamdani approach. Triangular membership functions (MFs) will be used for creating fuzzy sets and the Mean of Maxima (MoM) method will be used for defuzzification (**chapter 6**).
 - Outcome: ML model for stroke prediction with high precision and accuracy using the Boosted decision tree binary classification model. Also two FE systems to model patient health using the Mamdani approach.
 6. Use case scenarios for each stakeholder will be explained indicating critical functions they need to carry out to ensure the smooth transition of information through the data pipeline (**chapter 7**).

Outcome: Standard process that users can follow to effectively utilize the system
 7. Applying the system to a series of clinical trials and tests to gauge its performance compared to other systems (**chapter 8**).
 - Outcome: Overall system performance and results.

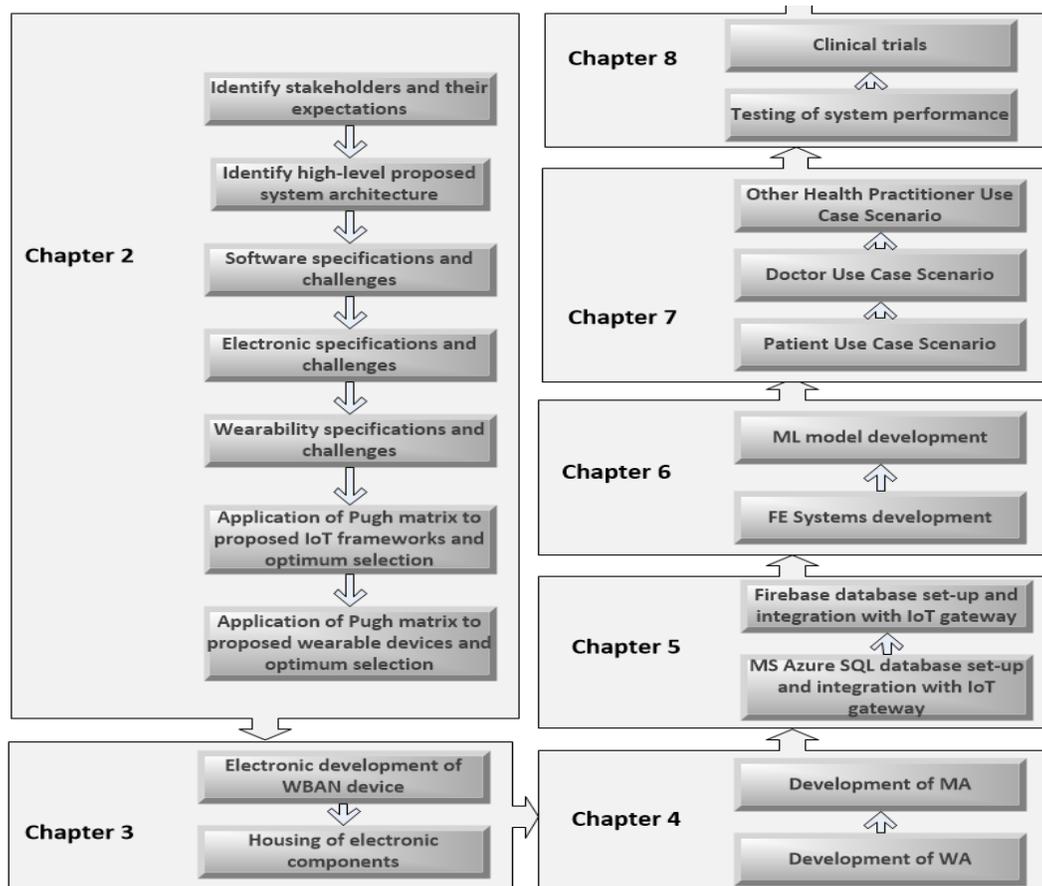


Figure 1-14: Dissertation Objectives & Outline

1.6. Chapter Summary

This chapter began by explaining the concept of IoT systems and how when integrated with powerful ML and FL models, can revolutionize healthcare telemonitoring systems. The Cisco annual report was used to show that the expected growth of M2M solutions including IoT was expected to grow by 33% to 55% by the year 2023 thus indicating the need for sectors like healthcare to get on board utilizing such solutions. The report further showed that around 70% of the population is expected to have mobile connectivity by the year 2023 which highlighted the suitability of using smartphones as IoT gateways. This findings by Cisco was further supported by Grandview research forecasting a CAGR of 19.9% in the IoT healthcare market for the period of 2019-2025. The potential of telemonitoring with smart wearable devices was also identified as a powerful pillar of 5IR which could potentially revolutionize the healthcare industry. The possibility of using these systems to reduce emergency response time, for the early identification of disease as well as to reduce the burden on healthcare facilities was highlighted. The proposed research topic was then introduced to be the development of an integrated IoT system utilizing a combination of inputs to provide monitoring and diagnostics that was an extension of previous research. An overview of the topics to be discussed was then highlighted. The chapter ended with a literature review explaining the limitations of previous research and how the proposed solution aims to bridge the gaps in these shortfalls.

Chapter 2: Requirements Analysis, Design Specifications & Concept Development

Chapter 1 examined the various types of telemonitoring systems that have been researched and designed as well as the need for improved healthcare telemonitoring systems. It was found that a gap exists to use multi-sensory inputs to create new FL and ML models that can be integrated into an IoT framework for remote prediction and monitoring. However before a working system can be developed, the following has to be established:

- Identification of critical stakeholders and their requirements need to be examined and formulated into a proposed architecture. This is to take into consideration ease of use and also to ensure that a system which meets a genuine need in the healthcare industry is designed.
- Background information of IoT architectures need to be examined to gain an understanding of the most suitable technologies in existence.
- The background information will then serve as insight to establish the critical design requirements and challenges that need to be understood so that solutions can be implemented. The following areas need to be carefully considered before conceptual designs can be developed: 1) software specifications and challenges, 2) electronics specifications and challenges and 3) wearability specifications and challenges.
- Evaluation of the various concept developments can then be carried out to eventually identify the most suitable design to progress with.

2.1. Stakeholder Expectations

In order to create a successful IoT telemonitoring system, careful attention needs to be given to identify the key requirements that the system should satisfy. These key requirements are often determined by the end users and stakeholders utilizing the system. The following section therefore aims to identify the key stakeholders of the intended IoT telemonitoring system as well as their requirements with which the system needs to be designed around. In order to obtain valid information, health care practitioners and patients were interviewed to gain an understanding of their perspective. The results are summarized below.

2.1.1. Patient

The patient includes all registered users forming a part of the system's patient database. These patients don't necessarily only include at risk individuals but also those wanting to take a better hold of their health status. From the patient's point of view, they are concerned with the accuracy and timely display of their health status. In addition, they require a system that will continuously

transmit their data to a network of doctors for monitoring, so that ambulatory services can be dispatched in the case of an emergency or so that doctors can pick up on early warning signs of life threatening conditions. They also require a system that will predict and alert doctors of impending dangerous health calamities. This is so that their doctors will not miss the early warning signs under their often busy and stressful working conditions.

2.1.2. Physician/Doctor

The Physician includes any individual that forms a part of the system's doctor database. This can include doctors from all specialties that are registered health practitioners with the Health Professions Council of South Africa (HPCSA). A doctor is concerned with the reliability of the telemonitoring system and the accuracy of data obtained. They need to be able to quickly access patient data and easily navigate through the applications to identify issues. Hence the front end designs of these applications needs to ensure simplicity while still maintaining a feature rich experience. The doctor is also concerned with being able to track the patient in the case of an emergency so that ambulatory services can be dispatched to prevent fatalities. While real time monitoring is massively important to spot health conditions with their patients, doctors are also concerned with historic data to spot trends and anomalies within their patients. Hence the long term storage and access of information presented in an easy to digest manner is critical for proactive health care management.

2.1.3. Other Healthcare Personnel

Other healthcare personnel includes nurses, radiologists or other health practitioners who have been given privileges to access a specific patient's data by the physician in charge of that patient. As with the doctor, the supporting health practitioner is concerned with quickly accessing the patient's data. With the fast pace nature of health care, information is required quickly. Hence reliability of the system and ease of use is of utmost importance to the healthcare staff. Also as mentioned previously, data integrity is also of massive importance to gauge the actions that needs to be taken.

2.2. Design Specifications and Challenges

The previous section addressed the key stakeholders and their requirements so that future design considerations can be aligned to end user goals and expectations. This section will now examine the fundamental structure of most IoT frameworks. This knowledge will then be utilized to identify the design specifications and challenges that need to be overcome when designing a successful IoT based telemonitoring system.

2.2.1. Telehealth, Telemedicine and Telemonitoring

Telehealth refers to the technologies and services which enables health care over a distance. Telemedicine which falls under the umbrella of telehealth differs in that its main focus is to utilize remote technology to aid health practitioners to treat patients from a distance. Telemedicine can be further expanded into three main categories viz. 1) store-and-forward (information storage and transmission to doctors without the need for patient visits), 2) telemonitoring (remote monitoring) and 3) real time interactive services (Yolanda Smith, 2021). Figure 2-1. Illustrates the telehealth taxonomy.

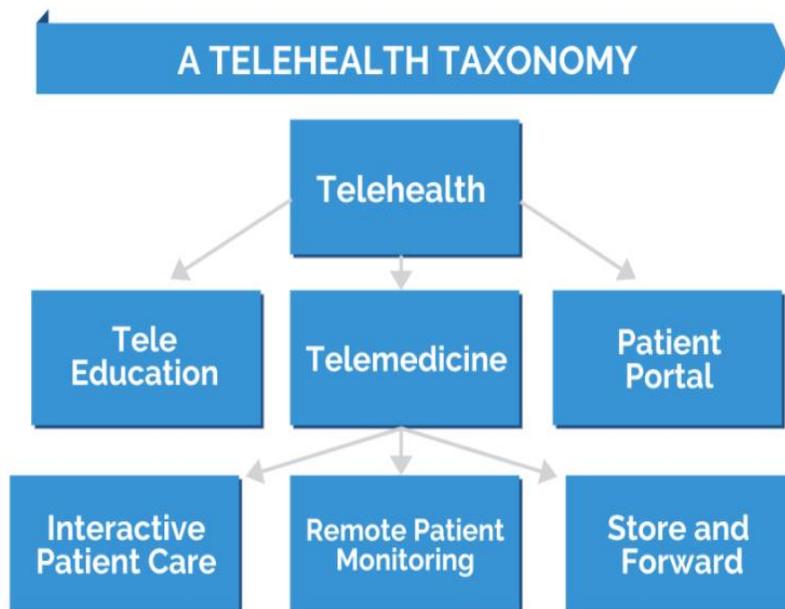


Figure 2-1: Telehealth Taxonomy (Elena Muller, 2021)

Telemonitoring is required to meet challenges such as demands placed on Western healthcare systems as a result of chronic conditions among the elderly, and the added pressure for quality patient care that revolves around patient-centeredness (Christensen, 2018). Telemonitoring technology allows for the successful transmission of critical patient physiological data such as blood pressure, heart rate, oxygen saturation etc. directly to caregivers via a MA or WA (G.F. Gensini, 2017). These technologies utilize sensors which can be embedded in smartphones or other wearable devices to capture, store or respond to data being retrieved from physical settings (e.g. a patient's vital signs). The end result being a more proactive approach to healthcare (J. Mathew, 2018).

2.2.2. The IoT Infrastructure

IoT can be defined as a detailed network infrastructure comprising of a multitude of real-world objects all of which require a careful integration of communication, sensory, networking and information processing technologies. IoT can be broken up into M2M or M2H connections. The former being based on no human interactions and machines communicating with each other to make smart decisions and the latter being human engagement to interpret the IoT output (Hulft, 2018).

2.2.2.1. The 5 layers of IoT

The IoT infrastructure can be broken up into several layers (R. Mehtaa, 2018):

- 1) **Perception layer** – this layer, also called the ‘device layer’ is where sensor devices collect, process and securely transmit data to the network layer. Several device embedded sensors identify and collect critical information based on the application requirements.
- 2) **Network layer** – also referred to as the ‘transmission’ layer, the core purpose that this layer serves is to securely transfer collected sensor data to the information processing system. This is either achieved through wireless or wired technologies including 3G, Wi-Fi, and Bluetooth etc.
- 3) **Middleware layer** – collects the data from the network layer and uploads it into a database where processes and decision making can take priority. It is composed of a variety of different technologies including cloud computing, databases and big data processing (Khanna, 2017).
- 4) **Application layer** – this layer allows for the conveying of useful information to the end users and often involves a user interface that allows for interactions with the IoT interface. The application layer is often integrated with a software application that can either be mobile or web based.
- 5) **Business layer** – the IoT business layer is where the information from the application layer can be analyzed and where decisions can be taken to improve the business KPI’s. For non-business related applications – the business layer can be thought of as just taking action based on the information provided by the IoT system.

2.2.2.2. IoT Architectures

Based on the layers identified above. Various IoT architectures can be incorporated based on the needs and requirements of the system (Hulft, 2018).

- 1) **Device to cloud** – direct connection and transfer of data from sensor to cloud

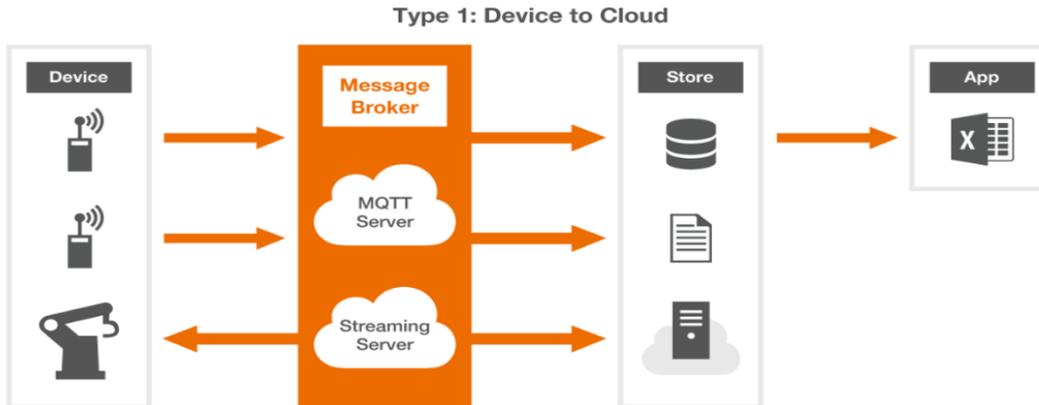


Figure 2-2: Device to cloud architecture

- 2) **Use of an IoT gateway** – with gateway type architecture, a middle gateway layer intersects the sensor and cloud. Due to the primitive nature of sensors, this gateway is required for the successful transmission of data to the cloud. One such benefit of using a gateway type architecture is that the use of short range communication like Bluetooth can be used to reduce the strain on the device.

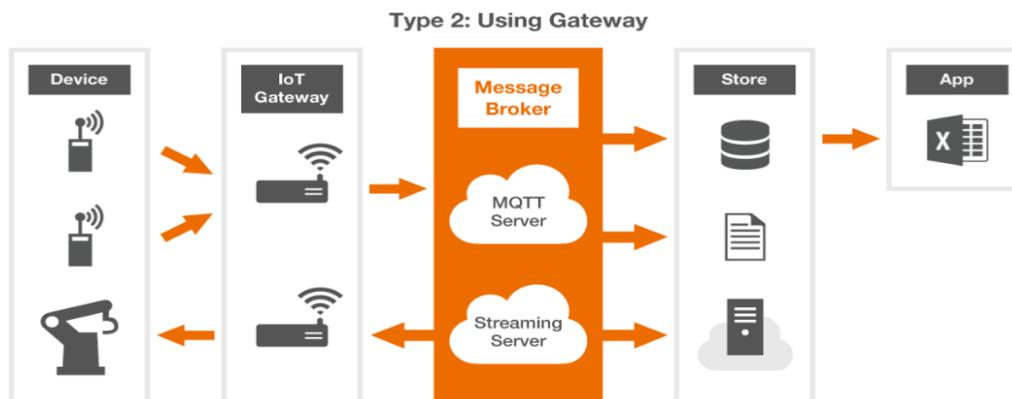


Figure 2-3: Gateway IoT Architecture

2.2.2.3. Mobile as a Gateway Architecture

With the above scenarios, the gateway can either be a server or smartphone. With a mobile being used as a gateway for IoT, the idea lies in the mobile being at the center of the IoT ecosystem and all other IoT devices forming the peripherals. In the image shown in Figure 2-4, sensor based devices collect information from real world entities and convey this information to the smartphone

via protocols such as RFID, Bluetooth, Wi-Fi and NFC. This transmission happens via data packets, and when a data packet is received by the smartphone, it establishes an internet connection and transfers the data to a cloud platform where the data gets housed in a database. Various methods of connection can be utilized to establish a secure connection for transmission such as a Hypertext Transfer Protocol (HTTP) POST to the specified destination (Softweb solutions, 2021). An example of this set-up is shown in Figure 2-4.

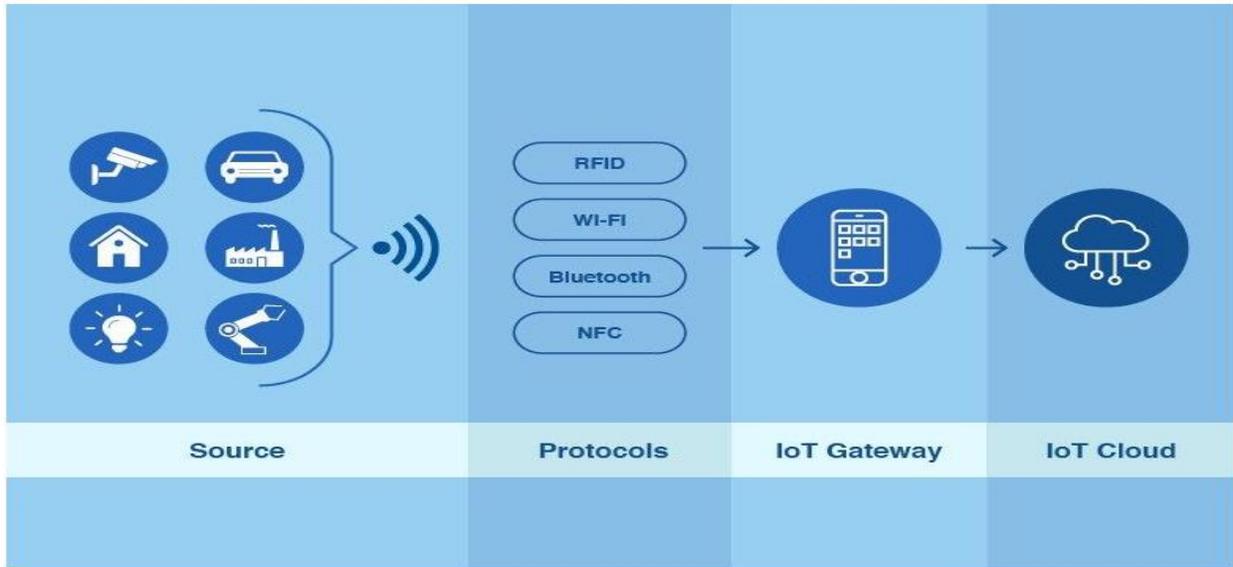


Figure 2-4: Mobile Gateway as an IoT Architecture

2.2.3. Software Specifications and Challenges

The software used in an IoT telemonitoring system is the backbone of its functionality. Choosing the right type of software is an integral component in terms of speed, data integrity, efficiency and future scalability. The following section highlights critical software challenges that need to be considered when designing the proposed IoT system:

2.2.4. Electronics Specifications and Challenges

The electronics which include the microprocessor and its corresponding sensors are also fundamental components required to complete the data pipeline. Correct wiring of the sensor network with the microprocessor will ensure physiological patient data is extracted in real time and transmitted to the network and application layer for patient monitoring. The following section discusses the main electronic device requirements and challenges:

2.2.4.1. MCU

When selecting the microcontroller for the telemonitoring system, careful consideration should be given to the size, speed, device ports, energy usage and memory. The MCU selected should have sufficient processing power to run the code. In addition, a small enough controller should be selected to allow for the wearability criteria to be satisfied. There should also be sufficient communication ports for the peripheral devices and I/O ports for configuring the electrodes.

2.2.4.2. IoT Gateway

The IoT gateway is the central link between the WBAN and cloud interface. The IoT gateway selected for the intended telemonitoring system should be able to store and process large amounts of info locally. It should have Bluetooth or similar short range communication capabilities such that it is able to collect data from the WBAN network. Then it should be able to integrate with a cloud network to transfer data to a server based database for access by health practitioners. The IoT gateway should be energy efficient, reliable and small enough to be carried around with the patient to continuously receive, store and transmit data to the cloud. It should be able to easily integrate with cloud interfaces without incurring excessive costs for subscription or software based tools. Lastly it is preferred that the IoT gateway have a front end design for patients to monitor and interact with the sensor data from the WBAN.

2.2.4.3. Battery

The battery selected should be small enough yet have sufficient amperage and voltage to power the circuit. The battery should be rechargeable to allow for frequent charging and re-use considering the real time nature of device monitoring. With respect to charging, the battery should have a balanced charger to prevent short-circuiting.

2.2.4.4. MCU Software

The software chosen should be capable of programming the chosen MCU. In addition, it should allow for the programming of interrupts and timers for the transmission of data packages sent to the MCU via Bluetooth. The program developed should also minimize MCU flash memory and have fast execution time. Due to the complexity of the code, a debugger is also required to speed up the design process through efficient troubleshooting.

2.2.4.5. Cloud Software

Cloud integration forms an integral component of any IoT based telemonitoring system. Therefore careful consideration needs to be taken when selecting the intended platforms. For storing of information, suitable databases need to be selected. NoSQL databases should be used for

storage of schema less data (such as patient registration information) as it is cost effective and easier to use with non-complex querying requirements. Scalability is often easier and less cost intensive with NoSQL databases which allows for an increase in users on the platform. Schema data from the sensors should be stored in a SQL database where ACID (Atomicity, Consistency, Integrity, Durability) properties are a concern and semi-structured data is being used.

The cloud software should also consider interoperability between the IoT gateway and sensing layers (mobile and web). This needs to be facilitated by a suitable connection SDK or backend software which is either part of the cloud platform or independent. Again cost and efficiency is a driving factor for this selection. It is also necessary to take into consideration community support. Well supported cloud software is easier to implement and communicates ease of use. Lastly to satisfy ML requirements, a cloud integration platform with ML capabilities is required to allow for ease of model development and integration with mobile and web apps via POSTS requests.

2.2.4.6. MA Software

Software selected should allow for programming of an Android app since most users have android based smartphones. A suitable mockup program should be used to determine the necessary application activities required to achieve optimum functionality. The programming language selected for the mobile app should have a large community support and usable syntax for ease of troubleshooting. The software should have high processing power and sufficient in built functions to allow for full mobile functionality. A suitable IDE with debugger, interpreter and compiler is also necessary.

2.2.4.7. WA Software

A suitable web programming software and run time environment is required. This is to allow for the creation of dynamic web pages with backend functionalities to draw data from cloud databases. HTML should be used for web structure development and CSS with suitable frameworks to initiate the front end design. These technologies are widely used and considered the standard mechanisms for creating reliable web based applications.

2.2.4.8. Sensors

The sensors selected should be suitable for the intended measurements and suitable for integration with the MCU. Accuracy of measurements is also a very important criteria for each sensor. For ease of use, the sensors should also have an embedded signal conditioning block which reduces the complexity of amplifying or filtering signals.

2.2.4.9. Short Range Communication

Transmission of data to the gateway device is a crucial step in the IoT framework. A short range communication protocol needs to create a paired connection with the chosen IoT gateway and then transmit data continuously. Hence, indicator lights are required to indicate each stage of the connectivity process i.e. connected, disconnected and pairing. The size of the short range communication device is another important criteria that needs to be considered together with software and equipment compatibility. Baud rate (rate of data transmission) should be adjustable

2.2.4.10. Haptic Feedback and Indicators

The patient needs to be aware of impending dangers based on readings received by the sensors. Therefore indicators and haptic feedback devices are required to warn users when they are under strain or are reaching dangerous conditions. Patients need to also be made aware of when to take readings. The haptic feedback and indicators therefore play an important role in early warning detection and therefore need to satisfy the criteria of being obtrusive i.e. blink brightly, vibrate etc. to get the patient's attention even when they are pre-occupied.

2.2.5. Wearability Specifications and Challenges

The wearability specification of the proposed monitoring system is not easily quantifiable. The following criteria needs to be met:

- The device needs to be robust to withstand continuous wears but also comfortable to ensure ongoing use at least for testing purposes.
- As far as possible the appearance of the WBAN should be unobtrusive which would otherwise deter users from wearing the device.
- The above criteria must be achieved while also maintaining the signal quality.
- Sensor placement is also an important consideration when designing the wearable strap/casing.

2.3. Proposed System Description

The techniques, methods and tools together with their applicability were based on a detailed literature review carried out in chapter 1.3 and stakeholder analysis in chapter 2.1. The WBAN device needs to contain the appropriate sensors to capture patient physiological data. This is dependent on the patient performing the relevant initiation processes e.g. setting Bluetooth, Wi-Fi etc. The sensors will make contact with the patient's skin and the corresponding data will be sent via Bluetooth (short range communication) to the patient's smartphone. The corresponding information will then be stored locally in the patient's phone database. The smartphone will serve as the IoT gateway which will create a link between the locally stored data and cloud platforms. The patient can then choose to transfer data to the cloud via a Wi-Fi connection and this can be

subsequently accessed by registered doctors via their smartphones or through the created WA. The MA will make use of the patient's phone GPS tracker and will send this info to the cloud when prompted by the user. This will enable real time tracking of the patient by hospital personnel. The patient will be able to view health stats from the patient interface of the MA. FL models imbedded in the phone application will calculate patient status and subsequently transfer to cloud storage for viewing by doctors on their mobile phones or through the developed WA. ML models will be developed on a cloud service and accessed by the MA through a POST request. The resulting output will also be stored on the cloud database for viewing by health personnel. All of the predictive modelling evaluations can also be viewed by the patient on the mobile patient interface. An overview of the proposed system architecture based on stakeholder requirements and system specifications is shown in Figure 2-5. Figure 2-6 also shows the flow of information using a uniformed modelling language (UML) diagram.

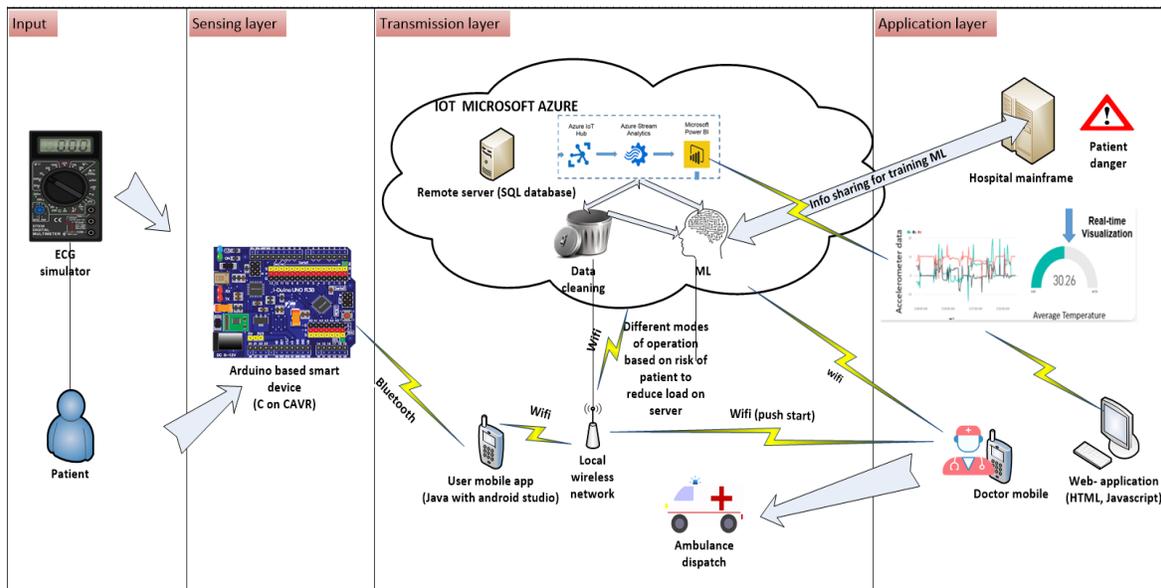


Figure 2-5: Proposed Smart IoT Telemonitoring System

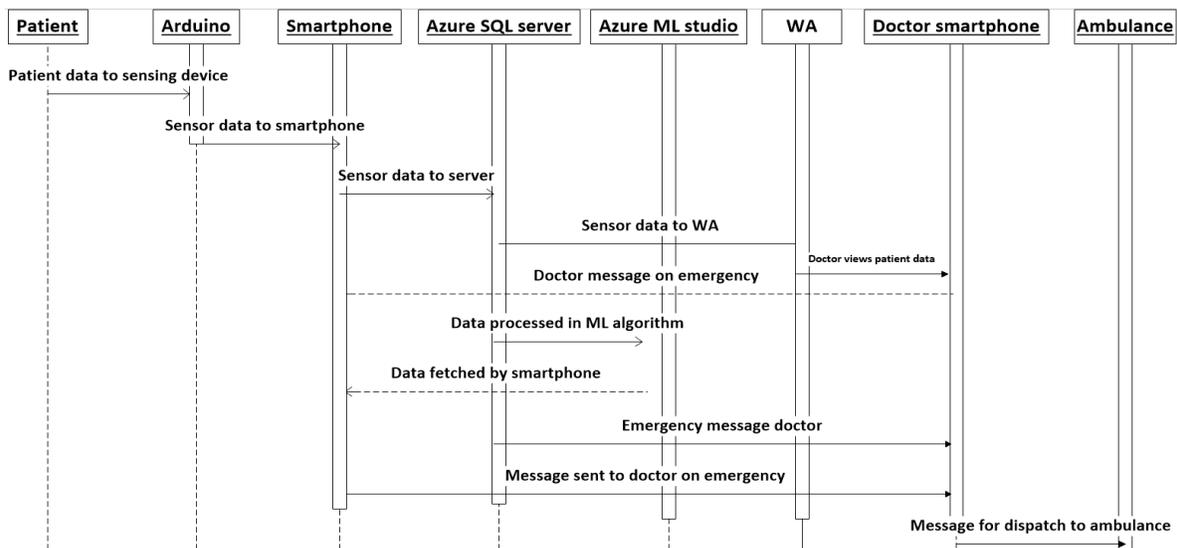


Figure 2-6: UML Diagram for Proposed Telemonitoring System

2.4. Concept Development and Selection

The previous sections identified and gauged the requirements, specifications and challenges required to design an effective IoT based healthcare system. This together with the objectives mentioned in section 1.5, can be used to identify and discuss possible IoT based telemonitoring systems. This section brainstorms and scores several concept designs using the Pugh matrix based on three areas 1) physical characteristics, 2) Usability and 3) IoT connectivity. The outcome is the selection of an optimum concept design which can be developed in further detail.

2.4.1. The Pugh Matrix

The Pugh matrix proposed by Pugh is an effective means of comparing the effectiveness of various design concepts based on a predetermined set of criteria. The main steps in developing the Pugh matrix and identifying the best design concept is highlighted below:

Step 1: involves designing a criteria for selection based on internal and external stakeholder requirements.

Step 2: selection of a design which serves as the baseline. Here the apparent best solution was chosen to be the benchmark to determine if alternative designs outscore it. An “S” is used to indicate the benchmark score. For the case of the IoT gateway selection, the Smartphone (Samsung Galaxy Note 10 Lite) was chosen as the benchmark.

Step 3: All other designs are compared against the benchmark and scored for each item on the criteria. “S” indicating no difference, “+” indicating the design is more effective than the benchmark and “-“, indicating that the design is less effective than the benchmark. “++” and “--“can be used to indicate higher levels of discrimination.

Step 4: Add all the “+” values and “-“values for each design and the highest score wins.

Step 5: Check the above scoring by using a weighting for the criteria. Multiply the weighting by the respective score and sum up. Once again the highest weighted “+’s” depicts the most suitable design.

The preceding sections will look at the matrix application for two fundamental design considerations i.e. the IoT gateway selection and the selection of the wearable design concept.

2.4.2. Proposed IoT Infrastructure Designs

One of the fundamental components of the IoT infrastructure is the selection of a suitable IoT gateway as mentioned in section 2.2.2. The Pugh matrix seen in Figure 2-7 was used to evaluate three different IoT gateway options i.e. mobile as a gateway (Samsung Galaxy Note 10 Lite for comparison), Raspberry Pi as a gateway and the Arduino Nano as a gateway. The scoring from the matrix clearly shows that the best option is mobile as a gateway, with the Raspberry Pi option having a score of -8 in comparison to the benchmark (Samsung Galaxy Note 10 Lite) and the Arduino Nano scoring -48 in comparison to the benchmark (Samsung Galaxy Note 10 Lite). Three characteristics were used to compare the various IoT gateways available i.e. physical characteristics, Usability and IoT connectivity. Within the domain of physical characteristics, the reliability of the device was considered the most important since the IoT gateway needs to be

physically durable for continuous uses while also being physically capable in terms of hardware capability to support the intended application. Battery life was also very important and scored second in terms of weighting since the continuous transmission of data was paramount to achieve the goal of real time monitoring. Size of device and affordability scored lower in weighting because it is assumed that the customer would still purchase the product if it were larger or more expensive on condition that it prolonged their life span. Under the area of usability, processing power scored the highest weighting of “5” since without a good enough middleware processor, the entire IoT infrastructure would collapse. On the other hand, ease of programming and sensor integration were considered a “nice to have” but not a deal breaker, hence they had lower weightings. Front end usability was given a moderate weighting since it is important, however complexities could be ironed out with further training or adjustment in the interface from market feedback during commercialization. All properties within IoT connectivity were considered relatively important. However ease of integration with the cloud platform and data handling costs were given a higher weighting as these would be crucial to ensuring transmission to healthcare professionals. Cost of data handling was also considered of high importance since it would result in major operating cost issues during the scale up and commercialization phase of the product deployment. Ability to store data locally and wireless capabilities were considered important however scored a “4” in terms of weighting since bare minimum requirements would be good enough to satisfy this criteria e.g. range of Bluetooth and size of storage does not need to be superior.

			Design Concepts			
			Weight	Smartphone (Samsung Galaxy N Lite) as a gateway	Raspberry Pi as a gateway	Arduino Nano as a gateway
Pugh concept selection matrix						
IoT connectivity	Ease of integration with cloud e.g. databases or IoT hubs	5	S	S	-	
	Cost of data handling	5	S	++	--	
	Ability to store data locally	4	S	-	--	
	Wireless capabilities	4	S	S	-	
Usability	Ease of programming language	2	S	S	-	
	Ease of integration with sensors	3	S	S	S	
	Processing power	5	S	-	--	
	Front end usability	3	S	-	--	
Physical Characteristics	Affordable	2	S	+	++	
	Size of device	3	S	+	++	
	Battery life	4	S	-	--	
	Reliability	5	S	-	-	
TOTAL +				0	4	4
TOTAL -				0	5	14
TOTAL SCORE				0	-1	-10
WEIGHTED TOTAL +				0	15	10
WEIGHTED TOTAL -				0	23	58
WEIGHTED TOTAL SCORE				0	-8	-48

Figure 2-7: Pugh Matrix for IoT Gateway Selection

In terms of IoT connectivity, the Raspberry Pi option scores the same as the mobile option for cloud integration as it is possible to create connections with databases and IoT hubs easily. Raspberry Pi also has a built in Bluetooth module as does a mobile phone which makes short range communication possible. In terms of local storage, a Raspberry Pi does have sufficient capabilities (32 GB) however this falls short in comparison to the impressive storage space found within the smartphone (128 GB). With regards to the Arduino Nano, it fails in all of the criteria. Due to the absence of an operating system and built in Wi-Fi and Bluetooth module which need to be added on as auxiliary equipment. The Arduino Nano onboard memory is minimal (32 KB) which means that it is unable to store large amounts of data from the sensors.

With respect to usability, programming language is subjective however the Python coding for the Raspberry Pi module was considered of equal complexity to Java required for the MA to be used on the smartphone. This factored in language popularity and the availability of resources for troubleshooting. The Arduino Nano however used “C” on the Codevision interface which had minimal online resources and a small audience of users therefore it scored lower in comparison to the benchmark. All options are the same in terms of integration with sensors and are able to connect easily through a plug and play approach. In terms of processing power, the Raspberry Pi does have computational power (512 MB RAM) however not nearly as powerful as a smartphone (6 GIG RAM). The Arduino Nano is however not effective in terms of processing power which is why it scores poorly in the area (2KB SRAM).

Physically, the Arduino Nano scored well in terms of size (45mm by 18mm) which is much smaller than a smartphone or Raspberry Pi (66mm by 30.5mm). The Raspberry Pi Zero and Arduino Nano are also much cheaper in comparison to a smartphone. In terms of battery usage, the smartphone is the biggest consumer of power, however it has a big enough battery with energy optimization capabilities which ranks it higher than the two other design options. The smartphone also wins in terms of reliability since it is a commercialized product as compared to the Raspberry Pi and Arduino Nano which are often suited for prototype designs.

2.4.3. Proposed Wearable Designs

As discussed in section 2.2.5, the wearable device has a unique set of challenges which need to be satisfied to ensure the effectiveness of the IoT telemonitoring system. To overcome these unique challenges, three design concepts were evaluated using the Pugh matrix. The results are shown in Figure 2-8. The first is the baseline concept which is based on a wrist strap design. A chest and waist strap were the other design concepts examined. The results show that the baseline is the winning concept and will be the most suitable for the intended application. The second best design is the waist strap (score of -7) with the worst suited being the chest strap (score of -63).

			Weight	Design Concepts			
				Wrist Strap	Chest Strap	Waist Strap	
SELECTION CRITERIA		Pugh concept selection					
		EMG Sensor	Placement	5	S	--	-
			Accuracy	5	S	++	+
			Ease of access	3	S	--	-
		ECG Sensor	Placement	5	S	--	-
			Accuracy	5	S	++	+
			Ease of access	3	S	--	-
		Pulse Rate Sensor	Placement	5	S	--	-
			Accuracy	5	S	++	+
			Ease of access	3	S	--	-
		Temperature Sensor	Placement	5	S	--	-
			Accuracy	5	S	++	+
			Ease of access	3	S	--	-
		Humidity Sensor	Placement	5	S	--	-
			Accuracy	5	S	++	+
			Ease of access	3	S	--	-
		Usability	Willingness to use	4	S	--	+
			Prolongued use possible	4	S	--	+
			Ease of application and removal of device	3	S	--	++
			Comfort	5	S	--	+
Physical Characteristics	Affordable	3	S	S	S		
	Non - obtrusive	3	S	+	+		
	Size of device	3	S	-	-		
	Easy to see visual indicators	3	S	--	--		
	Reliability	5	S	+	-		
		TOTAL +		0	12	11	
		TOTAL -		0	31	13	
		TOTAL SCORE		0	-19	-2	
		WEIGHTED TOTAL +		0	58	47	
		WEIGHTED TOTAL -		0	121	54	
		WEIGHTED TOTAL SCORE		0	-63	-7	

Figure 2-8: Pugh Matrix for WBAN Selection

Each WBAN option was weighed up against placement, accuracy and ease of access of sensors. Placement and accuracy had the highest weightings of “5” since the positioning and accuracy are the most important factors for reliable communication of data to doctors. Ease of access is important however patients will not be accessing the sensors all the time so it was not given the highest weighting. In terms of usability, comfort was scored the highest weighting of “5” as the device needs to be worn 24/7. Willingness to use and possibility of prolonged use were also considered important yet had lower weightings since these are factors which can be rectified during commercialization. Ease of application and removal of device scored a weighting of “3” since this is done rarely. In terms of physical characteristics, reliability scored the highest with a weighting of “5” since a WBAN needs to be durable for continuous use and capable of protecting the components within it. Affordability, obtrusiveness, size of device and ease of seeing visual indicators were considered medium concerns hence were given a weighting of “3.”

The chest strap scored low for placement of certain sensors as readings would not be possible from the chest. These include the humidity sensor and pulse rate sensor. Some sensors however, like the ECG would get optimum readings when placed on the chest. The design also scores low in terms of usability because it is much more uncomfortable to use as it requires wires and sensors to run across the chest. Although it is much less obtrusive than the other designs as it is used under the clothes, it prevents the user from seeing visual indicators when abnormal conditions are detected. The waist strap was a close runner up to the wrist strap. It has a small housing and users can easily see warning alerts. The design is also less obtrusive as it can be worn like a belt with the sensor housing depicting a belt buckle. The design scores in terms of comfort but falls short with respect to placement of sensors as many sensors would not be able get readings from the waist. The best design was the wrist strap which will be further explained in the preceding chapters. Here the sensors will be housed in a casing enclosed within a wearable strap. All readings can be taken with sensors obtaining measurements from the arm area. The device is also something most people will be inclined to wear as it depicts a typical wearable device.

2.5. Chapter Summary

The user requirements from the stakeholder’s perspective were examined based on interviews with concerned stakeholders. This together with the IoT frameworks were used to establish the specifications and challenges to address when designing the ideal IoT system. This included areas of software development, electronic design and wearability. A proposed high-level design was then presented together with a UML diagram. The concept developments were thereafter put through a Pugh matrix to identify 1) the optimum IoT gateway and 2) the optimum WBAN device. The criteria assessed for the IoT gateway was physical characteristics, IoT connectivity and usability. While for the WBAN device, the Pugh matrix scored conceptual designs based on physical characteristics, usability and sensor characteristics. The chapter concluded with the identification of the wrist WBAN set-up with smartphone IoT gateway integration as being the optimal conceptual designs to take forward.

Chapter 3: Sensing Layer - WBAN Device

Chapter 2 looked at the various proposed wearable designs and showed through the use of a Pugh matrix that the best possible WBAN device would be in the form of a wearable arm strap. The chapter further elaborated on the various IoT frameworks characterized by their difference in IoT gateways and once again through a Pugh matrix approach scored an IoT system using a smartphone gateway as the most suitable for the proposed design. This section will delve into the detailed design of the WBAN selected in chapter 2, paying careful attention to the design criteria set forth in chapter 2.

3.1. Sensing Layer Characteristics

With an IoT based framework, the sensing layer is often composed of several layers of sensors based on the application. In the healthcare sector, the IoT sensing layer is composed of sensors that measure various physiological parameters. These parameters are obtained non-invasively through various different sensors either directly, as is the case with an electrocardiogram (ECG) or indirectly through methods like IR where the output ADC signal is fed into a correlation that determines the desired parameter for e.g. sugar levels in blood for the diagnosis of diabetes. Below is a list of common physiological readings that are possible within the health telemonitoring spectrum:

3.1.1. ECG

The ECG is a means of visually viewing the complex electrophysiological events that are occurring within the cardiac tissues of the heart over a period of time. In order to maintain a healthy body, the heart needs to be able to sustain regularity in its contraction and relaxation cycles. An ECG is therefore of paramount importance to monitor and ensure that deviation from a healthy heart function is indicated to health practitioners for them to take corrective action (Becker, 2006).

Depolarization and repolarization can be defined as the action potential (AP) caused by ions moving across the cell membrane. This is what causes the contraction of the cardiac cells or muscles. Repolarization is the opposite and refers to the return of the ions to their rest state resulting in relaxation of the heart muscles.

A normal ECG pattern can be shown by Figure 3-1. The waveform is constructed from various electrical signals that occur within the heart muscle tissues. The P wave is used to illustrate the electrical signal originating from the atria (chambers of the heart). There is a slight delay in electrical activity illustrated by the PR segment which can be attributed to the Atrioventricular (AV) node slowing down depolarization to ensure that the atria unloads blood into the ventricles before it contracts. The QRS is then the largest cycle and represents depolarization of the heart ventricles. Repolarization of the myocardium is indicated by the ST segment (Price, 2010).

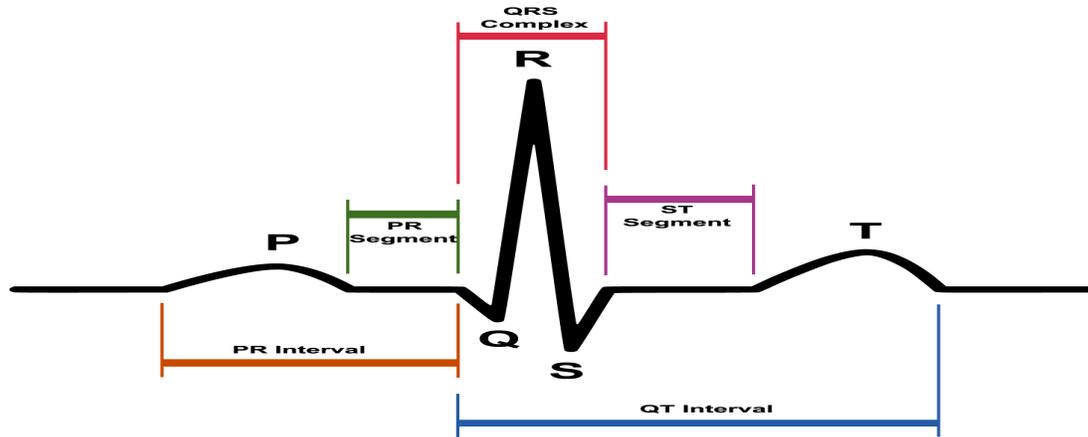


Figure 3-1: Normal ECG Wave

As the ECG is plotted over time in milliseconds, the various segments have a normal range which indicates a healthy heart function.

PR interval – 120 to 200 ms

QRS duration – Up to 120ms

QT interval – Up to 440 ms

3.1.2. Electromyography (EMG)

EMG is another fundamental measurement for health monitoring. It is the visual representation of the muscles response or electrical activity as a muscle stimulus is inflicted. An EMG is carried out when a health practitioner suspects that the patient may be suffering from a nerve or muscle disorder (Mayoclinic, 2019).

Figure 3-2 shows an example of patient with normal EMG. At rest there is no AP, however during stimulation of the muscles a waveform appears. The size and shape of the EMG signal is dictated by the AP. Hence a stronger AP will mean that the muscle is more responsive to nerve stimulation. Here muscle fibre size is correlated to the AP rate and amplitude of the AP (ebme, 2021).

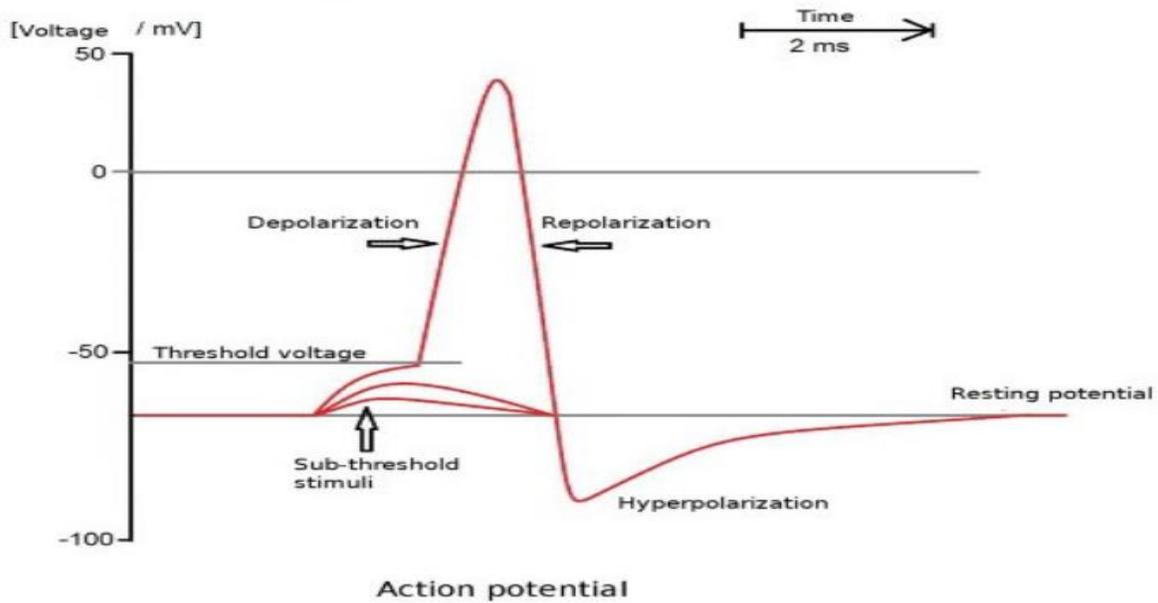


Figure 3-2: Normal EMG Wave

3.1.3. Oxygen Saturation & Pulse Rate

A pulse oximeter sensor is a device that measures both blood oxygen levels as well as pulse rate through the fingertip by means of light beams. The oxygen saturation (SpO_2) gives information regarding the oxygen levels in the blood stream. The use of light beams to approximate the output variables allows the device to work non-invasively (U.S. food and drug administration, 2021). Pulse rate can be defined as the number of successful beats the heart is able to accomplish in a minute (John Hopkins Medicine, 2021) also referred to as beats per minute (BPM).

Normal ranges: $SpO_2 = 95\% - 100\%$ (John Hopkins Medicine, 2021)

Pulse rate = 60 BPM – 100 BPM (U.S. food and drug administration, 2021)

3.1.4. Humidity

Humidity and temperature parameters can be useful in the grander scheme of a health telemonitoring system. When combined with the patient's location through GPS – doctors will be able to make correlations about how humidity affects the body temperature (BT) of a patient thus indicating potential health hazards such as sun stroke. A study on health effects has shown that humidity can be the cause of an increased risk of respiratory infections and allergies (Arundel et

al., 1986). According to the study (Arundel et al., 1986), relative humidity in the range of 40%-70% can hinder the survival and infectivity of bacteria and viruses.

The effect of environmental temperature can also have an adverse effect on the human body, especially if it is left unmonitored. Relative humidity can also have an effect on temperature resulting in extreme cold or extreme heat effects. Risk level temperatures are shown below (Elaine, 2017):

Between 32 °C – 40 °C, cramps and exhaustion is possible

Between 40 °C – 54 °C, heat exhaustion is likely

Over 54 °C, the likelihood of heat stroke is probable

3.1.5. Body Temperature

The importance of regulating body temperature is of utmost necessity which is why it forms an integral part of a well-integrated telemonitoring system. Limit alarms can be used to indicate when thresholds have been reached to warn both patients and doctors of impending health issues. Keeping the body temperature close to 37 °C helps to maintain cell temperature for optimal bodily reactions (BBC, 2021).

Abnormal Ranges: Any temperature above 38 °C means fever (WebMD, 2021)

A drop in temperature below 35 °C means hypothermia (WebMD, 2021)

3.1.6. IR Sensor & Receiver

An IR sensor and receiver is a two part device that works together. When it comes to medical measurements, an IR sensor and receiver can work together with laws such as the Beer Lambert law to non-invasively determine the concentration of various substances within the blood. An IR LED transmits an IR signal which the receiver then picks up. These changes in the attenuation of the light signal will then give an indication of the concentration of specific components in the blood stream. According to Beer Lambert's law, there is a correlation between the attenuation of light through a substance and the properties of the substance. By mapping the changes of light reflecting through different blood concentrations a variety of substances and their concentration can be detected in the blood stream e.g. blood sugar levels to detect diabetes (Ionescu, 2018).

Glucose balance within the body is maintained by an intricate balancing act between the chemicals Insulin and Glucagon. When Glucose is too high, Insulin plays a role in reducing it. Conversely, Glucagon helps to increase Glucose levels when it detects an abnormal level. During Diabetes, the body's ability to produce insulin is inactive and hence results in an imbalance in glucose levels. A high glucose level indicates a condition known as hyperglycemia and a low

glucose level indicates hypoglycemia. The various ranges of glucose levels and what it means is indicated below (Stoppler, 2021):

- 72 mg/DL – 99 mg/DL is the normal range
- 200 mg/DL – 400 mg/DL hyperglycemia
- <70 mg/DL - hypoglycemia

High blood sugar can be dangerous and overtime can cause the occurrence of non-communicable diseases such as stroke, eye damage and kidney failure.

3.2. High Level Specifications

Now that the characteristics and importance of the sensors used within the sensing layer have been identified, the high level specification of the WBAN can be determined. It is known that the device needs to be in the form of a wearable arm band. Within the band needs to be an enclosure to house all the sensors. The band needs to fit securely so that sensors make adequate contact with the skin. The sensors should be optimally located to ensure the proper reading is attained. The device should be as small as possible, hence a small MCU and sensors are required to obtain readings. A Bluetooth module is required to transmit the data to the patient's mobile phone. The device should have both haptic and visual feedback to alert the patient of impending health issues and also when to take readings.

3.3. Electronic Design

The electronic design refers to the integration of all sensory components with the chosen MCU. Whilst the software integration refers to the programming of the MCU to activate the electronic components. This section will elaborate on all of the sensors utilized in the prototype design along with the design decisions that led to the completed electronic system. Thereafter the software to control the MCU will be elaborated on.

3.3.1. MCU

A MCU is required to receive the digital signal from the WBAN, convert it to an analogue signal and then transmit the data to the smartphone MA using the UART protocol. When choosing the suitable MCU, various physical and performance characteristics need to be evaluated to determine a fit for purpose solution. The MCU selected was the Arduino Nano with the Atmega 328P chipset. The Arduino Nano was suitable due it being low voltage, compact in size, lightweight and designed with sufficient digital and analogue pins to enable connections of various sensors. It supports an input voltage of 7 V-12 V and has an onboard 5 V and 3 V regulator if required. This MCU is also easily available and a cheaper choice compared to its predecessors

that have more computational power and are much larger such as the Arduino Uno and Arduino Mega. Another fundamental requirement was the availability of TX and RX pins to enable the transmission of data through the Universal Asynchronous Receiver and Transmitter (UART) protocol. The Atmega 328P is a high performance and low energy chipset with a 10 bit analogue to digital converter (ADC) which means it has a longer battery life and high enough precision for the intended application. Although the chipset is running on an 8 bit system with a low end random access memory (RAM) and processing power/clock speed, it satisfied the requirements of the proposed system as most of the data processing is done on the MA which also reduces energy usage. The MCU is also able to reach baud rates of up to 115200 which provides a fast enough transmission without errors on the receiving end. Table 3-1 indicates the required specifications along with the chosen MCU capabilities:

Table 3-1: Requirements and Specifications of Chosen MCU

Specification Description	Requirement	Microcontroller capabilities
Low supply voltage	3.3V - 5V	3.3V and 5V output pins
Low power consumption	11-20mAh	19mA
10 bit ADC channels	4	8
Digital pins	7 (4 of which are PWM, 1 TX, 1 RX)	14 (6 of which are PWM, 1 TX, 1 RX)
Serial communication protocol	UART	UART, I2C protocol, SPI protocol
Flash memory	<32KB	32KB
RAM	<1KB EEPROM	2KB SRAM, 1KB EEPROM
Weight	<10g	7g
Dimensions	As small as possible	18mm × 45mm
Cost	<=R200	R150
Clock speed	18MHZ	18MHZ
Baud Rate	9600 bauds	max of 115200 bauds

The analogue and digital pin connections to the MCU is highlighted in the Table 3-2:

Table 3-2: Analogue and Digital Pin Connections on MCU

Pin	Connection Description
TX1	MCU TX1 connects to Bluetooth RX input
RX0	MCU RX0 connects to Bluetooth TX output
D2 and D3	Used to set-up interrupts
D5	Output of D5 connects to source pin of vibration module
D6	Output of D6 connects to voltage pin of LED
D10	Output of ECG sensor electrode connects to input of D10
D11	Output of ECG sensor electrode connects to input of D11
D13	Out of D13 connects to voltage pin of IR LED
A0	Analogue output of ECG sensor connects to input A0 of MCU
A1	Analogue output of pulse rate sensor connects to input A1 of MCU
A2	Analogue output of EMG sensor connects to input A2 of MCU
A3	Analogue output of IR receiver connects to input A3 of MCU
A6	Analogue output of temperature sensor to input A6 of MCU
A7	Analogue output of Humidity sensor to input A7 of MCU
Vin	Output of battery to input Vin of MCU
Vref and 5V	5V pin or arduino to Vref pin of arduino

3.3.2. Bluetooth

The Arduino Nano does not come equipped with a built in Bluetooth module, therefore a standalone unit was purchased to satisfy the UART capabilities required. The Bluetooth module will be used to transmit and receive data to and from the MA using the MCU. The HM-10 low energy module, HC-05 and HC-06 were considered for use as a Bluetooth. The HC-05 was an older model and the HM-10 was much more expensive with the low energy requirement not being an immediate necessity. Therefore the HC-06 was selected as it is a common of the shelf (COTS) product that is easy to set-up and integrate. The HC-06 is a class 2 Bluetooth device which means that it can operate in ranges of up to 2 m. For the intended application, this is satisfactory as the IoT gateway is the patient’s smartphone which is usually in the patient’s possession. The device has a reasonable power and voltage requirement and utilizes a method called frequency hopping spread spectrum (FHSS) technique to inhibit interference with neighboring devices. In addition, the device is cheap, easily available, small and lightweight which fits the purpose of a WBAN. One of the drawbacks of the HC-06 is that it can only be operated as a slave, however for the system employed, the HC-06 is only required to accept a connection from the smartphone. The Android based application requires a radio frequency communication (RFCOMM) or Serial Port Profile (SPP) which is available on the HC-06 module.

Figure 3-3 depicts the connection between an Arduino Nano MCU and the HC-06 Bluetooth module.

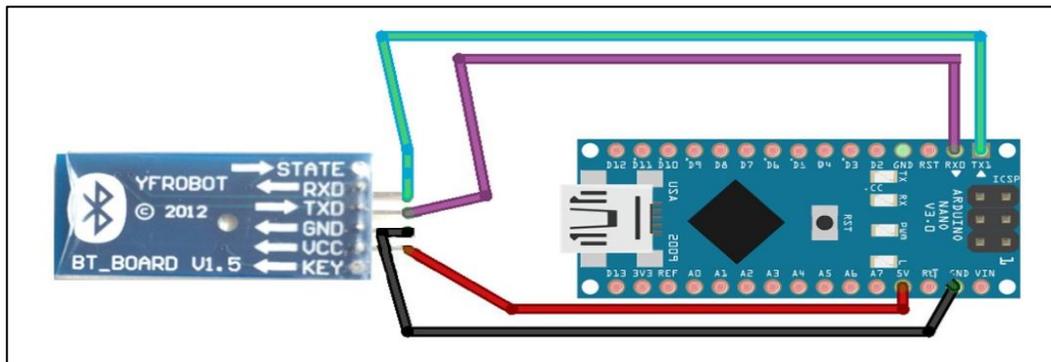


Figure 3-3: HC-06 Connection with Arduino Nano MCU

Table 3-3 shows the system requirements for a Bluetooth device and the HC-06 capabilities:

Table 3-3: Requirements and Specifications of Chosen Bluetooth Module

Specification Description	Requirement	HC06 capabilities
Operating range	<5m	10m
Power consumption	<20mA	0.5mA for standby, 8.5mA active
Bluetooth Socket Requirement	RFCOMM/SPP	RFCOMM/SPP
Price	<R200	R120
Weight	<5g	3.24g
Size	As small as possible	37mm × 16.96mm
Mode	Slave	Slave

3.3.3. Body Temperature Sensor

The LM35 temperature sensor was utilized for measuring body temperature which would be used as an input for the FL and MEWS model and to monitor the patient. The LM35 temperature was selected over the MAX30205 temperature sensor since it is much cheaper with the required level of accuracy. The LM35 is low cost (due to wafer-level trimming), compact (20 mm × 5 mm), low energy (<60 μA) and easily accessible with analogue output capabilities. It is typically used to measure room temperature, however it is possible that it can be adapted to measure body temperature through the direct contact mechanism as opposed to the IR mechanism of operation. The response time is relatively fast when the sensor is already at ambient temperature and the temperature rise is gradual. The LM35 has an advantage over other linear scale temperature sensors calibrated in Kelvin as one can easily get a centigrade reading without having to subtract a large constant voltage from the sensor output. The sensor is also able to achieve typical accuracies of $\pm \frac{3}{4}^{\circ}\text{C}$ over the temperature range of -55 °C to 150°C. The low output impedance of the sensor circuit also helps to keep noise out which results in a stable output voltage linearly proportional to the centigrade temperature.

The LM35 temperature sensor has 3 pins. The ground, source and voltage. The sensor requires a 4 V – 20 V input which is supplied by the LM7805 voltage regulator (5 V) which is connected to the output of the 7.4 V lithium battery. The ground is connected to the lithium battery ground and the analogue pin is connected to one of the Arduino ADC channels. The sensor connection is shown in Figure 3-4:

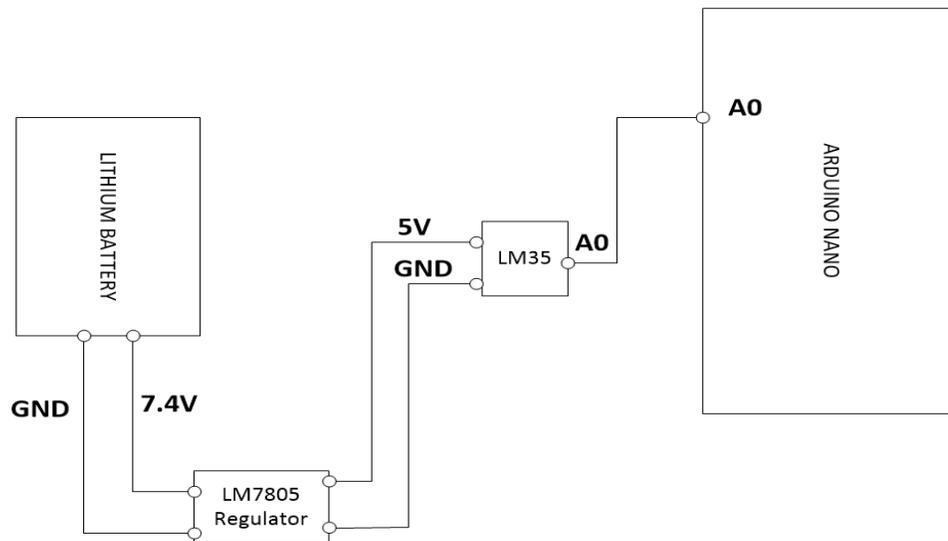


Figure 3-4: LM35 to MCU connection

The resulting ADC output is then plugged into equation (3.1) to calculate the output voltage in volts:

$$V_{out} = ADC\ Reading \times \frac{V_{ref}}{ADC\ Resolution} \dots \dots \dots (3.1)$$

For the Arduino microcontroller operating with a 10 bit ADC, the ADC resolution is given be 2^10. Hence the equation (3.1) can be simplified to give equation (3.2):

$$V_{out} = ADC\ Reading \times \frac{V_{ref}}{1024} \dots \dots \dots (3.2)$$

The measured temperature is then given by equation (3.3) where the output voltage is measured in mA. This equation was then used in the MCU code to calculate the temperature output of the sensor. This equation is further analyzed in the results section to determine its efficiency.

$$Temperature\ (^{\circ}C) = \frac{V_{out}}{10} \dots \dots \dots (3.3)$$

3.3.4. Humidity Sensor

A humidity sensor was added to the WBAN due to the importance of tracking the effect of humidity on the human body. The humidity output would be used for input into the FL model. Apart from high humidity resulting in low energy levels and lethargy, it can also result in hyperthermia in extreme cases as a result of the body's inability to dispose of heat energy. The humidity sensor chosen for the application was the HIH-4000 since it was the most compact, accurate and reliable compared to similar sensors like the KS0430 Keystudio module. The following characteristics contribute towards the choice of using the HIH-4000 module:

- Ultra-low power requirement of less than 200 μA (requirement is <1 mA)
- The element of the sensor has a 3 layer design which protects against wetting, dirt, dust, chemicals etc. (requirement is water proof and dust proof)
- There is an almost linear voltage vs % relative humidity (RH).
- The sensor is relatively easy to set-up.
- Requires a low voltage supply in the range of 4 V – 5.8 V (requirement is 3.3 V to 5 V)

The HIH-4000 has a ground, voltage and analogue output pin. As with the LM35 sensor, the ground and voltage is connected to the ground and output pin of the LM7805 (5 V) voltage regulator respectively while the output analogue pin is connected to one of the Arduino Nano ADC channels. An 80 KOhm resistor is required across the ground and output of the HIH-4000 which acts as a load resistor to stabilize the output value. The set-up is shown in Figure 3-5.

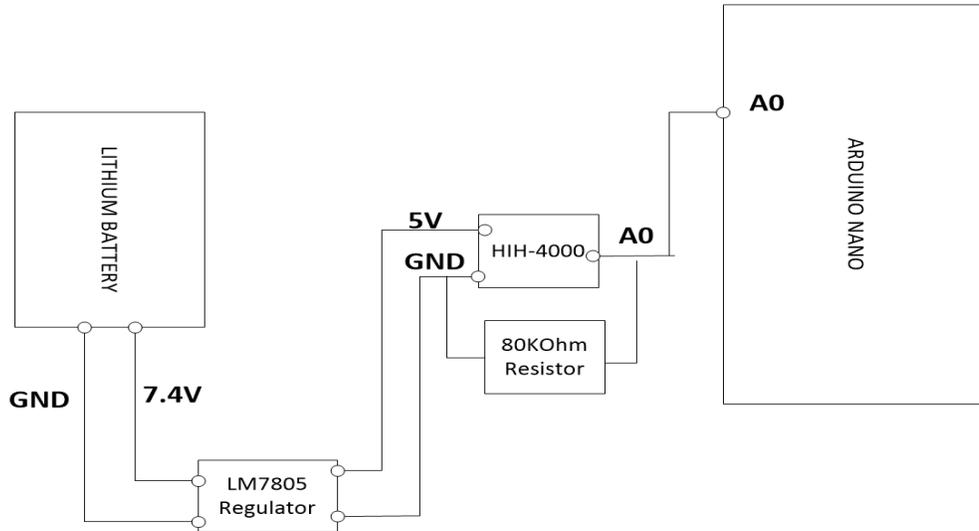


Figure 3-5: HIH-4000 Connection with MCU

The ADC output can then be used to determine the output voltage at 25 °C where V_{supply} is the sensor 5V supply and sensor RH is the measured sensor ADC value.

$$V_{out} = V_{Supply} \times (0.0062 \times Sensor\ RH + 0.16) \dots\dots\dots (3.4)$$

The above equation can be manipulated to give the RH at 25 °C:

$$Sensor\ RH = \left(\frac{V_{out}}{V_{supply}} \times 0.0062 \right) - 25.81 \dots\dots\dots (3.5)$$

The true RH can then be calculated by taking into consideration the effect of temperature on humidity using equation (3.6) where T is the temperature at the measured sensor RH:

$$True\ RH = \frac{Sensor\ RH}{1.0546 - 0.00216\ T} \dots\dots\dots (3.6)$$

To determine if the effect of temperature was considerable on the humidity, a graph was drawn (Figure 3-6) showing the change in RH at different temperature values between 0 °C – 120 °C. The results showed that there was a difference of 1.1 % – 5.7 % between the humidity value at

25 °C and the humidity value at different temperatures. The higher temperatures have a more significant effect on temperature. Since the range of temperatures experienced in the SA climate do not reach such extremes, it was decided that the equation for RH at 25 °C would be a sufficient indicator for the intended application and could be used in the MCU code.

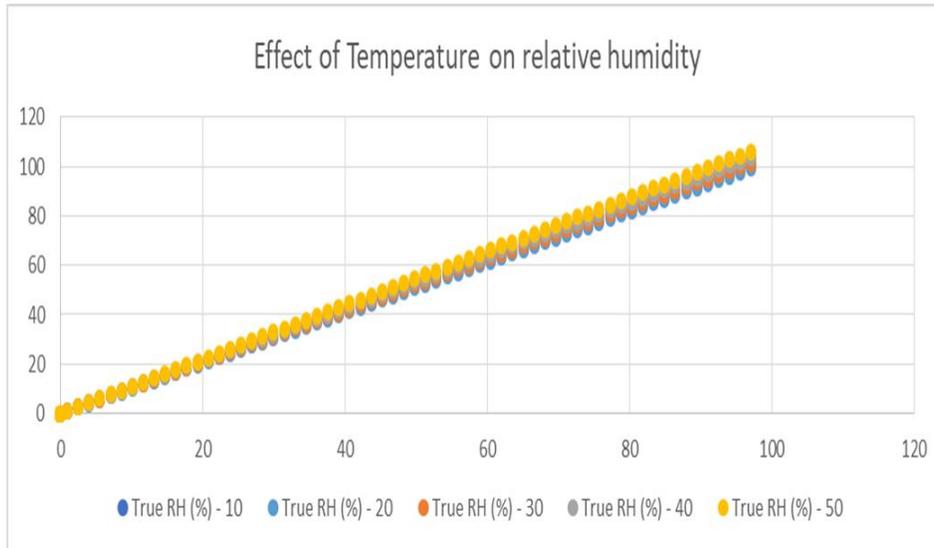


Figure 3-6: Effect of Different Temperatures on Relative Humidity

3.3.5. ECG Sensor

The ECG sensor used for determination of the patient's ECG signal was the Keystudio AD8232. The other option was the Mikroelektronika ECG module, however this was much more expensive and required an adapter for use on the Arduino Nano MCU. The AD8232 is a low power consumption (170 μ A) single lead heart rate monitor that has an operating voltage of 3.3 V. The requirement for such as sensor was current <1 mA and voltage between 3.3 V and 5 V. The AD8232 has a built in circuit designed to extract, amplify and filter small biopotential signals even in noisy conditions. In addition, the AD8232 has a leads off detection feature which indicates when all three electrodes are not connected hence when readings need to be retaken, this is a major requirement to ensure that the user is aware when the sensor is disconnected.

The AD8232 has a six pins. The SDN pin was not used as this would power down the device when connected to the digital pin of the Arduino. The ground and 3.3 V pin of the ECG sensor is connected to the ground and 3.3 V pin of the LD33CV (3 V) regulator respectively, which is connected to the output of the lithium battery. The output LO+ and LO- pin of the AD8232 is also connected to the digital pins of the Arduino and is used to activate the leads off detection feature. The block flow diagram showing the connections can be seen in Figure 3-7.

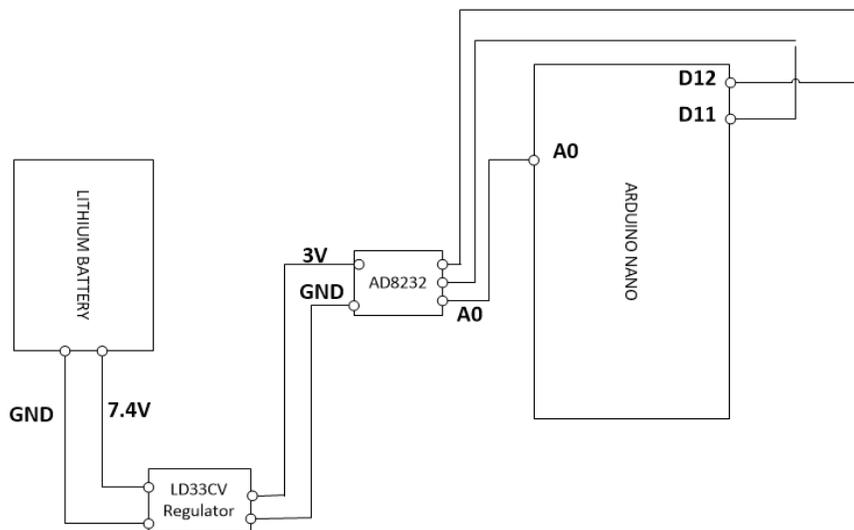


Figure 3-7: AD8232 Connection to MCU

The chosen reference voltage for the system was selected as 5 V, however the AD8232 is the only sensor that operates on a voltage of 3.3 V. As a result of this, the 5 V reference will not give the correct output peaks. To correct the issue, the MCU code was tweaked to recalculate the ADC output from the sensor. Using equation (3.1), for each ADC output, the corresponding output voltage was calculated using the known reference of 5 V. The calculated voltage was then substituted back into equation 3.1 to calculate the corrected ADC output now using a reference voltage of 3.3 V.

3.3.6. EMG Sensor

The Myoware muscle sensor was selected to determine a patient's EMG waveforms which would be used to identify muscular issues. The other options for EMG were sensors like the DFRobot SEN0240, however these sensors were much more expensive and did not have electrodes directly on the sensor. The Myoware sensor filters and rectifies the electrical activity from the muscular area it is located on. Once again, signal amplification and filtering is paramount to ensure data is correctly processed. It then outputs a voltage indicating the level of activity experienced by the muscle it is measuring. More activity will give off a higher voltage. The sensor has the placement of electrodes directly on the sensor which prevent noise artefacts caused by long cables. The forearm was selected as the area for muscle measurement as it allowed for a perfect positioning based on the wrist strap WBAN concept.

The sensor operates with a supply voltage of between 3.1 V to 5 V therefore a 5 V supply was selected which was in accordance with voltage requirements. The ground and voltage pins were

connected to the ground and output pins of the LM7805 regulator respectively while the analogue output pin was fed into the ADC of the Arduino Nano. The connection is shown in Figure 3-8.

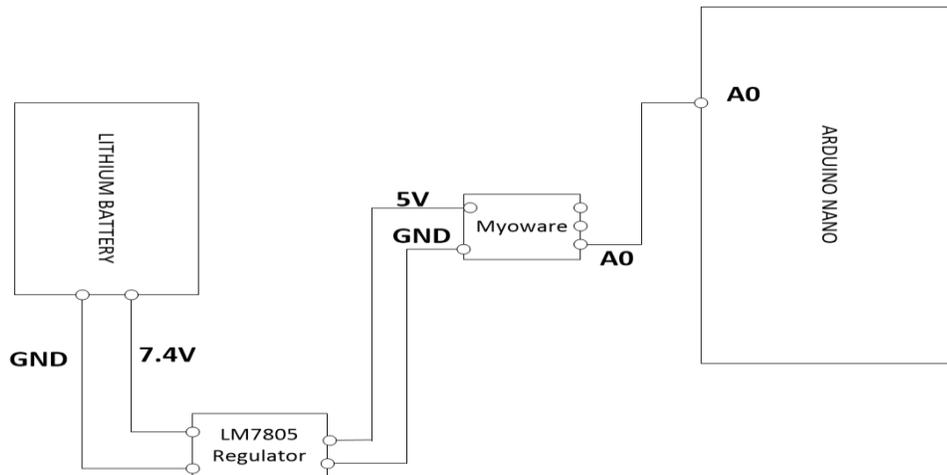


Figure 3-8: Myoware EMG Connection with MCU

3.3.7. Pulse Rate Sensor

The Keystudio analogue output pulse rate sensor was used as the heart beat measurement tool. Other options considered were the MAX30100 range sensors, however in terms of prices and ease of set-up – the Keystudio sensor proved a better choice. The Keystudio sensor is compact, affordable and lightweight which makes it suited for its function. The front of the sensor, as shown in Figure 3-9 is made up a small round hole and a square ambient light sensor. When a person places their earlobe or finger on the front side, the ambient light reflects off and enters through the round hole to be read by the sensor. The circuit connection is shown in Figure 3-10.

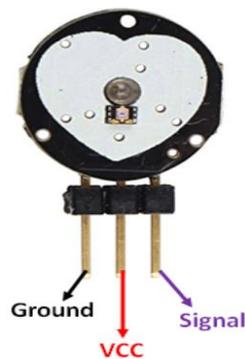


Figure 3-9: Pulse Rate Sensor

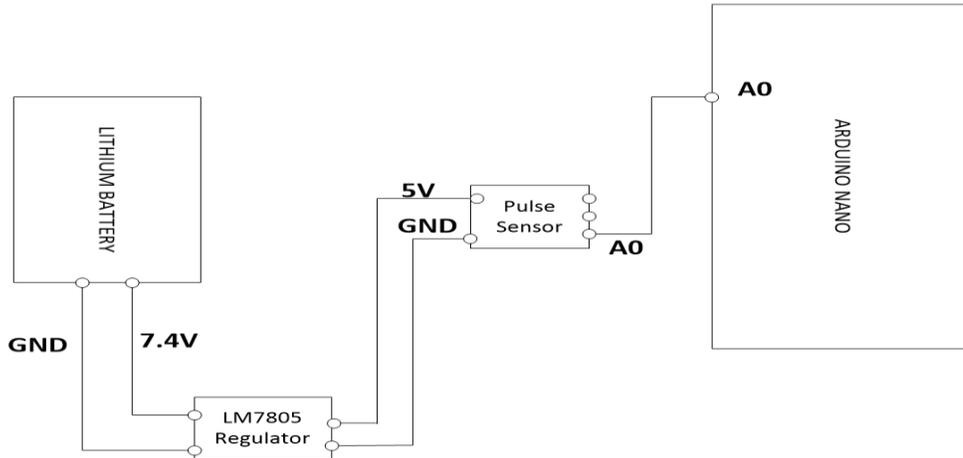


Figure 3-10: Pulse Rate Sensor Connection with MCU

In order to get the BPM, a threshold ADC output value needs to be selected to represent a peak/beat. A value of 512 was selected almost half way through the 1024 ADC resolution. The number of thresholds in a 10 second period is then calculated using timers on the MCU code and multiplied by 6 to give the BPM.

3.3.8. IR Transmitter and Receiver

The Keyes flame sensor was utilized as the sensor for measuring IR readings. Other options considered were PIR sensors however these required amplification circuits to be built which would not satisfy the size requirements. The Keyes flame sensor although traditionally used to detect fire through IR light was adapted to measure IR absorption when IR light passes through a patient's finger. The set-up includes an IR LED which gives of IR wavelengths of up to 940 nm. The index finger is placed between the IR LED and the flame sensor which can measure wavelengths in the range of 760 nm to 1100 nm which is a requirement for glucose detection. As the light shines through the finger it will then come out at a lower frequency based on the light that is absorbed by the constituents in the blood as indicated by Beer Lambert's law discussed previously. This can be used to measure different components in the patient's blood through calibration with a known sensing device. Glucose measurements were taken using the IR transmitter and receiver set-up and plotted against a calibrated glucometer. The resulting straight line equation was then determined which could be used as a predictor of a person's blood glucose levels. The equation and straight line derivation can be seen in the results section 8.1.3. Figure 3-11 shows the IR sensor connection with the MCU. The sensor uses a 5 V supply voltage which falls within the voltage requirements of 3.3 V to 5 V. The ground and voltage pins were connected to the ground and output pins of the LM7805 regulator respectively while the analogue output pin was fed into the ADC of the Arduino Nano. The IR Led was connected to a digital pin of the MCU.

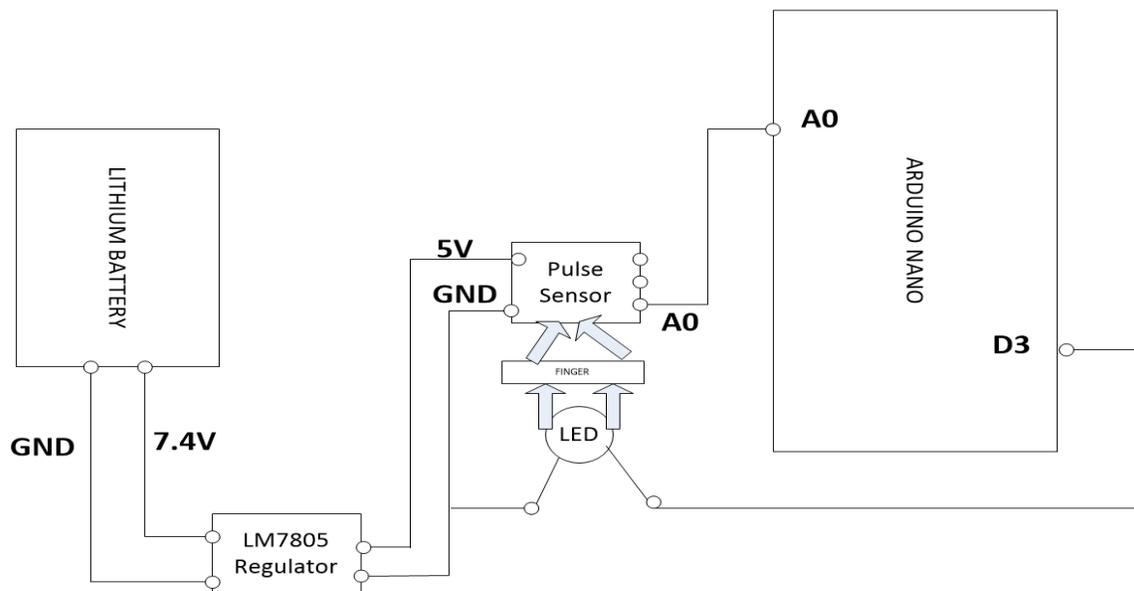


Figure 3-11: IR Sensor Connection with MCU

3.3.9. Lithium Battery and Balanced Charger

Based on the current requirements calculated in Appendix I, it was decided that a 450 mA Lithium battery would allow for sufficient running of the prototype for testing (at least 2 hours). A compromise on the current was made to satisfy the small size requirement. In addition, the Lithium battery chosen carried a voltage of 7.4 V which satisfied the maximum supply voltage requirement for the sensors and MCU. The system was designed so that the battery can easily be removed and recharged using the balanced charger purchased.

3.3.10. Veroboard Design

A PCB layout could have been used to optimize on space, however for the purpose of this dissertation, it was decided to develop the prototype using a veroboard layout while also maximizing on board space. The veroboard will be used to establish the connection between the various components and to hold the components in place. A veroboard with L-80 mm by W-50 mm was cut up and allowed for the connection of all the components. The stripboard layout showing the connection of the components can be seen in Appendix D. Figure 3-12 shows the veroboard with all the connections made with jumpers, ribbon cables and headers. To protect the end connections, ribbon cables were used to close off the exposed ends.

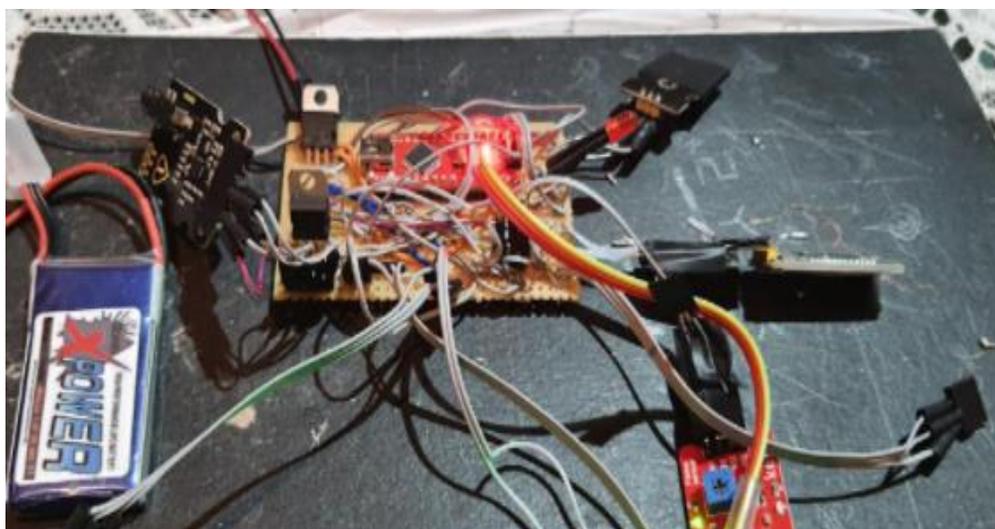


Figure 3-12: Components connected to Veroboard

3.4. MCU Software Design

The MCU software design includes all aspects of programming the Arduino Nano to read the ADC output of the sensors, control the haptic and visual indicators as well as conversion of the ADC output to a measurable variable. The code also focuses on creating strings with unique delimiters that are sent via the UART protocol in buckets, using interrupts and timers. This so that it can be processed by the MA. The MCU software design focuses less on heavy data processing and more on extracting and sending data from the WBAN network to the IoT gateway. The preceding sections will delve into each component in more detail.

3.4.1. Program Flow Initiation

The microcontroller is programmed using the CodeVision AVR software which is an IDE that consists of an automatic program generator called the CodeWizard AVR. Codevision is implemented using the C language and is compatible with the Atmega 328P chipset being used. Figure 3-13 highlights the process flow for initiating the program code using CodeWizard AVR. The process starts of by creating a new project based on the Atmega chipset properties. The relevant input and output ports are then set-up based on requirements. The interrupts, timers, USART and ADC settings are then enabled before generating the code for editing.

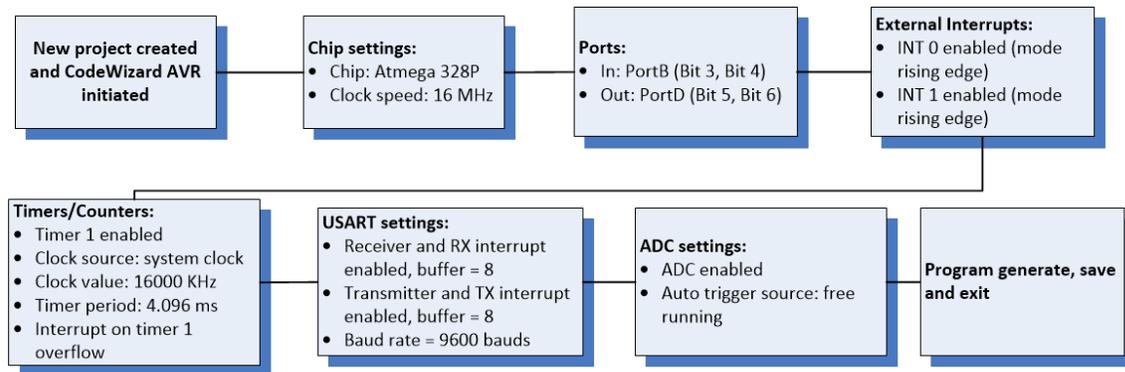


Figure 3-13: CodeVision AVR Wizard Set-up

3.4.2. Program Flow Operation

Once the initial program code is generated, the code can then be edited to perform the following functions:

- Read analogue sensor values.
- Convert ADC output to measurable parameters e.g. conversion of ADC output for the LM35 output to a °C reading.
- Set-up timers and interrupts to receive and send data through the UART protocol periodically.
- Control the haptic and visual feedback.

Figure 3-14 displays the operational flow for a single loop of the MCU. The process starts with a timer initiated loop. The loop is initiated after every 1000 ms. The loop will first start off by reading the sensor values through the read analogue function. Then converts this to a measurable parameter if required based on given correlations. In order to read the value via the serial output or MA, it is first converted to ASCII annotation before being transmitted. This process is accomplished through a series of IF statements that look at the integer value size and counts the number of thousands, hundreds, tens and unit values and prints each value as it is calculated. The *print* statement is then used to concatenate the measured value together with a “#” delimiter to separate sensor values, before sending to the MA using the UART protocol. Two other timers are used for the pulse rate sensor and the vibration motor. The vibration motor is used for reminding the patient to take readings and is set to go off after 30 min. It will then remain on for 5 seconds before resetting the timer to 0 for the next 30 min. The pulse rate sensor as mentioned in section 3.3.7 does not give a direct pulse output. A timer is used to calculate threshold ADC values over a 10 second period using the third timer. This value is then calculated over a minute

and communicated via UART as the BPM. The LED utilizes an interrupt routine to turn on based on info received from the MA when vital signs are determined to be above normal values defined.

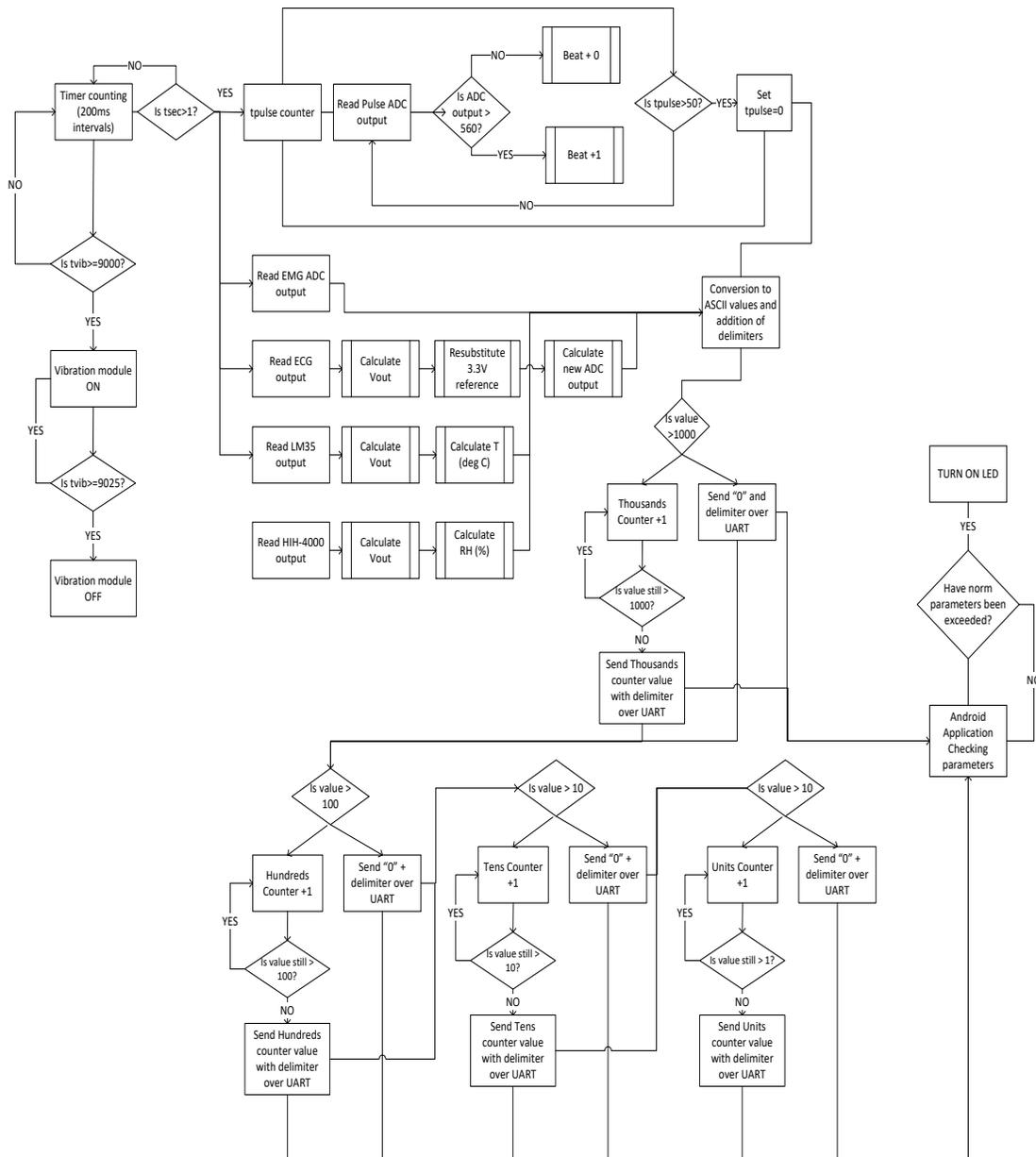


Figure 3-14: Single Loop Operational Flow MCU Code

3.4.3. ADC

The ADC settings were set to use the V_{ref} pin of the Arduino as the voltage reference which corresponds to a 5 V value. In addition, the auto trigger source was selected to be free running which means that the ADC is continuously doing conversions without having to be re-triggered.

This is desirable as the patient vitals will need to be checked on a continuous basis to flag potential health hazards.

3.4.4. Timers and Interrupts for Sending Data Packages

The speed at which data is received on the MA was controlled using a buffer on the MA which will be discussed in greater detail in chapter 4. In this way the parity between the MA and Bluetooth could be controlled through the use of delimiters. A 1000 ms timer was used to separate the transmission of different chunks of readings. The 1000 ms delay was trialed to be the minimum delay to ensure that the MA could process and split each chunk effectively without errors or crashes. Each set had a starting “#” delimiter and ending “#” delimiter. Which helped the android application identify the different sets and how they should be handled. Timer 1 overflow was set up for the MCU. The overflow value was calculated to be 50 using equation (3.7), where the clock frequency was set at 16000 KHz and n is equal to the 10 representing the 10 bit ADC system:

$$F_{Timer} = \frac{f_{clock}}{2^n} \dots\dots\dots (3.7)$$

3.5. WBAN Strap Integration

The WBAN strap includes the enclosures for housing of the electronic components as well as the Velcro strap used to secure the enclosure and sensors to the wrist and forearm of the patient. The smallest dimensions possible for the enclosure was calculated taking into consideration the dimensions as well as the optimum positioning of the sensors for accuracy of readings. The Velcro strap was to provide a firm enough hold of the enclosure while also providing enough pressure for some of the sensors to obtain an accurate enough reading from the surface of the skin.

3.5.1. Enclosure Selection and Sensor Positioning

Two commercial enclosures were selected to house the various components. A smaller enclosure houses the lithium battery, IR LED, sensor and Bluetooth and a larger enclosure was used to house the sensors, MCU, LED and vibration module. Some of the sensors are positioned outside of the casing so that readings can be taken. Various holes were made on the enclosure to allow for the electrodes to be visible for patient contact. The larger enclosure with the holes for the electrode pads, LED etc. is shown in Figure 3-15 along with the enclosure for the battery pack.

Two holes were cut up on the bottom of the larger enclosure for the two ECG electrodes. On the side of the enclosure is a hole for the HIH-4000 humidity sensor and the LM35 temperature sensor which requires an index finger contact. The smaller enclosure has an opening for the index finger

to allow for an IR reading to be taken as well as an opening to see the Bluetooth's LED so that patients can determine when pairing is successful. The top of the enclosure has a third opening for the ECG sensor ground electrode which requires the index finger contact. Also on the top is another opening for the LED to notify the patient of impending health issues. A layout diagram showing the positioning of the sensors can be found in appendix K.



Figure 3-15: Small Enclosure (black) and Large Enclosure (grey) with Holes for Electrodes

The ECG sensor needs to be in contact with a muscular region to measure contraction of the muscle fibres. It was therefore decided to position the sensor on the Velcro strap so that it could extend across the forearm region. A small Velcro strap was used to position the EMG sensor in place. Figure 3-16 shows the positioning of the sensor.



Figure 3-16: Positioning of EMG Sensor of Forearm Using Velcro Strap

3.5.2. Integration of WBAN Strap and Electronics

In order to obtain readings and ensure that the patient can comfortably use the device while moving around, the enclosure housing the electronic components needed to be strapped onto the patient's arm using a Velcro strap. The preceding subsections details how this was done.

3.5.2.1. Combining Electronics with WBAN Strap

As mentioned previously, the smaller enclosure which contains the battery was glued at a right angle to the larger enclosure with a small hole between the two enclosures to allow the battery connection with the veroboard. The battery and veroboard was then be placed in their respective enclosures taking sensor placement discussed into consideration. The enclosures were then closed and fastened with screws. Figures 3-17 and 3-18 show the components within their enclosure.



Figure 3-17: Electronics and Battery Pack within Smaller Enclosure

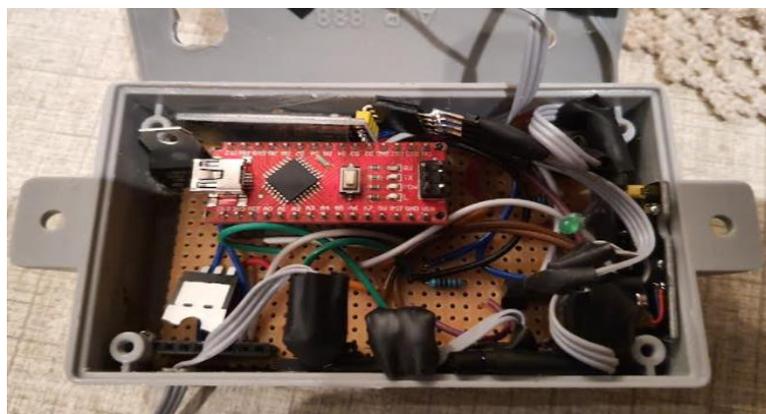


Figure 3-18: Electronics within Larger Enclosure

3.5.2.2. Prototype Fitting With Patient

The patient starts off by putting on the stretchable arm band which contains the EMG sensor positioned in place. The enclosures are then secured onto the patient's arm and the Velcro strap is placed across to hold the enclosure in place. Figure 3-19 highlights the proposed fitting.

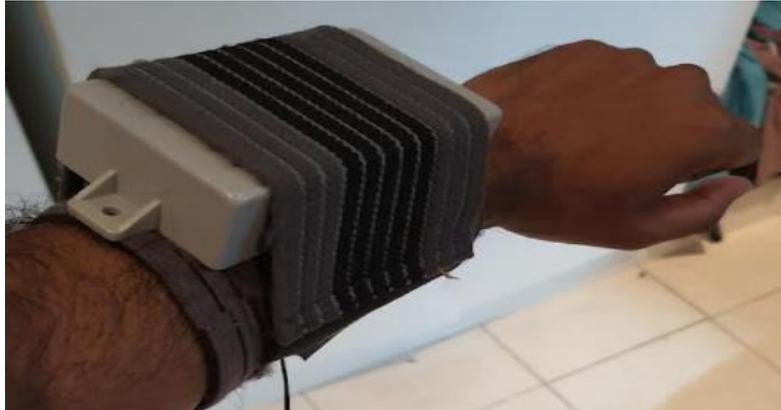


Figure 3-19: Fitting of WBAN on Patient's Arm

3.6. Chapter Summary

This chapter highlighted all details of how the WBAN layer was designed. Initially the characteristics of the sensors used within the WBAN were discussed. The expected output of the sensors, how they are interpreted as well as why they are important components of the sensing layer was discussed.

The chapter progressed with detailed descriptions of each component utilized within the WBAN. This included a discussion of the MCU connections with each sensor as well as MCU programming using the CVAVR software. The correlations to convert the ADC output to a valid sensory value were also discussed.

The chapter concluded with discussing the enclosures housing the sensors, how they are integrated with the WBAN strap and finally how the device fits onto the patient.

Chapter 4: IoT Gateway and GUI's

Chapter 3 looked at setting up the WBAN device which serves as the sensing layer to receive vital signs from the patient. The electronic components, MCU software and eventual WBAN prototype was explained in detail. This section will focus on the development of the MA which aids IoT gateway functionality as well GUI functionality. Thereafter the second GUI which is the WA will be explored. It is important to note that chapter 7 focuses on the use case scenarios for the WA and MA, however this chapter will look at the technical aspects and design considerations which facilitates the information flow through the IoT gateway and GUI's.

4.1. MA and IoT Gateway Development

The MA used in the system was developed in the Android studio IDE using the Java programming language. When designing the MA, a great deal of effort was made to ensure that the application satisfied the criteria of ease of usability, reliability and being a rich source of valuable information. These were the requirements identified by the key stakeholders in chapter 2. Since both physicians and patients have a different sets of requirements for the system, the MA needs to consider these differences and provide a different offering for each type of audience. The MA was therefore designed with two interfaces (a physician and patient interface). The preceding subsections will explain the activities within each interface in greater detail. Appendix C shows the MA UML diagram with the different activities that make up the application. Figure 4-1 shows the program flow which will be discussed in the preceding subsections.

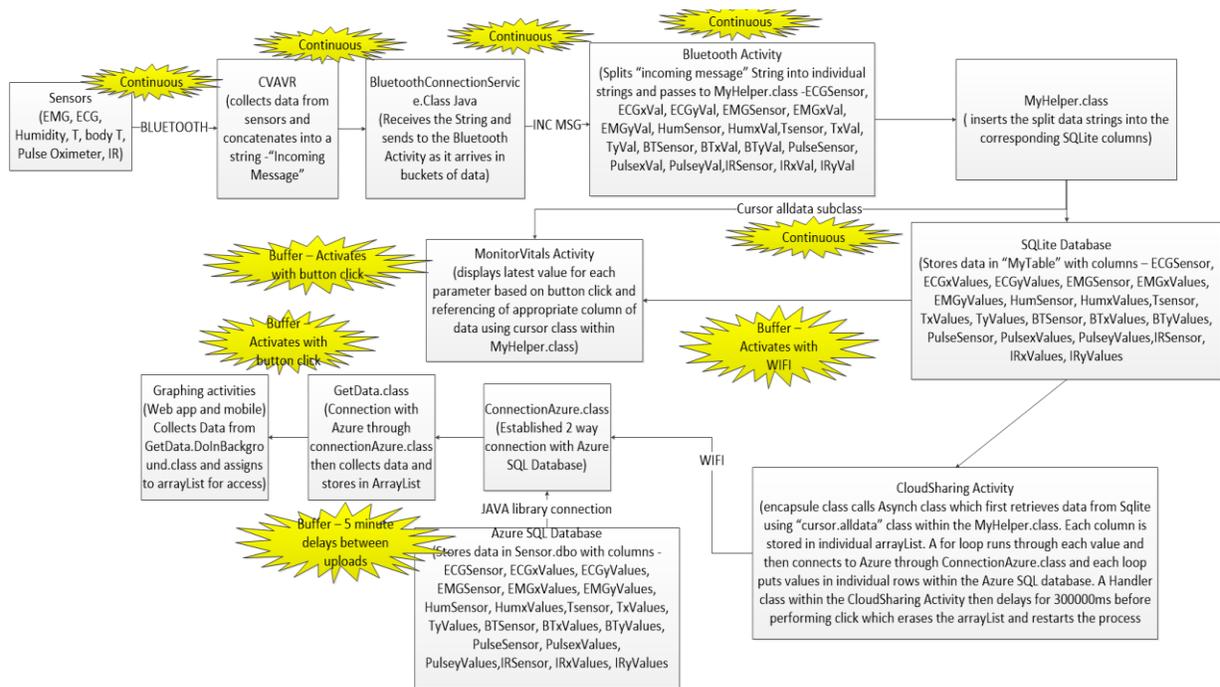


Figure 4-1: Operational Flow of MA

4.1.1. Patient Registration

In order for a patient to utilize the MA, he/she is required to first register on the application. The registration process was designed to extract as much personal information from the user that would help health practitioners gauge the effect of lifestyle choices and health statistics on the health profile of the patient. Data such as age, weight, smoking status, previous health conditions, emergency contact details etc. can help the physician react more effectively during emergency situations. In addition, these variables were later used as inputs to the ML model discussed in chapter 6. Once the patient starts entering their personal information on the registering pages, the data is then stored in hashMaps before a connection is established with Firebase and the data is uploaded to the cloud. The details of this transaction is explained in greater detail in chapter 6 and chapter 7.

4.1.2. Patient Tracking

The location manager application within Android Studio is used to identify the user's location using the phone's built in GPS settings. This prevented having to integrate a GPS module within the WBAN network which reduced costs and space consumption. In order for the process to work, the relevant permissions need to be set under the Android manifest file.

The ACCESS_FINE_LOCATION permission Android API needs to be granted to gauge the most precise location (within 3 square Km) whereas the ACCESS_COARSE_LOCATION grants permission to obtain an estimate location within 50 meters. On Android 12 (API level 31) or higher, users can only request that the MA provide approximate location (Android Studio, 2021). Therefore to avoid this mishap both the ACCESS_FINE_LOCATION and ACCESS_COARSE_LOCATION permissions should be set. ACCESS_BACKGROUND_LOCATION permission should be set and allows for continuous sharing of users location even when the MA is running in the background (Android Studio, 2021). This is important to ensure that user location is continuously being monitored in cases of emergencies.

Details such as longitude, latitude, altitude, speed and accuracy together with the patient's address is determined when the user activates the service on the MA. The service then runs in real time updating the patient's Firebase node each time a change is observed. This enables real time monitoring of the user's whereabouts by health practitioners.

4.1.3. Receiving Information via the RFCOMM Protocol

Figure 4-1 shows the operational steps undertaken by the MA when a patient utilizes the service. The process starts off by the patient registering or logging onto the patient interface. Once on the main interface, it is assumed that the patient performs all preliminary set-ups such as Bluetooth

and Wi-Fi enablement which is discussed in the use case scenarios in chapter 7. The Bluetooth activity allows the user to enable Bluetooth and discover neighboring devices through the *device_list_adapter class*. Search and pairing with other Bluetooth devices is made possible using *broadcast receivers*. In order for the code to work effectively, the following permissions require enablement under the Android MA manifest file:

The BLUETOOTH ADMIN permission is used to allow for the discovery of local Bluetooth devices. The BLUETOOTH_PRIVILEGED permission allows for successful pairing of devices (Android Studio, 2021). The BLUETOOTH permission is to allow general access to Bluetooth capabilities (Android Studio, 2021).

Information from the Bluetooth is received using the RFCOMM protocol via the phone’s Bluetooth and is processed using the *Bluetooth_connection_service class*. Data is stored in a byte which is converted to a string. A while loop then determines the start and end of a sensor sequence based on the “#” delimiter. Once a full sequence is detected it is then sent back to the *Bluetooth* class where the split function splits the sequence into individual sensor values based on the “-“ delimiter and split function. Prior to data being added to SQLite, the data will be checked against predefined thresholds to flag if the patient is having a medical emergency – in which case a short message service (SMS) will be sent to the doctor of the patient. This will allow for instantaneous alerts on problem cases.

4.1.4. Storage of Data in the SQLite Database

Through the use of a *helper Class*, the data is then uploaded onto a local SQLite database on the patient’s phone together with a created timestamp of the upload. The *helper* class accomplishes this through a *create_table* query which creates a new SQLite table with columns representing each sensor X and Y value. The *insert_data* function then adds the data to a matrix which is uploaded to SQLite using an *insert* query. The Schema shown in Figure 4-2 demonstrates the SQLite database structure after a successful upload.

	ECGSensor	ECGxValues	ECGyValues	EMGSensor	EMGxValues	EMGyValues	HumSensor	HumxValues	HumyValues	TSensor	TxValue
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1		E 2021/09042245	206.0		EM 2021/09042245	399.0		H 2021/09042245	632.0		T 2021/0904
2		E 2021/09042245	734.0		EM 2021/09042245	813.0		H 2021/09042245	359.0		T 2021/0904
3		E 2021/09042245	748.0		EM 2021/09042245	696.0		H 2021/09042245	163.0		T 2021/0904
4		E 2021/09042245	0.0		EM 2021/09042245	39.0		H 2021/09042245	547.0		T 2021/0904
5		E 2021/09042245	71.0		EM 2021/09042245	259.0		H 2021/09042245	655.0		T 2021/0904
6		E 2021/09042245	313.0		EM 2021/09042245	688.0		H 2021/09042245	462.0		T 2021/0904
7		E 2021/09042245	739.0		EM 2021/09042245	778.0		H 2021/09042245	193.0		T 2021/0904
8		E 2021/09042245	39.0		EM 2021/09042245	44.0		H 2021/09042245	472.0		T 2021/0904
9		E 2021/09042245	0.0		EM 2021/09042245	151.0		H 2021/09042245	659.0		T 2021/0904
10		E 2021/09042245	391.0		EM 2021/09042245	568.0		H 2021/09042245	537.0		T 2021/0904
11		E 2021/09042245	763.0		EM 2021/09042245	805.0		H 2021/09042245	262.0		T 2021/0904
12		E 2021/09042245	216.0		EM 2021/09042245	593.0		H 2021/09042245	172.0		T 2021/0904
13		E 2021/09042245	201.0		EM 2021/09042245	158.0		H 2021/09042245	364.0		T 2021/0904
14		E 2021/09042245	0.0		EM 2021/09042245	75.0		H 2021/09042245	611.0		T 2021/0904
15		E 2021/09042245	205.0		EM 2021/09042245	391.0		H 2021/09042245	615.0		T 2021/0904
16		E 2021/09042245	732.0		EM 2021/09042245	793.0		H 2021/09042245	352.0		T 2021/0904
17		E 2021/09042245	760.0		EM 2021/09042245	700.0		H 2021/09042245	156.0		T 2021/0904
18		E 2021/09042245	139.0		EM 2021/09042245	291.0		H 2021/09042245	284.0		T 2021/0904

Figure 4-2: Schema for SQLite Database

4.1.5. Viewing of Vitals on the Patient Interface

The MA allows the patient to view data in the *monitor_vitals* Activity. This is accomplished through a *cursor* function within the *helper* class that selects all values from the SQLite table through a *select* query. The *monitor_vitals* Activity then calls the *cursor* function and pulls the last value off the SQLite table to display as text on the front end of the MA. The patient's body mass index (BMI), FES scores and ML stroke probability is also calculated by creating a connection with the Firebase database to collect input data and then to subsequently upload the calculated values to the Firebase database so that health practitioners can later access these scores. This functionality and integration with Firebase will be discussed in chapter 5.

4.1.6. Cloud Upload

Assuming Wi-Fi has been set-up, the patient can then initiate the upload to the MS Azure SQL database. The *cloud sharing* Activity utilizes a *connection* Class to create a data pipeline with the Microsoft Azure SQL database. This connection will be further explained in Chapter 9. The following permissions need to be set on the Android Manifest file:

The INTERNET and ACCESS_NETWORK_STATE is used for accessing of the internet and network respectively. The CHANGE_WIFI_STATE allows the MA to change the WiFi connectivity state. The CHANGE_NETWORK_STATE allows the MA to change the network connectivity state. The ACCESS_WIFI_STATE allows the MA to access information about the WiFi networks.

The cloud sharing Activity starts off by retrieving data from the SQLite table through the *cursor* function within the *helper* class. Each SQLite column is stored in an individual array list. Before upload can be done, a function within the activity checks if a table with the patient's ID exists in the SQL database. If the table does not exist, the user's Firebase UID is extracted and used to create a new table with the relevant columns using a SQL *create_table* query. If the table already exists, each new upload deletes records using a SQL *delete query* and refreshes with updated data. A FOR LOOP runs through each column and adds individual column values to each row in the SQLite database through and *update query* and *asynchronous task*.

ID	ECGxValues	ECGyValues	EMGxValues
864	2021/10241306	359	2021/10241306
865	2021/10241306	220	2021/10241306
866	2021/10241306	304	2021/10241306
867	2021/10241306	339	2021/10241306
868	2021/10241306	362	2021/10241306
869	2021/10241306	372	2021/10241306
870	2021/10241306	374	2021/10241306

Figure 4-3: Schema for SQL Database Table on Azure Showing Uploaded Data

The update query runs each time a complete loop of sensor values are retrieved, this ensures that there are no blank rows on the SQL table. Once the row update is complete (i.e. all values from the SQLite database have been transferred to the SQL database), the array lists are emptied and the process restarts until all values within the columns have been successfully updated. There exists a *handler* class that waits for 5 minutes for the *asynch* task to complete its process of uploading. After every 5 minutes, a *perform_click* function restarts the upload process so that there is a 5 minute delay between continuous updates. To control the size of the local SQLite database on the user's phone, a *resize* function instance from the *helper* class is used to check if the rows in the SQLite database exceeds 5000 lines, if this is the case the oldest values are deleted using a *delete* query and the process of upload commences.

4.1.7. Granting Firebase Permissions

The *find_doctor* Activity works through the *RecyclerView_Adapter* and *OnBindViewHolder* functions which searches in the Firebase database for registered physicians. A class was created which allows for the basic physician information such as their specialty, location, mobile and email to be made available so that a patient can decide whether a physician is a good fit for them. Once a patient decides to share information with a doctor, the doctor's Firebase UID is automatically added to the patient's node – thus allowing the selected doctor to gain access to the patient's data. Figure 4-4 shows an example node of a patient where a specific doctor has been granted access to data:

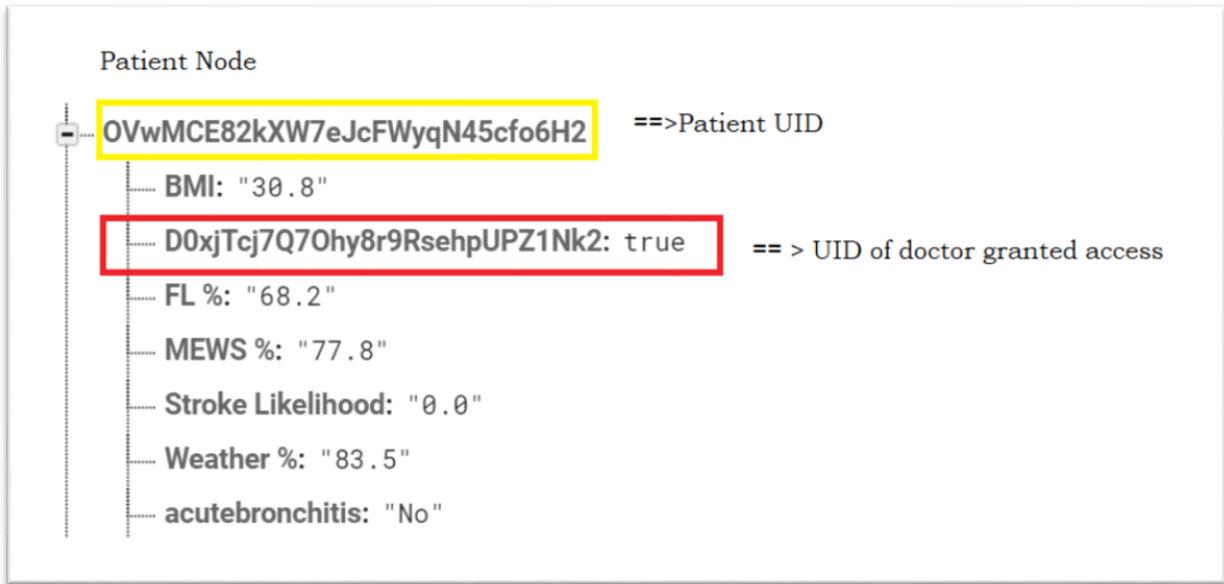


Figure 4-4: Sample Patient Node Showing Doctor Permissions

4.1.8. Physician Registration Process

The physician registration process is similar to that of the patient with the exception of the information collected during the registration process. The doctor registration process focuses on gathering information to validate the doctor and provide enough information for patient's to find physicians that may be able to help them. Hence the focus is on collecting information such as contact details, HPSA details and the physician's specialty area.

4.1.9. Displaying of Information on the Physician Interface

When the physician clicks on the *track_patient* Activity, a *RecyclerView_Adapter* and *OnBindViewHolder* function is used to search for a patient on the patient database. This is accomplished through a Firebase connection which searches for patients via their first names. The physician can then choose to either view the patient location or sensor values and diagnostics through the various Activities available. The location tracking feature connects with the patient Firebase database and retrieves the user's location from their node. The other activities for the sensor information and health status pull data from the MS Azure SQL database using a JDBC SDK which creates a connection with MS Azure using an API key. When patients upload their data, SQL tables corresponding to unique Firebase UID's are created. This UID is automatically selected when the physician selects a specific patient so the correct table is looked up in the SQL database. The SQL table columns are stored in individual *arraylists* and are then converted to integers before being plotted using the *MPAndroid* chart software. The X-axis labels are inserted

and formatted using the *LineChartXAxisValueFormatter*. The following dependencies need to be enabled in the *build.gradle* file of Android studio:

- MPAndroidChart:v3.1.0

4.1.10. Accessing Patient Timestamp Readings

The MP Android chart has x axis labels with numbers. These numbers are annotated to specific date and time values by extracting them from the SQL database through MS Azure connection and displaying them on a *ListView*. Ranges at which specific issues occurred can then be determined using these timestamps.

4.1.11. Integration with Models (FES, ML and MEWS)

The FL, ML and Modified Early Warning Score (MEWS) are first calculated on the MA before being uploaded to Firebase for access by the doctor interface. This prevents processor usage and time delays through duplication of calculations. The MEWS and FES models are programmed in the MA code. Inputs to the models are extracted from the Firebase database based on the logged in user and then plugged into the models to obtain an output (score and probability) which is displayed on the MA and uploaded to the cloud. The MA also takes the Firebase input data and through a POST request consumes the Azure ML classic studio model developed. The input data is posted and then in return a scoring and probability outcome is displayed in JSON format which is further formatted and displayed via a *TextView* on the MA.

4.1.12. MA Front End Design

The front end design of the MA was created to take into consideration ease of use and best practices. Typical application set-ups were utilized such as *registration* pages and *about* pages so that users could easily identify and navigate through activities. To visualize the mobile activities, wireframe diagrams were developed using a software called Balsamic Wireframes. Sample wireframes are shown in Figure 4-5.

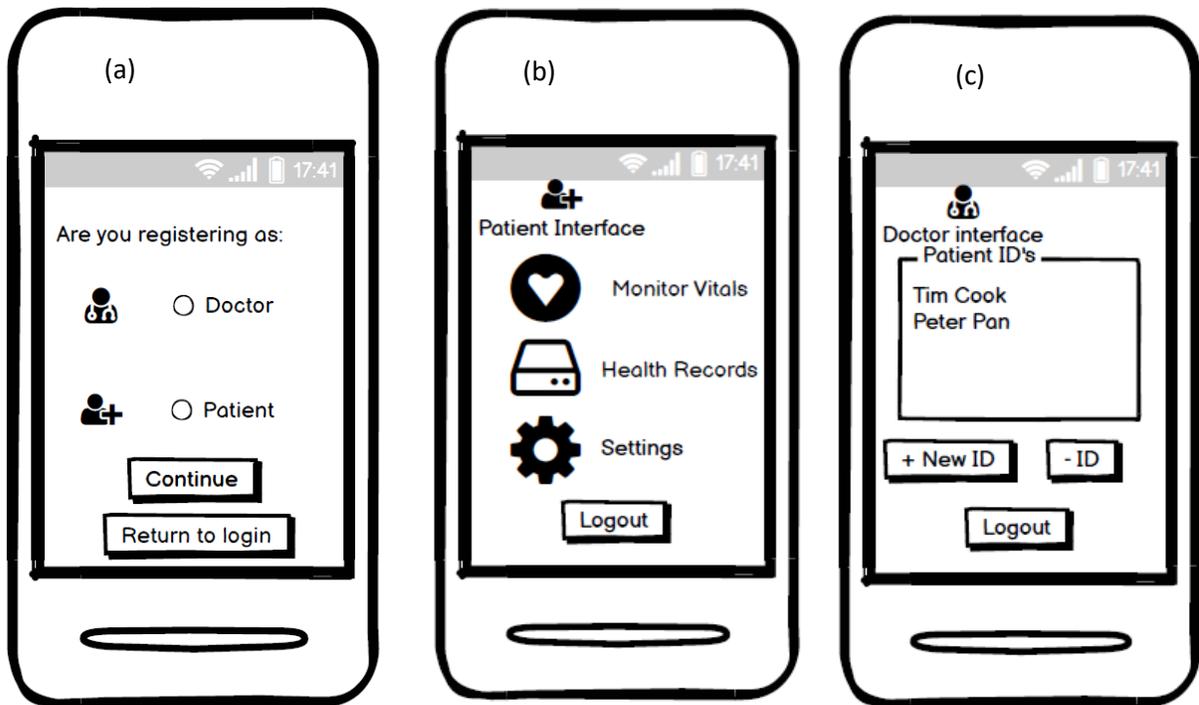


Figure 4-5: (a): Main Page Wireframe, (b) Patient Interface Wireframe, (c) Doctor interface Wireframe

The front-end design of the MA was developed using layout resources which consists of the buttons, lists and texts etc. that the patient uses to interact with the GUI and is written in Extensible Markup Language (XML). Android Studio enables a drag and drop approach which allows for simplicity in front-end design which is another reason it was chosen as the IDE for the design.

4.2. WA Development

The WA was created as a secondary application layer to be used on the hospital servers. This will enable accessibility to a wide array of health care workers including those that don't have access to the patient data via the MA. The WA was developed using Javascript running on the runtime environment node.js for the backend design and HTML, CSS and bootstrap for the front end design. The WA was designed to prioritize usability and provide a reliable and continuous supply of data to healthcare practitioners. The Visual studio source code editor was used to compile and develop the WA architecture. Within visual studio is a number of plugins that allows for easy integration with MS Azure. Figure 4-6 shows the general WA flow.

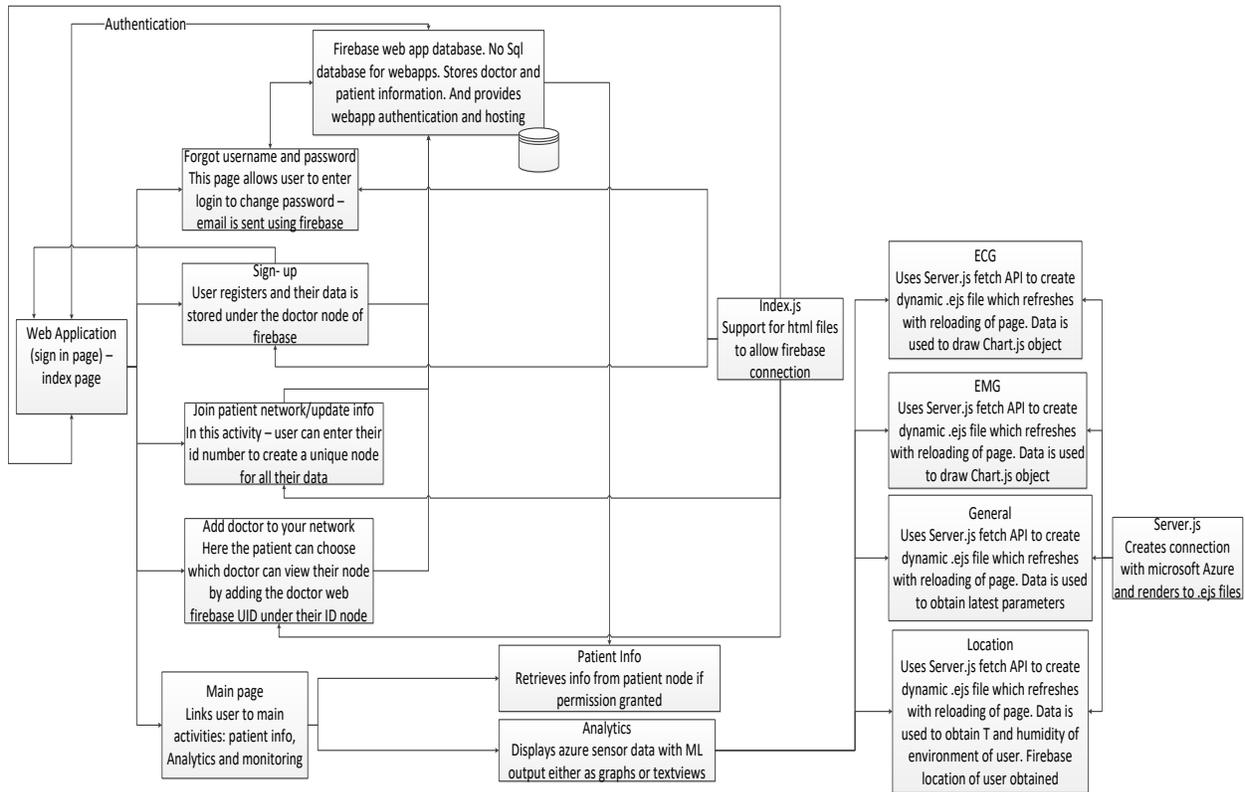


Figure 4-6: Flow of Information through WA

4.2.1. Back- End Design

The back-end design is achieved through the utilization of the node.js and *npm* package. Figure 4-7 shows the initial set-up process for the *node.js* server. The process starts by first downloading and installing the Node.js application which also installs with the *npm* packaging manager. The *http_server* package is then installed using the command prompt. Node.js needs to then be linked up with visual studio code editor. To do this, visual studio code is downloaded and installed followed by the plugins necessary for HTML, Azure and Node.js integration. An *index.js* file is then created to run all server code. To run the website on the browser, the server is started on the visual studio terminal using the *http-server* command.

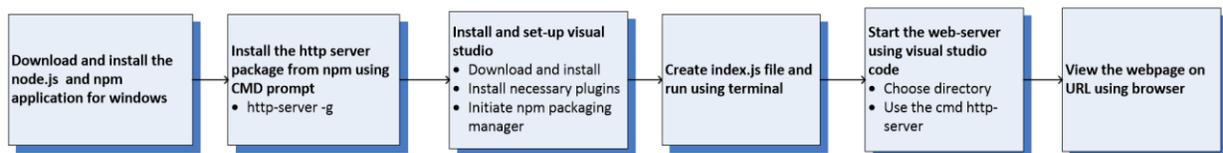


Figure 4-7: Node.js Server set-up

4.2.2. Integration with Databases

Since Firebase creates different types of databases for a MA and WA, a separate database for patients and doctors needed to be created for the WA. The WA therefore requires patient's to enter their data on the WA so that it can be viewed by physicians accessing the information. Here, the patient uses their ID number to create a unique node which the doctor can use to access their data. Like the MA, the WA also has the *registration/login* and *forgot password* pages for doctors using the system.

The WA was also integrated with the MS Azure SQL database to view patient readings on the *general patient info* page. Similarly EMG and ECG graphs can be viewed on the *ECG.js* and *EMG.js* pages respectively. To activate the server running these pages, a *server.js* file is set-up which uses POST requests to retrieve info from the Azure database. The servers can be activated on the command line in visual studio using the *node server.js* prompt. EJS (embedded Javascript) files were used to acquire sensor readings from the database, since they allowed for dynamic updating of the values each time the browser was updated. To access any of the EJS pages, there is a sequence that is followed to correctly start and end a connection. The first page asks the doctor to enter the patient ID before proceeding to the next page, if this matches the database – the next *transition.js* page is reached. Once completed, the doctor will be asked to close the session which successfully ends the connection.

4.3. Chapter Summary

Chapter 4 looked at the detailed front and back-end designs of the MA and WA. The use of the MA as an IoT gateway was also discussed. Details of how information flows from the WBAN to the SQLite database to the MS Azure SQL database was examined. In addition, the integration with Firebase for patient and doctor registrations was looked at.

The WA front end and back-end design was presented and an explanation of how doctors and other health practitioners can access information through the WA was also highlighted. The use of EJS pages for dynamic viewing of SQL data was touched on.

Chapter 5: Cloud Infrastructure

Chapter 4 discussed the development of the MA and WA and briefly on how a connection pipeline was created with the databases to upload and download information to and from the cloud through the GUI's. This chapter will focus on discussing the Firebase and MS Azure SQL database set-ups as well as the technicalities of how connections between the cloud and the MA and WA were established.

5.1. Firebase Integration

Firebase can be defined as a NoSQL database program that stores data in a JSON like format which uses the *backend-as-a-service* cloud model to take care of all backend database services. The data follows a tree like hierarchy with nodes/branches containing user data. It offers a variety of features ranging from user registrations, forgotten passwords retrieval and JSON rules for data security. The Firebase NoSQL database was chosen to register and store registration information of the users because of these easy to integrate features. The Firebase database service offers advantages over traditional app databases which requires backend code to be written to retrieve data. With Firebase, the client does all the work and all that is required is a SDK to provide a connection between Firebase and the MA. Figure 5-1 and 5-2 show the difference between a traditional set-up and Firebase set-up.

Traditional

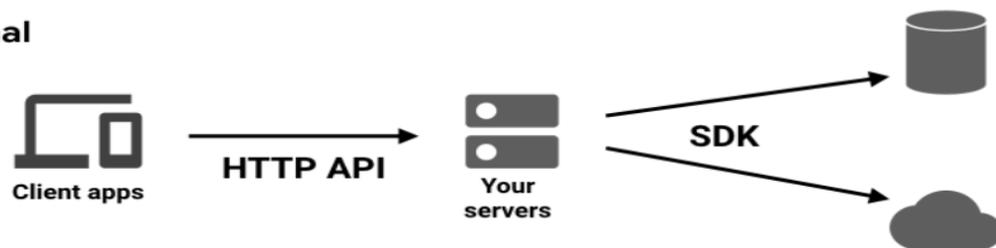


Figure 5-1: Traditional Database Set-up

Firestore



Figure 5-2: Firestore Database Set-Up

5.1.1. Setting up A Firebase Project

Before integrating Firebase with either a mobile or WA, a Firebase account needs to be created and preceded by setting up a Firebase project. The Figure 5-3 highlights the process that was followed:

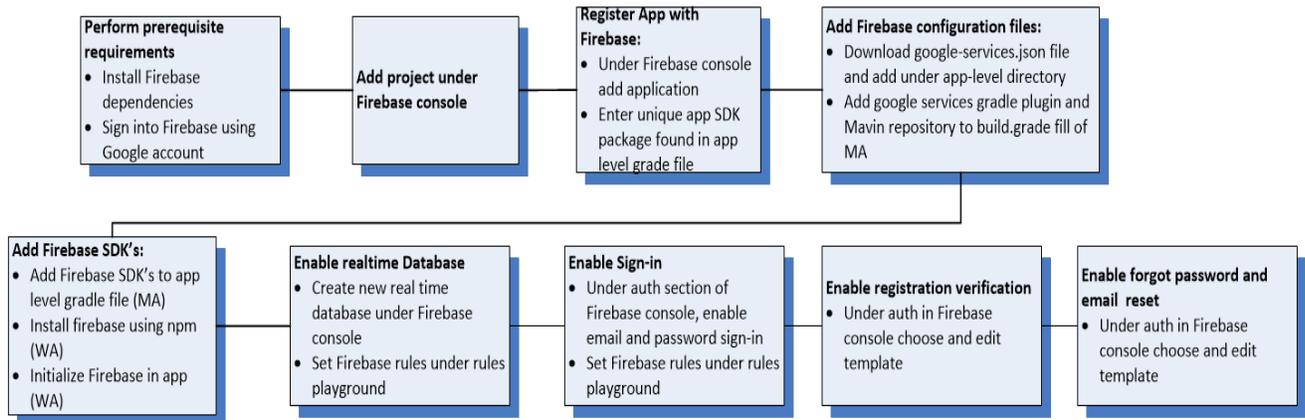


Figure 5-3: Firebase Console Set-Up

5.1.2. MA Integration

Firebase is integrated with the MA to perform the following functions:

- Register users through a unique authentication procedure.
- Store user data under nodes in the real time database.
- Retrieve data from the real time database.
- Set JSON rules to protect user data.

The Firebase registration process is initiated by first setting up a Firebase authorization and database reference instance in each of the *registration* Activities on the MA. A connection with the MA was established through the Firebase console. The database reference should have the correct child paths to ensure that the data is entered under the correct node. For sign-up, the *create_user_with_email_and_password* function will create a new user with unique UID. For logging in, a Firebase authorization instance is created and the *sign_in_with_email_and_password* function is used to successfully log a user into the application. In the case of an error, a toast message indicating the reason for the error login will be displayed on the user's MA.

The uploading of data into the Firebase database is similar to the uploading done during the registration process. The *getcurrentuserID* feature retrieves the logged in user and database reference instance with the correct child structure. Data can then be uploaded onto the correct

node. For retrieving data from Firebase, once again a database instance with the child structure can be created and a snapshot feature is used to extract the child data of a node into strings.

Since the MA utilized two databases, a secondary database needed to be declared so that the above functions could be carried out on the secondary database. Information such as project ID, application ID, API key and database URL needed to be set before the secondary database could be initialized. Thereafter the process for registering, logging, uploading and downloading information is the same as described for the primary database.

5.1.3. Web Integration

Firebase is integrated with the WA to perform the following functions:

- Register users through a unique authentication procedure.
- Store user data under nodes in the real time database.
- Retrieve data from the real time database.
- Set JSON rules to protect user data.
- Retrieve forgotten passwords.

The web integration process with Firebase typically follows a similar process to that of the MA. The exception is that the WA establishes a connection with the web based Firebase database by initializing a Firebase configuration on a JavaScript file. The configuration details will include:

- *apiKey*
- *AuthDomain*
- *projectId*
- *storageBucket*
- *messagingSenderId*
- *appld*
- *measurementId*

Once the connection is established and provided that the process of setting-up the Firebase console as highlighted in Figure 5-3 is established. The registration and receiving of information using the database is then possible. The registration process will use a *create_user_with_new_email_and_password* function to create a new user while the *signin_with_email_and_password* query will sign existing users in. The retrieving of information such as location and patient information from Firebase can be retrieved by creating a connection in a Javascript file and then using the *snapshot* feature of Firebase together with a database reference instance to extract the data into elements on the HTML pages. The process for uploading also requires a Firebase reference instance but instead uses the *update* functionality to upload or update a Firebase node and corresponding child node information. The Firebase reference root is constructed using the patient's ID number so that data is extracted from the correct nodes and child nodes. ID numbers are required since patients don't register on the WA,

instead they upload their information to the doctor's database using their unique SA ID numbers. This simplified the process since patients don't need to be viewing the data on WA.

5.1.4. JSON Rules

The Firebase database uses JSON rules and a user's unique UID to protect their data. These rules are highlighted in Figures 5-4 to 5-6. Each stakeholder (patient and doctor) have unique requirements with regards to data privacy. For the MA, rules are set for the patient database and doctor database. Within the patient interface are two nodes and Firebase allows each node to have specific security rules. The public user node within the patient interface is set that all authorized users can read and write to it. This allows doctors to search up patients. The user private node is set up to only allow physicians with permissions to access a patient's personal details. When the patient adds a doctor to their network, the doctor's unique UID is added under the private node of the patient. This enforces Firebase rule of only allowing UID's that fall under the private patient node to be allowed access to that nodes data. The physician Firebase data rules is set to allow all authorized users to access it. Since the physician's don't have personal information to protect, this is a suitable option.

For the WA, anyone can write on the Firebase node but only authorized users that have their ID in the created node can read it. This is to allow for patient's to enter their data onto the WA as they are not physically registered on the WA themselves.

```
//any authorized user can read and write on the patient public node - no private information

{
  "rules": {
    "Users_Public": {
      ".read": "auth != null",
      ".write": "auth != null"
    },

    //any authorized users can write on the private patient node
    //doctors that have been given access by the owner of the private patient node can access the info
    // and only auth users can read from patient node that is the patient him/herself

    "$UserID": {
      ".write": "auth != null",

      "$UserID": {
        ".read": "root.child('Users').child($UserID).child(auth.uid).exists() || auth != null",
      }
    }
  }
}
```

Figure 5-4: Public Patient Firebase JSON rules

```
//read and write set to public. Anyone can read and write, however no critical info  
// in cases of changes - they can always be corrected
```

```
{  
  "rules": {  
    ".read": true,  
    ".write": true  
  }  
}
```

Figure 5-5: WA Doctor Firebase JSON Rules

```
//anyone logged in or not can write on the patient node  
//but only authorized users that have their ID in the patient node can read
```

```
{  
  "rules": {  
    "Users": {  
      ".read": true,  
      ".write": true  
    },  
  },  
}
```

```
//any authorized users can write on the private patient node  
//only doctors that have been given access by the owner of the private patient node  
//can access the info
```

```
{  
  "$UserID": {  
    ".write": true,  
  },  
  "$UserID": {  
    ".read": "root.child('Patient').child($UserID).child('doctorid').val()===auth.uid",  
  }  
}
```

Figure 5-6: Private Patient Firebase JSON Rules

5.2. MS Azure Integration

Microsoft Azure was chosen as the cloud computing service to handle to the storage of sensory information from the WBAN. There are a variety of cloud services that currently exist however Azure was chosen as it offers ease of integration, a wide service offering, a variety of online resources to help understand the service offerings and a costing structure that helps control and monitor costs. Two services within the Azure suite were chosen to be incorporated into the design of the IoT system. The first being the ML studio classic and the second being the Azure SQL database. The SQL database is used to store information received from the MA while the ML studio classic has a drag and drop configuration to help create, deploy and consume ML models. The use and creation of these models are explained in greater detail in chapter 6. The sections below will highlight the set-up and integration of Microsoft Azure such that it is able to make a successful data pipeline with both the MA and WA.

5.2.1. Setting-up Microsoft Azure

The process in Figure 5-7 highlights the procedure undertaken to set-up the MS Azure account to fulfill the requirements. The process starts off with registering, choosing of the plan and thereafter setting up the services to integrate with the MA and WA. The standard subscription plan was chosen for most of the services as it provided enough resources for the prototype design. A standard SQL database with 2GB of data and 5 DTU's was sufficient to test the prototype. A server for the database was then created under the *created_resource* group. A resource group in Azure is essentially a container that houses all resources used within an Azure account. And resources are the services like SQL and ML which were utilized in the design. Once the server was created, the corresponding firewall rules were set to allow access to the data by users based on their IP addresses. For the ML modelling, Azure's ML classic studio was utilized as it features a multitude of models and data manipulation techniques through a modularized design which enables efficient modelling through a drag and drop approach.

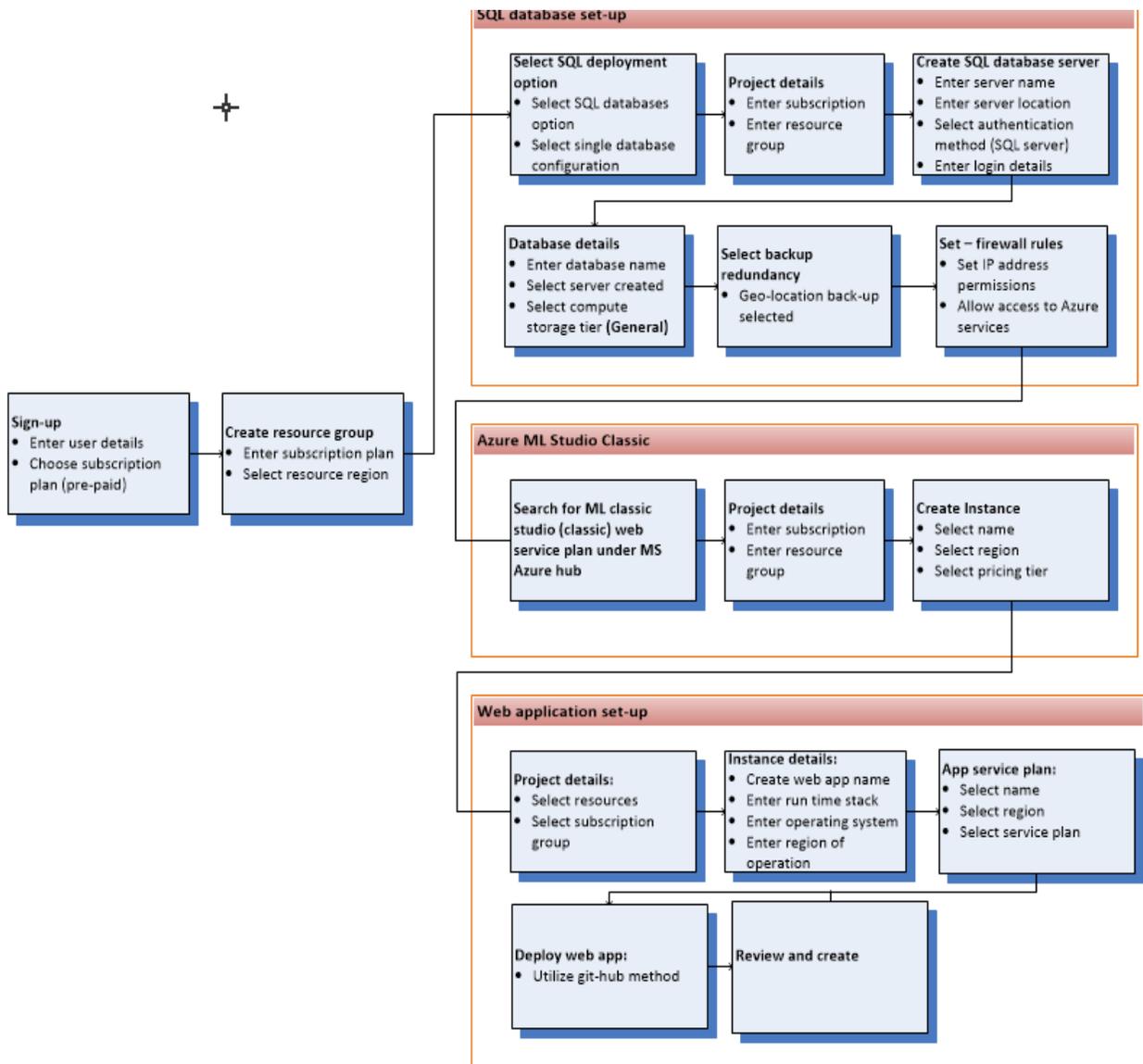


Figure 5-7: Setting-Up MS Azure Services

5.2.2. Mobile Integration

The connection between the Microsoft Azure SQL database and the MA was accomplished through the JDBC API. There are a variety of other methods to create a connection such as through a WA service on Azure, however JDBC method is both open source and does not impose any fees. The first version was released in 1997 with the aim of creating connections with SQL databases as well as creating queries and commands. Figure 5-8 shows the process of integration and communication to and from the SQL table and MA using the JDBC SDK. The process starts by first creating a connection with the MA and SQL table. Thereafter SQL queries can be used to create the SQL table using the unique user ID, retrieving data for viewing on the doctor interface, transmission of data from SQLite to Azure SQL or clearing of tables before each upload. On the MA, a *ConnectionAzure* class was created which is used in Activities where queries on the SQL database needs to be run.

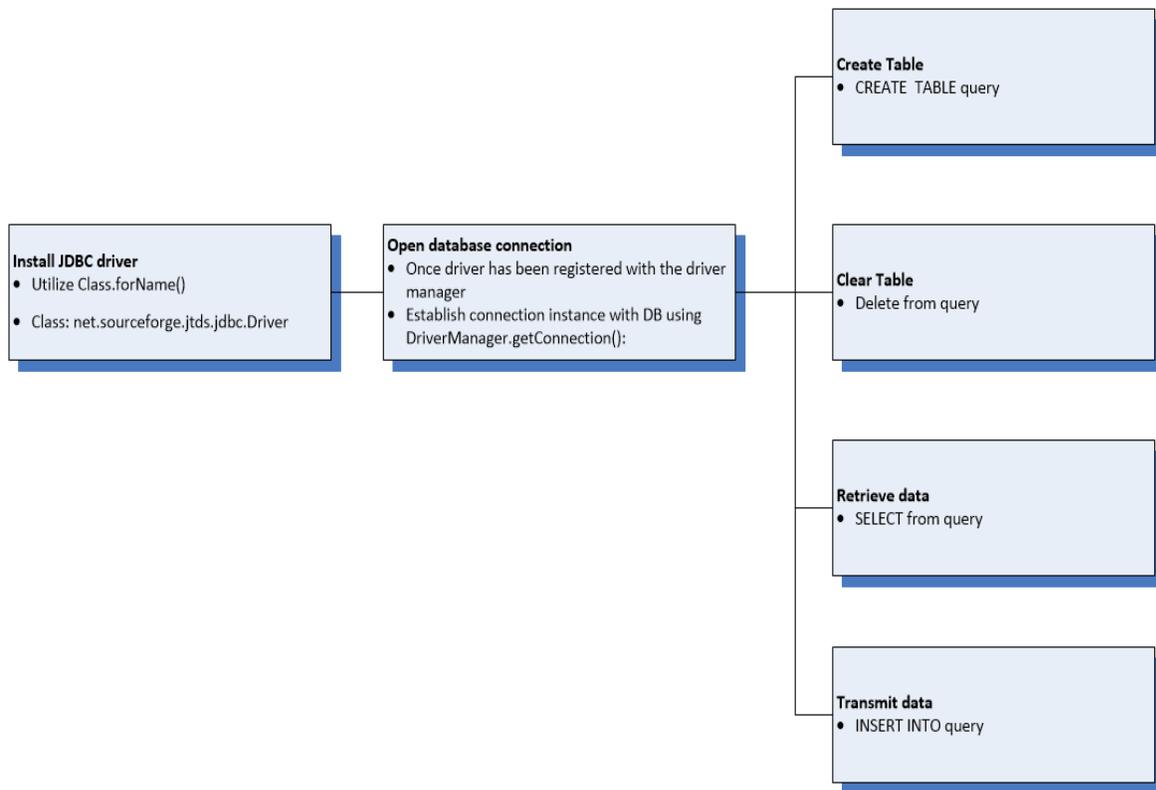


Figure 5-8: Creating a Connection between MS Azure and MA Using JDBC SDK

5.2.3. Web Integration

The web integration of MS Azure with the WA required the use of a template engine since dynamic pages are required to communicate the change in database information when refreshing the browser. The EJS express.js template engine was utilized to carry out the task. The first step involves using *npm* to install EJS on the console. A *server.js* file is then created in which the view engine (EJS) is set and a configuration is created and used to establish the SQL connection. The following configuration information was used to establish the connection:

- *User*
- *Password*
- *Server*
- *Database*

An EJS file is then created with the body similar to a HTML page. A route is then created in the *server.js* file which allows rendering of the EJS file as HTML each time a POST request is made.

5.3. Chapter Summary

Chapter 5 looked at the detailed design and configuration of the network layer. The advantages of Firebase was discussed as well as how a communication was established with the GUI's using Firebase instances referencing specific node addresses. Firebase rules to protect user information using JSON format was also discussed.

The rest of the chapter then looked at the integration of MS Azure with the MA and WA. The JDBC SDK used to establish a connection between the MA and MS Azure SQL database was also discussed. The use of POSTS requests to consume ML models in Azure was also touched on. Lastly the integration of the WA with Azure using Node.js and EJS concluded the chapter.

Chapter 6: Modelling

“All models are wrong, but some are useful”

(Carlson & Carlson, 2005)

An engineering model can be defined as a mechanism to simplify a complex problem through a set of rules that aim to describe the relationship of different variables within the problem (IGI Global, 2021). The previous chapters have looked at how the entire IoT system was designed and integrated. This section aims to add intelligence to the IoT system by mapping the relationship between input patient vitals and output (patient health status). Two approaches to modeling will be utilized: 1) A ML approach 2) A FL approach using the Mamdani approach.

6.1. ML using MS Azure Toolbox

The ML classic studio within MS Azure was utilized to develop and deploy the ML model used for stroke prediction in the developed IoT system. MS defines the suite as a GUI-based IDE that can effectively be used for the development and operation of ML models within the Azure space. This solution was chosen since it allowed for a variation of models to be developed and tested rapidly thus allowing the best solution to be achieved. The preceding sections delve into the details of the design process including how the model is eventually made ready to be consumed by the MA.

6.1.1. Background to ML Applications

ML is a subset of AI (Iriondo, 2018) which is concerned with the application of a variety of algorithms with the aim of achieving a predictive model based on a given data set. This data set is often large (Nichols et al., 2018) and allows systems to learn without being explicitly programmed to do so. Various types of ML models exist. These include: supervised, unsupervised, semi-supervised and reinforcement learning (Mohammed & Khan, 2017).

6.1.1.1. Structured and Unstructured Data

There are typically two types of datasets that are encountered when working with ML datasets. These include structured and unstructured data (Sarker, 2021). The explanations are given below:

Structured data – is usually data that is organized and which can be easily accessed by a computer. It can often be stored in rows and columns as part of a database schema. Examples of this type of data include: names, dates, addresses etc.

Unstructured data – A large part of data is categorized under this type. These types of data are often difficult to format, analyze and capture due to it lacking format and organization. Data under

this category can include images, sensor data, audio files, web pages etc. which often requires some form of pre-processing before it can be used.

6.1.1.2.. Evaluating ML Model Performance

The following metrics can be used to analyze the effectiveness of ML models. The theory in this section will be useful in understanding the analysis carried out later in this chapter and how the most effective model was chosen.

The first measure of performance used to evaluate ML models is the accuracy. Put simply, accuracy is the number of correctly classified cases over the total number of cases. The mathematical expression is shown in equation (6.1).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \dots\dots\dots (6.1)$$

Where:

True Positive (TP) – is the instances where the model correctly predicted the positive class.

True Negative (TN) – is the instances where the model incorrectly predicted the negative class.

False Positive (FP)/ Type 1 error – when the model incorrectly predicted the positive class.

False Negative (FN)/Type 2 error – when the model incorrectly predicted the negative class.

Accuracy is a good indicator of performance for a balanced dataset however hides the bias present in imbalanced sets – as with the model dataset used in the development of the stroke prediction model. Hence, other measures of performance are required to indicate model effectiveness. This is when precision and recall are introduced. Precision can be defined as the correctly predicted positive cases and is described by equation (6.2). E.g. a 0.94 precision means that the model is able to predict positive cases 94% of the time. So one could say that precision looks at quality of predictions rather than quantity often missing positive cases.

$$Precision = \frac{TP}{TP+FP} \dots\dots\dots (6.2)$$

Recall/sensitivity refers to the fraction of correctly predicted positive cases from the actual positive cases. Recall is described by equation (6.3). E.g. a recall of 0.94 means that the model identifies

stroke cases correctly 94% of the time. Recall looks at prediction quantity i.e. it maximizes on the number of TP predictions sometimes at the expense of incorrectly classifying a negative case as a positive one.

$$Recall = \frac{TP}{TP+FN} \dots\dots\dots (6.3)$$

In some cases there is a need for just one criteria to be satisfied, however in other cases a balance between precision and recall is required. Hence the F1 Score is introduced – which looks at the precision/recall tradeoff. A high F1 Score indicates a good balance between the precision and recall .The F1 Score is given by equation (6.4).

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \dots\dots\dots (6.4)$$

Other measures of performance can be gauged through graphical measures such as the area under the curve (AUC), the precision – recall curve (PRC) and the receiver operating characteristic (ROC) curve. An AUC of 0.87 for example indicates that 87% of the area is below the model curve. For binary classification models, the closer the AUC is to 1, indicates that the model is better at separating classes. The precision-recall curve given by Figure 6-1 indicates the balance between precision and recall. The curve needs to be as close to 1 as possible to indicate good precision and recall for the model.

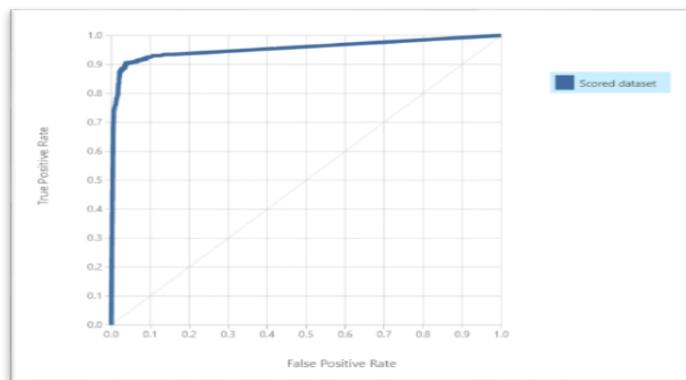


Figure 6-1: PRC Curve

The ROC given by Figure 6-2 needs to be close as possible to the left hand corner which indicates good model efficiency however the drawback of the ROC is that it does not take into consideration FN values.

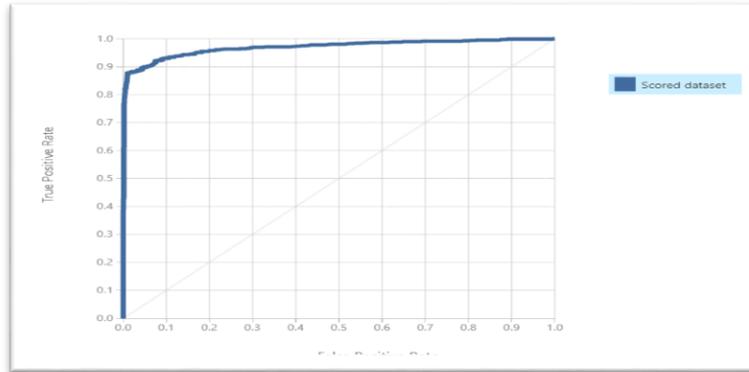


Figure 6-2: ROC Curve

Once a model has been developed it is also crucial to test for overfitting and under fitting. This is when the bias and variance tradeoff needs to be considered. A high variance results in overfitting. This is usually when the model learns too much from the given dataset and therefore falls short in prediction accuracy with a new test dataset. A model with a high bias on the other hand, under fits the data i.e. makes very simplistic assumptions about the data. A good model overcomes these two issues and seeks to find a balance.

6.1.1.3. Types of ML

Depending on the required output of the prediction model and input data provided, various types of ML models can be applied. These models can be categorized into four main types (Sarker, 2021):

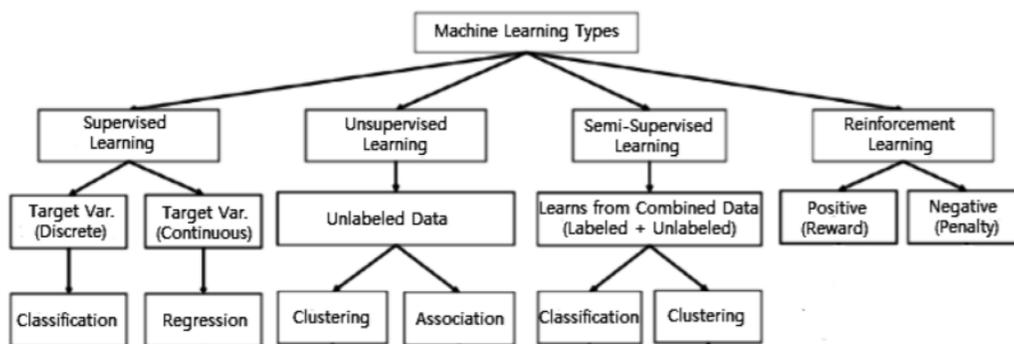


Figure 6-3: Various types of ML techniques

Supervised learning – with supervised learning models, sample input-output pairs in the form of a labeled training dataset is fed into the model. The model then develops patterns and is able to map a relationship between the input and output variables i.e. it infers a function which is

predictive in nature (He et al., 2015). Most supervised learning models either use techniques of classification (separation of data) or regression (fitting of data).

Unsupervised learning - this type of ML technique is centered on a data-driven process whereby the model learns independently with an unlabeled training set. This form of learning requires no human intervention (Han et al., 2011) and is often done for exploratory purposes for feature extraction, result groupings and making sense of trends and patterns in the data (Sarker, 2021). The most common learning tasks for models trained using this method are association rules, clustering, feature extraction etc.

Semi-supervised learning – is a mixture of the methods mentioned above. With this type of method, the model is often trained with both labelled and unlabeled datasets (He et al., 2015), (Sarker et al., 2019). The main goal of the semi-supervised model is therefore to give the model an edge over traditional supervised models to deliver a better outcome (Sarker, 2021). The most common methods employed within this category of learning is classification and clustering.

Reinforcement learning – reinforcement learning is when a software agent is able to automatically identify an optimal behavior within a given context or environment to improve its efficiency. A reward and penalty approach is taken where the agent learns to minimize risk and increase reward based on insights from environmental activists (Kaelbling et al., 1996).

6.1.1.4.. Binary Classification

Binary classification is a type of supervised learning that is trained to provide an outcome in one of two categories after it measures a series of attributes. An example of this would be in the field of medical diagnosis of a single condition where the outcome could be a1: 'diagnosis of the disease' or a0: 'no diagnosis of the disease' (Parmigiani, 2001). The most popular types of binary classification algorithms include Logistic Regression, k-Nearest Neighbors, Decision Trees, Support Vector Machines and Naïve Bayes (Rao et al., 2021).

6.1.2. Dataset for Developed Model

The dataset used for the stroke prediction model was extracted from the *Kaggle* data repository and represents a sample of 5110 people consisting of 2995 females and 2115 males. The dataset uses the variables shown in table 6-1 in order to predict the likelihood of stroke. The dataset consists of both attribute and numerical data.



Figure 6-4: Kaggle Dataset Statistics

Table 6-1: Dataset Description for Stroke Prediction Model

#	Variable (unit of measure)	Description	Type
1	id	Patient ID number	Categorical
2	gender	Patient sex ("Male", "Female" or "Other")	Categorical
3	age	Age of the patient	Numeric
4	hypertension	Patient hypertension status ("0" - No, "1" - Yes)	Categorical
5	ever_married	Marital status ("Yes" or "No")	Categorical
6	work_type	Patient occupation ("children", "Govt_job", "never_worked", "Private", "self-employed)	Categorical
7	Residence_type	"Rural" or "Urban"	Categorical
8	avg_glucose_level	Patient's blood sugar level	Numeric
9	bmi	Patient's body mass index	Numeric
10	smoking_status	"formerly smoked", "never smoked", "smokes" or "unknown"	Categorical
11	Stroke	Previous history of stroke ("0" - no stroke, "1" - previous case of stroke)	Categorical

6.1.3. Design Approach

The ML model was developed using the ML classic studio package within MS Azure. Due to the nature of the data, a binary classification model was trained using the labelled dataset to deliver an output of class/category (stroke or no stroke) along with a probability score. The process started by creating a new project and uploading the raw dataset. Data exploration was then done on the dataset to determine the further processing that would be required. The dataset was then pre-processed (meta-data editing and cleaning). Thereafter the data was split for testing and training. A 70% training and 30% testing split was used. The training set is then used to train, score and evaluate various different models. Here evaluation checks how effectively a model scores (predicts) from a given dataset after it has been trained. A *cross_validation* module was used to further evaluate the suitability of the chosen model based on the *k-folds* method. Once the model was satisfactory, it was then deployed to a webserver ready to be consumed by the MA through a POST request. Figure 6-5 gives a high level overview of the process:

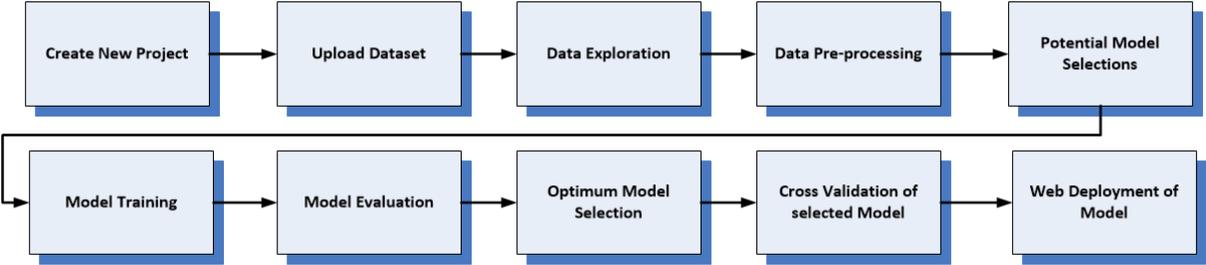


Figure 6-5: High-Level Overview of Model Development within MS Azure ML Classic Studio

MS azure ML studio, uses a work flow approach to create a process pipeline using modules to clean and model the input data. Each model can then be configured to convert and model the dataset as per requirements. The flow diagram in Figure 6-6 illustrates a schematic for the stroke prediction model developed. The following modules were utilized:

- Data exploration (*Summarize Data*).
- Pre-processing (*Edit Metadata, Clean Missing Data, Select Columns in Dataset, Synthetic Mining Oversampling Technique (SMOTE)*).
- Training (*Split Data, Train Model, Two-Class Decision Forest*).
- Evaluation (*Score Model, Evaluate Model, Cross Validate Model*).
- Deployment (*Web Deployment*).

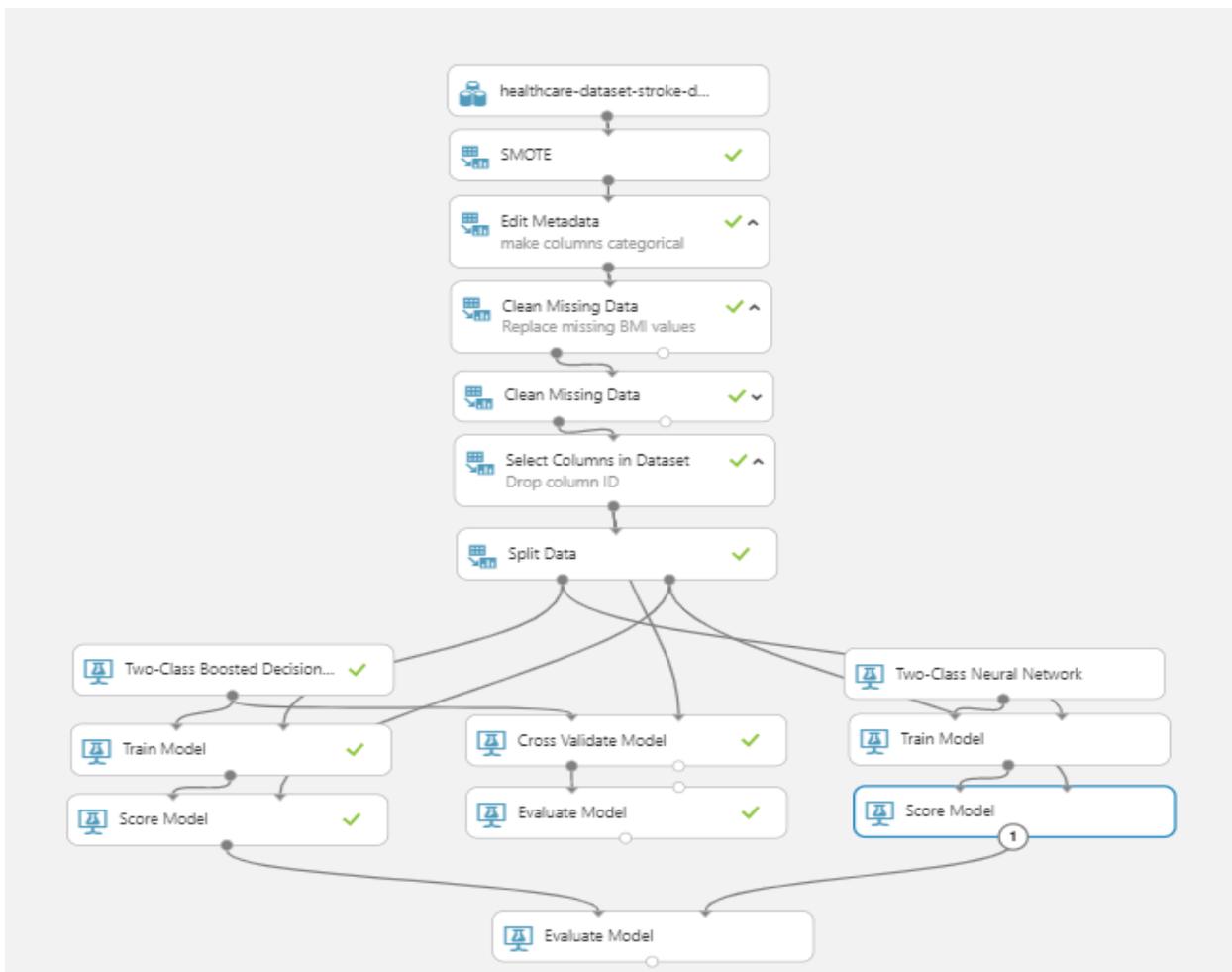


Figure 6-6: Work Flow of Stroke Prediction Model

6.1.4. Exploratory Research

Before processing the data, visualizations were used to identify trends, patterns and anomalies in the data set. Below illustrates the findings:

Using the visualize feature with MS azure ML studio it is possible to view statistical information of the various characteristics within the dataset. It was seen that the original dataset consisted of 5110 rows and 44 columns. The BMI column has the most missing values of 201 (Figure 6-8). Figure 6-7 shows that the dataset is also severely unbalanced with the TP stroke class being under-represented, with positive stroke cases accounting for only 4.9% of the dataset and non-positive stroke cases accounting for 95% of the dataset.

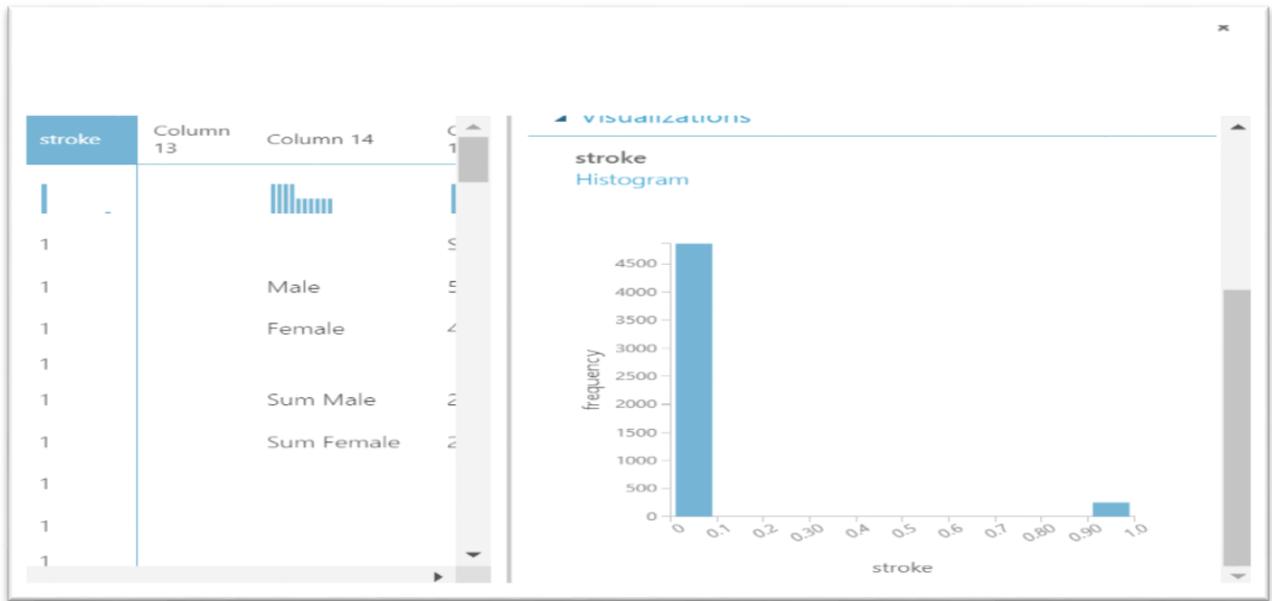


Figure 6-7: Histogram Showing Unbalanced Dataset

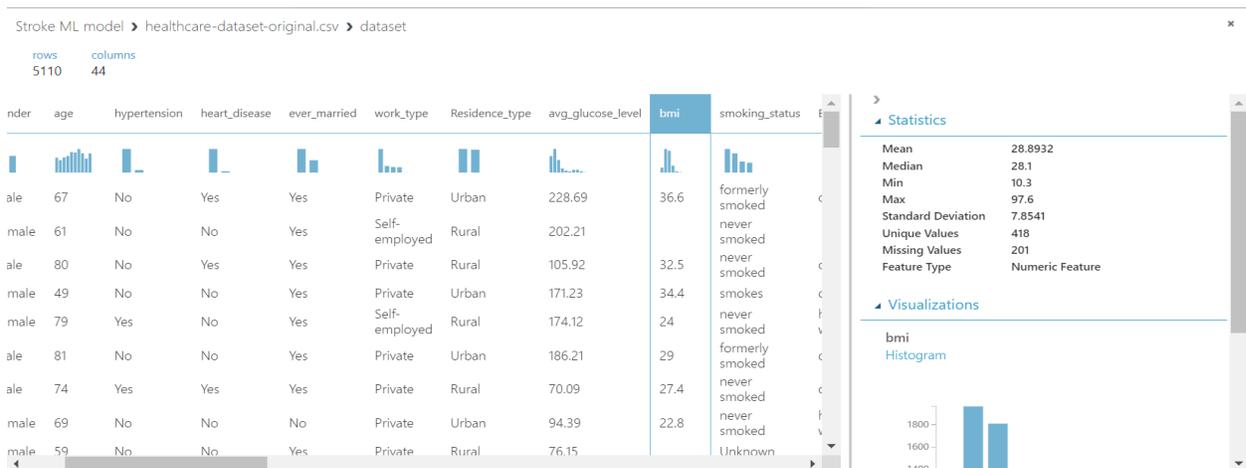


Figure 6-8: Extract from MS Azure Showing missing BMI Values

The attribute data was then assessed using the histograms (Figure 6-9 – 6-18):

Gender – of the men and woman who participated in the study, a higher percentage of men compared to woman experienced stroke.

BMI status – Feature addition was done on the dataset to determine the BMI status of the patient. The ranges below, taken from the CDC, were used to categorize patients based on where their readings fell in the below ranges.

- Below 18.5 (underweight)
- 18.4-24.9 (healthy weight)
- 25.0-29.9 (Overweight)
- >30 (obese)

The data shows that out of each of the weight categories, the cases of stroke was most prevalent in the overweight and obese categories.

Hypertension status – a greater percentage of people with hypertension experienced stroke compared to the group without hypertension.

Residence type – of the people living in urban areas, a greater percentage of these people experienced stroke compared to the rural group percentage.

Heart disease status – the percentage of people with heart disease that experienced stroke was higher than the group with no heart disease.

Marital status – the percentage of people married had higher cases of stroke compared to the group of unmarried individuals.

Work type – more self-employed people suffer from stroke compared to any other working groups.

Smoking status – most cases of stroke occurred in the group of individuals that formerly smoked.

Glucose status – feature addition was done once again with the glucose levels. Based on the prescribed ranges below, patients were categorized according to their likelihood of developing health conditions:

- Below 70 (hypoglycemia)
- >99 (diabetes)
- Any other value (normal)

The trend shows no apparent correlation between the glucose levels in a patient's blood and their likelihood of getting stroke.

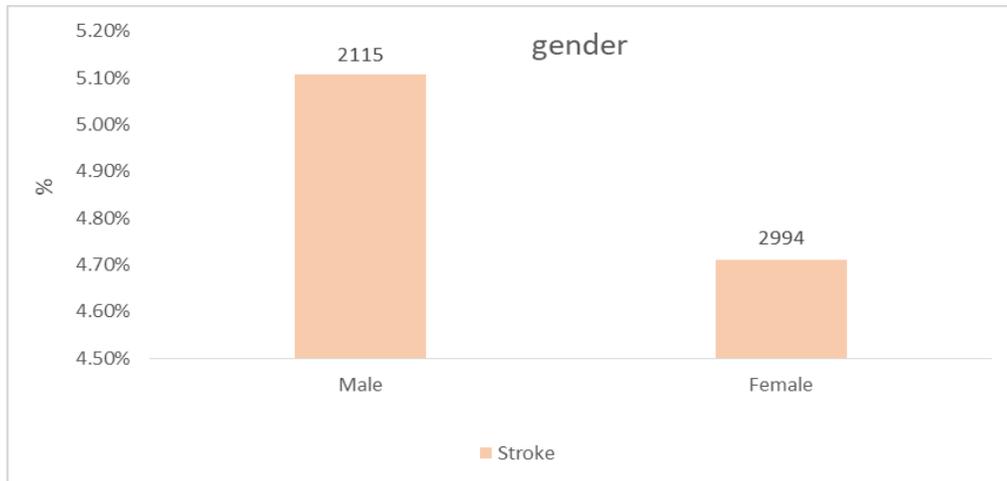


Figure 6-9: histogram of Gender Variable vs Stroke Probability

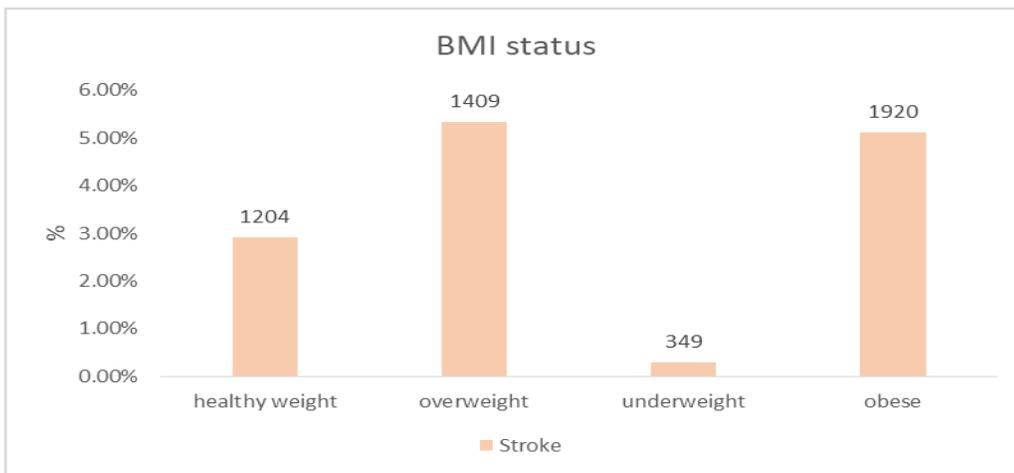


Figure 6-10: Histogram of BMI Status Variable vs Stroke Probability

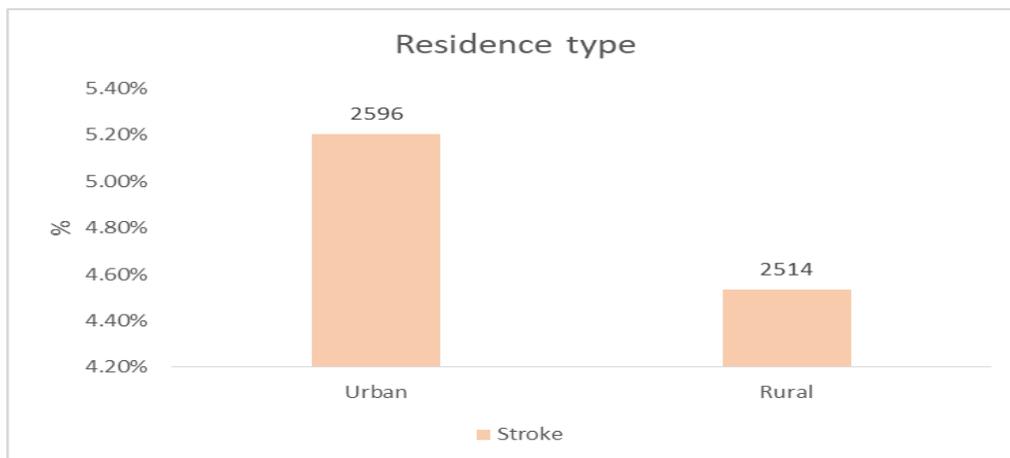


Figure 6-11: Histogram of Residence Type Variable vs Stroke Probability

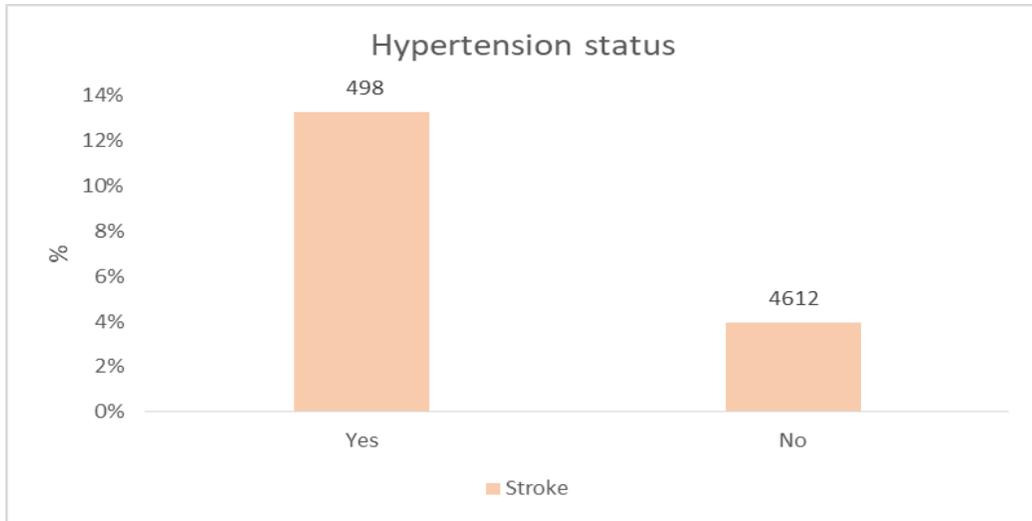


Figure 6-12: Histogram of Hypertension Status vs Stroke Probability

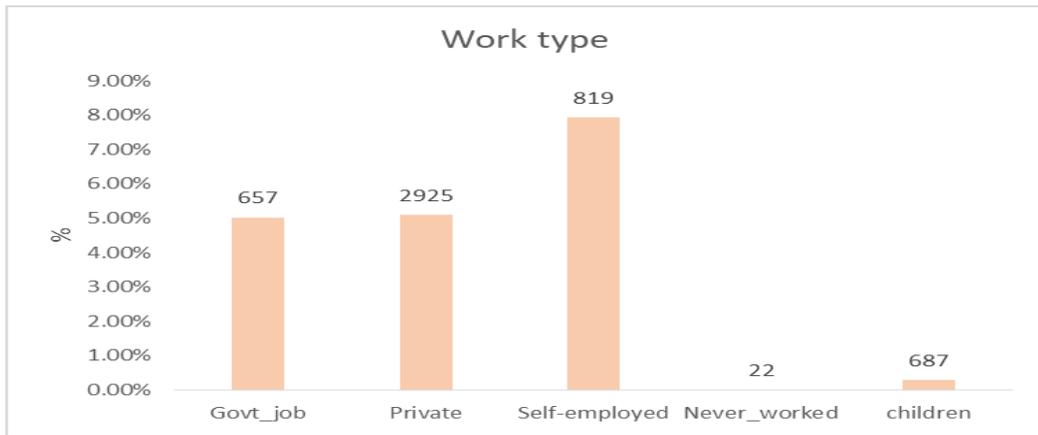


Figure 6-13: Histogram of Work Type Status vs Stroke Probability

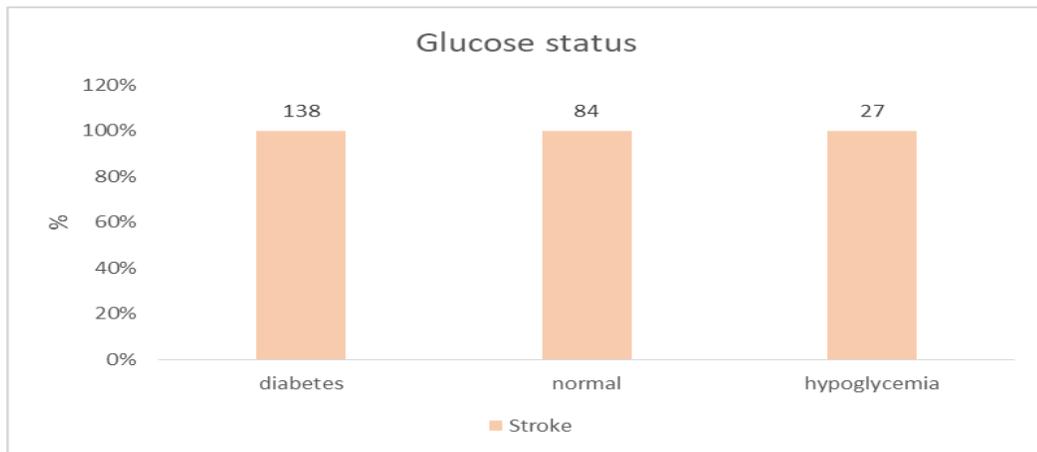


Figure 6-14: Histogram of Glucose Status Variable vs Stroke Probability

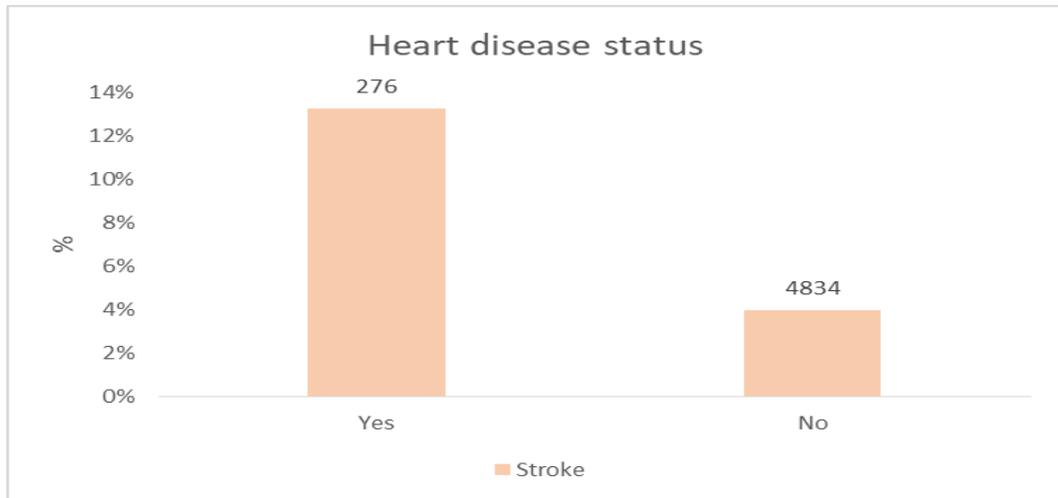


Figure 6-15: Histogram of Heart Disease Status vs Stroke Probability

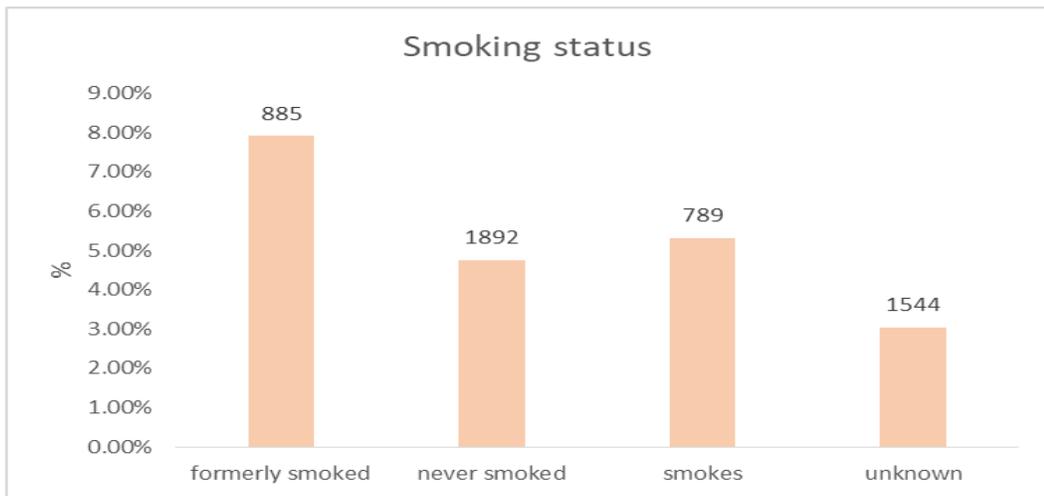


Figure 6-16: Histogram of Smoking Status Variable vs Stroke Probability

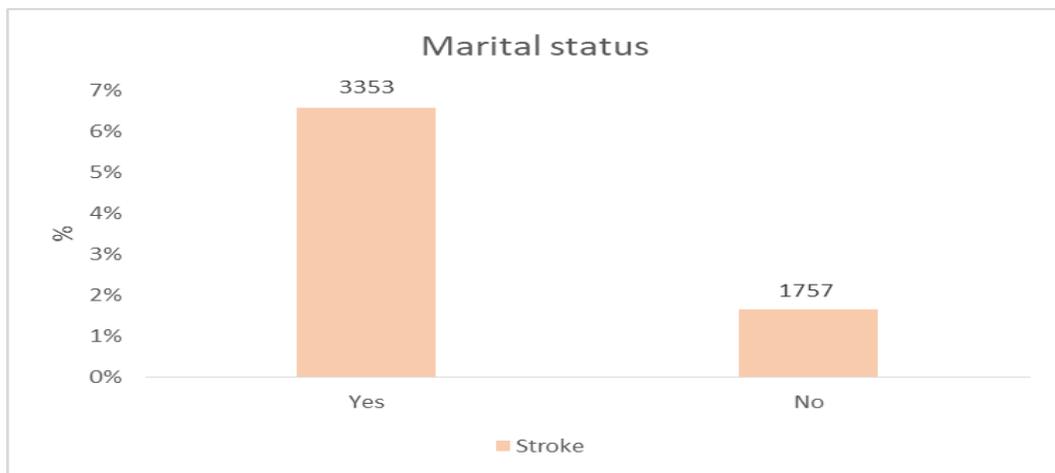


Figure 6-17: Histogram of Marital Status Variable vs Stroke Probability

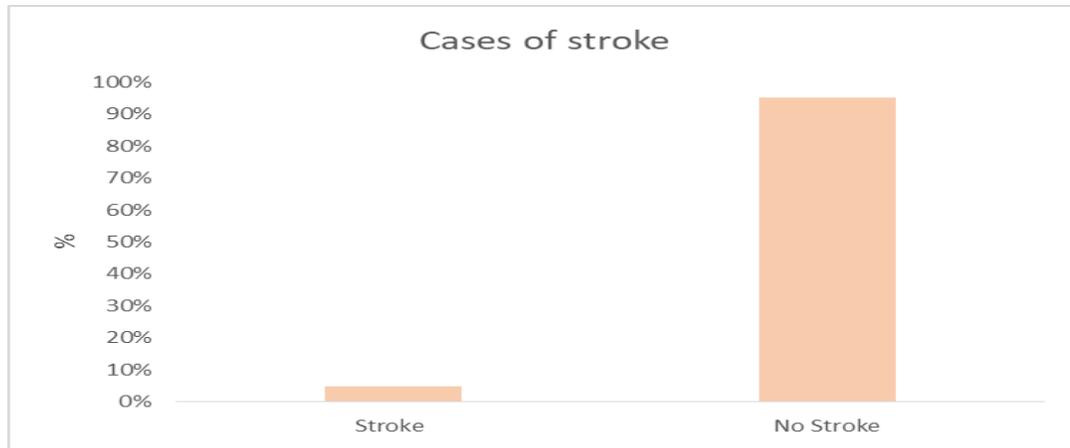


Figure 6-18: Histogram of Stroke Cases

The numerical data was evaluated using the box and whisker plots shown in Figure 6-19. As can be seen, patients who experienced stroke had a higher average age compared to the average age of the group that didn't suffer from stroke.

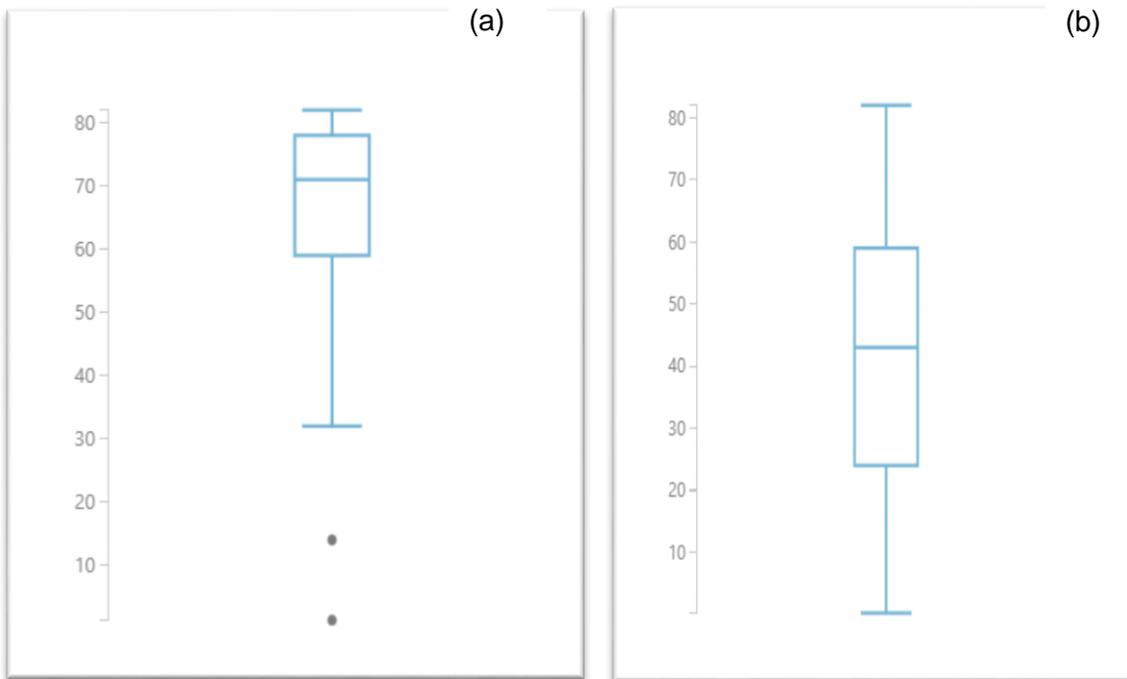


Figure 6-19: (a) Age of Stroke Patients Box Plot, (b) Age of Non- Stroke Patients Box Plot

The BMI feature has a large amount of outliers for those that didn't experience stroke. To rectify this, the values greater than 50 were converted to 50 as super obesity is capped at this value. The rectified box plot is shown in Figure 6-20(a). Subjects that suffer with stroke tend to have a higher mean glucose level compared to those that don't. Figure 6-20 (b) and Figure 6-20 (c) illustrate this point.

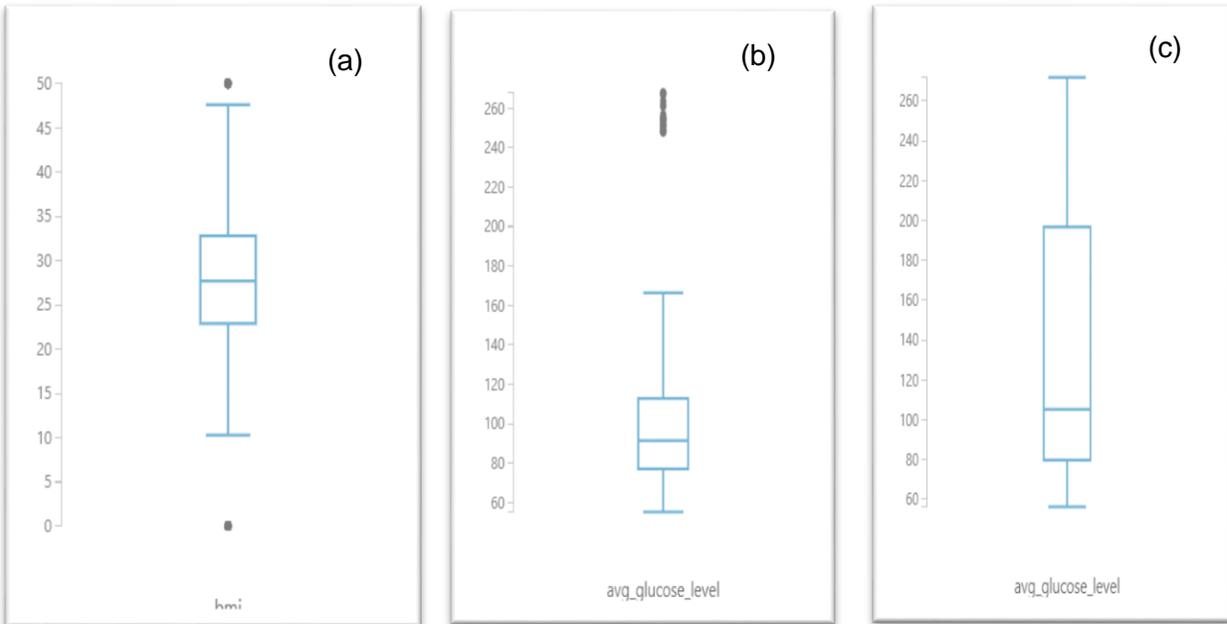


Figure 6-20: (a) Rectified BMI Box Plot, (b) Box Plot Glucose levels of Non-Stroke Patients, (c) Box plot of Glucose Levels of Patients with Stroke

Figure 6-18, shows the total percentage of individuals within the dataset diagnosed with and without stroke. It can be seen that the dataset is severely unbalanced, this means that as is the dataset will result in high precision and low recall models. However, for the problem at hand, the precision and recall are equally important. A high precision and low recall will result in more FP outcomes resulting in more patients visiting the physician when there isn't a problem.

A low precision and high recall will increase the FNs and could fail to predict those that are potentially at risk of stroke. Hence, as mentioned earlier in this chapter, a tradeoff between precision and recall is necessary in order to get a robust model.

6.1.5. Data Pre-processing

As can be seen, the dataset is severely unbalanced and required a re-sampling method. An oversampling method was chosen since the dataset was not large enough for under-sampling of the over represented class (the cases of no stroke). The SMOTE was used to create synthetic samples of the minority class i.e. positive cases of stroke. The sample size of the under-

represented class was successfully increased from 249 elements to 2490 elements. The pie charts, Figure 6-21 (a) and (b) illustrates the sample size split before and after over-sampling:

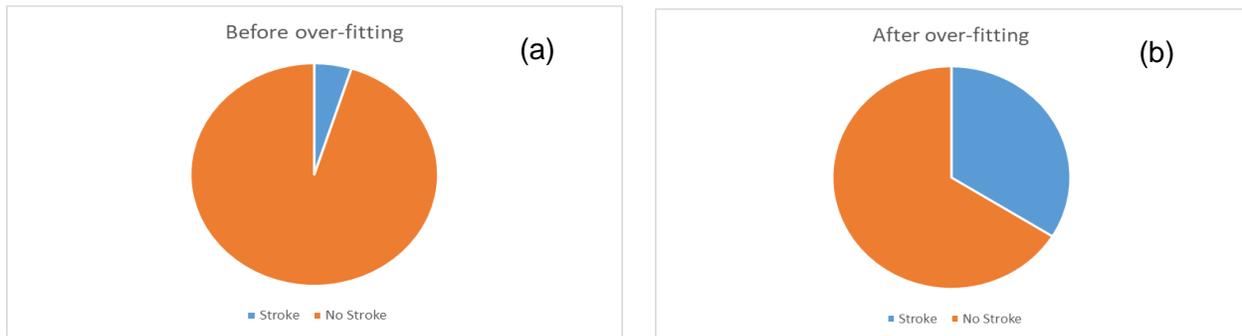


Figure 6-21: (a) Sample Size Split before SMOTE, (b) Sample Size Split after SMOTE

The second part of the data-processing stage included the classification of columns into numerical and categorical data as indicated in Table 6-1. Data imputation of the missing values was then done using the median value of the dataset. This was done for the BMI column which had the most missing values and consequently had a significant effect on the output. In addition, with feature addition of the BMI status, all missing status values were also filled in to complete missing values. Feature selection was then done to remove unnecessary columns. In the case of this dataset – only the Id column was removed since it had no effect on the output stroke prediction value. As mentioned previously, to remove outliers in the BMI column, values greater than 50 were capped at 50 which represents the super obesity mark.

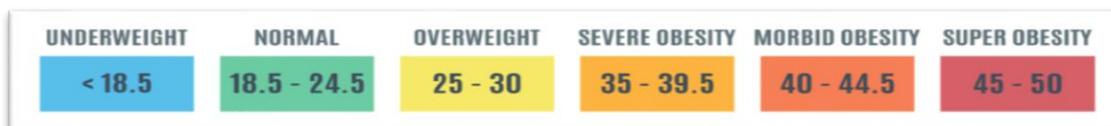


Figure 6-22: Obesity Ranges (Wright, 2021)

6.1.6. Model Evaluation

To evaluate and score the models, the data was split into a training and testing set (70/30) split. 70 % of the data was used to train the various models while 30 % was later used for the scoring of models. The details below indicate the various models with their respective scores.

The 6 different Binary classification models shown below were explored:

- Two-class Bayes point machine
- Two-class boosted decision tree
- Two-class decision forest
- Two-class logistic regression
- Two-class neural network
- Two-class support vector machine

6.1.6.1. Two-Class Bayes Point Machine

The first model evaluated was the Two – class Bayes point machine algorithm. The model utilizes a linear classification approach called the *Bayes point machine*. Due to its roots in Bayesian theory, the model is not prone to overfitting. Thirty iterations were chosen and the statistical summary can be seen in Figure 6-24. The model has a good accuracy and precision of 86 % and 97 % respectively, however falls short in terms of the recall which sits at 61 %. The model will diagnose 86 % of positive stroke cases correctly, however will only be able to successfully identify 61 % of strokes cases (high degree of FNs).

The AUC is 0.97 which indicates that the model is good at separating classes. The F1 score of 75 % indicates an imbalance between precision and recall although it is not very severe. At a threshold of 0.5, this imbalance can further be seen in the PRC. The ROC curve is relatively close to the upper left corner which indicates good efficiency however a drawback of ROC is that it does not take into consideration FNs and hence hides the faults in this model.

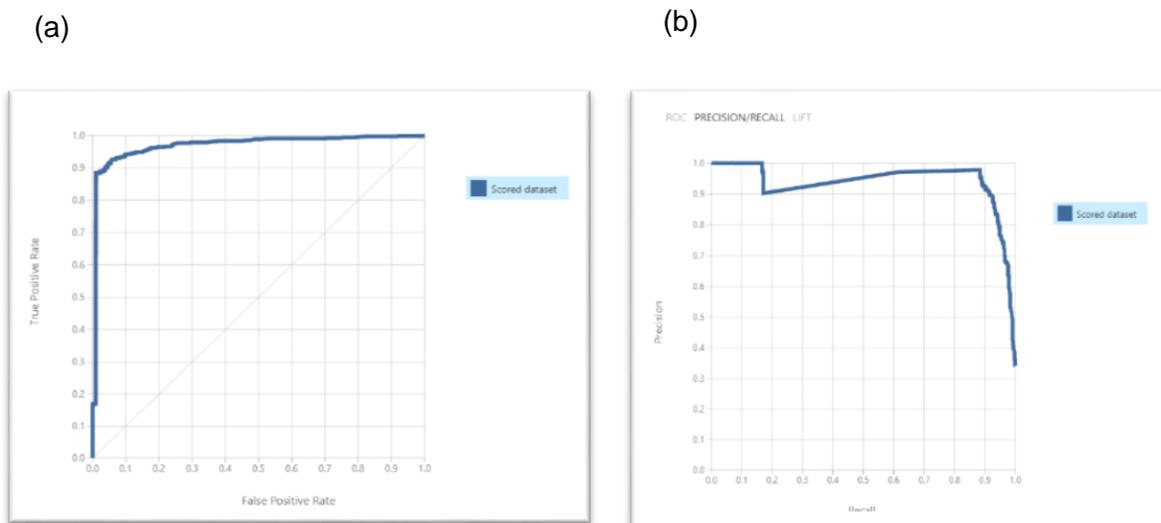


Figure 6-23: (a) ROC Curve Bayes Point Machine, (b) Precision/Recall Curve Bayes Point Machine



Figure 6-24: Bayes Point Machine Statistics

6.1.6.2. Two-Class Boosted Decision Tree

The next model evaluated was the two-class boosted decision tree. This is an ensemble model that makes use of collection of weak learners to improve the overall performance of the model collectively. A sequence approach is taken where each weaker model works to improve the error of the previous model. With gradient boosting the loss function optimization is achieved through the gradient descent approach. This model seems to have an overall excellent rating with regards to all parameters. The F1 Score of 91 % indicates a good balance between precision and recall which is evident by both these Figures being above 90 %. The AUC is 97 % which indicates an excellent ability to separate the classes.

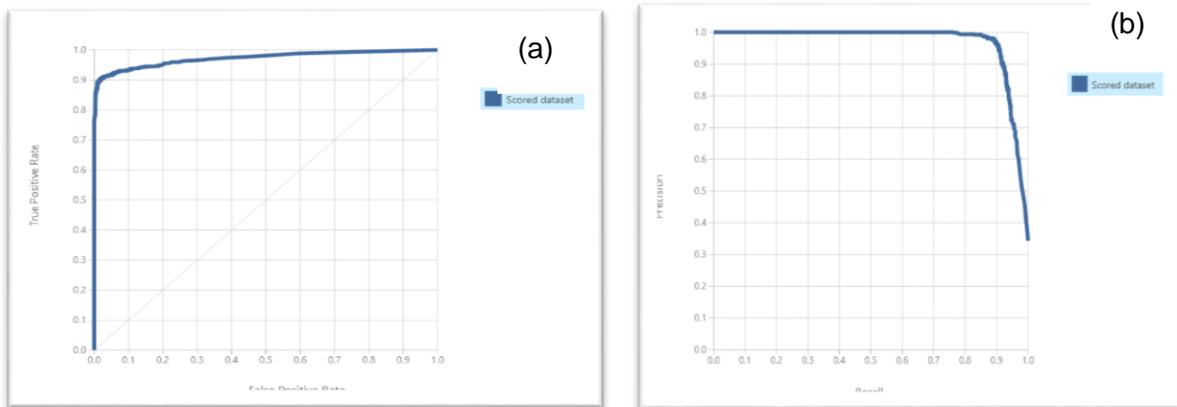


Figure 6-25: (a) Boosted Decision Tree, (b) Precision/Recall Curve Boosted Decision Tree

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
692	65	0.942	0.918	0.5	0.972
False Positive	True Negative	Recall	F1 Score		
62	1386	0.914	0.916		
Positive Label	Negative Label				
1	0				

Figure 6-26: Statistics Boosted Decision Tree

6.1.6.3. Two-Class Decision Forest

Two class decision forest is another ensemble technique that relies on the multiple decision trees to create a generalized model which uses a process of voting on the most popular output class. Trees with higher prediction confidence will have a higher weight. Results show that the model is not very effective for the intended dataset. Even though the accuracy is good at 82 %, the recall is very low at 67 % indicating once again a high degree of FNs which in the area of medical diagnosis cannot be compromised.

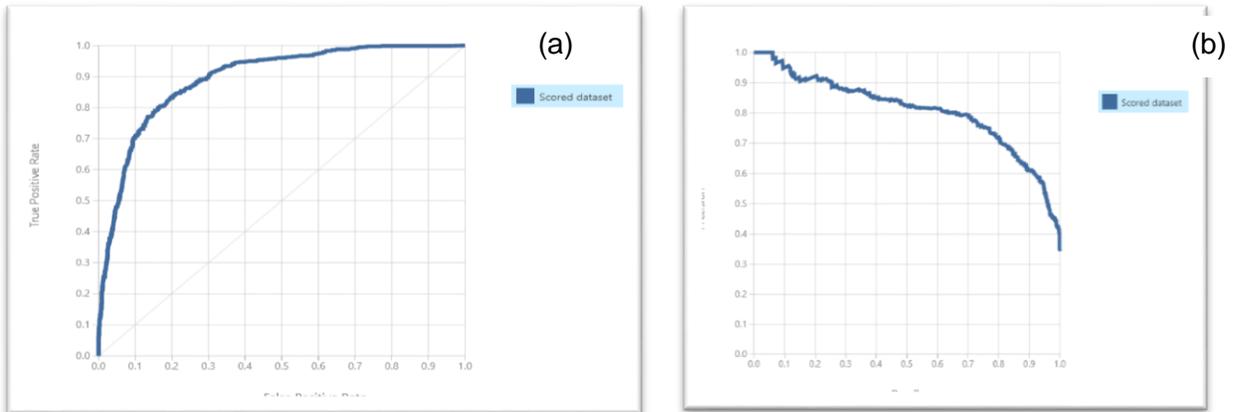


Figure 6-27: (a) ROC Curve Two-Class Decision Forest, (b) Precision/Recall Curve Two-Class Decision Forest



Figure 6-28: Statistics of the Two-Class Decision Forest

6.1.6.4. Two-Class Logistic Regression

The logistic regression model uses a supervised learning algorithm which means that the model requires a training set with the outcome. The probability of occurrence is predicted by fitting the data to a logistic function. The model scores well in most parameters and is possibly a good model for the intended application. Recall may have to be improved if used either through fine tuning of parameters or further data processing.

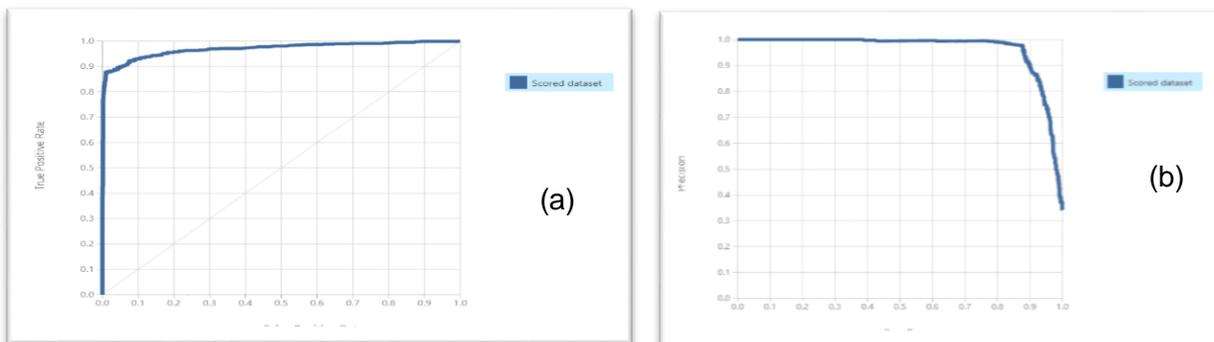


Figure 6-29: (a) ROC Curve Logistic Regression, (b) Precision/Recall Curve Logistic Regression

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
668	89	0.944	0.950	0.5	0.970
False Positive	True Negative	Recall	F1 Score		
35	1413	0.882	0.915		
Positive Label	Negative Label				
1	0				

Figure 6-30: Statistics Logistic Regression

6.1.6.5. Two-Class Neural Network

The two-class neural network like the two-class logistic regression model is a supervisor based algorithm that requires a dataset with an output column. The neural network approach makes use of a set of interconnected layers. The first layer consist of the input which is connected to the second output layer. Between the input and output layer is a series of hidden layers which allows for effective training of the neural network. One drawback of neural networks is its black box approach to prediction with little understanding of how the final outcome is achieved.

The two class neural network can possibly be used for the application with all parameters of interest scoring in the 90's.

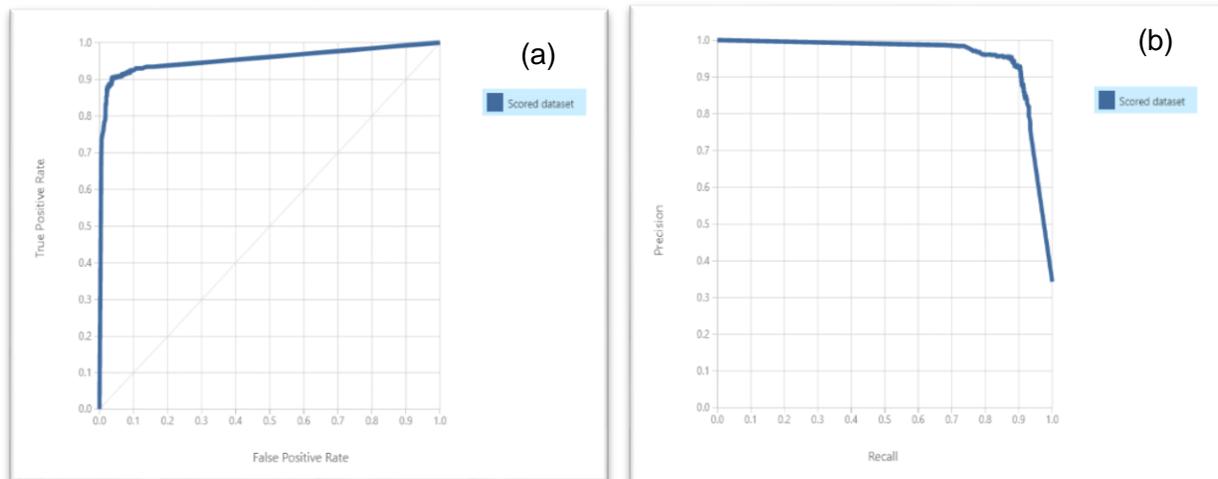


Figure 6-31: (a) ROC Curve Two Class Neural Network, (b) Precision/Recall Curve Two Class Neural Network

True Positive	False Negative	Accuracy	Precision	Threshold	<input type="range" value="0.5"/>	AUC
684	73	0.941	0.924	0.5		0.959
False Positive	True Negative	Recall	F1 Score			
56	1392	0.904	0.914			
Positive Label	Negative Label					
1	0					

Figure 6-32: Statistics Two Class Neural Network

6.1.6.6. Two-Class Support Vector Machine

The SVM model is a supervised learning algorithm which requires a labeled dataset. Quite simply the model works by creating a line or hyperplane which is able to effectively separate the data into classes. The model is a good model, but compared to other models identified, it falls short with regards to accuracy, precision and recall. At most the model is average and would not improve significantly with hyper tuning.

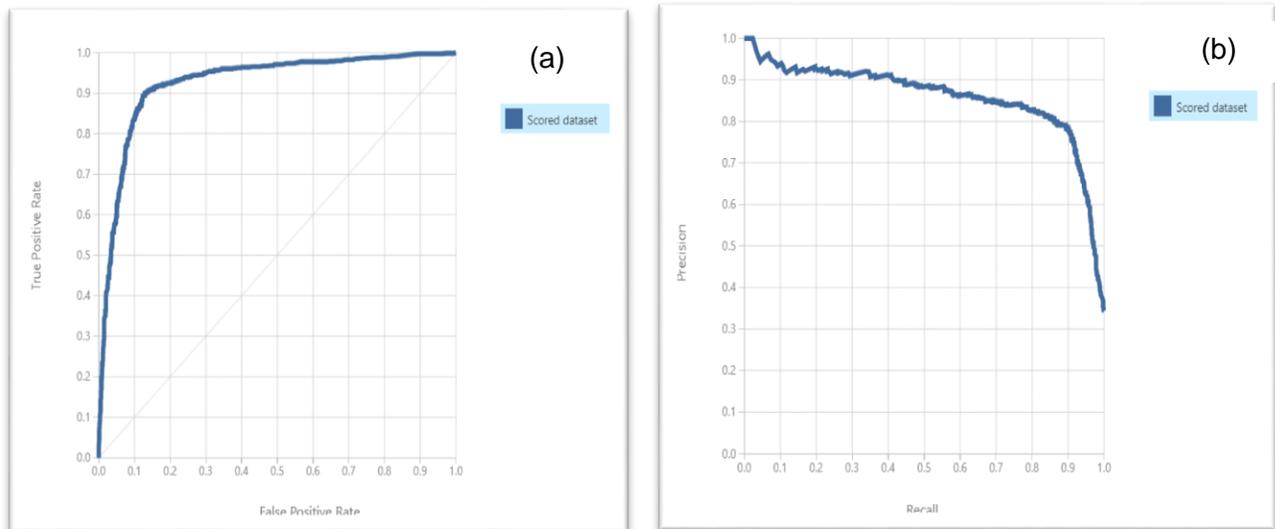


Figure 6-33: (a) ROC Curve SVM, (b) Precision/Recall Curve SVM

True Positive	False Negative	Accuracy	Precision	Threshold	<input type="range" value="0.5"/>	AUC
608	149	0.875	0.827	0.5		0.926
False Positive	True Negative	Recall	F1 Score			
127	1321	0.803	0.815			
Positive Label	Negative Label					
1	0					

Figure 6-34: Statistics for the SVM Model

6.1.7. Model Selection

Based on the evaluation of the various models, the most effective model was the two-class boosted decision tree. In the preceding sections, an explanation of how the model was further tuned and assessed for overfitting, and under fitting will be elaborated. Figure 6-35 shows the different statistics of each model in comparison to each other.

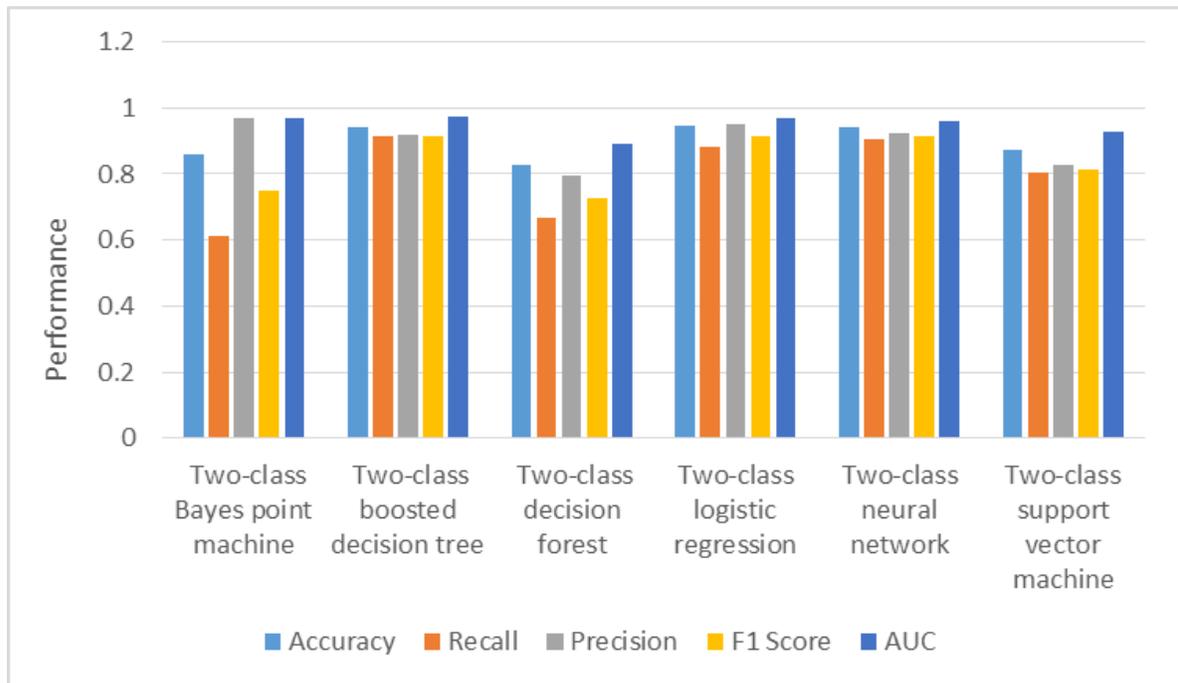


Figure 6-35: Graph Showing Different Performance Statistics of 6 Models

6.1.8. Cross validation

The *cross validation* module within MS Azure ML studio was used to perform a number of train-score-evaluate operations i.e. 10 folds automatically through different subsets of the input data. The first fold/subset is used for testing and the other 9 is used for training. The process is carried out 10 times before taking an average. As opposed to the previous evaluation method, the entire dataset is fed into the cross evaluation module together with the selected untrained model. Cross validation will ensure that there is no overfitting (high variance and low bias) of the data as it exposes the model to different sets of training data. The advantages of cross validation includes:

- Cross validation will use more of the test data during the different simulations which will ensure overfitting is reduced.
- It can be used to gage the quality of the dataset as well as how the model responds to different partitions of the dataset.

Looking at the output of the cross validation module, one can see that classification metric for each fold as well as the standard deviation and mean of all the folds. As seen in Figure 6-36, the accuracy statistic for each fold is constantly high for each fold with an overall average precision, accuracy and recall above 91 %. This implies limited variation and a good quality dataset.

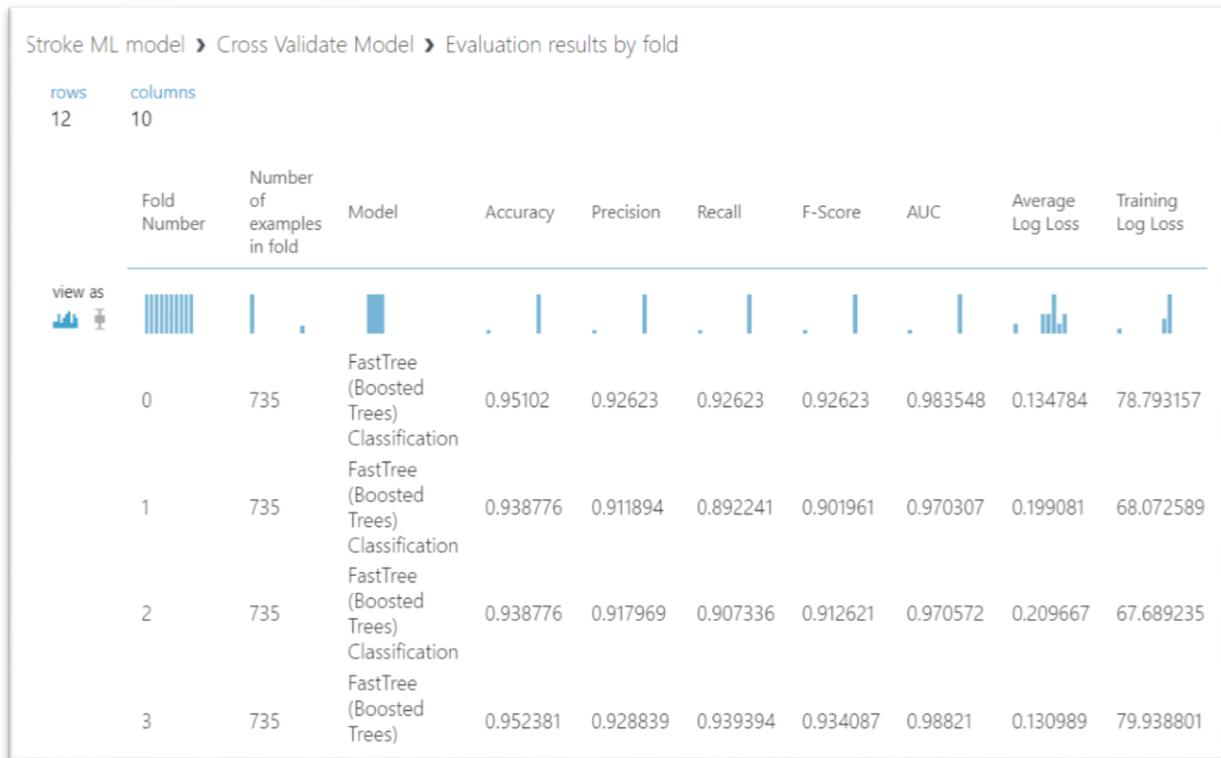


Figure 6-36: K-Fold Method Statistics

6.1.9. Test for Overfitting and under fitting

The first test of overfitting was achieved through the cross validation process which reduces variation by using more of the data for training the model. Another way to check the susceptibility of the model variance in the dataset is to score the model with the training set and the testing set and compare performance. As can be seen below, the model passes the test with the test score being slightly lowered compared to the training score yet still quite high with limited variation in scoring. This indicates that the model is not experiencing overfitting.

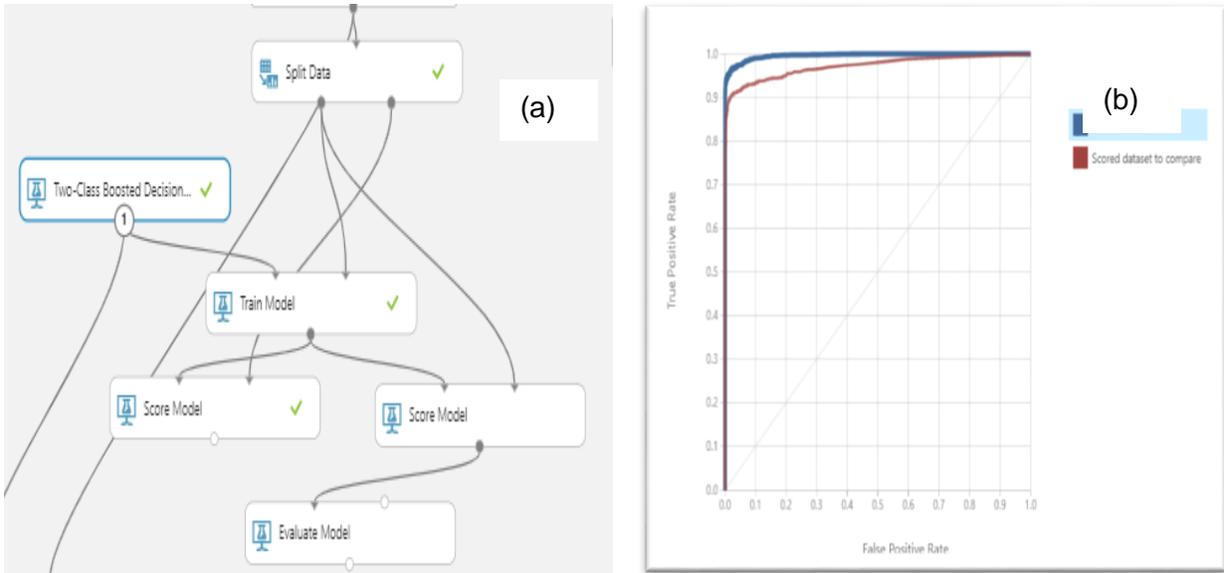


Figure 6-37: (a) Workflow of Test Overfitting and Under Fitting, (b) ROC Curve Comparing Training and Test Data with Model

True Positive 1651	False Negative 82	Accuracy 0.977	Precision 0.979	Threshold <input type="range" value="0.5"/>	AUC 0.996
False Positive 36	True Negative 3377	Recall 0.953	F1 Score 0.965		
Positive Label 1	Negative Label 0				

Figure 6-38: Training Set Model Statistics

True Positive 692	False Negative 65	Accuracy 0.942	Precision 0.918	Threshold <input type="range" value="0.5"/>	AUC 0.972
False Positive 62	True Negative 1386	Recall 0.914	F1 Score 0.916		
Positive Label 1	Negative Label 0				

Figure 6-39: Test Set Model Statistics

A test for under fitting can also be determined from fig 6.38 since under fitting would mean that the model would perform poorly on the training set because it is unable to capture the relationship between the input and output variables correctly thus indicating a high bias. This however is not seen when the model was trained with the training set, as all parameters have achieved scores above 95%.

6.1.10. Web deployment and Model Consumption

The deployment of the ML model is the final part of the ML model process and involves converting a training model to a predictive model. The first part of the process involved removing all redundant models and retraining the two-class gradient boosting model. The *set-up web service* module within Azure is then utilized which creates an *input and output* web service. A *select_columns_in_dataset* query can then be used to ensure that the model only outputs the score and probability. Once the model is deployed it will then generate a unique URL and API key which can be used to create POST requests.

6.2. FL Model

This section will focus more on explaining how the FES systems utilized in the IoT system were developed. The theory surrounding FL systems will be explained followed by the application of 2 FES models used for the prediction of patient health status.

6.2.1. Fuzzy Logic Explained

Fuzzy logic is a predictive methodology that allows for the description of systems that have uncertainty and imprecision. FL essentially represents the inputs of a system as linguistic variables to produce a certain output. It basically works through assigning partial truth values between true and false which becomes useful when certain systems cannot be described through traditional logic of 0 and 1 (Torres & Juan, 2005). In areas like medical diagnosis, it is more useful to consider intermediate logical values to make sense of the situation. Therefore in the medical field, FL plays a very important role (Torres & Juan, 2005).

All FL systems consists of a knowledge base which is essentially made up of an IF-THEN rule base. These rules together with what is referred to as MFs establish reasoning with the data. MFs are curves which enable crisp input functions to be mapped out to their corresponding membership values of between 0 and 1 also known as their respective fuzzy sets (Section, 2020). The most common types of MF's that exist are Gaussian, triangular and trapezoidal in nature (Erdal Kayacan et al., 2016). Fuzzy logic is however dependent on rules being created, so often expert knowledge is needed to create rules that make practical sense.

6.2.2. Fuzzy Logic Architecture

Figure 6-40 illustrates the basic architecture surrounding a Fuzzy Logic system (Manshahia & Singh, 2018). At the start of the process, an input crisp variable is converted into fuzzy values through the information stored in the knowledge base. The fuzzy input together with the rule base established will produce a fuzzy output. The last step of the process is defuzzification which entails converting the output fuzzy values back into crisp values.

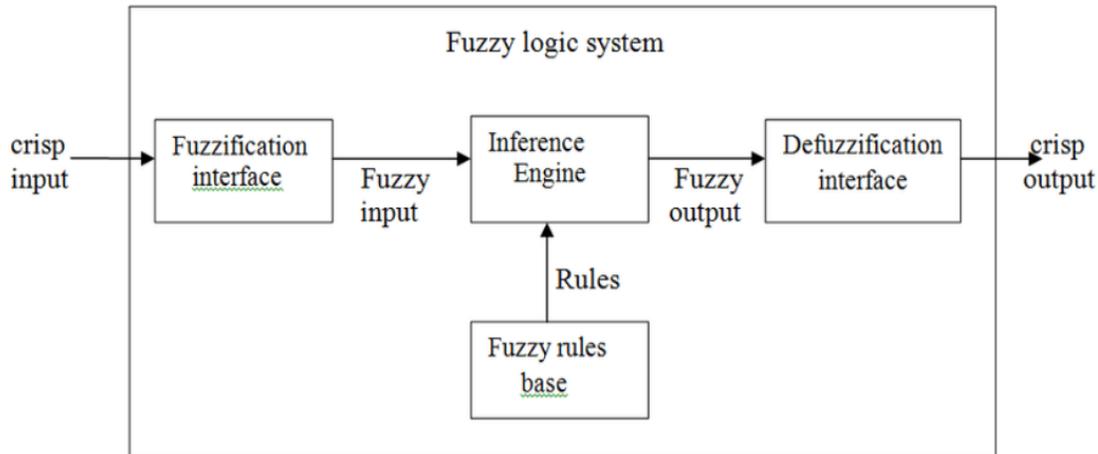


Figure 6-40: Fuzzy Logic Block Diagram

6.2.3. Types of fuzzy inference systems

There are two types of inference systems that are typically used in the fuzzy logic space. These include the Mamdani-type and the Sugeno-type. The Mamdani-type inference system expects that the outputs of the membership functions be fuzzy sets. The difference with the Sugeno-Type inference system is that the output membership functions are linear or constant (Kalogirou, 2014).

One of the benefits of the Mamdani type system is the intuitive and simplistic rule bases. They tend to be useful in applications requiring tacit and specialist type knowledge. One such example is in the field of medical diagnostics where health practitioners can intuitively identify optimal rules to improve the accuracy of the fuzzy model (Mathworks, 2021).

6.2.4. Approach

The FES models were developed using the Mamdani approach. The development steps followed the typical FL model set-up as shown in Figure 6-41. The process starts off by converting the input variables (crisp format) to fuzzy sets using MFs. Here the triangular membership functions were used to describe the linguistic variable ranges for each measured input quantity. A rule

based was then developed with a combination of linguistic variables and their respective outcomes. The outcome membership function was then developed. The defuzzification method used was the MoM method. The model was programmed on the Android IDE.

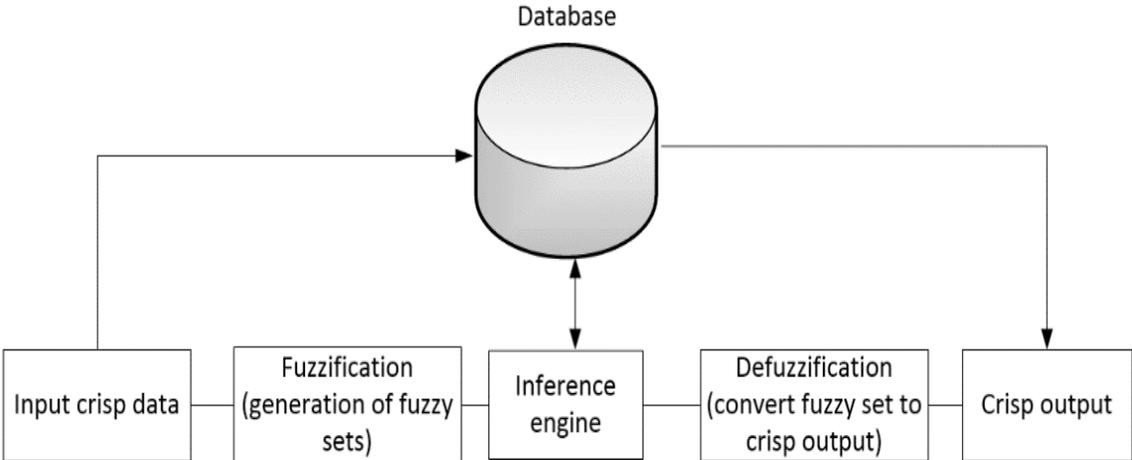


Figure 6-41: Typical FL Model Development Process

The operational flow diagram showing the typical process for both FES models developed can be seen in Figure 6-42.

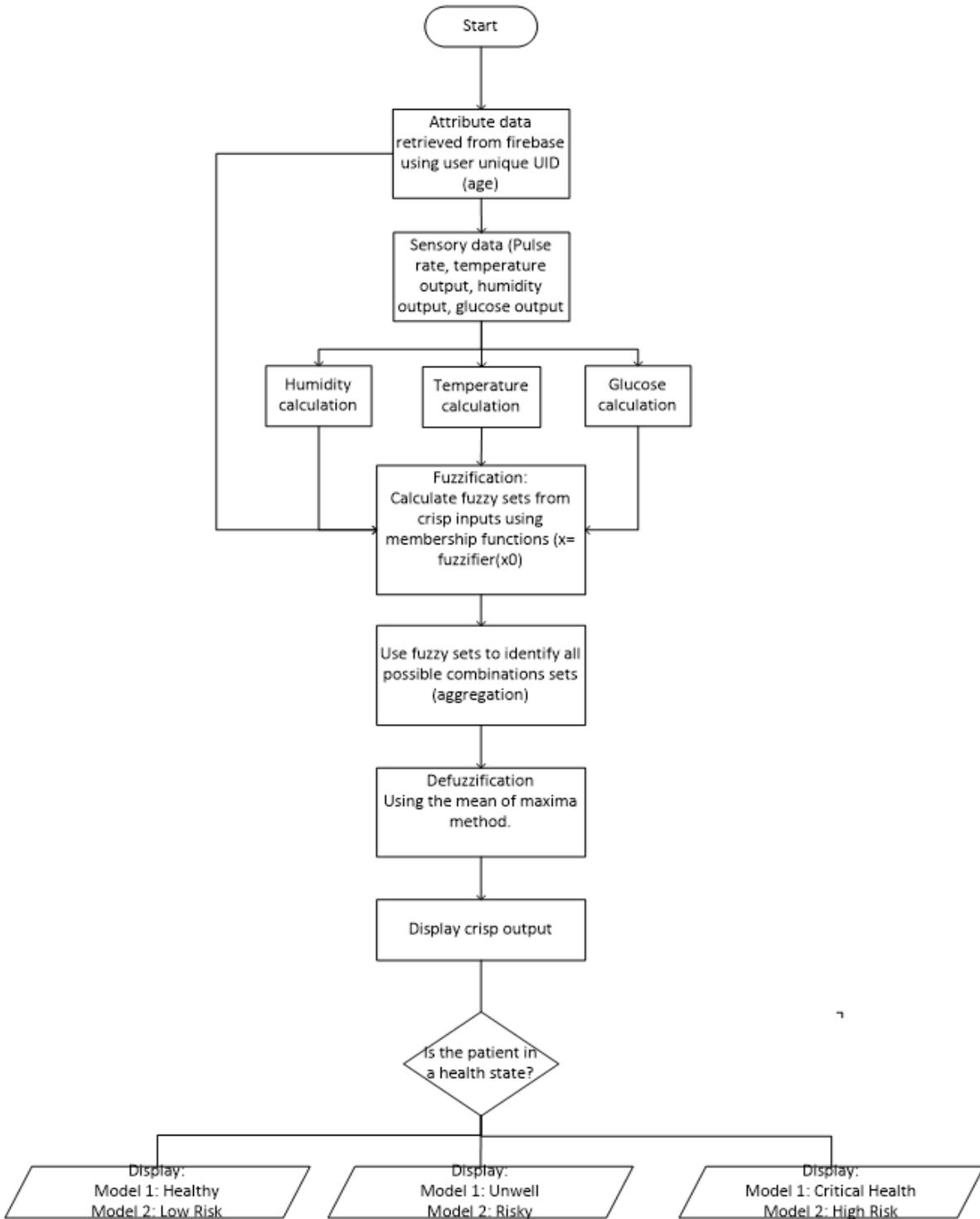


Figure 6-42: Operational Flow FES

The process starts by retrieving data from Firebase and SQL and converting ADC values to suitable outputs if required e.g. LM35 temperature conversion. The input values are then plugged into the triangular MFs to output a fuzzy set. Each fuzzy set combination is then determined. During defuzzification, the minimum of each combination is determined. The maximum of all minimum values is then taken and the corresponding fuzzy set that it belongs to it is identified. The fuzzy set rule is then activated and score is determined based on the rule. The maximum value is then equated to the output MFs which operates in the range of the maximum value. The output is then averaged and this serves as the probability rating for the scoring identified. The preceding sections will explain the process in detail.

6.2.5. Input, Output and Linguistic Variables

The first step of the process involved identifying the input variables (also called crisp variables) that go into the fuzzification process. This included the sensory information from the WBAN network. The linguistic variables which describe each input variable was then determined based on different expected ranges of the input variables. Two models were developed: 1) A FL model to indicate general patient health status and 2) A FL model to determine the risk factor associated with the patient’s environmental conditions.

Table 6-2: Input Field and Corresponding Linguistic

Input Field	Range	Linguistic Variable
Model 1		
Pulse	$0 \leq x \leq 65$	Bradycardia
	$60 \leq x \leq 80$	Normal
	$80 < x \leq 100$	
	$5.3 \leq x \leq 20.0$	Tachycardia
Body Temperature	$0 \leq x \leq 36.5$	Cold
	$36.0 \leq x \leq 37.0$	Normal
	$37.0 < x \leq 38.0$	
	$37.5 \leq x \leq 42.0$	Hot
Glucose	$0 \leq x \leq 4.1$	Low
	$3.9 \leq x \leq 4.7$	Normal
	$4.7 < x \leq 5.5$	
	$5.3 \leq x \leq 20.0$	High
Model 2		
Humidity	$0 \leq x \leq 45.0$	Low
	$40.0 \leq x \leq 50.0$	Normal
	$50.0 < x \leq 60.0$	
	$55.0 \leq x \leq 100.0$	High
Age	$13 \leq x \leq 19$	Adolescent
	$16 \leq x \leq 30$	Adult
	$30 < x \leq 45$	
	$40 \leq x \leq 100$	Senior Adult
Body Temperature	$0 \leq x \leq 36.5$	Cold
	$36.0 \leq x \leq 37.0$	Normal
	$37.0 < x \leq 38.0$	
	$37.5 \leq x \leq 42.0$	Hot

6.2.6. Fuzzification with MFs

The fuzzification process involved developing the relevant MFs to convert the Crisp input variables to its corresponding fuzzy set. Each variable as listed in Table 6-2 consists of 3 or 4 fuzzy sets which are represented using the triangular MF. Below lists the MFs and their corresponding fuzzy sets.

6.2.6.1. Model 1

Pulse Rate

$$\mu_{Low}(x) = \left\{ \frac{65-x}{65}, \quad 0 \leq x \leq 65 \right. \dots\dots\dots (6.5)$$

$$\mu_{Normal}(x) = \begin{cases} \frac{x-60}{80-60}, & 60 \leq x \leq 80 \\ \frac{100-x}{100-80}, & 80 < x \leq 100 \end{cases} \dots\dots\dots (6.6)$$

$$\mu_{High}(x) = \left\{ \frac{x-95}{140-95}, \quad 95 \leq x \leq 140 \right. \dots\dots\dots (6.7)$$

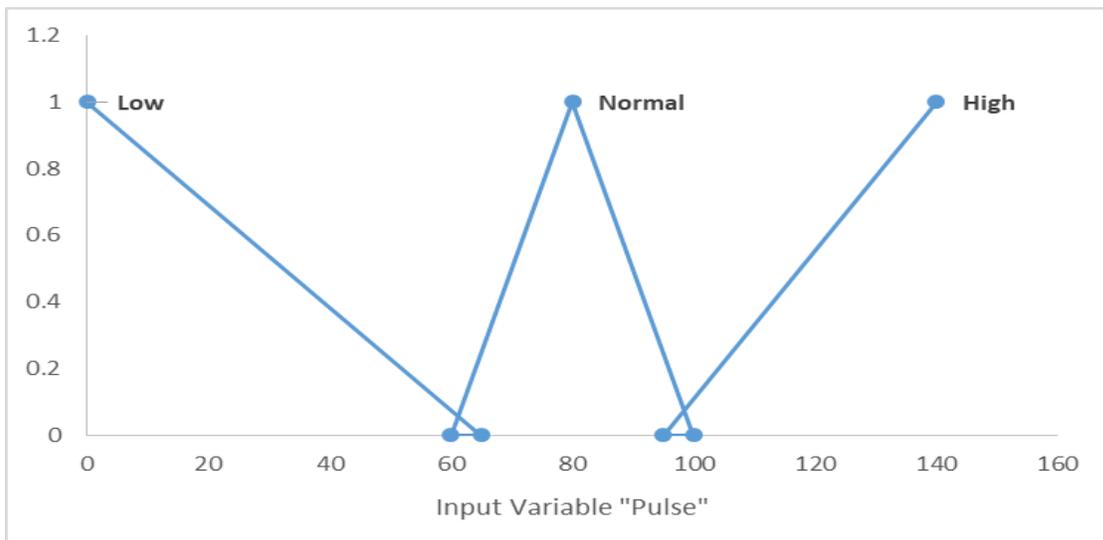


Figure 6-43: Triangular Membership Function Pulse Rate

Body Temperature

$$\mu_{Cold}(x) = \begin{cases} \frac{36.5-x}{36.5}, & 0 \leq x \leq 36.5 \end{cases} \dots\dots\dots (6.8)$$

$$\mu_{Normal}(x) = \begin{cases} \frac{x-36.0}{37.0-36.0}, & 36.0 \leq x \leq 37.0 \\ \frac{38.0-x}{38.0-37.0}, & 37.0 < x \leq 38.0 \end{cases} \dots\dots\dots (6.9)$$

$$\mu_{Hot}(x) = \begin{cases} \frac{x-37.5}{42.0-37.5}, & 37.5 \leq x \leq 42.0 \end{cases} \dots\dots\dots (6.10)$$

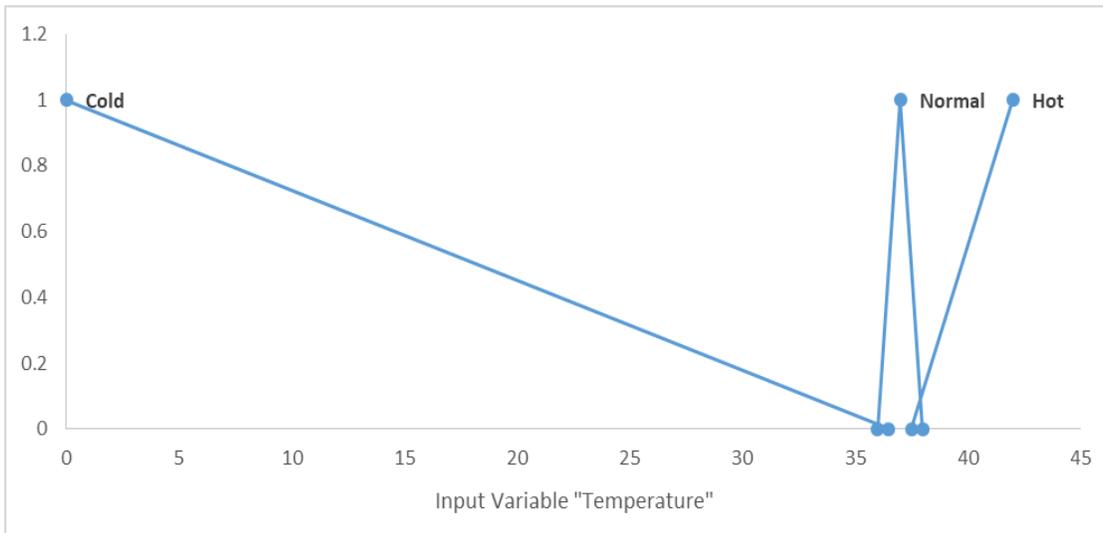


Figure 6-44: Triangular Membership Function Body Temperature

Glucose

$$\mu_{Low}(x) = \begin{cases} \frac{4.1-x}{4.1-0}, & 0 \leq x \leq 4.1 \end{cases} \dots\dots\dots (6.11)$$

$$\mu_{Normal}(x) = \begin{cases} \frac{x-3.9}{4.7-3.9}, & 3.9 \leq x \leq 4.7 \\ \frac{5.5-x}{5.5-4.7}, & 4.7 < x \leq 5.5 \end{cases} \dots\dots\dots (6.12)$$

$$\mu_{High}(x) = \begin{cases} \frac{x-5.3}{20.0-5.3}, & 5.3 \leq x \leq 20.0 \end{cases} \dots\dots\dots (6.13)$$

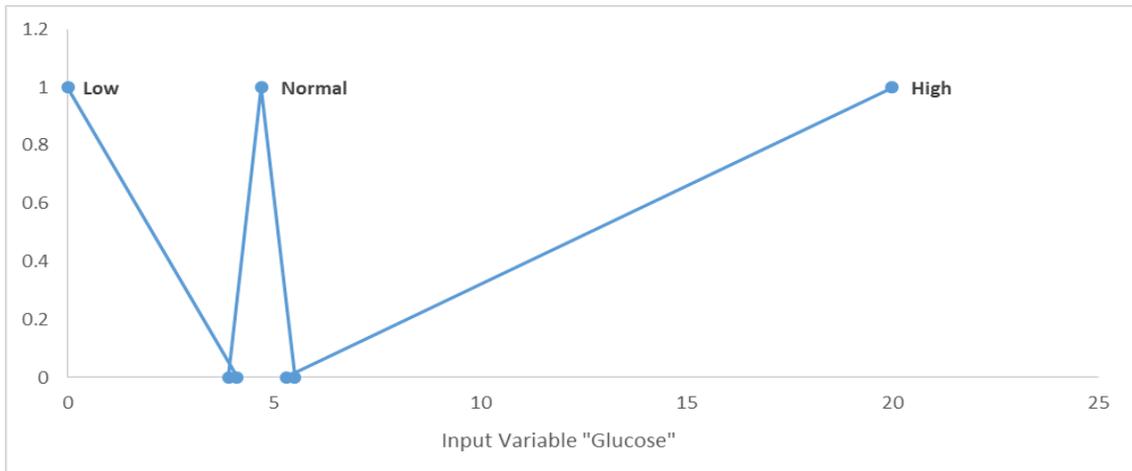


Figure 6-45: Triangular Membership Function Glucose Levels

Age

$$\mu_{Adolescent}(x) = \begin{cases} \frac{19-x}{19-13}, & 13 \leq x \leq 19 \end{cases} \dots\dots\dots (6.14)$$

$$\mu_{Adult}(x) = \begin{cases} \frac{x-16}{30-16}, & 16 \leq x \leq 30 \\ \frac{45-x}{45-30}, & 30 \leq x \leq 45 \end{cases} \dots\dots\dots (6.15)$$

$$\mu_{Senior Adult}(x) = \begin{cases} \frac{x-40}{100-40}, & 40 \leq x \leq 100 \end{cases} \dots\dots\dots (6.16)$$

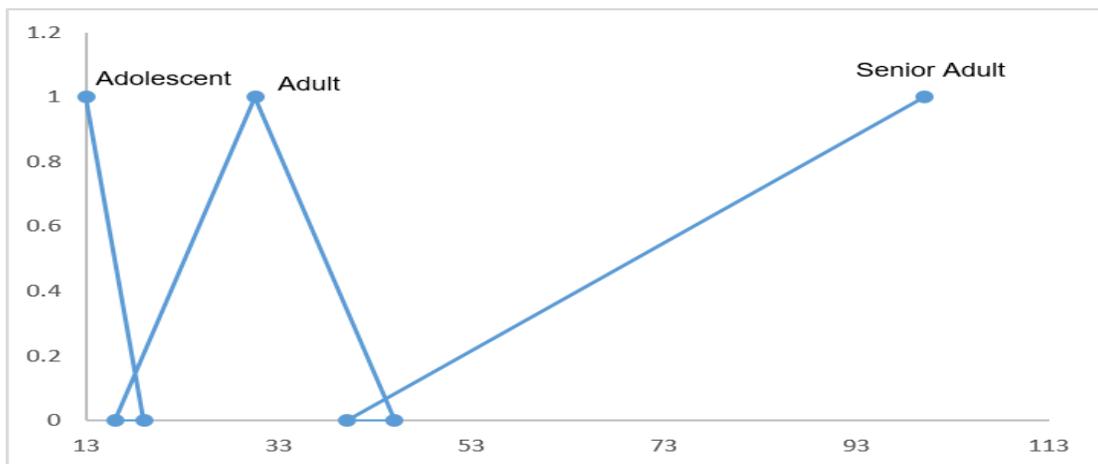


Figure 6-46: Triangular Membership Function Age

Output Function

$$\mu_{Healthy}(x) = \left\{ \frac{50-x}{50}, \quad 0 \leq x \leq 50 \right. \dots\dots\dots (6.17)$$

$$\mu_{Unwell}(x) = \begin{cases} \frac{x-45}{60-45}, & 45 \leq x \leq 60 \\ \frac{75-x}{75-60}, & 60 < x \leq 75 \end{cases} \dots\dots\dots (6.18)$$

$$\mu_{Critical\ Health}(x) = \left\{ \frac{x-70}{100-70}, \quad 70 \leq x \leq 100 \right. \dots\dots\dots (6.19)$$

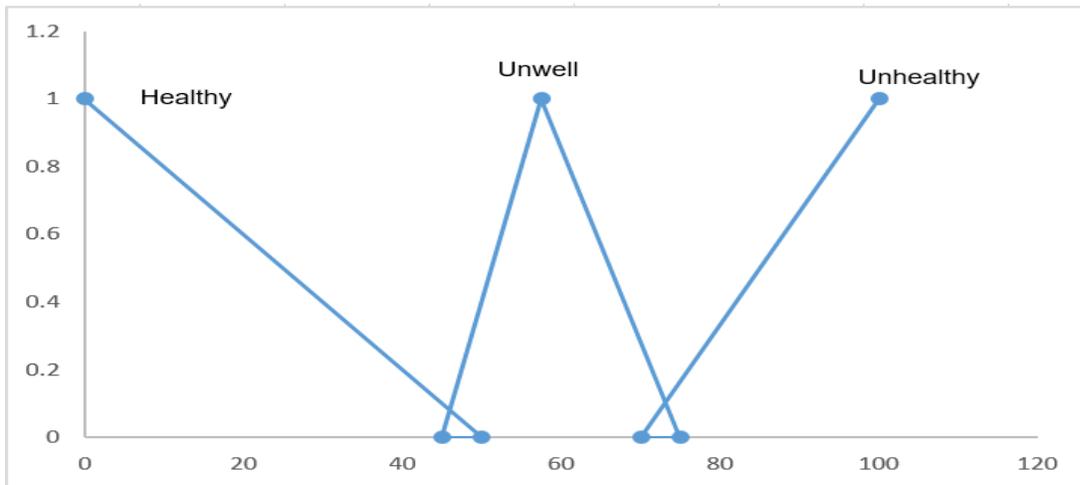


Figure 6-47: Triangular Membership Function Output

6.2.6.2. Model 2

Humidity

$$\mu_{Low}(x) = \left\{ \frac{45.0-x}{45.0}, \quad 0 \leq x \leq 45.0 \right. \dots\dots\dots (6.20)$$

$$\mu_{Normal}(x) = \begin{cases} \frac{x-40.0}{50.0-40.0}, & 40.0 \leq x \leq 50.0 \\ \frac{60.0-x}{60.0-50.0}, & 50.0 < x \leq 60.0 \end{cases} \dots\dots\dots (6.21)$$

$$\mu_{High}(x) = \left\{ \frac{x-55.0}{100-55.0}, \quad 55.0 \leq x \leq 100.0 \right. \dots\dots\dots (6.22)$$

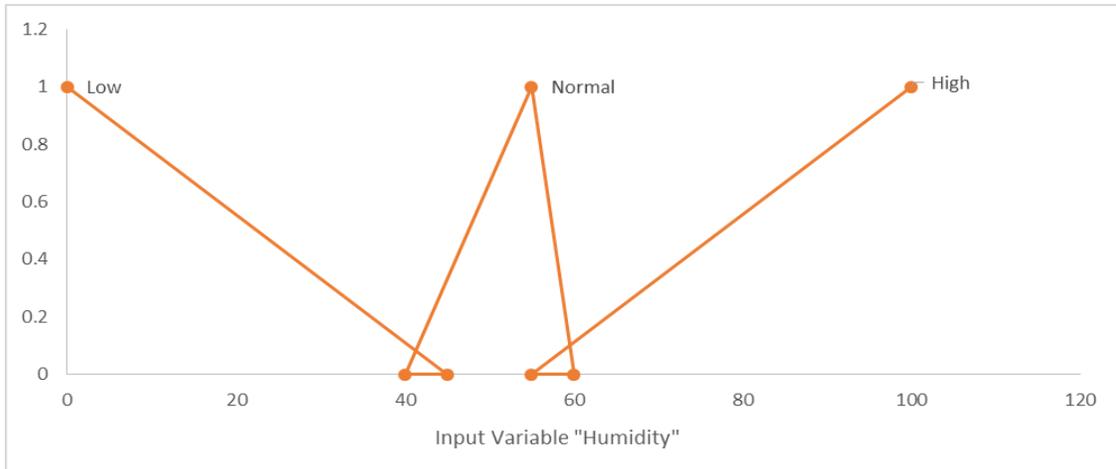


Figure 6-48: Triangular Membership Function Humidity

Age

$$\mu_{Adolescent}(x) = \begin{cases} \frac{19-x}{19-13}, & 13 \leq x \leq 19 \end{cases} \dots\dots\dots (6.23)$$

$$\mu_{Adult}(x) = \begin{cases} \frac{x-16}{30-16}, & 16 \leq x \leq 30 \\ \frac{45-x}{45-30}, & 30 \leq x \leq 45 \end{cases} \dots\dots\dots (6.24)$$

$$\mu_{Senior Adult}(x) = \begin{cases} \frac{x-40}{100-40}, & 40 \leq x \leq 100 \end{cases} \dots\dots\dots (6.25)$$

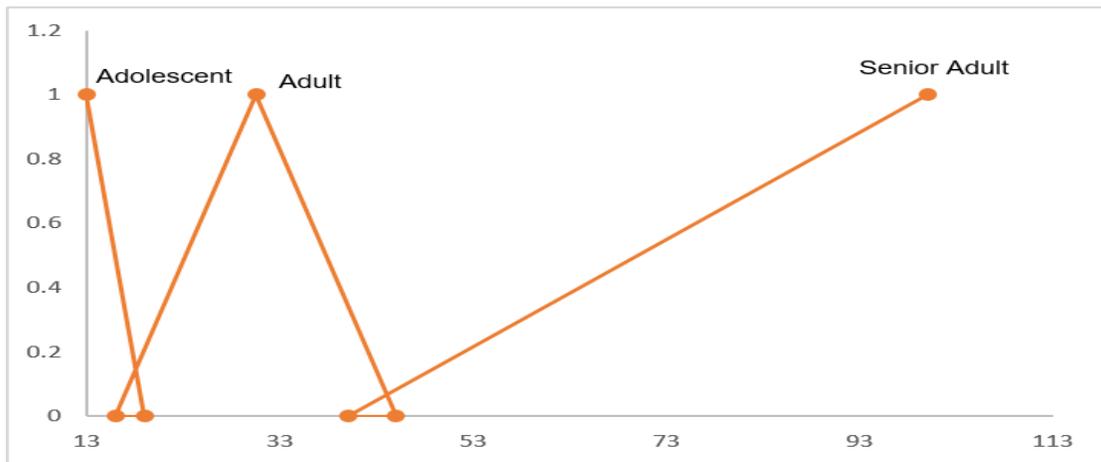


Figure 6-49: Triangular Membership Function Age

Body Temperature

$$\mu_{Cold}(x) = \begin{cases} \frac{36.5-x}{36.5}, & 0 \leq x \leq 36.5 \end{cases} \dots\dots\dots (6.26)$$

$$\mu_{Normal}(x) = \begin{cases} \frac{x-36.0}{37.0-36.0}, & 36.0 \leq x \leq 37.0 \\ \frac{38.0-x}{38.0-37.0}, & 37.0 < x \leq 38.0 \end{cases} \dots\dots\dots (6.27)$$

$$\mu_{Hot}(x) = \begin{cases} \frac{x-37.5}{42.0-37.5}, & 37.5 \leq x \leq 42.0 \end{cases} \dots\dots\dots (6.28)$$

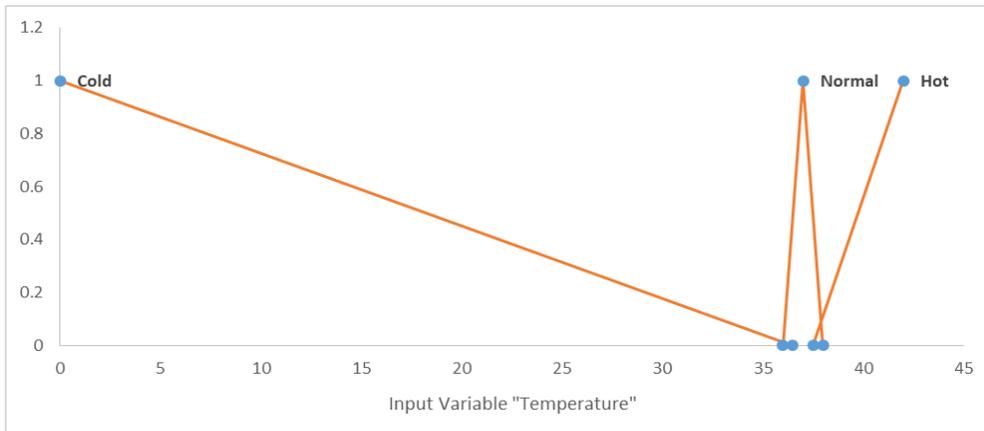


Figure 6-50: Triangular Membership Function Body Temperature

Output Function

$$\mu_{Low Risk}(x) = \begin{cases} \frac{50-x}{50}, & 0 \leq x \leq 50 \end{cases} \dots\dots\dots (6.29)$$

$$\mu_{Risky}(x) = \begin{cases} \frac{x-45}{60-45}, & 45 \leq x \leq 60 \\ \frac{75-x}{75-60}, & 60 < x \leq 75 \end{cases} \dots\dots\dots (6.30)$$

$$\mu_{High Risk}(x) = \begin{cases} \frac{x-70}{100-70}, & 70 \leq x \leq 100 \end{cases} \dots\dots\dots (6.31)$$

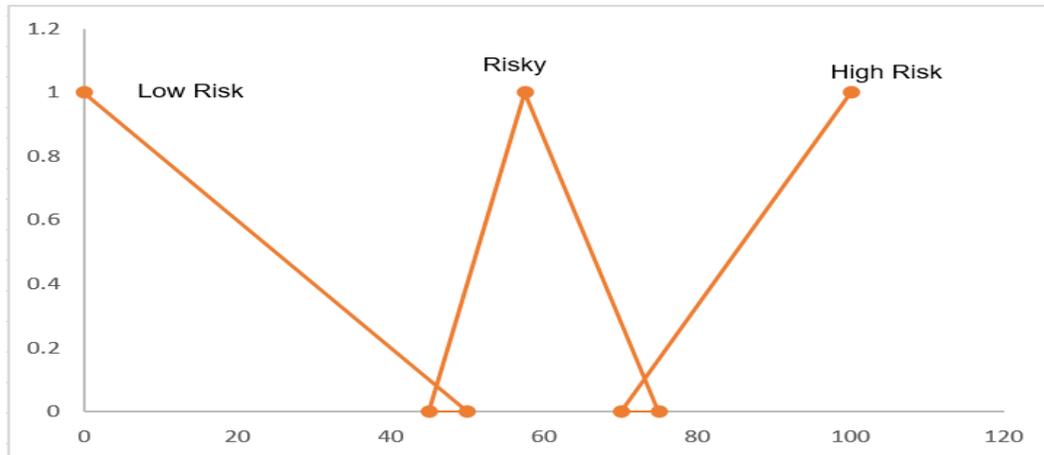


Figure 6-51: Output Membership Function Model 2

6.2.7. Developing a Rule Base

The rule based forms an intricate component for FL predictions. For the systems employed based on 3 and 4 input variables each having 3 linguistic variables there were 27 and 81 possible rules. Table 6-3 and Table 6-4 show sample rules for model 1 and model 2 respectively. A full list of the rules can be found in appendix L.

6.2.7.1. Model 1

Table 6-3 shows a sample set of the rule base for model 1. Rule 1, 12 and 15 can be interpreted as follows:

Rule 1: If body temperature = cold or glucose levels = low, pulse = low and age = adolescent, then output = critical health. This essentially means that if an adolescent's body temperature or glucose is low together with a low pulse this rule will activate and give a critical health output.

Rule 12: If body temperature or glucose levels = normal, pulse = low and age = Senior Adult then the overall output is critical health. This means that although one out of the 3 measured variables is outside the limits, the scoring is critical since the patient is a senior adult.

Rule 15: if body temperature or glucose levels = normal, pulse = normal and glucose = normal then the output is healthy. This implies that all measured variables of the patient is within normal ranges and therefore they are in a healthy state.

Table 6-3: Sample Rule Base Model 1

Rule No	Temperature or Glucose	Pulse	Age	Output
1	Cold/Low	Low	Adolescent	Critical Health
...				
12	Normal	Low	Senior Adult	Critical Health
...				
15	Normal	Low	Senior Adult	Healthy
...				

6.2.7.2. Model 2

Table 6-4 shows a sample set of the rule base for model 2. Rule 1, 12 and 15 can be interpreted as follows:

Rule 1: If Humidity = low and age = adolescent and body temperature = low, then output = risky. This essentially means that if an adolescent has a low body temperature and is in low humidity conditions they are potentially in a risky situation. Due to their age, they are not classified at a higher risk rating.

Rule 12: If Humidity= normal and age = normal and body temperature = normal then the overall output is low risk. This means that because the person's body temperature is normal and they are in normal humidity conditions, they are at low risk of experiencing adverse effects due to the environmental conditions.

Rule 15: if Humidity = high, and age group =adolescent and body temperature = normal then the output is high risk. This implies the patient needs to move to better conditions as the environment is having an adverse effect on their health status.

Table 6-4: Sample Rule Base Model 2

Rule No	Humidity	Age	Body Temperature	Output
1	Low	Adolescent	Cold	Risky
...				
12	Normal	Adolescent	Normal	Low Risk
...				
15	High	Senior Adult	Cold	High Risk
...				

6.2.8. Rule Evaluation

This part of the process involves extracting the relevant Crisp variables from the SQLite database and determining their relevant fuzzy sets using the defined membership functions. IF AND statements were utilized within android studio to determine which membership function would be required to convert the necessary crisp inputs into fuzzy sets. The example below illustrates the rule evaluation principle for model 1:

Given temperature = 36.1 °C, pulse = 70 BPM, glucose = 4 mmol/L and age = 21. The corresponding membership functions are as follows: 37 °C is located between sets cold and normal giving membership values of 0.01 for the cold set and 0.1 for the normal set. The pulse of 70 BPM lies between the normal membership function with a value of 0.5. The glucose level of 40 mmol/L lies between the low and normal range membership functions with output values of 0.025 and 0.125 respectively. Lastly the age lies in the *Senior Adult* range giving output values of 1.

6.2.9. Defuzzification

Once the necessary fuzzy sets are determined they would need to be converted back to crisp outputs using a relevant defuzzification method. The MoM, as shown in equation (6.32) was utilized to carry out this process.

$$x^* = \frac{\sum_{i=1}^n \bar{x}_i}{n} \dots\dots\dots (6.32)$$

Where x^* represents the crisp input and \bar{x}_i is the sum of crisp values whose MFs reach the maximum.

In order to identify the points to utilize in the above equation, the AND (minimum) and OR (maximum) needs to be determined from all fuzzy output sets. The possible combinations need to first be calculated to carry out this process. Two variables were utilized to hold the values of the fuzzy sets. The second variable is used for the instances of a value falling between two ranges. If the value however only falls between one of the ranges, then the second holding string will be set to a value of 2000, a random out of range number which will eliminate holding string 2 combinations.

Based on the holding values, 16 possible combinations are possible for model 1 and 8 possible combinations for model 2. This is considering that with any given crisp input value, it is only possible for it to fall between a maximum of two ranges e.g. low and normal or normal and high. So for model 2, letting the *holding_string 1* = a and *holding_string 2* = b with position 1, 2 and 3 corresponding to humidity, age and BT for model 2, the possible sets are as follows:

- If humidity = "a-humidity" and age= "a-age" and BT = "a-BT"
- If humidity = "a-humidity" and age = "a-age" and BT = "b- BT"
- If humidity = "a-humidity" and age = "b-age" and BT = "a- BT"
- If humidity = "b-humidity" and age = "a-age" and BT = "a- BT"
- If humidity = "a-humidity" and age = "a-age" and BT = "a- BT"
- If humidity = "a-humidity" and age = "a-age" and BT = "a- BT"
- If humidity = "a-humidity" and age = "a-age" and BT = "a- BT"
- If humidity = "a-humidity" and age = "a-age" and BT = "a- BT"

Where "a", "b" and "c" can be any linguistic variable based on the input crisp value e.g. humidity: a-humidity= "Low", age: "a-age" = "Adolescent" and BT: a-BT= "cold." The Figure below shows an extract of the programming for the temperature MF of model 2. The range of the input crisp value is determined and a value of "A", "B" or "C" is given based on a lower range, normal range and high range. The holding string is prefixed with the word "*Lingusitic*."

```
//Humidity
if (Humidity <= 39.9) {
    LinguisticH1 = "A";
    LinguisticH1Val = (45.0 - Humidity) / (45.0 - 0.0);
    LinguisticH2 = "A";
    LinguisticH2Val = 2000.0;
} else if (Humidity >= 40.0 && Humidity <= 45.0) {
    LinguisticH1 = "A";
    LinguisticH1Val = (45.0 - Humidity) / 45.0;
    LinguisticH2 = "B";
    LinguisticH2Val = (Humidity - 40.0) / (50.0 - 40.0);
}
```

Aggregated pairs are then determined as shown below, where Group 1 for example indicates a scenario for input values which fall in low ranges of humidity, age and BT i.e. A = Low Humidity, A = Adolescent and A = Cold. The min value is then calculated followed by the max output of all min values. Here the 2000 value eliminates all duplicate sets so that the correct max value can be obtained.

```

Group1 = LinguisticT1 + LinguisticP1 + LinguisticG1;
MinGroup1 = Math.min(LinguisticT1Val,Math.min(LinguisticP1Val,LinguisticG1Val));
CheckG1 = Math.max(LinguisticT1Val,Math.max(LinguisticP1Val, LinguisticG1Val));
MinGroup1S = String.valueOf(MinGroup1);

if (CheckG1<2000)
{
    MF= Math.max(MinGroup1,MinGroup1);
    Flag = "Okay";
}

if(CheckG2<2000)
{
    MF=Math.max(MinGroup2,MF);
    Flag = "Okay";
}

```

Finally, the output is fed back into the membership functions and the MoM method is used to determine the crisp output. The max value is equated back to its fuzzy set and this set is used to determine which MF to use.

```

//Defuzzification

if (Stat2.equals("AAA")) {
    Status2 = "Risky";
    HealthStat2 = (((60 - 45) * Statval2) + 45) + (75-(Statval2*(75-60))) / 2;
    Holding(); //done
} else if (Stat2.equals("AAB")) {
    Status2 = "Low Risk";
    HealthStat2 = 50 - (50 * Statval2);
    Holding(); //done
}

```

Model 1 will follow a similar process but will have 16 possible combinations possible for aggregating the fuzzy output sets generated.

6.3. Chapter Summary

Chapter 6 described the modelling integrated with the IoT telemonitoring system. The chapter discussed the development of the stroke prediction model using MS Azure ML classic studio which obtained scores for accuracy, precision, recall and AUC all above 92% using the decision tree binary classification model. Six different models were tested and this scored the best. The chapter discussed the use of K-folds method to test for overfitting.

The Chapter then proceeded to discuss the development of two FES models. The first one using humidity, age and body temperature to gauge environmental risk on patient health and the second using body temperature, pulse, age and glucose to gauge general patient health. The FL systems design methodology was described i.e. the use of triangular MFs for the inference rules and defuzzification using the MoM method.

Chapter 7: Use Case Scenarios

The previous chapters described how the entire IoT system was designed and integrated. This chapter will now look at how the various stakeholders engage with the system to ensure a smooth data pipeline. This chapter will further outline how effectively the system meets stakeholder requirements.

7.1. Patient Use Case Scenario

The following use case scenario will highlight how the patient will engage with the system and their role in ensuring the transition of data through the IoT system. The use case will start by explaining the sequence of activities that the stakeholder would need to carry out while engaging with the system and the subsequent data flow in each scenario. Figure 7-1 illustrates a use case diagram for a patient accessing the system.

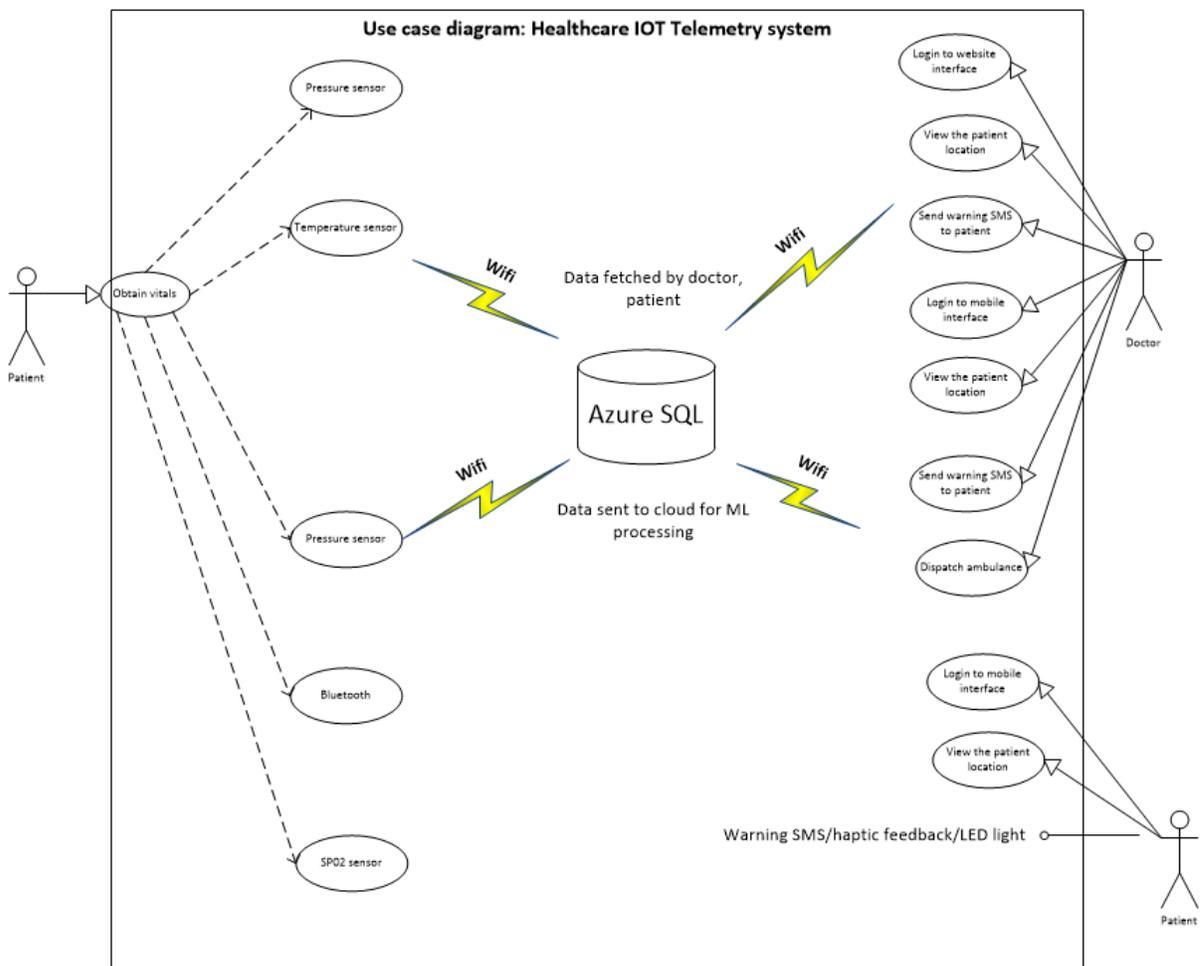


Figure 7-1: Use Case Diagram for Patient Accessing IoT system

7.1.1. Overview

The patient use case illustrates a typical scenario of a patient accessing the IoT platform i.e. the relevant steps to set-up and ensure the successful transfer of data through the data pipeline. This use case focuses on the following actions from the patient’s perspective:

- 1) Initial MA set-up – preliminary actions that need to be performed by the patient to collect data from the WBAN and store it locally on the phone before migrating the data to the cloud. These include MA registration, setting up the Bluetooth and Wi-Fi communication and setting up the GPS tracker.
- 2) Initial WA set-up – preliminary actions that need to be performed by the patient to ensure the flow of information from the cloud database to the WA
- 3) Patient viewing of physiological data – how the patient can access their physiological data and health status through the patient interface of the MA.

7.1.2. Sequence of Steps

The patient should ensure that the battery in the enclosure is fully charged using the balanced charger and will ensure that this is done before proceeding. The patient will put on the wearable device which consists of the various sensors responsible for measuring patient vitals. When the patient is comfortable with the positioning of the device, they can then proceed to the next step. The device is designed to remain in the “ON” status to ensure that sensors are stabilized from the start of taking the readings. If the user is already a registered user they can proceed to the *patient_login* Activity as shown below and enter their credentials before clicking on the login button.

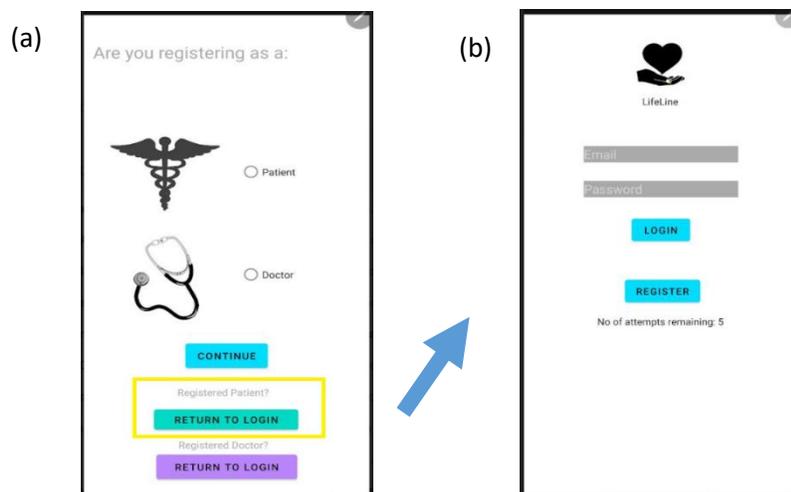


Figure 7-2: (a) Registration Screen, (b) Login Screen

If the patient is not registered on the MA platform, they will then proceed to register as a “patient” on the MA. To do this, they should ensure that their mobile or Wi-Fi connectivity is enabled. The application will then ask the user to enter all of their user information. Sample registration pages are shown below:

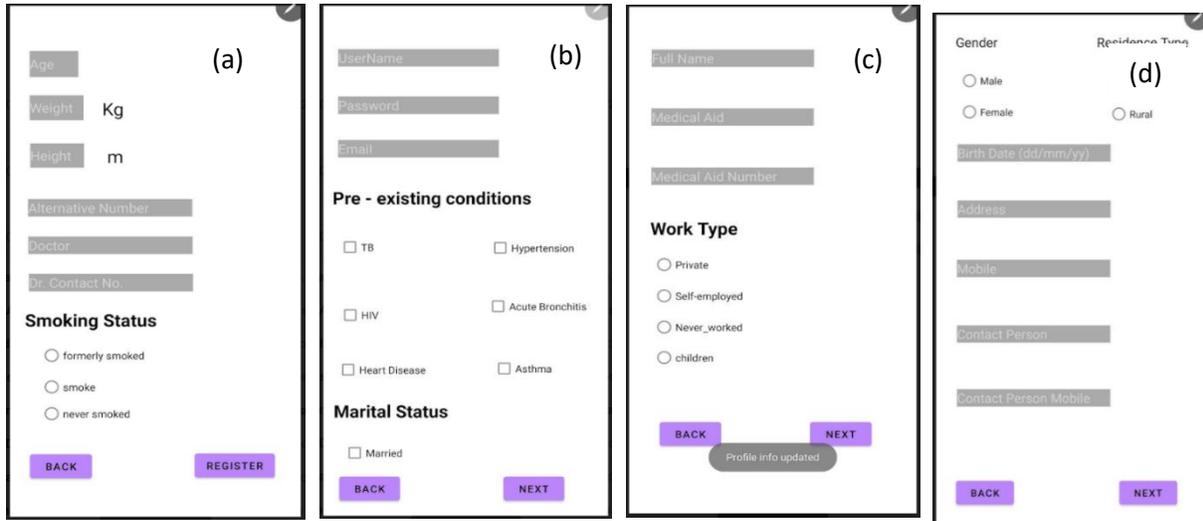


Figure 7-3 : (a) to (d) – registration pages

Upon completion, all the data will be stored in *Hashmaps* on the patient’s phone before being uploaded to the Firebase NoSQL patient database. The user will be successfully registered, as seen in Figure 7-4.



Figure 7-4: Registration Complete Page

The patient will subsequently be logged in and taken to the *patient_interface* Activity of the MA. Once in the *patient interface*, the patient needs to then proceed to the *connect* Activity where they can set-up the Bluetooth, GPS and cloud sharing functionalities:

- **Bluetooth enablement** – the *Bluetooth devices* Activity will allow the user to create a connection to transfer data between the WBAN and the MA. Clicking on the *enable* button will prompt the user to turn on Bluetooth with the message shown in Figure 7-5:

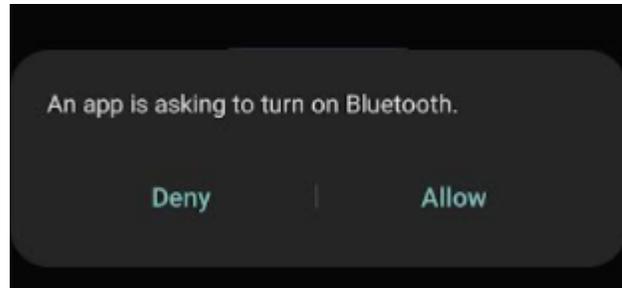


Figure 7-5: Bluetooth Prompt

After allowing this, the patient can then click the *discover* button to search for the WBAN Bluetooth device (HC06 Bluetooth module) as shown in Figure 7-6:

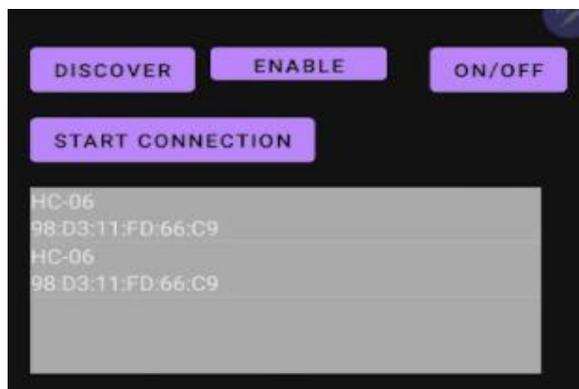


Figure 7-6: Bluetooth Discovery

The user needs to then click on the device and subsequently click the *Start Connection* button. This will enable the *broadcast_receiver* function to pair and create a connection with the Bluetooth device. To ensure successful connection, the patient can view the Bluetooth LED on the WBAN device. A fast flashing of the LED indicates the Bluetooth default state, upon pairing the flashing slows down and when a successful connection has been established the LED stays permanently on without flashing. Upon completion of connection, the MA should start receiving strings of data from the WBAN and this can be seen on a *textview* on the MA as shown in Figure 7-7.



Figure 7-7: Screen showing Smartphone receiving data from WBAN

- **GPS enablement** – In order to ensure GPS tracking of the patient’s location, the application makes use of the built in GPS within the phone. The patient can enable the app to access their location and subsequently store the data into a Firebase database node under the patient’s profile. By clicking on the *GPS Activity* “ON” button, the user can either switch on or turn off the location tracking capabilities of the system. Figure 7-8 indicates the screen output when a patient has the GPS turned off and turned on.

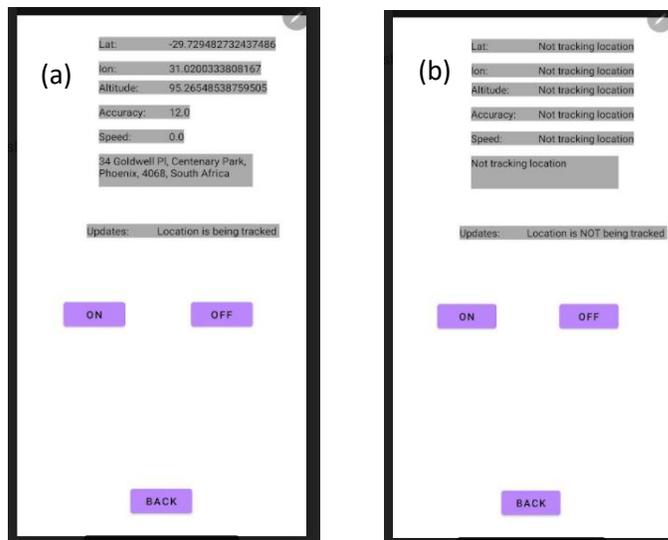


Figure 7-8: (a) GPS tracking enabled, (b) GPS tracking disabled

- **Cloud sharing enablement** – The *cloud_sharing* Activity will allow the user to transmit data stored in the phone’s SQLite database to the Microsoft Azure SQLite database.

Clicking on the *transmission* button will initiate the upload process with a *Textview* illustrating that the upload is a success (Figure 7-9).



Figure 7-9: Cloud Sharing Transmission Screen

Although system is designed to allow all registered doctors to read information stored in the Azure SQL database, as a start to data security – Firebase database security was enabled to protect the user’s primary registration data. Under the *find_a_doctor* activity, the patient is able to search for specific doctors that belongs to the health network. They can then choose to invite a doctor, which will transition them to a new screen as shown in Figure 7-10 (b) where they will be prompted to share their information. Upon accepting, they will then get a pop-up indicating that the doctor selected has been granted access to their Firebase data.

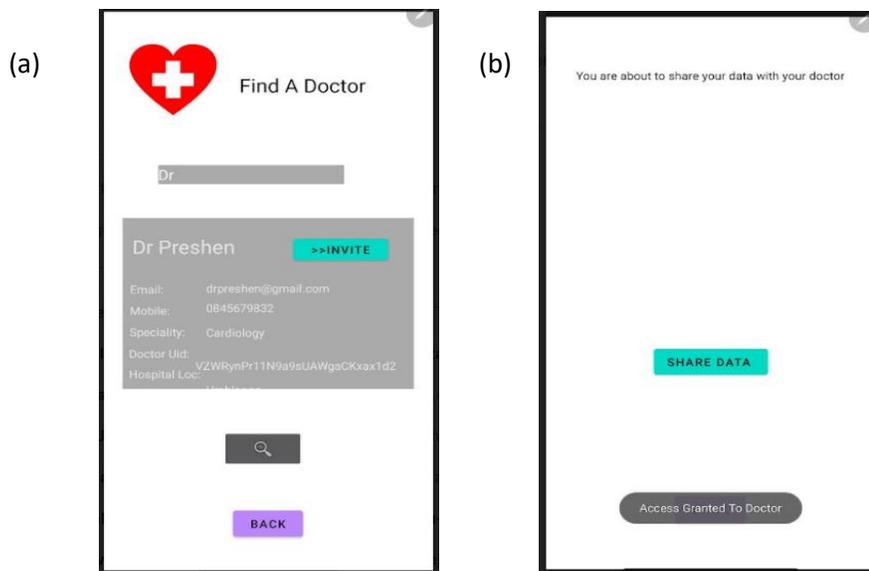


Figure 7-10: (a) Searching for Doctor on Database, (b) Sharing info with Doctor Screen

It is at this point that the user can start viewing data being received from the WBAN over Bluetooth. By navigating to the *monitor_vitals* Activity, the user will be able to view their physiological data

by clicking on the *getdata* button. The units of measurement as well as the condition i.e. status is also shown like Figure 7-11 illustrates.

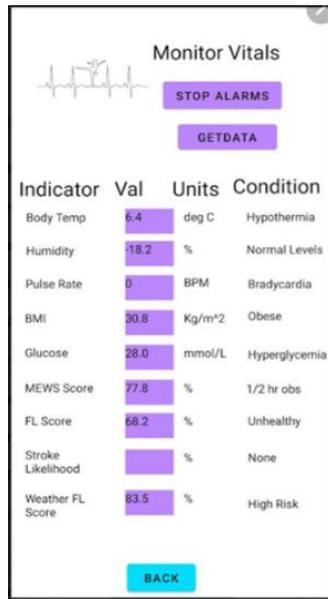


Figure 7-11: Patient Monitoring Vitals Screen

To complete the data pipeline requires establishing a connection between the Azure SQL database and the WA. This is done by the user first registering their information on the WA. This will include the user entering the doctor's unique database key (with whom they want to share information) - this key should be provided by the doctor. They will also enter their unique SA ID which will store their information under a node hierarchy starting with their ID.



PATIENT INFO UPDATE

Full Name

ID

DatabaseKey

Gender

Age

Weight

Figure 7-12: Screen for Patients to Add Their Data to WA Database

When the patient enters the doctors unique database key, the doctors ID is then stored under the specific patient's node (Figure 7-13). Rules in JSON format, then allow only doctor's with ID's under the patient's node to view the patient's data. A typical patient node and the respective screen for the user to add doctor permissions (Figure 7-14) is shown:



Figure 7-13: Firebase Node with Doctor ID Who Patient Has Chosen to Share Info With

If the patient already has their personal information stored in the database, but just wants to grant permission to another doctor, they can do so via the `add_doctor_to_your_network` html page. This will take the doctor ID and add it under the patient's ID node. The doctor will then be able to access that patient's node information since the JSON rule will now permit it.

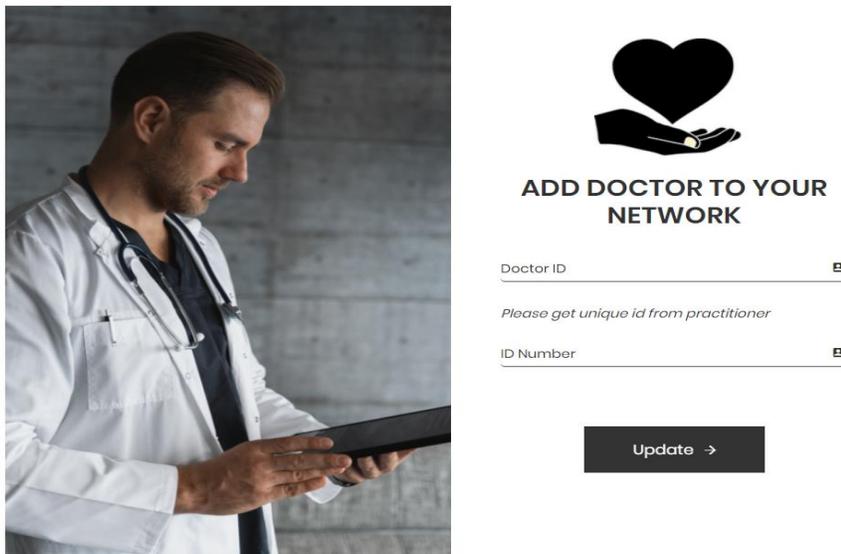


Figure 7-14: Screen on WA for Patients to Add Doctors They Want on Their Network

7.2. Doctor Use Case Scenario

The following use case scenario will highlight how the doctor will engage with the system and their role in ensuring the transition of data through the IoT system. The use case will start by explaining the sequence of activities that the stakeholder would need to carry out while engaging with the system and the subsequent data flow.

7.2.1. Overview

The doctor use case scenario explains the sequence of activities that a physician will need carry out when utilizing the developed health care system. This use case will explain the following:

- 1) Doctor registration on both the mobile and WA
- 2) The viewing of patient info on both the mobile and WA

7.2.2. Sequence of Steps

The doctor can start accessing the patient's information through the doctor interface on the MA. If the doctor is already registered, they can navigate to the *login* Activity by clicking on the *Return to Login* Button. Once the button is clicked, they will be redirected to the login page. Here they can enter credentials to log in to the doctor interface. An illustration of this process is shown in Figure 7-15:



Figure 7-15: (a) Main Page, (b) Doctor Login Page

If the doctor isn't registered, they will then need to navigate to the registration page and complete details before they are transitioned to the doctor interface. Screens are shown below:

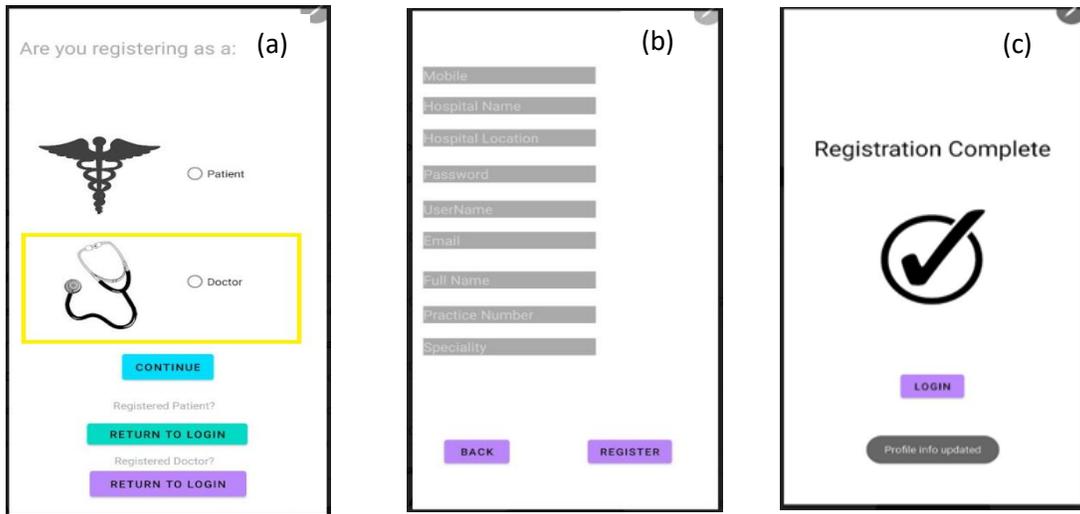


Figure 7-16: (a) Doctor Registration selection, (b) Doctor details Registration Activity, (c) Registration Complete Activity

Once registered, the information as with the patient will be uploaded to a Firebase database and thereafter transition the doctor to the doctor interface shown in Figure 7-17. Here, the doctor can either choose the *patient_info* Activity, *timestamps* or the *track patient* Activity.

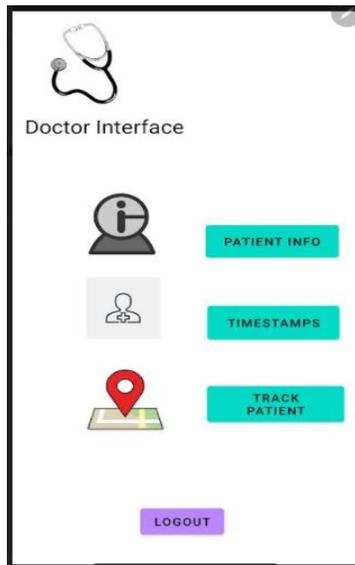


Figure 7-17: Doctor Interface Activity

The *track_patient* Activity will allow the doctor to access patient information in the form of graphs and *Textviews*. Upon clicking the Activity, the doctor can then search for a specific patient registered using their first name. When a patient is located in the database, a general description of the patient pops up together with the option for the doctor to choose between the 1) the *Loc* Activity which allows the doctor to track the patient's location, 2) the *Gen* activity for viewing the general patient data like temperature and BT, 3) The *EMG* Activity for viewing the patient's EMG data and 4) The *ECG* Activity for viewing of the patient's ECG data.

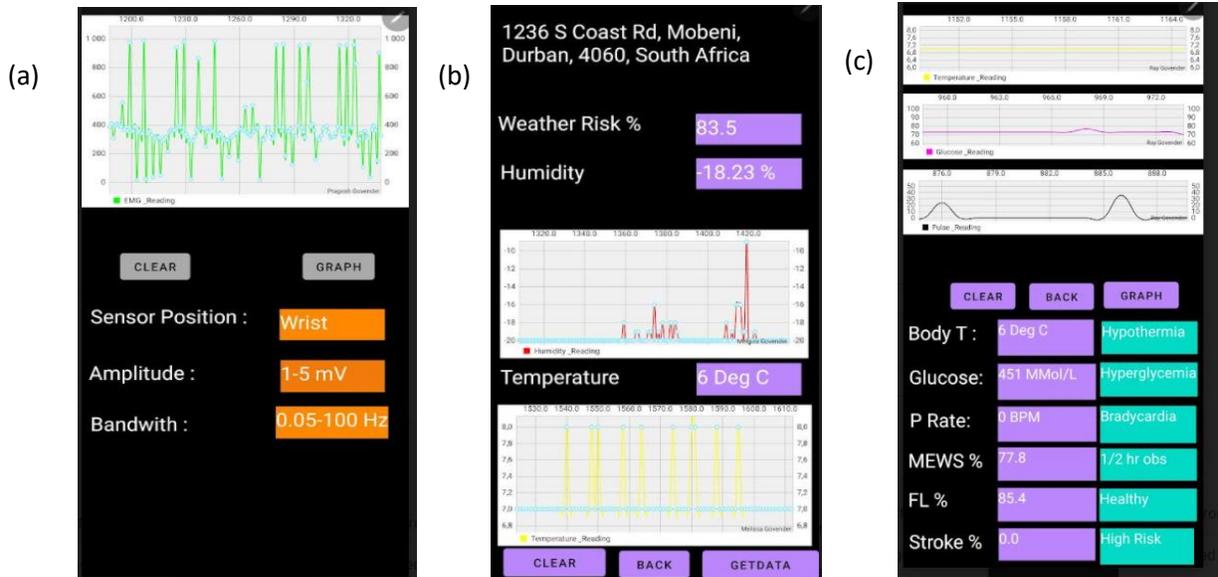


Figure 7-18: (a) ECG monitoring Activity, (b) Location Tracking and Environmental Monitoring Activity, (c) General Patient Info

Due to the importance of doctors being able to access historic data of patients to identify anomalies and trends in their health status, timestamps are extremely important. Details such as date of data points and its corresponding time will give insight into what triggered an event. These timestamps were too large to fit adequately on a mobile interface hence a timestamp screen was developed. Using this screen, a doctor is able to identify a number range for specific data points and through a *listview* under the *timestamps* Activity, identify the corresponding date and time range. The illustration of the screen is shown in Figure 7-19.

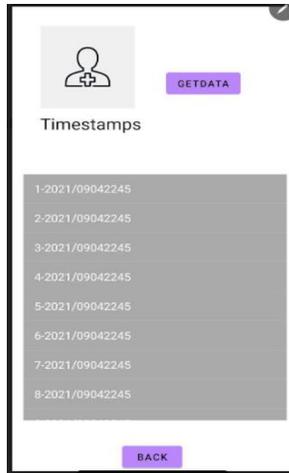


Figure 7-19: Timestamps Activity

To access data on the WA, as with the MA, physicians will need to first register on the WA. Here they will be prompted to enter all of their relevant data including practicing number and specialty service. Once registered, they will be directed to the login screen. Examples of registration pages are seen in Figures 7-20 and Figure 7-21.

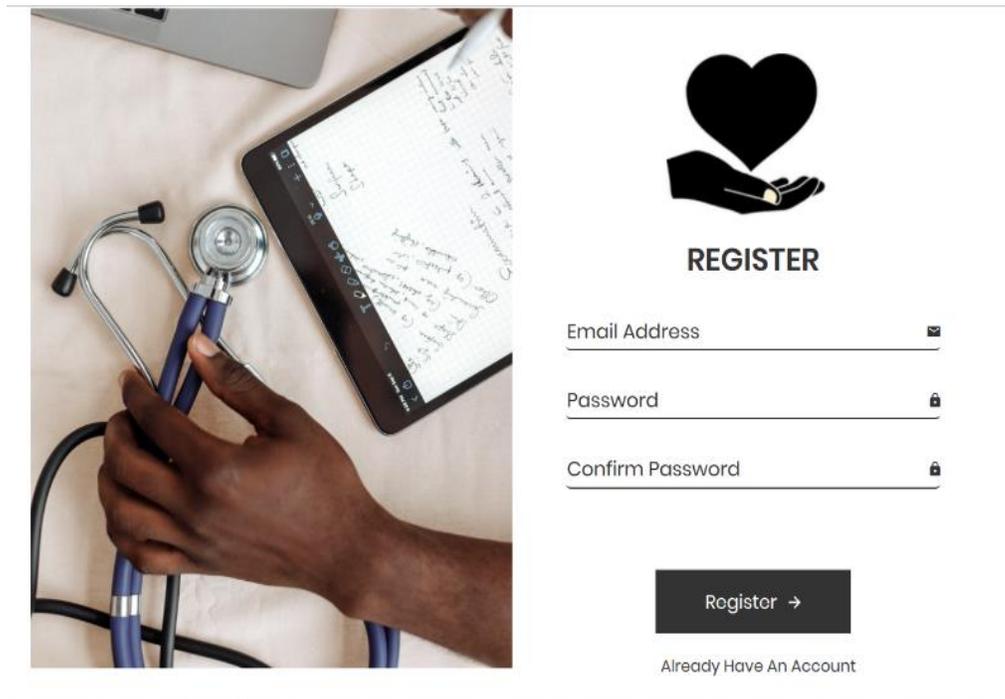


Figure 7-20: Doctor Registration Page WA

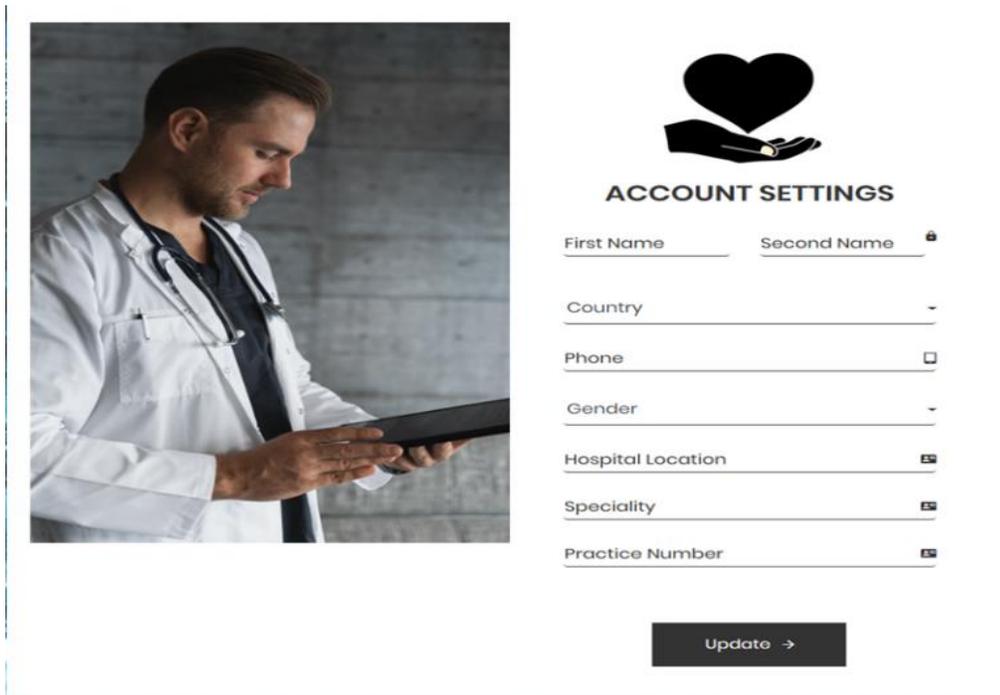


Figure 7-21: Doctor Registration Details Page

If the user already has an account, they can click on *already have an account* on the registration page. Alternatively they can navigate directly to the login page from the main menu. The login page for entering credentials is shown below.

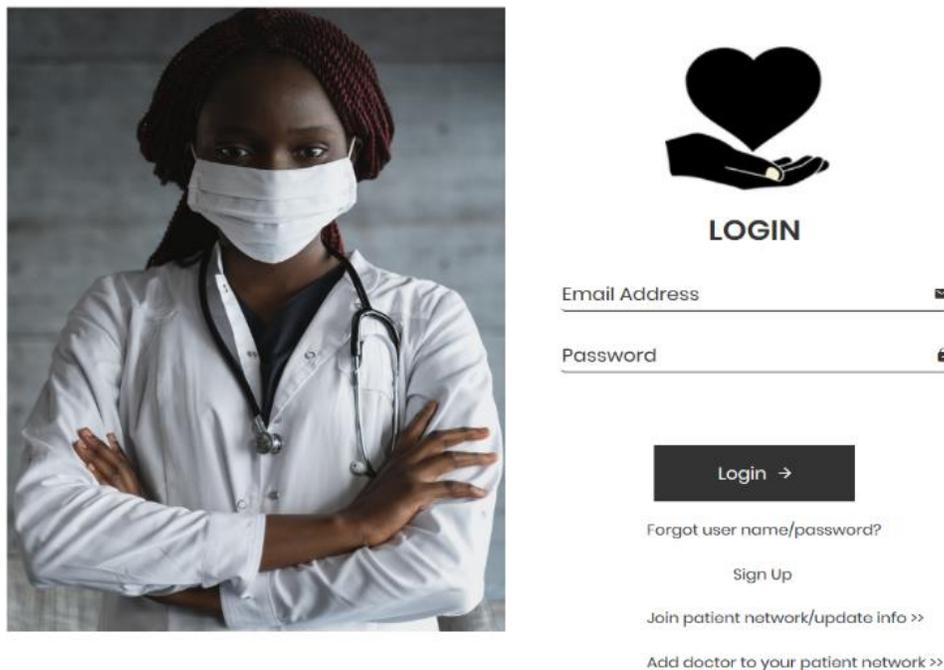


Figure 7-22: Doctor Login Page

If the user has forgotten their password, they can click on the “forgot password” link on the login page where they will be redirected to the *forgot_password* Page. Here a link will be sent from Firebase prompting user to change their password.

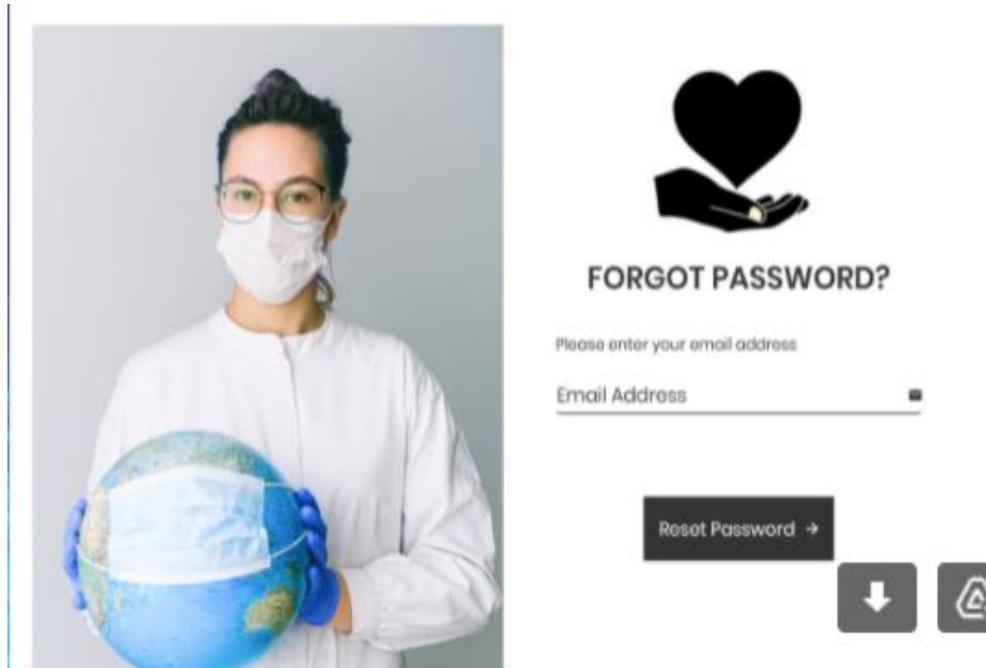


Figure 7-23: Doctor Forgot Password Page

Once logged in, the physician is able to choose between the *patient_info* screen and the *monitoring* screen. The *patient_info* screen will allow the physician to access general patient info stored within Firebase. Again only doctors with permissions can view a specific patient’s data. Once clicked, the doctor will be required to enter the patient SA ID as shown in Figure 7-24. The patient info Page is shown in Figure 7-25.

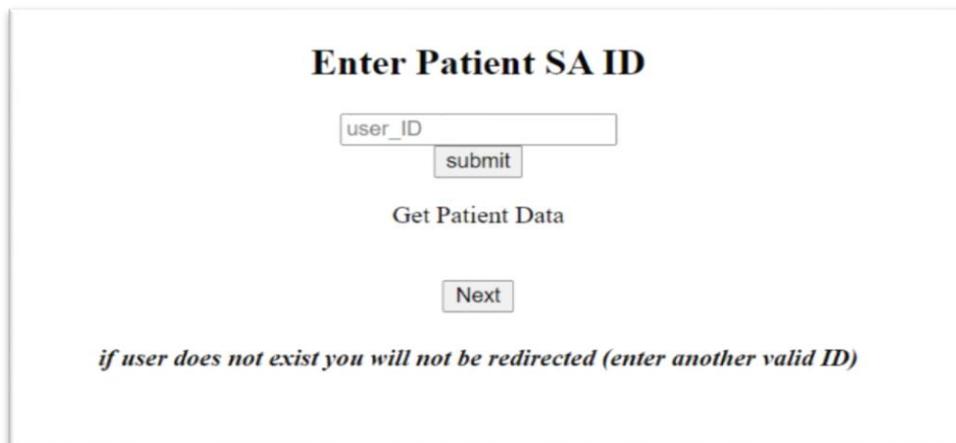


Figure 7-24: Screen for Doctor to Capture ID for “Patient Info”

Enter Patient SA ID to search Patient Info

Full Name: Peter Sinclair
Unique Key:
Gender: male
Age: 29
Weight: 67
Height: 1.8
Date of Birth: 17 September 1990
Mobile: 031 5673345
Alternative Number: 074 5589234
Family Doctor: Dr Rene Goller
Family Doctor: 031 4536789

Figure 7-25: Patient Info Page

If the doctor were to navigate to the *monitoring* screen he/she would be able to access three screens showing patient data. 1) The *location* screen for patient tracking, 2) The *general* screen for patient PR, Temp etc. 3) The EMG screen for EMG graph and 4) The ECG screen for the ECG graph. Figure 7-26 shows an example of how the doctor will access the ECG screen of the patient. The *EMG* and *general_info* screen follow a similar procedure. ID number of the patient needs to be entered and submitted, then the *next* button needs to be clicked – if the user exists then the physician will be navigated to the relevant screen as depicted in Figure 7-26 and 7-27. The *close off session* button needs to be clicked before exiting the screen.

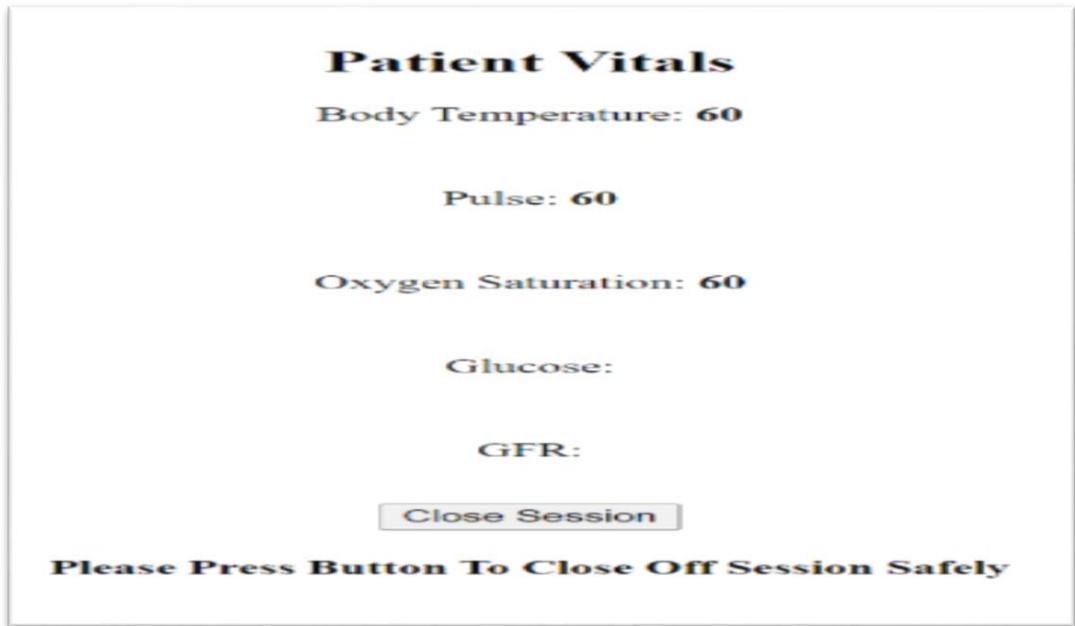


Figure 7-26: Patient Vitals Screen WA

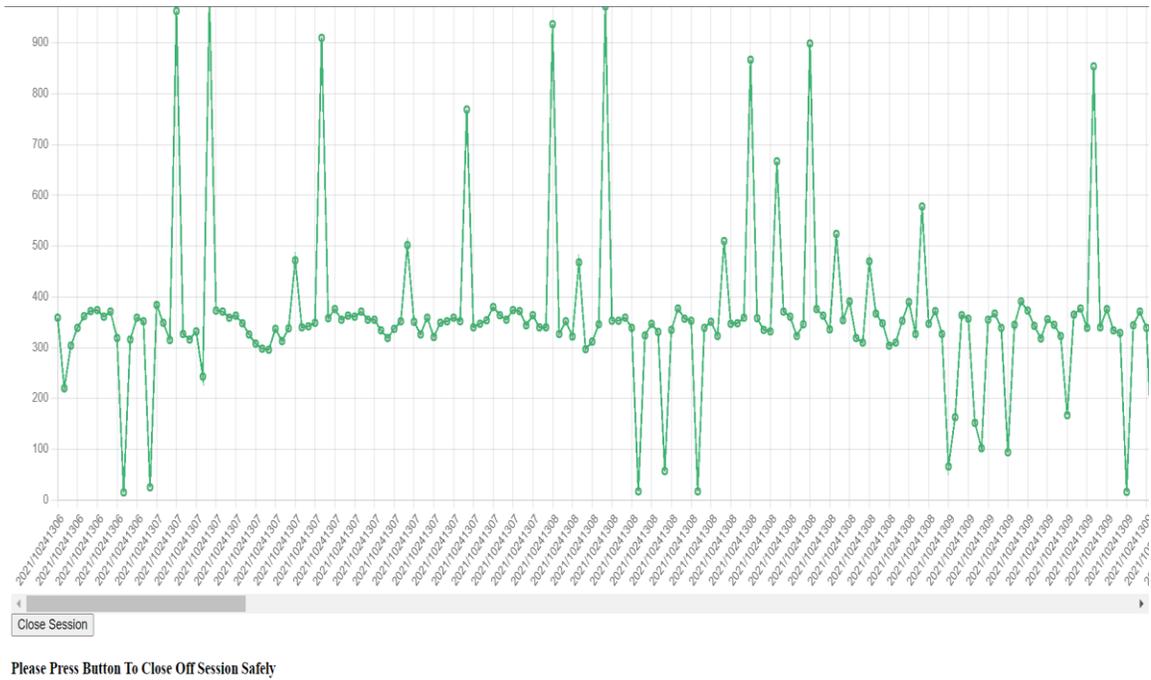


Figure 7-27: Example of ECG Graph on WA Page

The doctor will also be able to track the user location on the screen shown below. Again ID number of the user will enable their location to be extracted from Firebase and displayed on the webpage.



Figure 7-28: Patient Location Tracking on WA

7.3. Other Health Practitioner Use Case Scenario

There will be cases when the doctor requires the assistance of supporting staff such as nurses and radiographers. This will require them having access to patient data, hence the reason for the supporting health practitioner use case. This use case scenario is much simpler than the other use cases. Here nurses etc. will be able to access patient records as per instruction from the patient's doctor. No registration of supporting health practitioners is required. Instead, they can access the patient information through the registered doctor's interface. This will usually be through the WA hosted on the hospital server.

The supporting staff member will attain the doctor's credentials. They will then be able to log into the doctor's account and view the relevant patient's data of whom the doctor is treating.

7.4. Chapter Summary

This chapter looked at the use case scenarios of various stakeholders. The first use case scenario related to the patient accessing the IoT system. The registration process for the user and their duties in terms of ensuring data transmission from the WBAN to the smartphone was illustrated.

Thereafter the patient's role in enabling the transmission from the smartphone to the cloud was also elaborated on. The process of patients adding their data to the WA database was explained together with the procedure for the patient to access their health data via the MA.

The second use case scenario was for the physician. The registration process of the physician was explained followed by the process physicians should use to read patient data off the MA and WA. This included the very important demonstration of how doctors can track the location of patients during emergency cases. The process of how other health practitioners can access the patient's data was also briefly discussed.

Chapter 8: Results

The previous section looked at the use case scenario of each stakeholder. This was to gain an 1) understanding of how the system works, 2) to ensure that the system acts the way it was designed 3) to show the role of each user to ensure that data flows smoothly through the data pipeline and 4) to demonstrate system usability. With that established, the results section will now look at the effectiveness of the system in comparison with other known standards and systems. This section will be broken up into 4 parts:

- **Sensor Testing** – will look at sensor performance and calibration.
- **The model testing** – will look at the performance of the FL systems and ML model.
- **The MA testing** – will explore the effectiveness of the MA designed.
- **Overall system effectiveness** – will examine the performance of the system as a combined entity as well as compare the system to others found in the commercial sector and literature.

8.1. Sensor Testing

The sensor layer testing will involve determining the effectiveness, accuracy and precision of the connected sensors to ensure that the data entering the system is reliable and accurate. The section will start off by looking at the response times of sensors and will proceed to determine how accurate the sensors are in comparison to calibrated equipment or known standards. Repeatability tests will be carried out to ensure that the sensor results are consistent. Three patients were used to carry out the testing. Their profiles are shown in Table 8-1. Each patient has varying attributes from age to pre-existing conditions and exercise regime.

Table 8-1: Profiles of Participants in Study

Profile Info	Patient 1	Patient 2	Patient 3
Gender	Female	Female	Male
Age	59	33	60
Medical History	None	None	Heart Disease
Exercise Regime	None	Active	Moderately Active
Smoking Status	Non-Smoker	Non-Smoker	Smoker

8.1.1. Temperature Sensor

The section looks at the LM35 temperature sensor. The sensor performance will be evaluated by first looking at the sensor response times to determine if the sensor is suitable for the given application. A comparison with a calibrated temperature sensor will then be done to establish a

correlation to improve the sensitivity of the LM35 readings. Thereafter a cost evaluation will be done to show how the LM35 compares to COTS products.

8.1.1.1. Temperature Sensor Response Times

To determine the LM35 temperature response time, the 3 mentioned participants in Table 8-1 carried out tests determine how long it takes for their temperature reading to reach stability. As can be seen from Table 8-2 and Figure 8-1, all patients took about 8 seconds to reach stable readings. It should be noted that these stable temperature readings highlight the lack of sensitivity of the LM35 temperature sensor for body temperature measurements. Expected normal body temperature for normal status patients should reach stabilization values of between 36.1 °C to 37.2 °C which is not seen in Figure 8-1 for any of the 3 healthy patients. The LM35 temperature sensor was therefore calibrated in chapter 8.1.1.2 to fix these inaccuracies. However, what can be seen in the profiles in Figure 8-1 is that with each patient the same trend is observed i.e. the eventual outcome was the same in terms of stability time. At around 8 seconds, the graph starts to flat line indicating a stable value has been reached. Considering that the LM35 is a low cost sensor, this response time is promising. Although the sensor could possibly take longer to reach extreme highs and lows, for the typical range of body temperatures – the stability time is reduced as the temperature is already at room temperature. This is due to sensors never being switched off in the WBAN. This reduces the response time and improves the system performance.

Table 8-2: LM35 Response Times

Time in seconds	T (°C) (Patient 1)	T (°C) (Patient 2)	T (°C) (Patient 3)
1	29.8	29.8	29.8
2	30.3	30.3	30.3
3	30.8	30.8	30.8
4	30.8	30.8	30.8
5	31.7	30.8	30.8
6	31.7	31.3	30.8
7	32.2	31.3	31.3
8	32.7	31.7	31.7
9	32.7	31.7	31.7
10	32.7	31.7	31.7

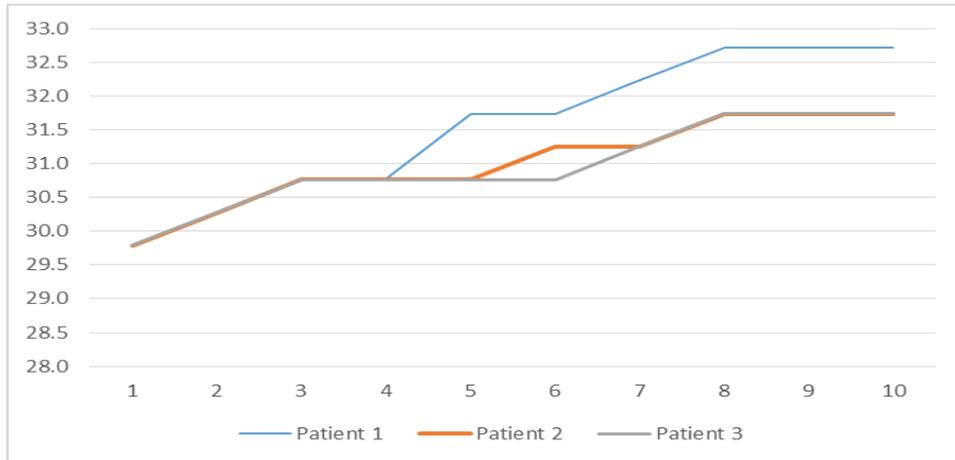


Figure 8-1: Graph of Temperature over time (seconds) for the LM35 Temperature Sensor

8.1.1.2. Temperature Sensor vs Calibrated Sensor

Body temperature readings were taken for the 3 patients at different intervals over several days. The Information was used to test repeatability as well as to compare the LM35 to commercial body temperature thermometers. The recorded results can be seen in Table 8-3.

Table 8-3: Body Temperature Readings of LM35 and Calibrated Sensor

Runs	Patient 1		Patient 2		Patient 3	
	LM35 Temp (°C)	Calibrated Thermometer (°C)	LM35 Temp (°C)	Calibrated Thermometer (°C)	LM35 Temp (°C)	Calibrated Thermometer (°C)
Run 1	32.7	35.9	30.8	36.2	30.8	35.8
Run 2	30.7	36.1	31.7	36.0	30.8	36.2
Run 3	31.2	36.3	32.7	36.1	30.8	36.2
Run 4	31.7	36.1	31.7	35.8	31.7	36.0
Run 5	30.8	36.2	28.3	35.9	32.7	36.2
Run 6	31.7	36.2	27.8	36.1	30.3	35.7
Run 7	29.3	36.1	27.3	35.7	30.3	35.6
Run 8	28.8	35.9	30.8	35.8	30.8	35.7
Run 9	29.3	36.1	29.8	35.9	28.3	36.0
Run 10	27.8	36.2	29.3	36.0	28.8	36.0

A 3D scatter plot of the calibrated sensor values (Figure 8-2), shows that most patients have BT temperature readings in the range of 36 °C (points clustered in this region) which is indicative of normal BT. The scatter plot also shows that the 3 patients share similar BT temperatures

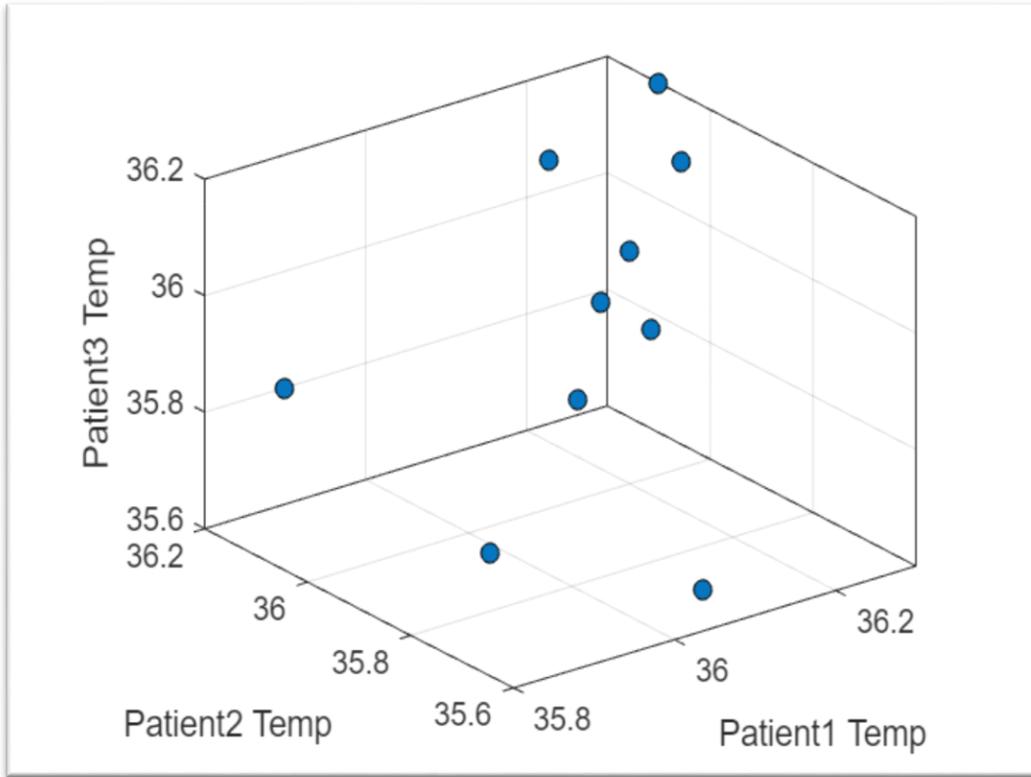


Figure 8-2: 3D Scatter Plot of Each Patient's BT Using a Calibrated Temperature Sensor

Table 8-3 above shows that the LM35 temperature sensor readings using the datasheet equation is not sensitive enough to be used for BT measurements, as it does not result in reliable enough readings. It was therefore decided to calibrate the LM35 sensor against a commercial thermometer to get a line of best fit. This was done using average ADC readings at various thermometer values (Table 8-4).

Table 8-4: ADC output Values and Thermometer Readings for Calibration of LM35

Average ADC output	Thermometer reading (°C)
56	35.7
58	35.9
59	35.9
60	36.1
63	36.2
64	36.2

The straight line equation (equation 8.1) depicts the established relationship between the input ADC signals from the LM35 temperature on the response variable i.e. the calibrated sensor BT value.

$$y = 0.0702x + 31.8 \dots\dots\dots (8.1)$$

Figure 8-3 with R² value of 0.96 was therefore used to calculate the temperature output from the sensor. The R² value is very close to 1 which indicates that the regression model fits the observed data well. 96 % of the variance of body temperature is explained by the variance of the ADC output. As can be seen the line fits the data well, as an equal amount of points lie on both sides of the straight line – this indicates no overfitting or under fitting of the data is taking place and also indicates that there is no bias and variance present. It can also be seen from Figure 8-3 that there is a positive correlation i.e. a positive slope between the body temperature and ADC output.

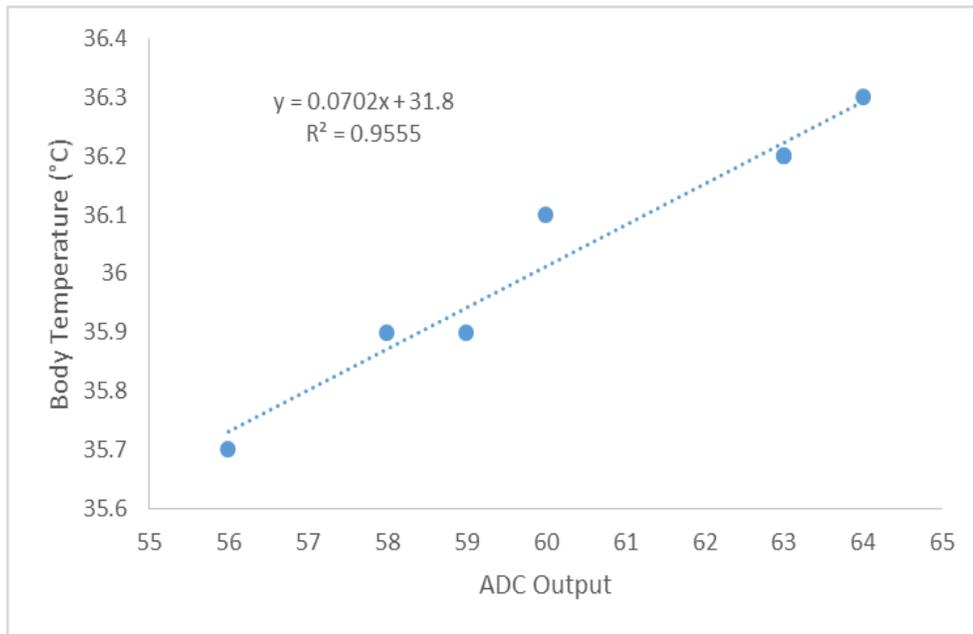


Figure 8-3: Calibration Curve for LM35 Body Temperature Sensor

Table 8-5 shows various readings of body temperature calculated using the new straight line equation and compares this with the calibrated thermometer. The statistical results after comparison can be seen in Table 8-6.

Table 8-5: LM35 Sensor Output Using Straight Line Equation vs Calibrated Sensor

Reading	Actual (LM35 with straight line equation)	Calibrated Sensor	Percent Error (%)	Standard deviation
1	36.6	35.9	1.9	0.5
2	36.2	35.6	1.7	0.4
3	36.3	36.2	0.3	0.1
4	36.4	36.0	1.1	0.3
5	36.6	36.1	1.4	0.4
6	36.4	35.8	1.7	0.4
7	36.3	35.8	1.4	0.4
8	36.5	36.2	0.8	0.2
9	36.3	35.8	1.4	0.4
10	36.3	36.2	0.3	0.1

Table 8-6: Statistics for Comparison of LM35 with Calibrated Temperature Sensor

Average	36.4
Sample Size	10
MAE %	1.2
Average Standard Deviation	0.4

Figure 8-4 shows a comparison between the LM35 body temperature sensor values (random samples from each patient) using the new equation developed vs the commercial body temperature sensor values. Table 8-6 shows there is a limited error between actual and expected values of 1.2 % which is less than the target of 5 %. In addition, the standard deviation is extremely low at 0.4 which indicates that the data is reliable and accurate. In Figure 8-4, the blue line depicted by the LM35 sensor has values slightly above the calibrated sensor values with random errors between readings.

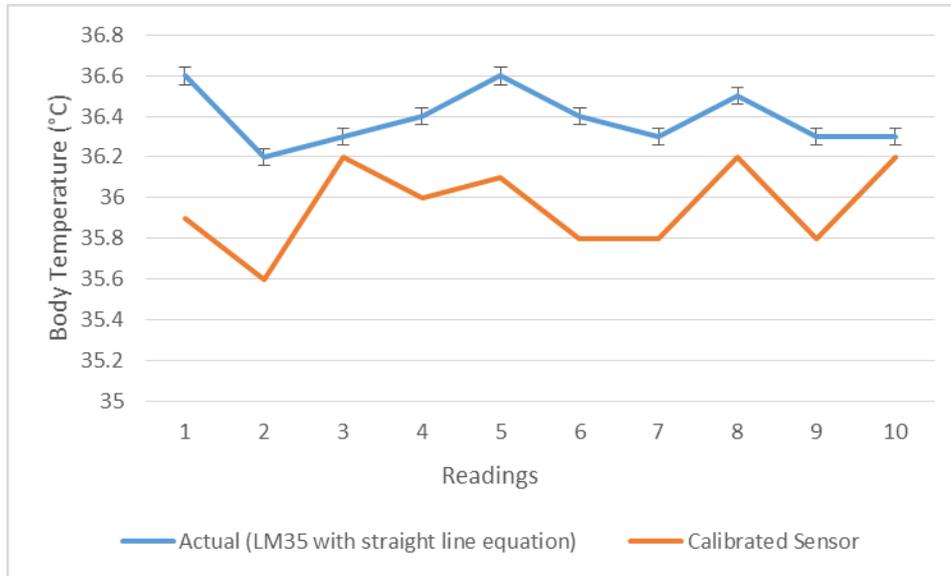


Figure 8-4: Temperature Output with Straight Line Equation vs Calibrated Sensor

Cost comparisons:

Table 8-7 and Figure 8-5 shows how the LM35 weighs up against other body temperature measuring devices in the market in terms of cost. As can be seen, there is a price difference in the range of 88 % - 98 %, which makes the LM35 an extremely affordable solution to body temperature monitoring. This is in addition to the fact that the commercial devices in the market are often difficult to carry around and much larger in size making them more obtrusive and uncomfortable to use. The LM35 however, is small, compact and easy to use when integrated with the WBAN. The *Beurer FT 70* Multi-functional thermometer is the most expensive at R1299 with the second cheapest being the *Beurer FT 09/1* Digital Fever Thermometer. The LM35 (represented as the current system) comes out on top at only R30. This sensor is cheap, easily replaceable and durable for the application

Table 8-7: Comparison in Price between Designed System and Commercial Products

Current System	Beurer FT 70 Multi-Functional Thermometer	Beurer FT 09/1 Digital Fever Thermometer FT 09/1 - White	Medic Thermometer B/o IR
R30	R1299	R250	R699

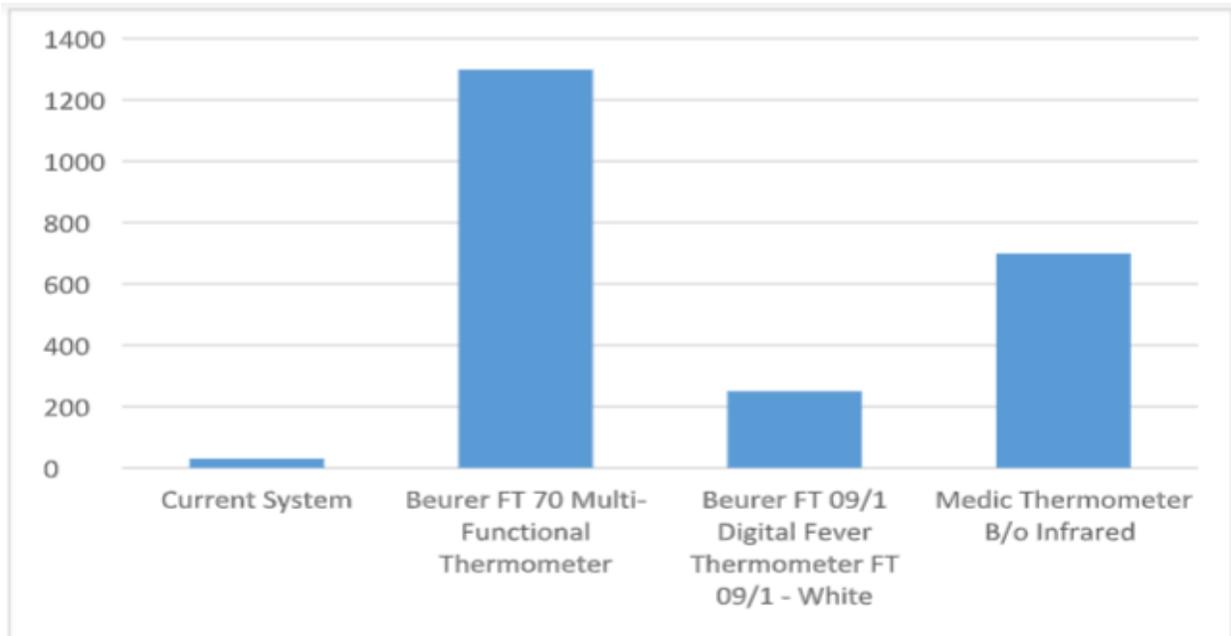


Figure 8-5: Graph Showing Price (Rands) of Current System vs Commercial Products

8.1.2. Humidity Sensor

The section looks at the HIH-4000 humidity sensor. The sensor performance will be evaluated by first looking at the sensor response times to determine if the sensor is suitable for the given application. A comparison with weather reports will then be done to establish the effectiveness of the sensor. Thereafter a cost evaluation will show how the HIH-4000 compares to COTS products.

8.1.2.1. Humidity Sensor Response Times

The response time for the HIH-4000 humidity sensor was obtained by running a test with steaming water. A starting point of 70 % humidity was chosen for each run which was the room humidity at the time. The sensor will always be at room humidity since it is never switched over. Then the time taken for each sample to reach 100 % humidity was measured. The results (Table 8-8 and Figure 8-6) shows that for run 1 and run 3, it takes the sensor approximately 5 seconds to reach 100 % humidity from the reference point while run 2, took around 7 seconds. This indicates a fairly rapid response time and validates the performance of the sensor. Since 2 runs reached stability much faster at around 5 seconds – it can be seen that the sensor is capable of performing better under most circumstances. Figure 8-6 also highlights the point that at higher humidity, the sensor will be able to stabilize much rapidly i.e. above 80 % humidity. While at lower humidity, the response time may be slower.

Table 8-8: Table of Response Times for the Humidity Sensor

Time in seconds	% Humidity (Run 1)	% Humidity (Run 2)	% Humidity (Run 3)
1	70.3	70.3	70.5
2	81.0	71.9	74.6
3	88.9	76.9	80.6
4	95.7	84.85	83.2
5	100	91.3	100.0
6	100	97.1	100.0
7	100	100	100.0
8	100	100	100.0

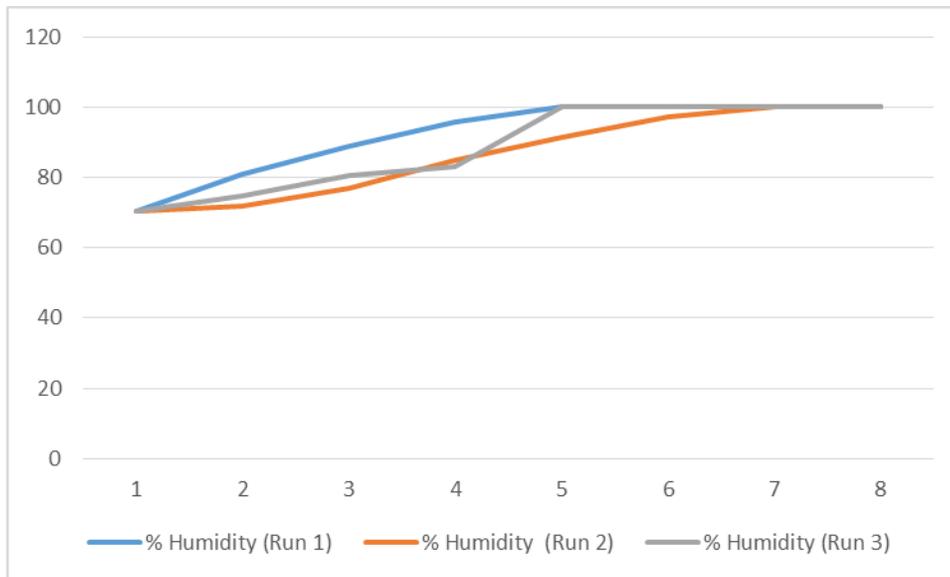


Figure 8-6: Graph of Humidity % over Time for the HIH-4000 Sensor

8.1.2.2. Humidity Sensor vs Calibrated Sensors

The comparison of the humidity sensor was done with weather report data. HIH-4000 readings were taken and compared with weather humidity readings for the same area. This was done for different time intervals and different days. As can be seen in Table 8-9 and Table 8-10, the RH of the HIH-4000 had a small error difference from the weather reports i.e. average % error of 3.3 which was lower than the target of 5%. Table 8-10 also shows that the standard deviation is extremely low at 0.04 which indicates high accuracy and reliability of the data. The slight deviations could be as a result of neglecting the effect of temperature on the true RH. In addition, the weather forecast gives an approximate RH which may fluctuate.

Table 8-9: Humidity Sensor Readings Compared to Weather

Humidity Sensor	Comparison		% Error	Standard Deviation
	Actual	Weather Forecast		
Run 1	62.0%	61.3%	1.1	0.00
Run 2	66.7%	69.0%	3.3	0.02
Run 3	76.0%	79.0%	3.8	0.02
Run 4	78.1%	82.0%	4.8	0.03
Run 5	86.3%	90.0%	4.1	0.03
Run 6	88.8%	92.0%	3.5	0.02
Run 7	52.0%	55.0%	5.5	0.02
Run 8	47.8%	50.0%	4.4	0.02
Run 9	63.0%	64.0%	1.6	0.01
Run 10	77.4%	78.0%	0.8	0.00

Table 8-10: Statistics for the Comparison of the Humidity Sensor with Weather Forecasts

Average	0.70
Sample Size	10.00
MAE (%)	3.28
Average std Deviation	0.04

Figure 8-7 graphically depicts a smaller deviation as compared to Figure 8-4 for the LM35 temperature sensor. However the average error between the LM35 and calibrated sensor was only an average of 1.2%. This can be attributed to the fact that the LM35 range is confined to the normal range of body temperatures which doesn't have as wide a range as the expected humidity sensor output. It is also important to note that the LM35 % error should be much smaller than the HIH-4000 as it is much more critical to know the effect of each degree change on BT than it is to know each % decimal change in humidity.

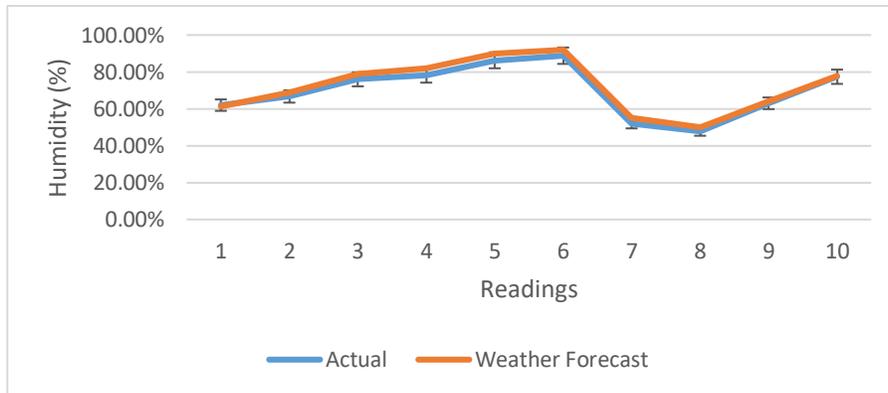


Figure 8-7: RH % readings for the HIH-4000 vs Weather Reports for the Same Area

8.1.3. Glucose Sensor

The Glucose readings of 3 patients were taken at different times of the day for multiple days (Table 8-11). Readings were taken at 3 intervals. The first being “before eating” which means a point when glucose levels should be lower. “During the day” when glucose levels is expected to be normal. Thereafter after eating when a spike in glucose levels is expected. The results tested repeatability and also helped to establish the glucose curve to calibrate the IR sensor.

Table 8-11: Glucose Readings for 3 Patients at Different Times of the Day

IR Sensor	Patient 3		Patient 1		Patient 2	
	Actual (mg/dl)	Calibrated (mg/dl)	Actual (mg/dl)	Calibrated (mg/dL)	Actual (mg/dl)	Calibrated (mg/dl)
After Waking/Before eating						
Run 1	997	46.8	994	36.0	992	30.6
Run 2	1007	77.4	1008	77.4	996	46.8
Run 3	994	43.2	992	37.8	992	32.4
During Day						
Run 1	996	50.4	1003	64.8	998	61.2
Run 2	1000	54.0	1003	68.4	997	52.2
Run 3	997	61.2	1001	57.6	1002	59.4
After Eating						
Run 6	1016	106.2	1016	117.0	1003	66.6
Run 7	1010	99.0	1001	70.2	996	50.4
Run 8	1007	88.2	1013	104.4	1003	66.6

A 3D scatterplot of the calibrated glucose readings from the commercial sensor is shown in Figure 8-8. As expected, after eating and during the day – the glucose readings of patients were usually in the high or normal range. With levels dropping when the patients hadn’t eaten. After eating, glucose levels would sometimes peak above 100 mg/dl while before eating to lows of 30 mg/dl.

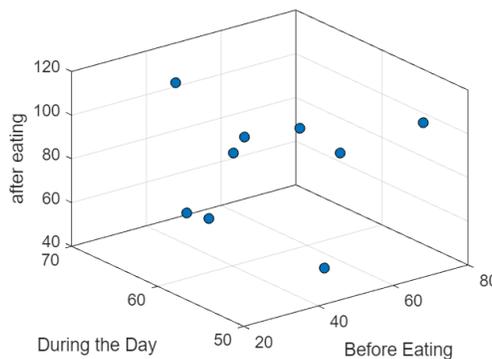


Figure 8-8: 3D Surface Plot of Glucose Readings at Different Intervals

The results in Table 8-11 further shows that as expected for the 3 participants, that the glucose levels after waking or when the patients were hungry, resulted in low blood glucose concentrations yet not in the norm of 72-126 mg/dl. The recommended range around 3 hours after eating should be less than 140.4 mg/dl which was the case for all test participants. During the day, all test participants usually had normal glucose levels, with the exception of times when participants did not eat enough in which case glucose levels dropped to lower ranges.

Using the ADC readings from the IR sensor at different calibrated glucometer values, Table 8-12 of ADC output vs glucose levels (mg/dl) was used to obtain a calibration plot and equation for the IR sensor. The equation for the straight line equation used to calibrate the IR sensor is shown below:

$$y = 3.24x - 3.1 \times 10^3 \dots\dots\dots (8.2)$$

Table 8-12: Glucose Sensor Calibration Table

ADC Output	Blood Glucose Levels (mg/DL)
992	30.6
994	36.0
998	61.2
1003	64.8
1003	66.6
1001	70.2

The developed graph of a straight line equation with R² value of 0.95 as seen in Figure 8-9. The graph shows a positive gradient, which means that as the ADC values from the IR sensor increases, the glucose levels in mg/dl increases as well. The R² value of 0.95 is very close to 1 and indicates a good fit for the data points.

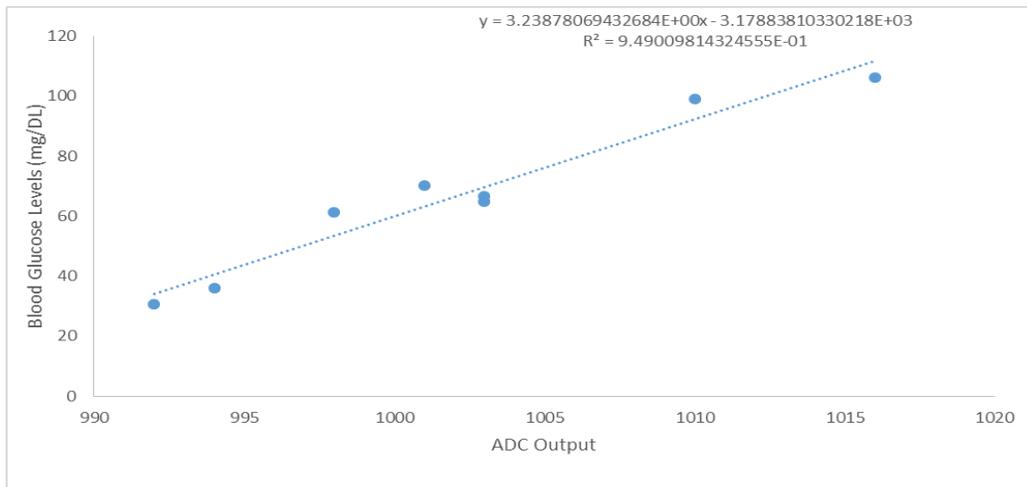


Figure 8-9: Blood Glucose Levels Calibration Fit Using Commercial Sensor

Random readings from each patient was then taken and plugged into the calibrated equation to get the glucose readings. These readings were then compared with the calibrated sensor as shown in Table 8-13.

Table 8-13: IR Glucose Readings Using Straight Line Equation Compared to Calibrated

Reading	Actual (IR sensor with Straight Line equation)	Calibrated Sensor	Percent Error (%)	Standard deviation
1	34.03	30.6	11.2	2.4
2	37.27	36	3.5	0.9
3	46.99	46.8	0.4	0.1
4	50.23	50.4	0.3	0.1
5	56.70	61.2	7.3	3.2
6	69.66	64.8	7.5	3.4
7	63.18	66.6	5.1	2.4
8	69.66	70.2	0.8	0.4
9	92.33	99	6.7	4.7
10	105.29	106.2	0.9	0.6

Table 8-14: Statistics for IR Sensor Compared to Commercial Sensor

Average	62.5
Sample Size	10
MAE (%)	4.4
Average std Deviation	2.4

As can be seen in Figure 8-10 the difference between the observed and predicted values are small and unbiased which indicates that the regression model fits the observed data well. Average standard deviation of 2.4 and MAE of 4.4% is shown in Table 8-14. These statistics are low which means points are clustered around the mean and this indicates that the data is more reliable

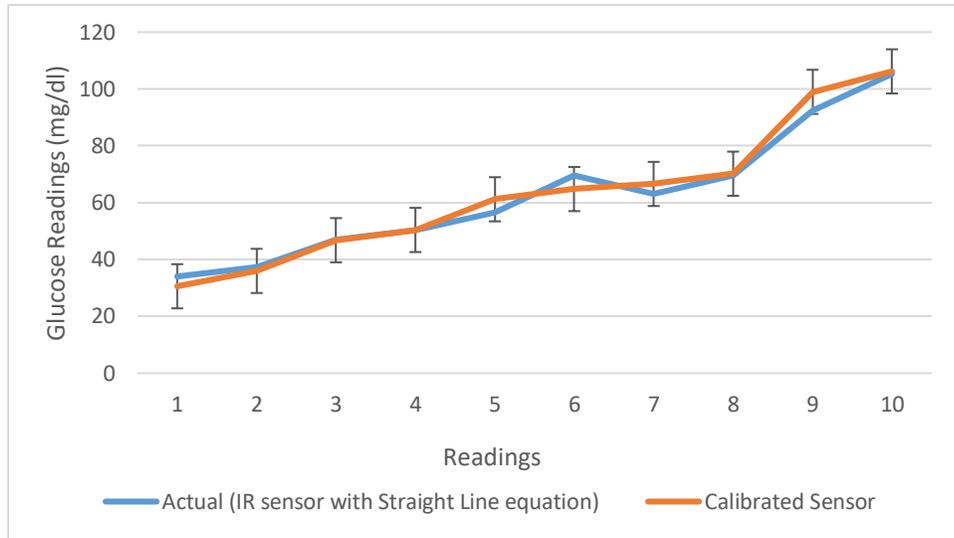


Figure 8-10: IR Sensor Output Relative to Commercial Sensor

The Clarke error grid analysis (Figure 8-11) was further used to validate accuracy of the glucose sensor system. This analysis method is used to validate the effectiveness and suitability of using commercial glucose sensors by comparing it to a reference reading (Daarani & Kavithamani, n.d.). In this case, the obtained values from the sensor reading will be compared to the commercial glucometer purchased.

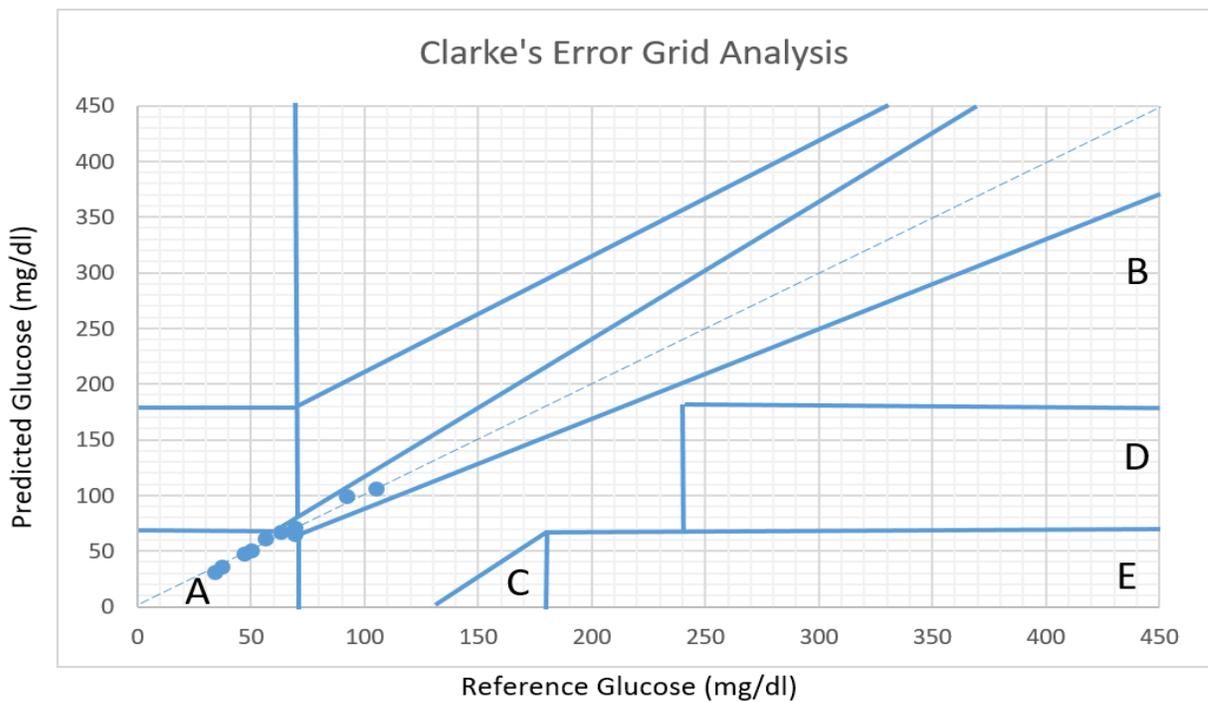


Figure 8-11: Clarke Error Grid Analysis

Table 8-15, shows the acceptable ranges of glucose readings relative to the reference. Points that fall in range A and B are acceptable, however if points in region C, D and E are in excess – it means that the glucose sensing device is not reliable for medical use. As seen in Figure 8-11 all points measured lie in the region A – which indicates that the sensor is accurate enough to be used for gauging blood glucose levels in patients. Region A, as described in Table 8-15 indicates that all measured glucose levels using the IR sensor is within 20% of the reference value (commercial sensor).

Table 8-15: Regions of Clarke Error Grid

Region A	Values within 20% of the reference value
Region B	Values are outside of 20% but would not lead to inappropriate treatment
Region C	Values leading to unnecessary treatment
Region D	Values indicating a potentially dangerous failure to detect hypoglycemia or hyperglycemia
Region E	Values that would confuse treatment of hypoglycemia or hyperglycemia and vice versa

Cost comparisons:

Table 8-16 shows how the IR costs of the prototype system designed compares to other IR based sensors on the commercial market. As seen in Table 8-16 and Figure 8-12, there is an 82 % - 92 % reduction in price range with the compared devices, making the designed solution far superior in terms of cost savings. This is in addition to the fact that the commercial devices in the market are often difficult to carry around and are much larger in size making them more obtrusive and uncomfortable to use. The current system (refers to the developed system), is only R200 while all other sensors are at least above R1000. Both Figures 8-5 and 8-12 therefore indicate the massive discrepancies between the costs of commercial products and the system designed – indicating a massive opportunity to reduce costs of wearable sensors while still maintaining sensor accuracy.

Table 8-16: Table Showing Cost of Glucose Monitoring Solution vs Commercial Products

Current System	Beurer GL 50 Blood Glucose Monitor	Abbott Freestyle Libre Blood Sugar/ Glucose Monitor	Accu-Answer Cholesterol, Glucose, Hemoglobin & Uric Acid Blood Test Meter
R200	R2500	R2199	R1099

The *Beurer GL 50 70* blood glucose monitor is the most expensive at R1299 with the second cheapest being the *Accu – Answer* system at R1099. The IR sensor (represented as the current system) comes out on top at only R200. This sensor is cheap, easily replaceable and durable for the application

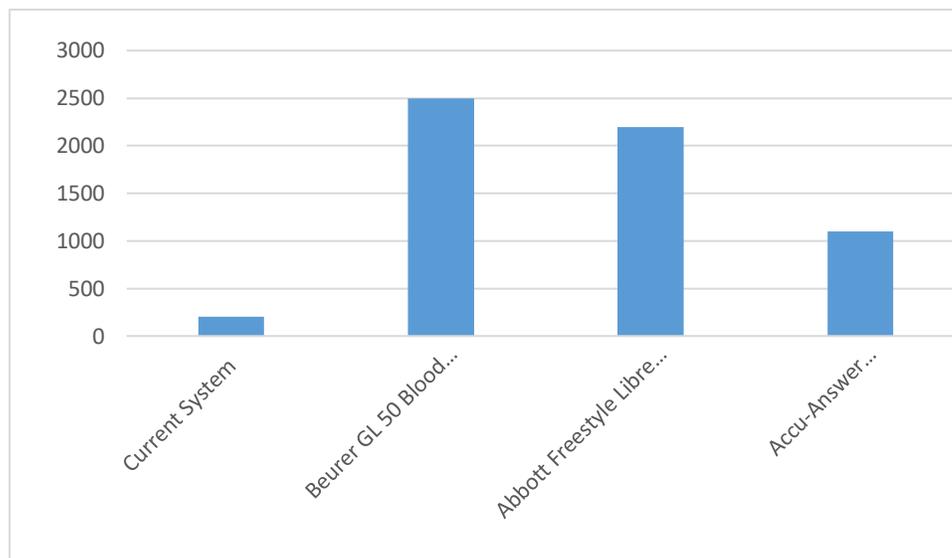


Figure 8-12: Cost Comparison (Rands) of IR System with Other Commercial Products

8.1.4. Pulse Rate Sensor

The comparison for the pulse rate sensor was done against a commercial pulse oximeter readings. The value obtained from the pulse rate sensor at intervals before and after exercise for the 3 patients were recorded in Table 8-17.

Table 8-17: Pulse Rate Sensor and Calibrated Sensor Values for different patients

Pulse Rate Sensor	Patient 3		Patient 2		Patient 1	
	Actual (BPM)	Physical measurement (BPM)	Actual (BPM)	Physical measurement (BPM)	Actual (BPM)	Physical measurement (BPM)
Before Exercise						
Run 1	72	73	78	74	66	64
Run 2	76	78	78	76	72	72
Run 3	74	72	96	92	66	69
After Exercise						
Run 4	96	100	128	136	96	86
Run 5	112	116	132	126	132	129
Run 6	123	119	126	119	126	130

As expected, pulse rates before exercise were lower than after exercise. Normal range before exercise (60 BPM – 100 BPM) was experienced by all participants. After exercise, all participants did experience higher pulse rates, while patient 3 and patient 1 – due their age sometimes saw pulse rates dip to the lower ranges of 86 BPM - 90 BPM which is expected for individuals in the age range of 50-60 years.

Readings of each of the patients were then compared to a calibrated pulse rate sensor and the results can be shown in Table 8-18. Table 8-19 shows that the % error is low at 2.7% which is well below the target of 5 %. In addition, a low standard deviation of 1.86 indicates that the data is reliable. The sensor readings sometimes results in deviations because of noise interference. This can be improved through placing the sensor in an enclosure to block out noise interferences.

Table 8-18: Pulse Rate Sensor comparison with calibrated sensor

Reading	Actual (IR sensor with polynomial equation)	Calibrated Sensor	Percent Error	Standard deviation
1	72	73	1.4	0.7
2	76	78	2.6	1.4
3	74	72	2.8	1.4
4	74	72	2.8	1.4
5	96	100	4.0	2.8
6	112	116	3.4	2.8
7	123	119	3.4	2.8
8	72	73	1.4	0.7
9	76	78	2.6	1.4
10	74	72	2.8	1.4

Table 8-19: Pulse Rate Sensor vs Calibrated Sensor Statistics

Average	84.90
Sample Size	10.00
Average error %	2.70
Average std Deviation	1.86

Figure 8-13 shows the graph with error bars of output IR glucose readings of implemented system, relative to the calibrated sensor values. As can be seen, the actual values are close to the calibrated sensor output.

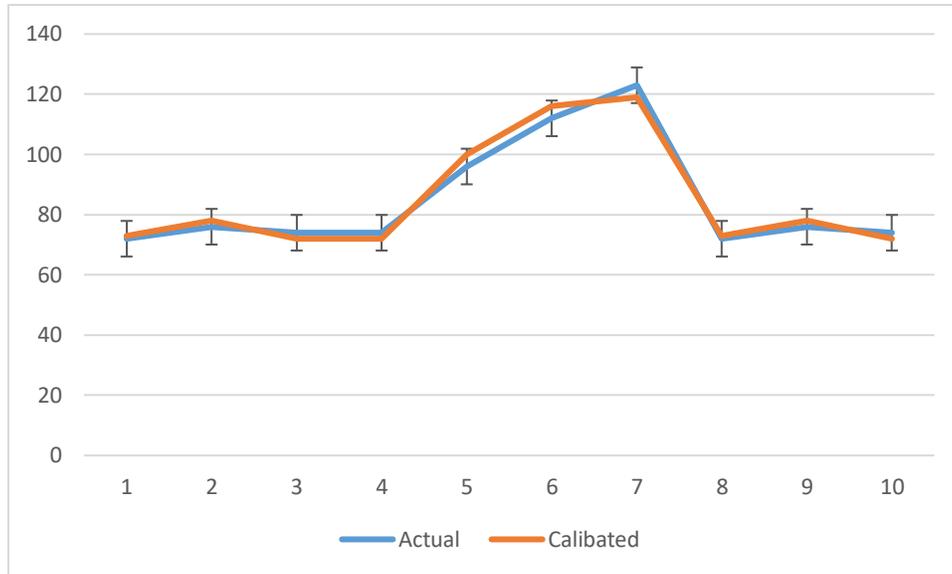


Figure 8-13: Pulse Rate Sensor vs Calibrated Sensor (BPM)

8.2. Model Testing

The sensor layer was tested and showed the effectiveness of the sensors in comparison to known standards and calibrated devices. This section will now independently test the performance of the models employed within the IoT system. This section will look at testing 1) The FL models using synthetic test cases and comparing with the intuition of a physician and also to literature and 2) the stroke ML model in comparison to literature standards.

8.2.1. MATLAB Simulations

After generating the models on Android studio, they were then developed on MATLAB to generate surface plots. This section will outline the process of re-creating the models in MATLAB and will thereafter discuss the surface plots generated.

8.2.1.1. Simulation Process

The simulation process was carried out using MATLAB's Fuzzy Logic toolbox. The first step of the process involves defining the input variables, output variables and inference system. This was done using the fuzzy logic designer screen. For both models 1 and 2, the Mamdani inference system was selected as with the model created for the MA. For each model, the input and output variables were named and the method of defuzzification was selected to be the MoM approach. Figure 8-14 shows the set-up for model 1. Model 2 followed a similar procedure with different input and output variables defined. As can be seen in Figure 8-14, the input variables pulse rate (PR), blood sugar (BS), body temperature (BT) and age were defined as inputs while the output

was defined as “health status.” Here a higher output of health status means a more high risk to the patient while a lower output indicates a positive status of health.

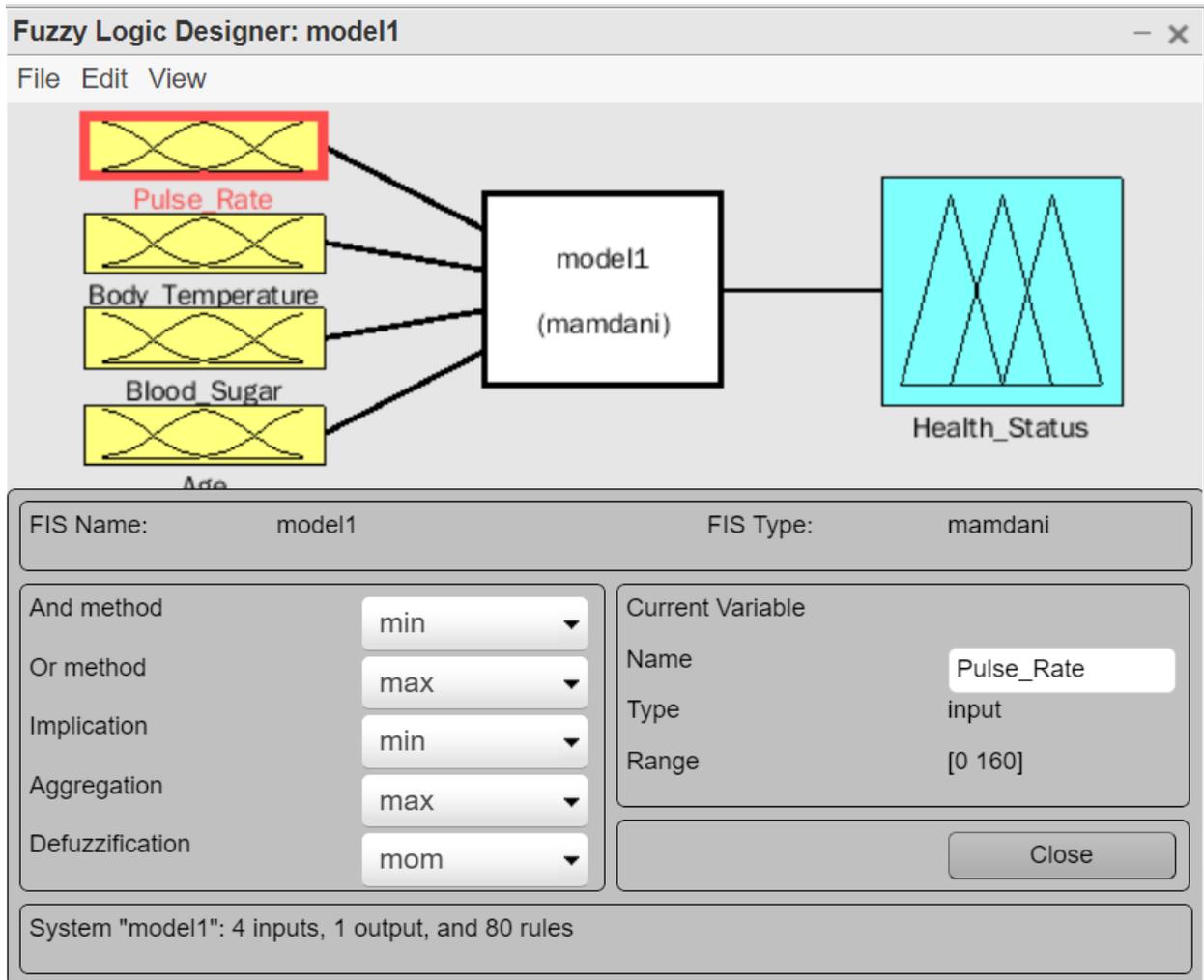


Figure 8-14: FL Designer Screen MATLAB (Model 1)

Once the general set-up was complete, the next step involved defined the fuzzy MFs using MATLAB’s MF editor. For each of the inputs and output of both model 1 and 2, the fuzzy membership functions were defined. This involved selecting the type of MF and its range. For all models and variables, the triangular MF was chosen as was the case for the FES developed for the MA. Figure 8-15 shows a typical MF set-up for the PR variable of model 1. As can be seen, a PR range of 1 BPM- 160 BPM was chosen. Thereafter the operating ranges of low, normal and high pulse rates were defined. The same approach was used for all variables in both models.

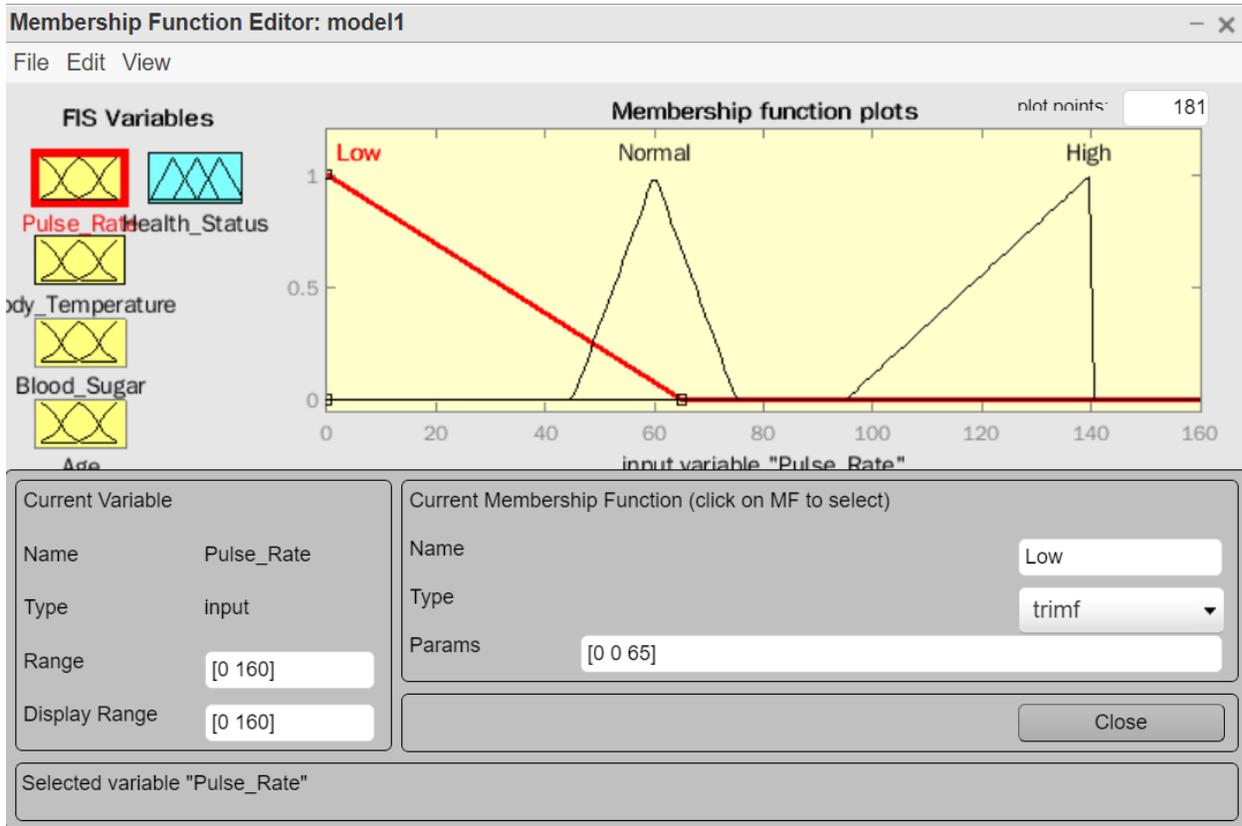


Figure 8-15: Membership Function Editor on MATLAB (PR Variable Model 1)

Once the membership functions were developed, the process of rule generation followed. This involved using MATLAB's rule editor. Here all possible rule combinations based on the set of input variables were used to create a rule base for the model to predict. Figure 8-16 shows the rule editor for model 1. Here, 81 rules were developed by selecting the ranges of each variable and thereafter the health status outcome expected. This rule base will serve as a reference for the model when new input variables are fed into the FES.

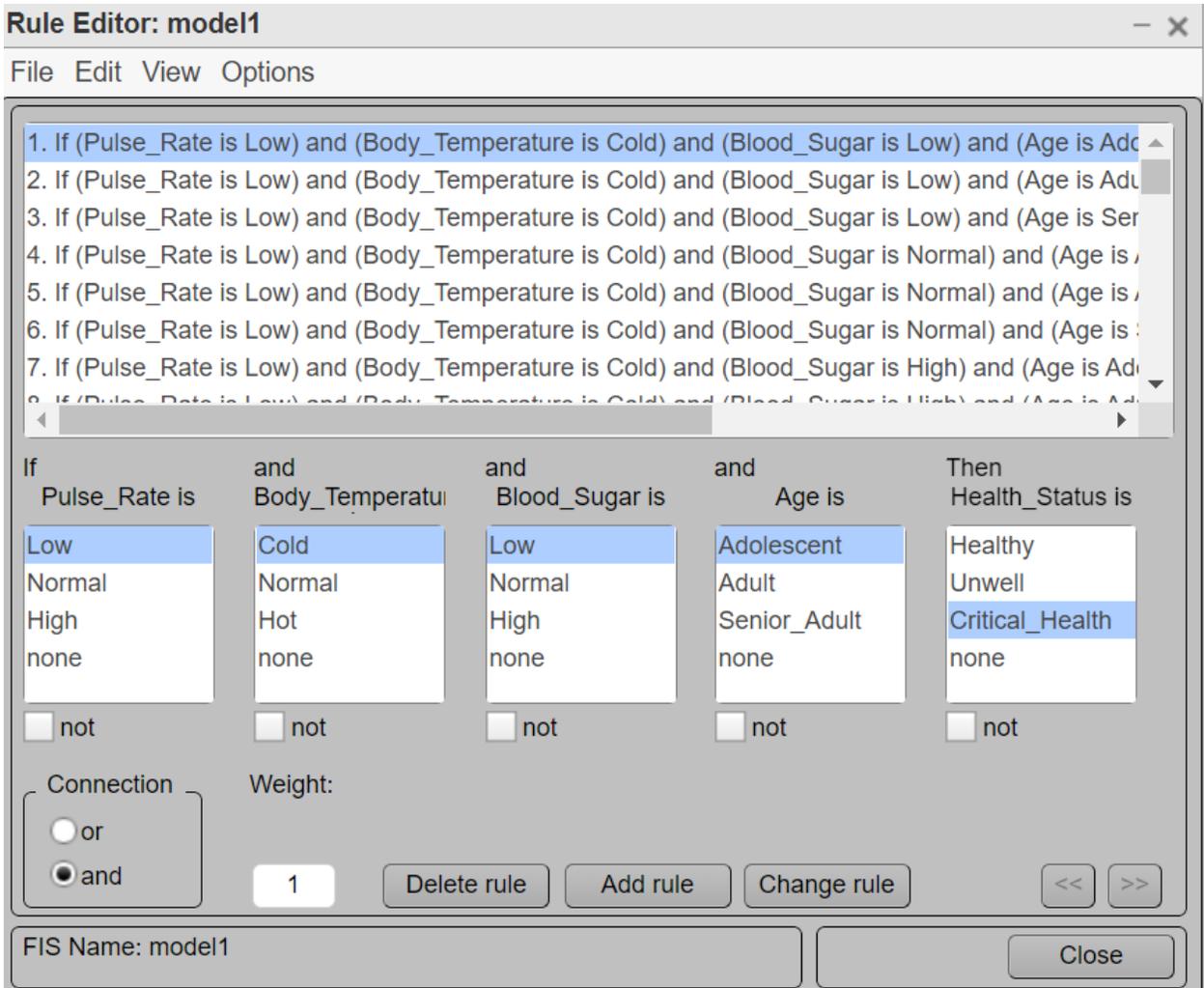


Figure 8-16: MATLAB Rule Generator (Model 1)

Figure 8-17 shows the rule viewer for model 1. By shifting the red line or manually changing input variables, the output of the model will change. This screen can be used to test the effect of the input variables (BS, PR, BT and Age) on the response variable which is the health risk of the patient.

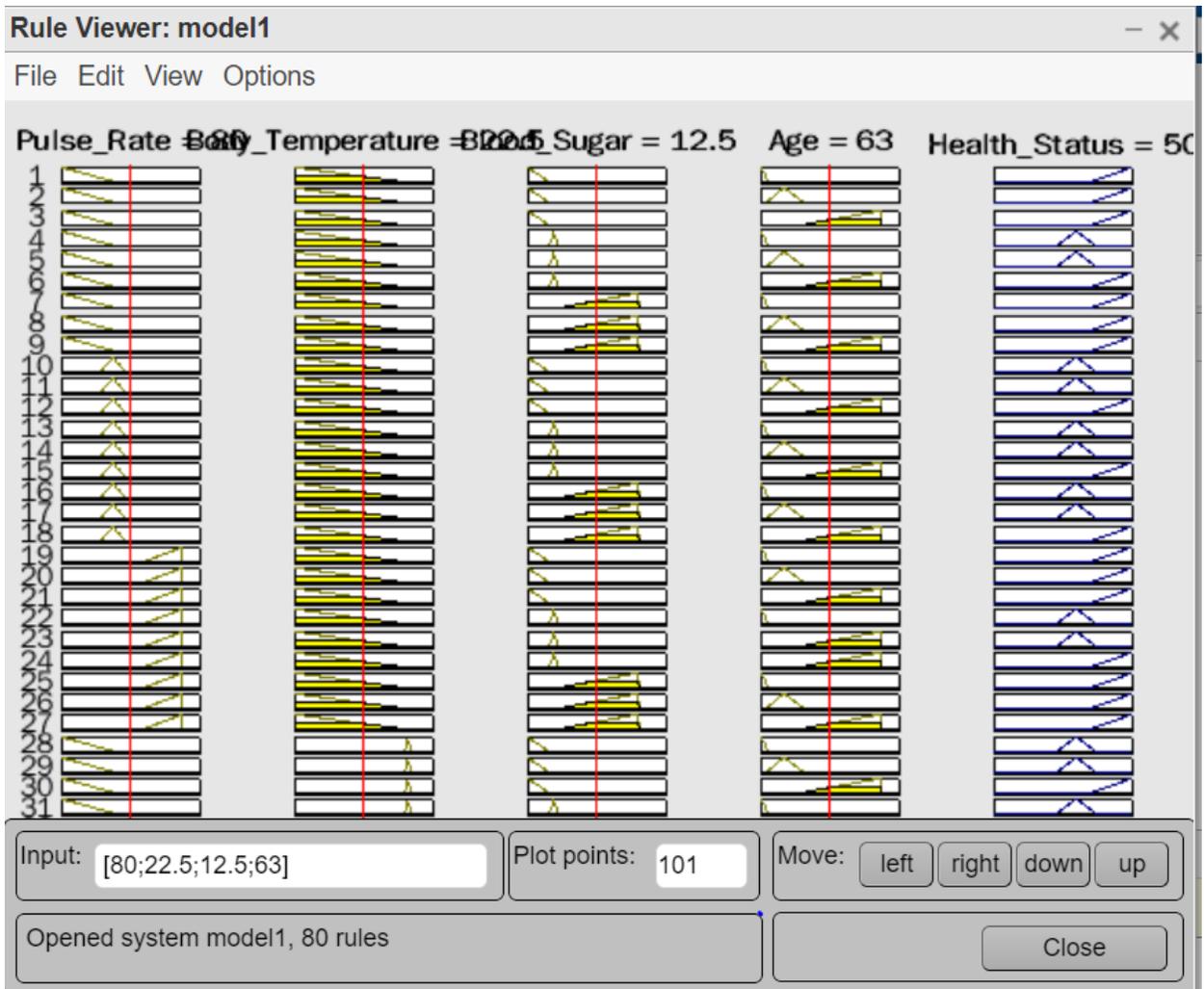


Figure 8-17: MATLAB Rule Generator (Model 1)

8.2.1.2. Surface Plots

Using MATLAB's surface viewer, it was then possible to generate surface plots for the different combination of input variables relative to the output for each of the models. This is the chosen method by MATLAB fuzzy logic suite to display the interaction between membership functions. A surface plot is used to determine the relationship between a response variable and two predictor variables. It consists of predictors on the x and y axis and a continuous surface on the z- axis representing the response variable. The results can be seen below, as mentioned previously – a lower output of the response variable “health status” is desired – as it indicates a patient is healthy and has low risk factors. A higher rating on the health status indicates a danger to the patient.

Model 1:

Surface and contour plots of BT and PR relative to the health status of the patient was plotted. The results can be seen in Figure 8-18. The plots show that at lower temperature patients' health become critical reaching values of over 90 %, however as temperatures reaches 36 °C, the start of the normal body temperature range, there is a dip in patient risk to around 80 %. The change at 36 °C can further be seen in the pseudo color chart on the y-axis with the change of color from yellow to green. After the normal temperature zone passes, lower temperatures tend to increase the health risk back to over 90 %. The plot also shows that the health status risk drops to zero between the normal PR range of 80 BPM - 100 BPM, however risk to patient tends to peak at values below and above this range. This can further be seen in the pseudo- color plot in figures 8-18(b). The blue area on the x-axis denotes the 60 BPM-100 BPM range and >150 BPM range.

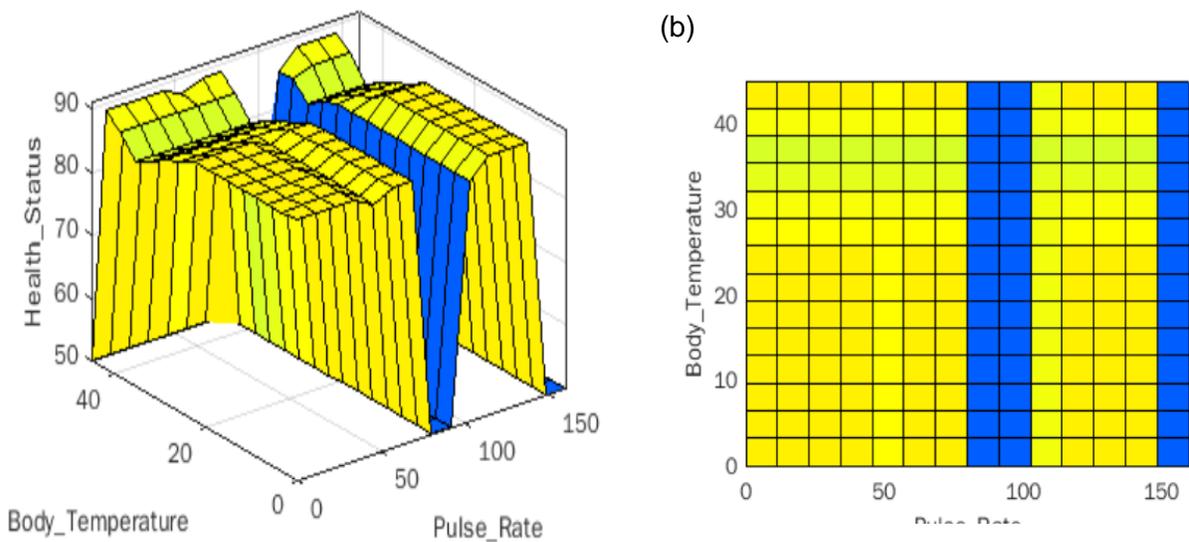


Figure 8-18: (a) Surface Plot BT and PR on Health Status, (b) Contour Plot of BT vs PR

A surface plot and color pseudo-color plot (Figure 8-19) can be used to show the impact of BS and PR on a person's health risk. As can be seen in 8.19, at sugar levels of about 21.7 mmol/L, health risk tends to drop from around 85 % to 50 %. This indicates that high BS poses a lower risk on patient health than low BS. At around 5-5.5 mmol/L, the normal range of BS, there is a slight dip on patient health risk from 90 % to around 87 %, however this is for pulse rates between 100 – 150 BPM.

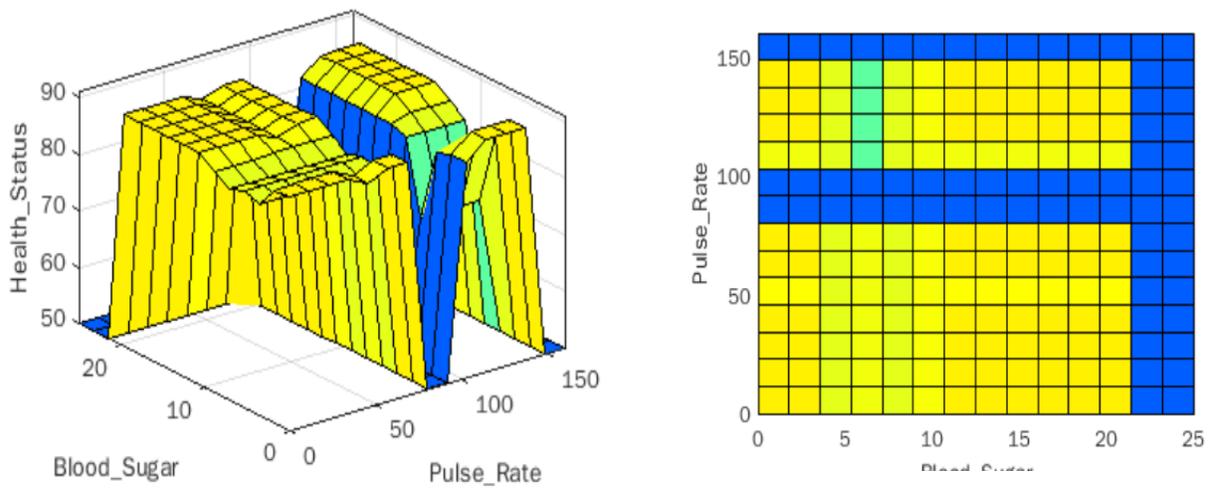


Figure 8-19: (a) Surface Plot BS and PR on Health Status, (b) Contour Plot of BS vs PR

Figure 8-20, shows the effect of age and PR on patient health risk. As can be seen from Figure 8-20(a) and figure 8-20(b) from around 60-100 BPM, there is a drop in health risk from 90 % to 0 %. At 60BPM, the health risk drops to 60 % and then the risk is further dropped to zero when the peak of 80 BPM is reached. Age seems to have minimal effect on the health risk. There is a slight dip from 90 % risk to 89 % for age groups 20-30 and 40-60.

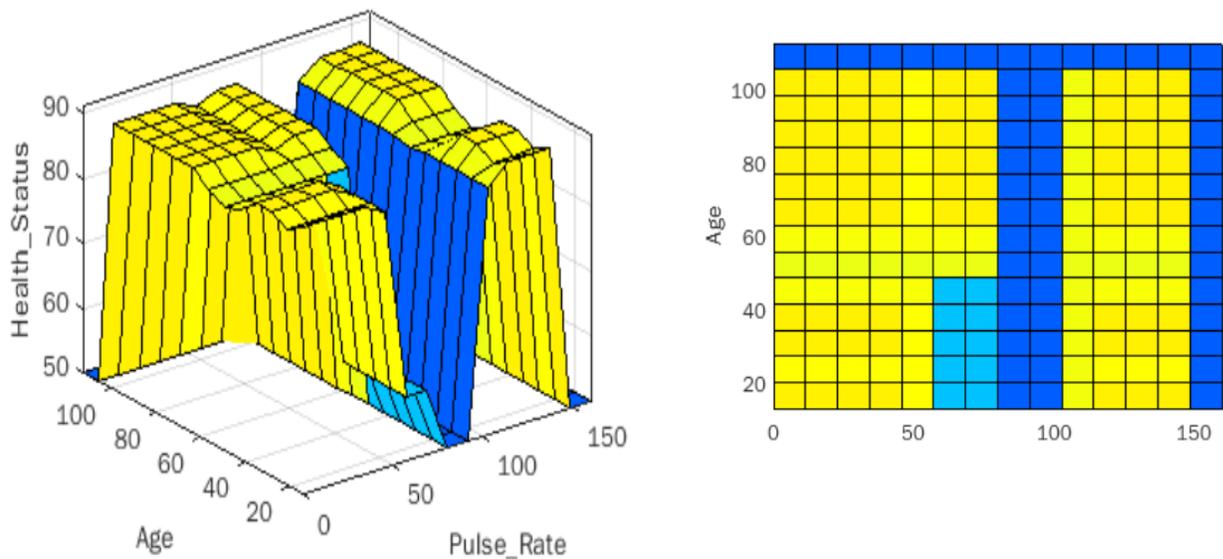


Figure 8-20: (a) Surface Plot Age and PR on Health Status, (b) Contour Plot of Age vs PR

Figure 8-21 shows the relationship between the age and body temperature relative to a patient's health risk. As can be seen, age and body temperature together have a constant effect on health risk – keeping it constant at 50%. The same can be said about the relationship between blood sugar and BT (Figure 8-23) as well as and age and blood sugar (Figure 8-22) which also show a 50% risk in patient health.

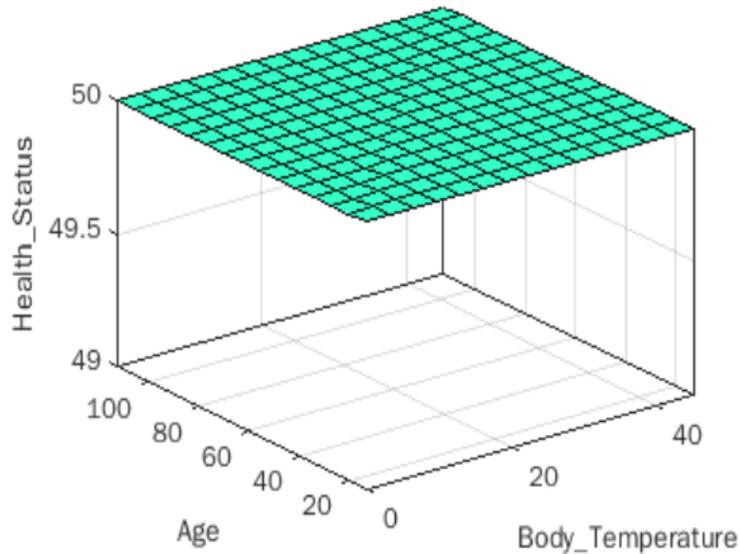


Figure 8-21: Surface Plot Age and BT on Health Status

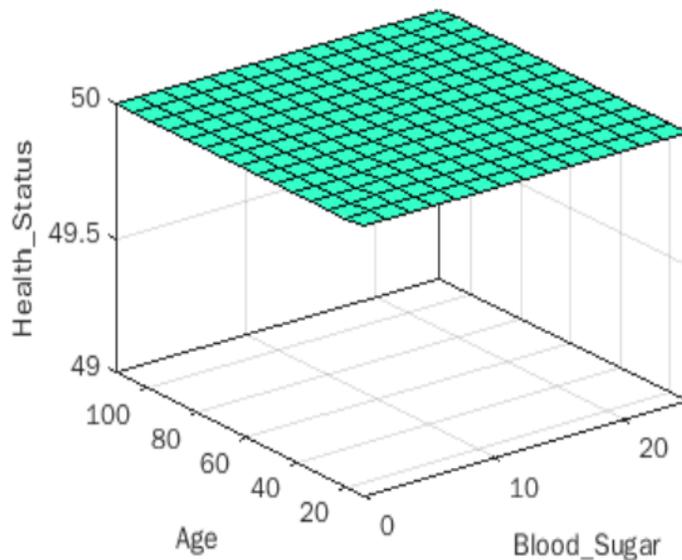


Figure 8-22: Surface Plot Age and BS on Health Status

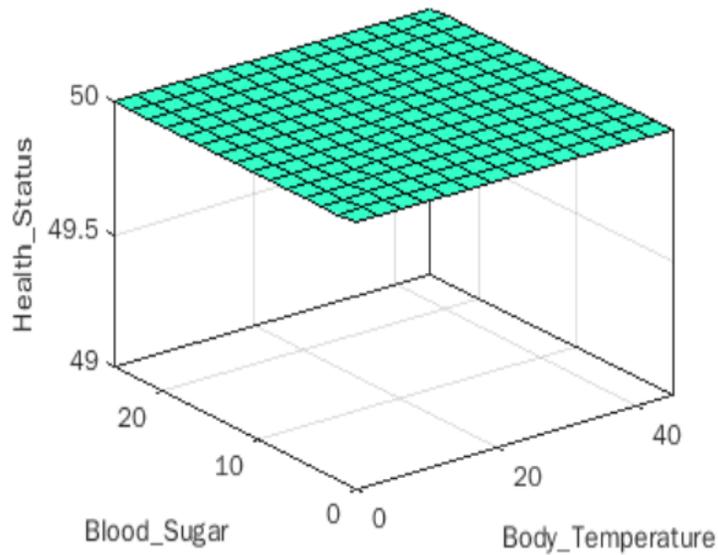


Figure 8-23: Surface Plot BS and BT on Health Status

Model 2:

Figure 8-24 shows the surface plot of humidity and age on the response variable rating for model 2. The graphs shows that at around 40-60 % humidity, the health risk of the patient drops from 90 % to 0 %. This is expected, as this is the normal range of humidity known to have no adverse effects on the patient. Once again, age has almost no effect on the health risk with risk rating dropping from 90 % to 80 % for the 20's age group and 86 % for the 40's age group.

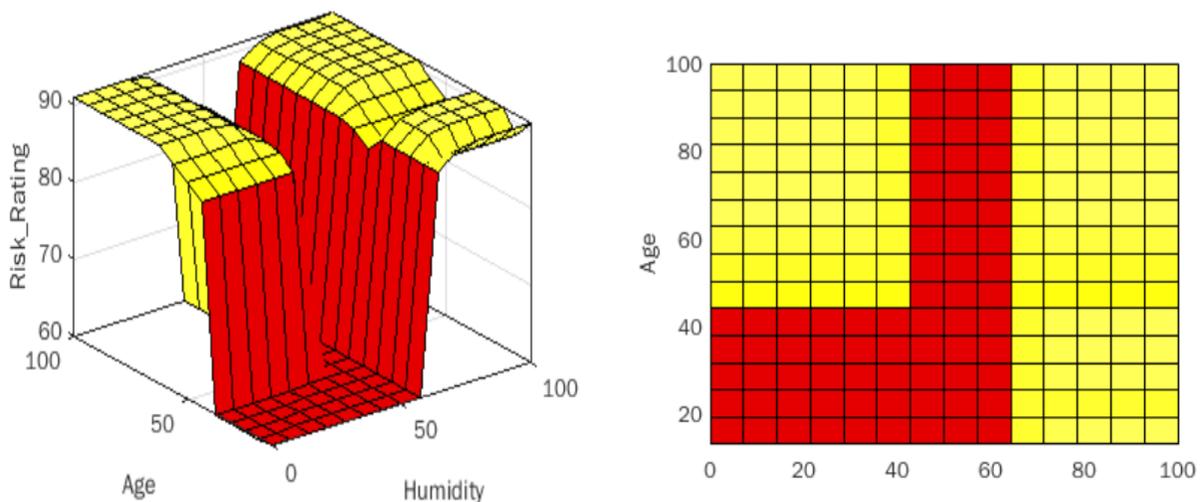


Figure 8-24: (a) Surface Plot Age and Humidity on Risk Rating, (b) Contour Plot

As seen in Figure 8-25, there is a strong relationship between the temperature and humidity on the risk rating of the patient. At around 40-60 % humidity and at temperatures between 0 °C to 40 °C, the risk rating drops to about 60 %. For the rest of the ranges of temperature and humidity, the risk rating remains high at over 80 %.

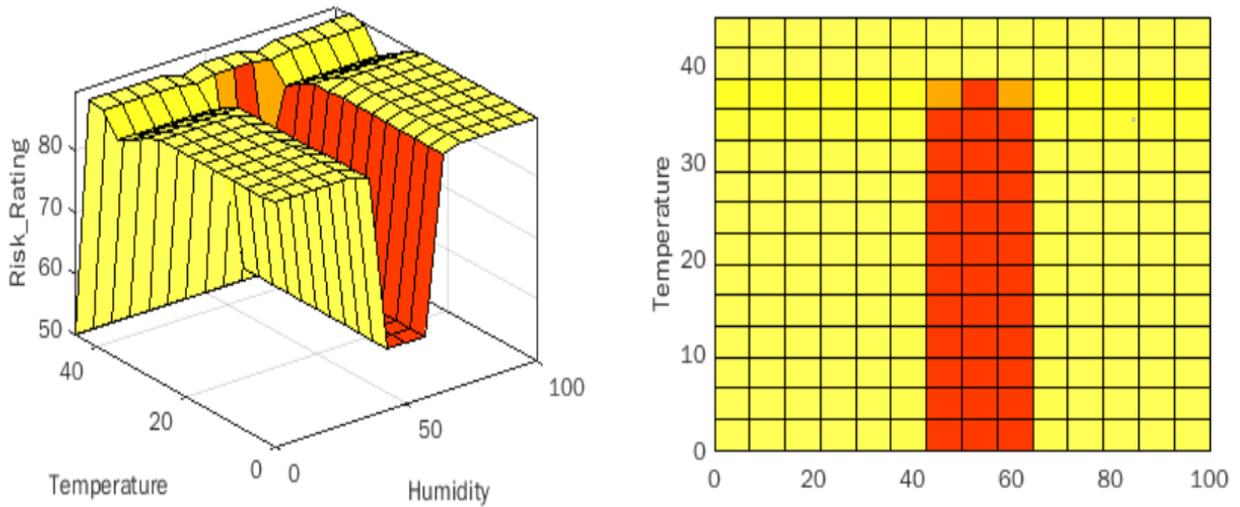


Figure 8-25 Surface Plot Temperature and Humidity on Risk Rating, (b) Contour Plot

Figure 8-26 shows that at age group 45 and above, with temperatures above 40 °C, risk rating on patient increases from 88 % to over 99 %. Showing a strong correlation between high temperature peaks and age peaks. For age groups 40 and below, the risk rating sits at 60 %, from age 40 to 45, this risk rating increases to 88 %.

– Again showing a strong correlation between increased age group and increased temperatures on risk rating

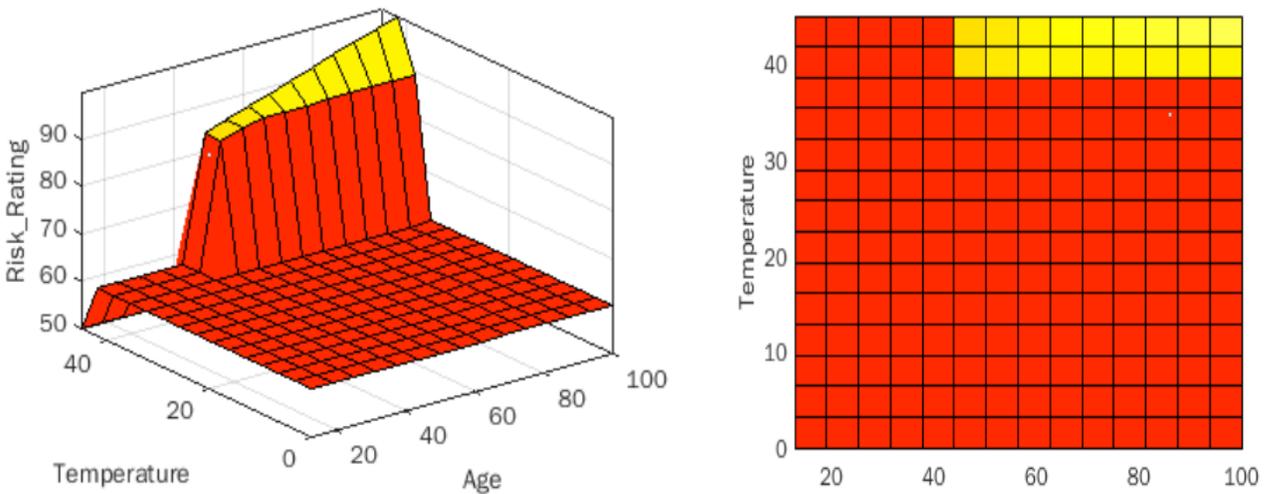


Figure 8-26: Surface Plot Temperature and Age on Risk Rating, (b) Contour Plot

8.2.2. FL Model Testing

The testing of the FL models was done using 15 synthetic test cases for each model. For each test case a score was obtained from the model and compared with the intuitive response of a physician and also MEWS rating for model 1. The comparison was then used to determine the statistics of the model and how it compares to the knowledge of a trained medical professional or a current risk rating method. The test will show if such systems can reliably be used to gauge patient health when doctors are not available.

8.2.2.1. FL Model 1 (BT, BS, PR and Age)

Table 8-21 shows a list of 15 test cases of hypothetical patient parameters designed to test the robustness of the designed models for different risk rating ranges. For each test case, the BT, BS and PR were input into the model to determine the status and percentage risk rating of the patient. This was then subsequently scored against the MEWS rating and the physician rating.

The MEWS is a system developed with the primary focus of helping health practitioners gauge the status of their patients by scoring the patient's vitals based on how severe they are. A score of 0-3 is used with the upper parameter 3 indicating highest severity. A higher MEWS total therefore indicates a higher severity of the patient's condition. If a patient has a MEWS score that is equal to or greater than 5, they are considered critical and in need of urgent medical attention (Al-Dmour et al., 2019). Another way to interpret MEWS scoring is by using the ranges shown below:

3 <Score >0 means 2 hour observations needed

Score = 3 means 1-2 hour observations needed

Score > 3 means ½ hour observations needed

For the purpose of comparison, these intervals will be used to designate test cases into critical health, healthy and unwell ranges as was the case for the physician's rating and the FL model rating. A score =2 will signify unwell range, a score >3 will signify critical range and a score between 0 and 3 will signify healthy status.

The MEWS score is calculated using the Table 8-20. Although the typical MEWS score takes into consideration factors such as blood pressure, PR, respiratory rate, BT and the Alert, verbal, pain, unresponsiveness (AVPU) – for the purpose of comparison, scoring will be done using a change of parameters in the MEWS rating as done by (Choudhury & Baruah, 2015). For the purpose of this work, the variables BT, BS and PR will be used to evaluate the MEWS score. Table 8-20 will be used for scoring.

Table 8-20: MEWS Score Parameters and Ranges

MEWS	+3	+2	+1	0	+1	+2	+3
BT		<=35	35.1-36	36.1-38	38.1-38.5	>38.5	
BS		<=40	41-50	51-100	101-110	111-130	>130
PR		<=4.1		4.2-5.5	5.6-10	10.0-20.0	>20

The MEWS score is calculated using the 3 variables in Table 8-20 above based on the ranges of the specific variables measured. Each of the patient's vital signs (i.e. BT, BS and PR) are cross referenced with Table 8-20 to give a score between 0-9. The total score for the patient is then referred to as the MEWS rating. For normal patients, a MEWS score of 0 is obtained. A percentage rating was then obtained to compare with the FL model and physician rating. This results can be seen in Table 8-21.

Table 8-21: FL Model 1 Performance Relative to MEWS and Medical Doctor Scoring

No.	Vital Signs				Using FL		Using MEWS		Medical Doctor	
	BT (°C)	BS (mmol/L)	PR (BPM)	Age	Status	Score (FL) %	Status	Score MEWS %	Physician Status	Score Physician (%)
1	39.9	12.3	110.0	14	Critical Health	80.0	Critical Health	55.0	Critical Health	86
2	40.3	11.7	125.0	32	Critical Health	80.0	Critical Health	66.7	Critical Health	90
3	41.5	12.0	55.0	45	Critical Health	72.5	Critical Health	66.7	Critical Health	75
4	34.2	4.8	72	80	Healthy	46.8	Healthy	22.2	Healthy	40
5	33.6	3.6	50.0	75	Critical Health	72.4	Critical Health	55.6	Unwell	45
6	36.6	4.3	75	21	Healthy	32.1	Healthy	0.0	Healthy	10
7	37.2	5.0	82.0	45	Healthy	45.8	Healthy	0.0	Healthy	15
8	36.5	4.9	93.0	88	Healthy	32.5	Healthy	0.0	Healthy	25
9	37.1	5.2	68.0	65	Healthy	31.3	Healthy	0.0	Healthy	20
10	37.0	5.1	77.0	15	Healthy	25.0	Healthy	0.0	Healthy	10
11	36.6	4.3	110.0	15	Healthy	33.3	Healthy	11.1	Healthy	10
12	40.3	5.0	118.0	18	Unwell	60.0	Critical Health	44.4	Unwell	60
13	41.5	4.9	85	32	Unwell	60.0	Healthy	22.2	Unwell	55
14	34.2	4.8	68.0	28	Healthy	46.8	Healthy	22.2	Healthy	45
15	33.6	5.1	50.0	33	Unwell	60.0	Unwell	33.3	Healthy	40

The comparisons were done between the model, the MEWS rating and the physician's response. The results can be seen in Table 8-22 and Figure 8-27. As can be seen the model scored equal accuracies and precision, 86.7 % and 83 % respectively, when compared to the MEWS rating and the physician rating. The high precision indicates that the model will be able to correctly diagnose patients with minimum FPs. The recall however differed between the comparisons with the doctor and the comparison with the MEWS rating. When the FL model was compared to the

doctor it achieved values of 100 % while this value dropped to 83 % when compared to MEWS. The recall is relatively high in both cases and indicates that the model will be capable of reducing the number of FNs. The difference in expected output with the MEWS could be attributed to the fact that the MEWS has a smaller range classification compared to the FL model. The high precision, recall and F1 Score means that the model will be able to adequately predict positive and negative risk scenarios in patients. The model can be improved through the use of medical doctors helping to solidify and develop the rule base for the model.

Table 8-22: Scoring Matrix for FL Model 1

Scoring metrics	Comparison with Doctor	Comparison with MEWS
Accuracy	86.7	86.7
Precision	83.0	83.0
Recall	100.0	83.0
F1 Score	90.7	83.0

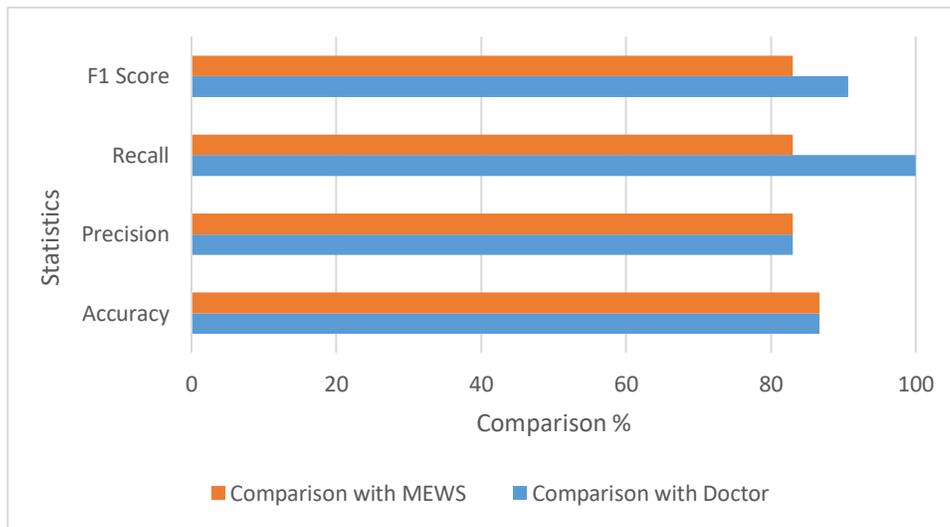


Figure 8-27: FL Model Comparison with MEWS and Physician

Figure 8-27 shows a graphical comparison of the % ratings of model 1 relative to the MEWS rating and physician scoring. As can be seen by the surface plot, Figure 8-28(a), the plot shows that the physician results increases with the FL model results – showing some alignment with these two ratings. The MEWS rating relative to the FL model shows that the two aren't always aligned. The plot shows creases on the x-axis as the FL model % increases. This can be attributed to the MEWS rating dropping to 0 % as can be seen in the straight line plot.

The line plot, Figure 8-28 (b) shows that there is a bigger difference between the FL model score % compared to the physician vs The FL model score % compared to MEWS. There is basically more alignment with the physician and FL model than the MEWS and FL model. There are also cases where the MEWS scores a test case as zero risk even if there is a slight risk. This can again be attributed to the stringent criteria of MEWS which is binary in nature. A rating either gets a 1 or a 0– no in-between. This is what adds to the attractiveness of using a method like FL – since medicine is not a precise science and many scenarios aren't a straightforward “yes” or “no”.

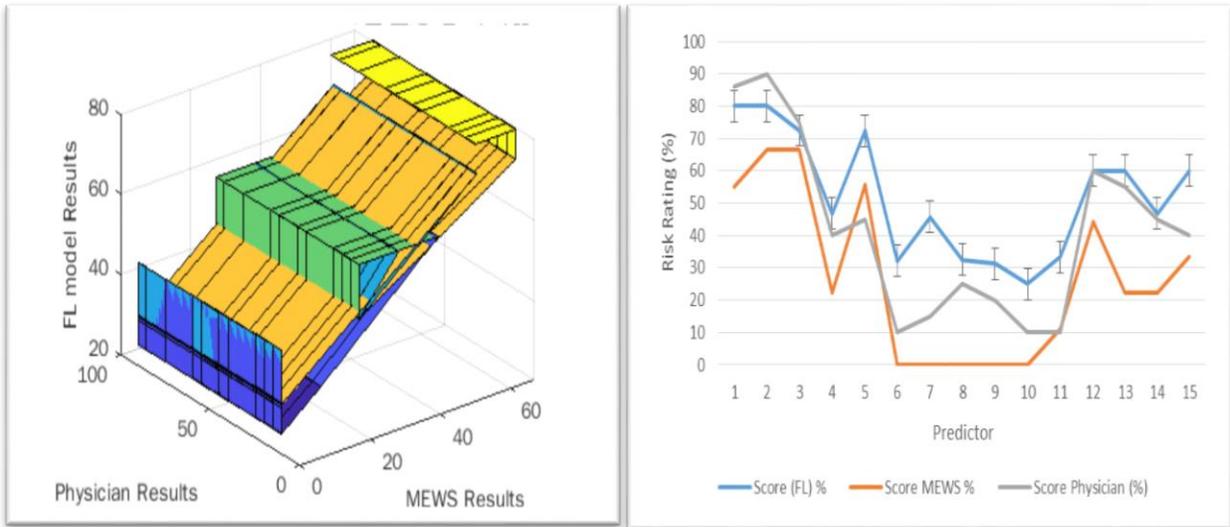


Figure 8-28: (a) Model 1 Surface Plot Comparison with MEWS and Physician FL Model Comparison with MEWS and Physician (b) Line Graph Comparison of Model 1 with MEWS and Physician Rating

8.2.2.2. FL Model 2 (Humidity, Age, Body Temperature)

Model 2 was also tested in a similar manner to model 1. Here, 15 hypothetical test case simulating patient parameters were used to develop predictions using model 2. Then these results were subsequently compared to the predictions made by a physician. The results of the status and scoring by both the model and physician can be seen in Table 8-23.

Table 8-23: FL Model 2 Performance Relative to Physician Diagnosis

No.	Vital Signs			Using FL		Medical Doctor	
	Hum (%)	Age	BT (°C)	Status	Score (FL) %	Physician Status	Physician Score (%)
1	30.0	15	34.2	Risky	60	Risky	70
2	25.0	21	37.3	Low Risk	32.1	Low Risk	10
3	80.0	16	33.2	High Risk	72.7	High Risk	85
4	79.0	25	38.5	High Risk	76.7	Risky	50
5	82	47	37.1	Low Risk	44.2	Low Risk	20
6	73.0	55	39.3	High Risk	77.5	Low Risk	45
7	82.5	60	33.5	High Risk	72.5	High Risk	85
8	25.0	73	32.5	High Risk	73.3	High Risk	95
9	19.0	82	38.3	High Risk	75.3	Low Risk	20
10	30.0	72	31.2	High Risk	74.4	High Risk	100
11	47.2	15	37.2	Low Risk	16.2	Low Risk	20
12	49.3	62	36.8	Low Risk	31.7	Low Risk	10
13	47.5	10	36.9	Low Risk	12.5	Low Risk	10
14	52.3	62	37.3	Low Risk	31.7	Low Risk	10
15	52.5	66	37.1	Low Risk	28.3	Low Risk	10

The comparisons were done between the model and the physician's response. The model statistics were then calculated and the results can be seen in Table 8-24. The model performed relatively well with an accuracy and precision above 80% and recall and F1 Score of 71% and 76.5% respectively. The high precision and recall means that the model will be able to adequately predict positive and negative risk scenarios in patients. The model can be improved through the use of medical doctors helping to solidify and develop the rule base for the model.

Table 8-24: Scoring Matrix for FL Model 2

Scoring metrics	Score FL% compared to Medical Doctor
Accuracy	80
Precision	83
Recall	71
F1 Score	76.5

As can be seen in Figure 8-29, the doctor and physician more or less follow identical paths, however there are cases where there is a huge discrepancy between the predicted results relevant to the physician benchmark. Standard error bars on the predicted values can also be seen.

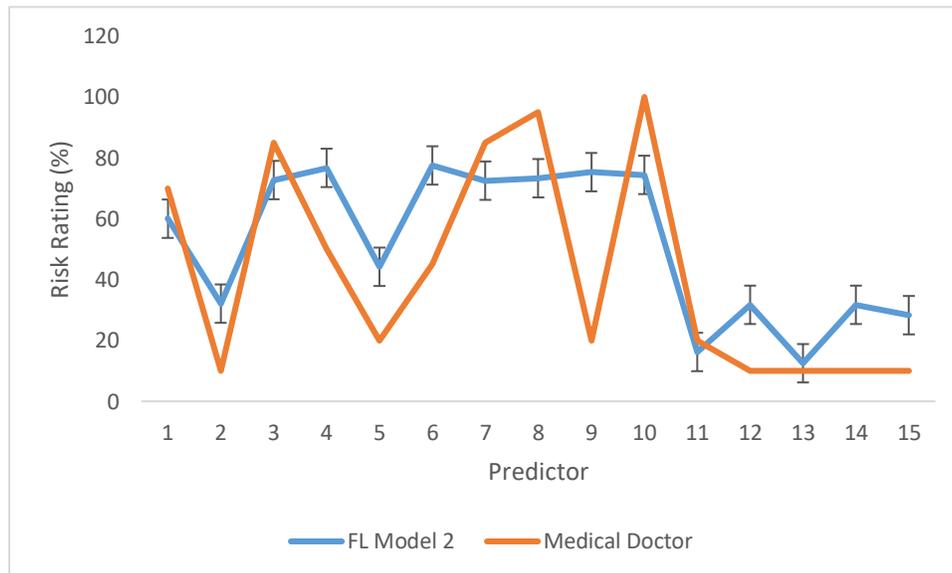


Figure 8-29: Line Graph Comparison of Model 2 with Physician Rating

8.2.2.3. Literature Comparison of FL Models

After comparing the model against medical doctors, the models were also compared to similar works in literature. The results of comparison can be seen in Table 8-25. The models either performed better or in the same range as the models examined. In the case of the model developed by (Jindal et al., 2020), the assistance of doctors establishing the rule base may have been a reason why accuracy achieved surpassed the models developed in this dissertation. Also a clearer definition of the critical inputs of renal cancer is known compared to the current models' risk rating features. Rian et al. (2019) was able to develop a similar system to rank patients health

risk and was able to get a range of accuracies ranging from 50-80 % whereas the developed model reached a constant range of accuracy irrespective of risk rating ranges.

Table 8-25: Comparison of FL Models with Similar Literature Models

Paper	Output linguistic variables	Output	Membership functions	Defuzzification method	Accuracy
Designed FL Models	1) Healthy, Critical health, unwell 2) Risky, Normal, Critical Risk	1) Health Status, 2) Weather Risk	Triangular	MoM	1) 88% 2) 80%
(Duodo et al., 2014)	Malaria free, uncomplicated Malaria, complicated Malaria	Malaria Diagnosis	Triangular	Centre-of-gravity	76.9%
(Rian et al., 2019)	Healthy, Unwell, Not Healthy	Health Status	Trapezoidal	Centroid	50%-80%
(Jindal et al., 2020)	Renal Cancer, No Renal Cancer	Renal Cancer Diagnosis	Gaussian	Centre-of-gravity	96%

8.2.3. ML Model

The stroke prediction model developed was compared to similar ML approaches used in literature. The accuracy of 3 models were compared to the designed model and shows that the stroke model developed outperforms the literature ML models. The ML model developed is 11-18 % more accurate than the other models discussed. The model developed by (Gangavarapu Sailasya et al., 2021) utilized the same dataset used for the developed model, however a lower accuracy of 82 % was obtained. An under sampling rather than an oversampling technique was used to balance the dataset. The results therefore show that for the given dataset, oversampling techniques, as was used, results in a higher model performance. Results of comparison can be seen in Table 8-26.

Table 8-26: Comparison of ML model Performance with Similar Literature Model Performances

Paper	Participants	Features	ML model	Accuracy
Stroke Model developed	10346	13	Two class boosted decision tree	94 %
(Gangavarapu Sailasya, et al., 2021)	5110	12	Naïve Bayes Classification	82 %

(Chun, et al., 2021)	503842	9	Gradient Boosting	76 %-80 %
(Qin, et al., 2021)	3035	17	SVM	83 %

8.3. MA Testing

The MA performance is vital as it serves as both the IoT gateway and one of the GUI's. This is difficult to measure as the effectiveness of the MA will differ depending on the phone the user has. Hence to determine the differences in performance, two phones will be used for testing. A Samsung Galaxy Note 10 Lite and a Huawei P Mate. The Galaxy Note 10 Lite will represent the superior performance phone while the Huawei will represent a lower performance device.

8.3.1. Battery Performance Testing

The first test carried out to test MA performance was the battery test. A graph of battery usage in mAh was plotted over time for the two different phones. Results are seen in Figure 8-30.

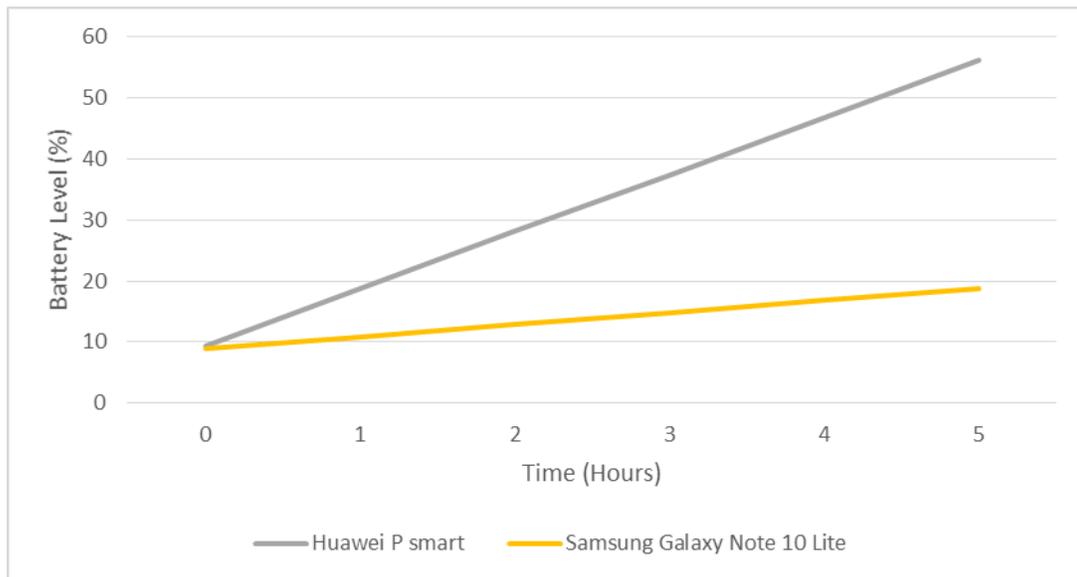


Figure 8-30: Battery Level % over Time for Different Phone Models Using MA

The average usage on the two smartphones per hour was used and extrapolated over time hence the reason for the constant rise in battery level. The results indicate that both phones when using the MA, utilize a low percentage of battery power over an hourly period. However, there is a substantial difference in mAh consumption over a 5 hour period between the two phones. The performance of the Samsung Galaxy Note 10 Lite (orange line) far outweighs the Huawei P smart

(grey line) with the Huawei P smart using almost 3 times as much battery power as the Samsung Galaxy Note 10 Lite. This could be attributed to that fact that the Samsung has better battery optimization capabilities (present in newer phones) which allows it to consume far less mAh compared to its counterpart. Most new phones however have this ability and therefore proves that the MA will perform exceptionally well in newer phones.

8.3.2. RAM Usage Testing

The next test carried out to test the MA performance was a test of its memory usage over time. Figure 8-31 shows a plot of memory in megabytes over time in hours.

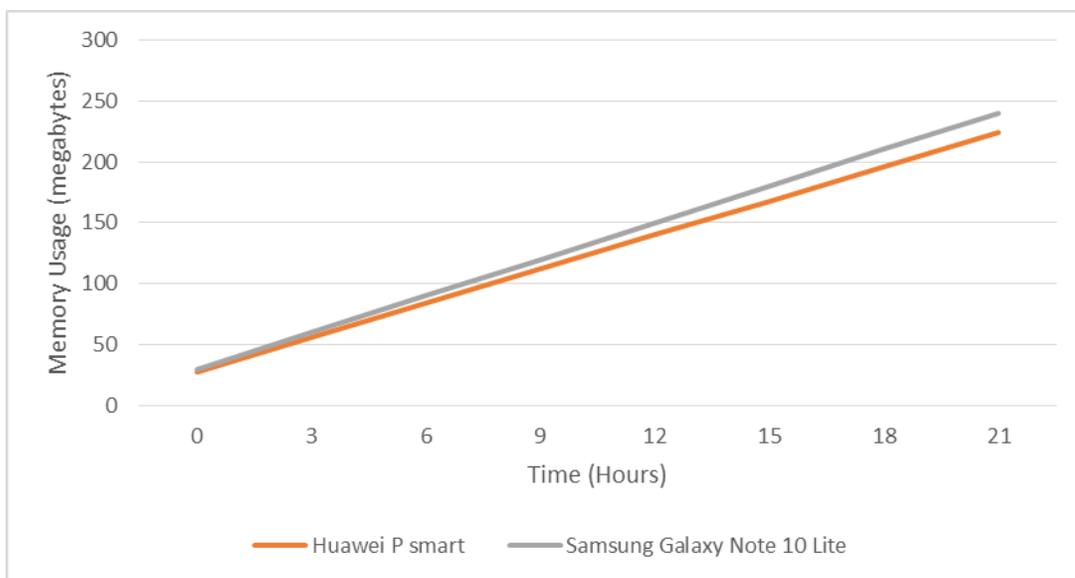


Figure 8-31: Memory Usage (megabytes) over Time for Different Phone Models Using MA

The results show that on average both phones utilized around 30 MB of RAM per hour. This in general indicates that the MA is a low consumer of memory usage considering that it is run continuously in the background and the fact that it uses a lot of computational power during data processing. It can be seen in Figure 8-31 that over a 21 hour period, both phones utilize about 250MB of RAM.

8.4. Overall System Performance

The overall system performance will look at whether all modules within the system speak to each other effectively and determines whether the system performs as per design requirements. In addition, a comparison against other similar systems found in literature and the commercial market will be done to determine how the system performs.

8.4.1. System Integration Testing

Software testing was done across the entire system to determine bugs and possible issues within the code. With complicated systems, it is an excellent method to determine if the system meets functional requirements and is working effectively. To effectively examine all aspects of the code within the system, the testing process is divided into different levels:

- **Unit testing:** functional level testing done on sections of code to determine if the code is working correctly
- **Integration testing:** testing between interfaces within the systems to determine if all components are effectively connected and work seamlessly
- **System testing:** Involves testing and integrated system with all of its components to ensure that it sufficiently meets specifications
- **Acceptance testing:** ensures that applications meet all requirements put forward
- **Performance testing:** is essentially a speed check of different components viz. computer, software or network or device. This can be expressed in the unit of measure of response time or millions of instructions per second etc.

Based on these types of testing, a combination of different test cases were developed to test the robustness of different components of the system. Tables 8-27 & 8-28 show the results of the tests performed. The tests were carried out on a Samsung Galaxy Note 10 Lite for the MA and Google Chrome for the WA, which represents the average capability of a typical smartphone and web browser.

The results show that the WBAN, MA, WA and cloud interfaces are integrated well as all requirements of the system are met. Table 8-27 shows that the response times are fast enough for the intended application. The 4000 ms response time for the models is relatively fast considering the complexity of the models. In addition, the longer times can be attributed to the time delays put in place in the MA to prevent premature loading of null values into the cloud.

Table 8-27: System Response Time Testing

Test Objective	System component	Time (ms)
Transmit data to MA	WBAN and MA	1000
Time taken for patient's location to appear on Firebase	MA & Cloud	8000
Time taken for webpage to load	WA & Cloud	2000
Time taken for models to calculate	MA & Cloud	4000
Time taken for data to display on patient interface	MA & Cloud	4000
Time taken for data to display on doctor interface	MA & Cloud	6000

Table 8-28 shows various scenarios developed to test the effectiveness of the overall system at critical points of operation. The Table lists the expected outcome of each test, the system or components it will employ and test, the input and out specifications to carry out the test. As can be seen, all tests carried out was a pass, indicating that the system works effectively.

Table 8-28: System Validation Testing

Test Objective	System component	Test Condition	Input Specification	Output Specification	Pass/Fail
MA successfully identifies and pairs with WBAN	WBAN and MA	User is on "Bluetooth Devices" activity	User clicks on discovered device and "start connection" button	HC-06 light stop blinking indicating pairing successful – data starts transmitting on listview	Pass
WBAN Strings display correctly	WBAN	User is wearing device	User pairs WBAN and MA	Data string has delimiters in the required order	Pass
Warning LED's work when required	WBAN and MA	User is wearing device and Bluetooth enabled	User has abnormal vitals	LED goes off	Pass
Vibration motor goes off every 30 min	WBAN and MA	User is wearing device and Bluetooth enabled	30 min elapsed	Vibration motor goes off for 5 minutes	Pass
Data uploaded correctly onto SQLite	WBAN and SQLite	User has receiving data from WBAN	Input string from WBAN	Data uploaded in correct columns in SQLite	Pass
Navigation from login activity to main activity (doctor/patient)	MA & WA & Cloud	User is on patient login activity	User enters credentials and clicks on login	User directed to main activity	Pass
Successful Registration process (doctor/patient)	MA & WA & Cloud	User is on registration page	User enters details for registration and clicks on register	User directed to login page and registration info stored in Firebase	Pass
Location tracking working on patient interface and Firebase updated	MA & Cloud	User is on "GPS" activity	User clicks "ON" button	User's location will be visible under their node in Firebase	Pass
Data pulling through from SQLite database (patient interface)	MA & Cloud	User is on "monitor vitals" activity	User clicks on "get data" button	Data displays on Textview	Pass
Data pulling through from Firebase and SQL (doctor interface)	MA & WA & Cloud	User is on "Track Patient" activity in MA or "Monitor Vitals" page of WA	User selects the parameter of user to track and clicks retrieve button	Graphs and textviews are populated	Pass
Model Scores are Being calculated & uploaded onto Firebase	MA & Cloud	User is on Monitor Vitals activity and clicks retrieve button	User clicks on "get data" button	User MEWS, FES and ML models scores displayed on textview. Firebase database connections working	Pass
Data uploaded to MS Azure SQL	MA & Cloud	Patient in "cloud sharing" activity	Patient clicks on "transmission button"	Data is uploaded onto SQL	Pass

8.4.2. System Comparison

The uniqueness of the WBAN compared to other literature or commercial WBAN's is the multitude of sensors it is composed of which helps gauge a patient's health profile from a variety of viewpoints. The increased sensors also help model different scenarios which help doctors and patient's gage trends through a systems thinking approach i.e. seeing the connections between different variables holistically rather than independently. The next advantage of the WBAN system is its integration with an IoT network that allows doctors and patients to share information. Although this may have been explored in literature, commercial products often don't incorporate this feature – which often detaches the doctor from the equation leaving information in the hands of the user who often lacks medical experience. Lastly the incorporation of predictive modelling is also a feature which has been explored in literature but not incorporated in commercial products. Even in literature, a lot of model development in the area of medical diagnosis has been explored, however many of these implementations have lacked the consumption phase which actively utilizes the model. The prototype addresses this issue by utilizing an IoT data pipeline to consume the FES and ML models developed – which means the bridging of research to practicality is achieved.

Tables 8-29 and 8-30 compares two critical components of an effective IoT healthcare telemonitoring system: 1) the price range, 2) the number of sensors/features respectively.

Table 8-29: Price Comparison of Current System with Other Commercial Systems

Current System	Samsung Galaxy Fit2 Fitness Tracker	Polaroid Single Touch Active Watch	Apple Watch Series 3 GPS
R1130	R 1399	R550	R3999

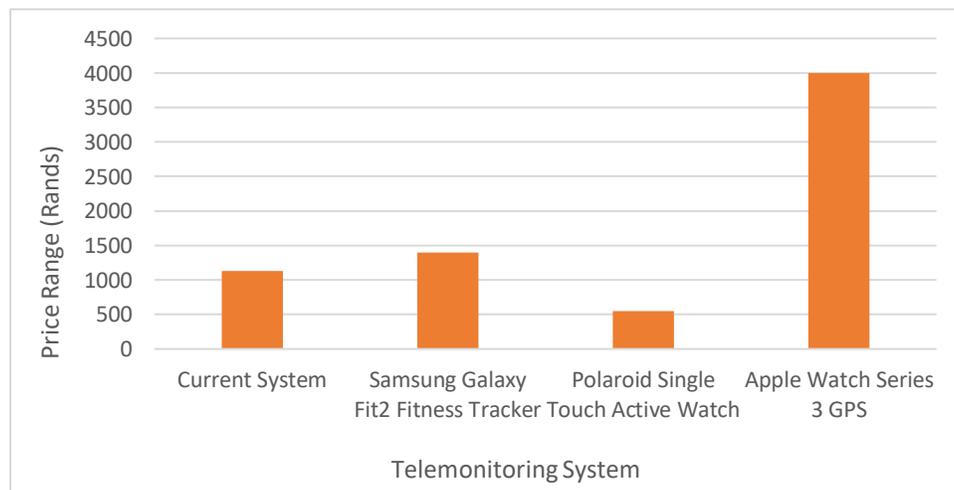


Figure 8-32: Bar Graph Showing Price of Designed System Relative to Commercial Products

As can be seen from Table 8-29 and Figure 8-32, the current system costs excluding manufacturing costs are low considering the vast features it has at its disposal. The *Polaroid* is much cheaper however only possesses about 40 % of the features that the designed system has. Table 8-30 shows that in comparison to other literature systems and commercial systems, the designed system is also far more superior in terms of offering a feature rich experience. The designed system has 15 out of 21 of the features that are common in most wearable health devices as denoted by the green crosses. This Table further highlights some of the contributions of this dissertation which are essentially the features of the developed prototype. 1) A multi-sensory WBAN was developed for remote monitoring, 2) a MA and WA was developed as GUIs, 3) Cloud integration was incorporated into the design and 4) Models were developed that allowed for vitals to be analyzed to identify risk factors.

Table 8-30: Feature Comparison of Current System with Other Telemonitoring Systems

Feature	Current System	Samsung Galaxy Fit2 Fitness Tracker	Polaroid Single Touch Active Watch	Apple Watch Series 3 GPS	(Simeone et al., 2021) (Literature)	(Hameed et al., 2020) (Literature)
ECG	x			x		
EMG	x					
Pulse Rate	x	x	x	x	x	x
Body Temp	x				x	x
Blood Pressure			x		x	x
Pedometer			x			
Sleep Tracker		x				
Humidity	x				x	
Stress Tracker		x				
Accelerometer			x	x		
Barometer				x		
BMI	x					
MEWS Rating	x					
FES	x				x	x
ML	x				x	x
GPS	x			x	x	
MA GUI	x	x	x	x	x	x
WA GUI	x					
Cloud Integration/IoT	x	x		x	x	x
Doctor Interfacing	x				x	
Alerts	x	x	x	x	x	x
Total	15	6	6	8	11	8

8.5. Chapter Summary

Chapter 8 looked at a holistic evaluation of the developed IoT telemonitoring system. The chapter began by discussing the effectiveness of the sensing layer and showed that the sensors satisfy requirements in terms of response times as well as sensor accuracy. The temperature and glucose sensor system was also compared and calibrated with commercial sensors. Calibration curves for both the glucose and temperature sensors were developed and showed minimum error between actual and expected results. A cost comparison of the LM35 and glucose sensor was also done with commercial products. Results showed that the cost effectiveness of the utilized sensors far outweighed commercial products.

The chapter then looked at testing the 3 models developed. A comparison between the models were done using benchmarks such as the MEWS rating and the expertise of a physician. Results showed that both fuzzy logic models were able to perform reasonably well with accuracies of around 80 %. Interesting insights using surface plots were understood after running the models on MATLAB's FL suite. The models including the developed ML model were also compared to literature models and showed that the FL models were in most cases better or equal in standard to similar models developed in literature. The developed ML stroke model achieved accuracy scores of >90 % which surpassed most models developed in literature.

The chapter then progressed to testing the MA performance. Here the RAM usage and battery utilization was tested for different phone models when using the developed MA. Results showed that the battery usage was minimum with Ram usage of around 30MB for both phones. The chapter then concluded by discussing the overall system performance. Overall system response times for the transmission of data were recorded. A due diligence exercise of testing each feature of the system was also recorded and showed a 100 % pass rate. The overall system was also compared to commercial and literature systems in terms of features and cost and showed that for the price tag that the developed system outweighs the similar systems designed. This was highlighted by the fact that the current system had 15 out of the 21 common features found in smart wearables. The chapter also ended with a reference to the objectives and research contributions and how the designed system met the defined criteria.

Chapter 9: Conclusion and Recommendations

This dissertation presented the detailed design of an IoT based healthcare telemonitoring system backed by ML and FL based models to help doctors handle patient diagnostics. The system using a smartphone as an IoT gateway, its two application layers and database integration was discussed. This chapter will further elaborate on the contributions and conclusions derived from the development of the designed healthcare system. In addition, opportunities for expansion of this system will be discussed under the recommendations section.

9.1. Dissertation Summary

This dissertation began by exploring IoT systems and its growth through the increase usage of Wi-Fi capabilities. It was seen that there is a massive opportunity to implement further IoT systems in the healthcare industry. It was further noted that IoT systems using smartphones was a growing trend since smartphone usage is estimated to reach 2/3 of the population by 2023. A literature review of similar IoT and telemonitoring systems were then explored and identified a gap in the sensory layer as well as the way intelligent models were integrated within the system. It was identified that an improvement in the features of the WBAN i.e. the incorporation of additional sensors and capabilities would allow for a more user rich experience allowing doctors and patients to have a more holistic view of the incoming healthcare information. These additional features would then allow for the development of improved models encompassing a variety of input variables. A further gap was identified, which showed that although non-communicable disease models for e.g. heart disease prediction models were incorporated into IoT systems – there was a severe shortage of integrating other non-communicable disease models like stroke prediction models within these IoT systems.

An opportunity to create a smart IoT system using a multi-sensory WBAN which overcomes the above shortcomings was therefore identified. To understand the full requirements of designing such a system – a detailed evaluation of the stakeholder needs as well as the design specifications and challenges in terms of software, electronics and wearability were examined. These requirements and considerations were then filtered down throughout subsequent detailed design stages. A high level proposed system was then highlighted showing how a WBAN would integrate with a smartphone as an IoT gateway and utilize a cloud platform to exchange information to healthcare professionals via a WA and MA interface.

The detailed designed stages were then presented, starting with the construction of a Pugh matrix to identify the optimum IoT gateway and WBAN configuration. The results showed that a smartphone IoT gateway with wrist positioned WBAN would be the optimal architecture for the proposed design. The detailed design was then explained starting with the development of the WBAN sensor layer i.e. the electronic and software design as well as enclosure and fitting on the patient.

The IoT gateway and GUIs were then discussed. This included the discussion on the development of the WA and MA and its role in the transfer of information from the WBAN to the cloud service as well as their use as GUIs for health care personnel to view patient data stored in the cloud. The integration of the MA and WA with the cloud databases were also discussed. This included an overview of the set-up and operation of the cloud interfaces to collect information from the IoT gateway and transmit it to the application layer.

An explanation of the models developed were then presented. This included 2 FL models using the Mamdani approach with triangular MFs and the MoM method for defuzzification. A binary classification stroke prediction model was also presented which utilized MS Azure's ML classic studio framework to develop the model. Various ML models were evaluated and showed that a boosted decision tree was the best classification model based on accuracy, precision, recall and F1 score.

Use case scenarios evaluating various stakeholders and their role in information transfer throughout the system was also discussed. The process of transmitting and receiving information for each stakeholder was explained in detail to highlight the effectiveness of the system.

The dissertation concluded with the results showing the effectiveness of each sensor in the WBAN sensor layer. The glucose and LM35 temperature calibration was also performed. The 2 FL models were tested against a medical doctor's evaluation and achieved accuracies of 88 % for Model 1 when evaluated against the MEWS and physician's rating. Model 2 achieved an accuracy of 83 % when compared to a physician's rating. The FL models and ML model were also evaluated against similar literature models which showed the effectiveness of the system. The overall system effectiveness in terms of response times, cost and features were tested and compared to literature and commercial products and showed that the developed system outperforms similar systems in terms of cost and features.

9.2. Conclusion

The objectives as outlined in chapter 1.5 as well as contributions identified in chapter 1.4 of this dissertation were achieved. Firstly a WBAN consisting of a multitude of sensors (ECG, EMG, Glucose, Pulse Rate, Humidity, and Temperature) was developed. The developed WBAN satisfied the wearability, software and electronic specifications as outlined in section 2.2. A Pugh matrix showed that to satisfy this criteria, a wrist based WBAN would be the optimal design. This configuration allowed for the stakeholder requirements to be satisfied in terms of the device being non-obtrusive, durable and accurate in terms of sensor readings (due to optimal placement of sensors). A MCU with enough ADC and digital connections was selected and ensured that the criteria of size and functionality was met.

The development of an IoT pipeline was satisfied through the use of a Firebase NoSQL database, Azure SQL database and a mobile IoT gateway. According to the Pugh matrix, to satisfy the

requirements of the IoT gateway, a mobile based gateway would be optimal. It is able to store large amounts of data, communicate via short range communication (Bluetooth) with the WBAN and long range communication (Wi-Fi) with the cloud platforms. It is also possible to use it as an application layer so that patients can view data being received and communicated. The Firebase database was used for the registration process since it was well equipped to handle authentication processes which ensured user security. Furthermore user registration information was in a structured single line data which meant that the Firebase NoSQL database was well equipped to store and process this information using its node configuration. For the sensory data which was tabular rows of data, the SQL capabilities within Azure made it a good choice to store this type of data. Connections between the mobile IoT gateway was established using a JDBC for Azure SQL and another API connection for Firebase. These connections further enabled the connection with the doctor MA and WA. This enabled the completion of an IoT pipeline from local storage (SQLite) on the smartphone to the cloud databases to the doctors MA or hospital WA. A multi-sensory remote monitoring solution as outlined in the contributions (chapter 1.4) was therefore fulfilled.

The viewing of data using application layers were satisfied through a development of a MA and WA. Both these application layers satisfied the requirements put forward i.e. ease of use while offering a feature rich experience to its users. The application layers allow both doctors and patients to interact with data and make conscious decisions about health. The application layer for the doctor allows for viewing of historic data graphically over a period of time with multiple sensor inputs. This further iterates the uniqueness of the system in providing a holistic and systems perspective to health care professionals accessing the data. The WA was successfully developed and can be hosted on a hospital server so that access can be given to healthcare professionals such as nurses, radiologists etc. who don't have access to patient data via the MA.

The Last objective of this dissertation was to add intelligence to the IoT system. This was accomplished through the development of 3 models and two techniques. The FES approach using the Mamdani technique and MoM method for defuzzification was utilized to develop a model to predict patient health risk and a model to detect the effect of environmental factors on human health. Both these models were compared to literature and subjected to a comparison with trained medical professionals and the MEWS rating. Results showed that the models fared well in its predictive abilities. Furthermore these models highlighted the unique abilities and contributions identified for the WBAN i.e. the unique combination of sensor inputs in models to predict health risks to patients ensuring a proactive approach to healthcare and reducing delays in treatment. A stroke prediction model was developed using the MS Azure ML Classic studio using the binary classification Boosted decision tree. This model was selected as the optimal after comparison with 7 other classification models. The Boosted decision tree resulted in an accuracy, FL score, precision and recall all above 90%. After development the model was successfully consumed by the MA allowing for remote prediction of stroke in patients. This model satisfied the intended contribution of creating a model that is actually utilized/consumed within an IoT system.

9.3. Recommendations

As data security mechanisms to protect user data was not extensively looked at, future work includes research into improved security to protect patient data within the MS Azure SQL database. This to ensure that only authenticated users are able to access specific patient info.

The research also looked at creating an independent IoT technology that operates outside of the confines of a hospital. Therefore there may be merit into looking at how this system can be further integrated into hospital based IoT systems. Here lies the opportunity of training ML models with historical data within the hospital databases.

Another possible opportunity of future work could entail further expanding the capabilities of the sensing layer either through new capabilities or improvement of current capabilities. While this research focused on a wearable wrist device, opportunities still exist for auxiliary devices such as a head piece capable of measuring EEG data parameters.

Azure SQL and Firebase were chosen as the database cloud platforms, however there may be opportunities to experiment with different cloud platforms to test scalability cost and data security potential.

Lastly as the extent of ML algorithms were not explored extensively in this research, there exists opportunities to further examine different models that provide unique predictions based on user parameters. Herein also lies the opportunity to test different ML platforms and features to compare factors such as ease of use, speed of deployment and cost efficiency.

References

- Abdul-jabbar, H. M. & Abed, J. K., 2020. Real Time Pacemaker Patient Monitoring System Based on Internet of Things. *IOP Conf. Series: Materials Science and Engineering*.
- Abed, J. K. A. a. H. M., 2020. Smart monitor of pacemaker patient by using iot cloud in real time. *Indonesian Journal of Electrical Engineering and Computer Science*, 18(1), pp. 158-166.
- Al-Adhab, A. et al., 2016. *IoT for remote elderly patient care based on Fuzzy logic*. Yasmine Hammamet, Tunisia, IEEE.
- Alarcón-Paredes, A., Francisco-García, V. & Guzmán-Guzmán, I. P., 2019. An IoT-Based Non-Invasive Glucose Level. *Applied sciences*, 9(15), pp. 1-13.
- Android Studio, 2021. *developers*. [Online]
Available at: <https://developer.android.com/training/location/permissions>
[Accessed 10 12 2021].
- Apar, V. & Dr Autee, R. M., 2019. Development of portable device for measurement of blood glucose , temperature and pulse-oximeter using Arduino. *International research journal of engineering and technology (IRJET)*, 6(12), pp. 85-89.
- Armstrong, S., 2007. Wireless connectivity for health and sports monitoring: a review. *British journal of sports medicine*, 41(5), pp. 285-289.
- Arundel, A. V., Sterling, E. M., Biggin, J. H. & Sterling, T. D., 1986. Indirect health effects of relative humidity in indoor environments. *Environmental health perspective*, Volume 65, pp. 351-361.
- Aspuru, J. et al., 2019. Segmentation of the ECG Signal by Means of a Linear Regression Algorithm. *Sensors*, 19(4), pp. 775-791.
- BBC, 2021. *Why do we need to maintain a constant internal environment? - OCR 21C*. [Online]
Available at: <https://www.bbc.co.uk/bitesize/guides/zqdg7p3/revision/4>
[Accessed 15 September 2021].
- Beach, C. et al., 2018. An Ultra Low Power Personalizable Wrist Worn. *IEEE*, Volume 6, pp. 44010-44021.
- Becker, D. E., 2006. Fundamentals of Electrocardiography Interpretation. *Anesthesia Progress*, 53(2), pp. 53-64.
- Bellagente , P. et al., 2016. The “Smartstone”: using smartphones as a telehealth gateway for senior citizens. *IFAC conference paper*, 49(30), pp. 221-226.
- Boulos, M. N. K., Wheeler, S., Tavares , C. & Jones , R., 2011. How smartphones are changing the face of mobile and participatory healthcare: an overview, with example from eCAALYX. *Biomedical engineering online*, 10(24), pp. 1-14.
- Brezulianu, A. et al., 2019. IoT Based Heart Activity Monitoring Using Inductive Sensors. *IoT sensors in E-health*, 19(15), pp. 1-16.

- Buczak, A. L. et al., 2015. Fuzzy association rule mining and classification. *BMC medical informatics and decision making*, 15(1), pp. 47-58.
- Burt , A. & Volchenbom, S., 2018. *How Health Care Changes When Algorithms Start Making Diagnoses*. [Online]
Available at: <https://hbr.org/2018/05/how-health-care-changes-when-algorithms-start-making-diagnoses>
[Accessed 18 September 2021].
- Carlson, R. & Carlson, J. E., 2005. *Design and Optimization in Organic Synthesis*. 2nd ed. Netherlands: Elsevier.
- Chamundeeswari, G., Srinivasan, S. & Bharathi, S. P., 2019. Low Cost Ecg Monitoring using Internet of Things. *International Journal of Recent Technology and Engineering*, 8(3), pp. 351-354.
- Chandini, H. P. & Kumar, M. S. B., 2018. ECG Telemetry System for IOT Using Rasberry Pi. *International Journal of Engineering Research & Technology (IJERT)*, 6(13), pp. 1-4.
- Christensen, J., 2018. The Emergence and Unfolding of Telemonitoring Practices in Different Healthcare Organizations. *International journal of environmental research and public health*, 15(1), p. 16.
- Chun, M. et al., 2021. Stroke risk prediction using machine learning: a prospective cohort study of 0.5 million Chinese adults. *Jamia*, 28(8), p. 1719–1727.
- Chun, M. et al., 2021. Stroke risk prediction using machine learning: a prospective cohort study of 0.5 million Chinese adults. *Journal of the American Medical Informatics Association*, 28(8), pp. 1719-1727.
- Cisco, 2020. *Cisco Annual Internet Report (2018–2023) White Paper*, s.l.: Cisco.
- Cleveland Clinic, 2021. *Hyponatremia*. [Online]
Available at: <https://my.clevelandclinic.org/health/diseases/17762-hyponatremia>
[Accessed 16 September 2021].
- Craig, J. & Patterson, V., 2005. Introduction to the practice of telemedicine. *Telemed Telecare*, 11(1), pp. 3-9.
- Daarani, P. & Kavithamani, A., 2013. Blood glucose level monitoring by noninvasive method using near infra red sensor. *International journal of latest trends and engineering an technology*, 1(3), pp. 141-147.
- Desai, R., Deshmukh, V., Balkhande, B. W. & Bhoir, M., 2021. Heart Disease Prediction System using Fuzzy Logic. *International Research Journal of Engineering and Technology (IRJET)*, 8(3), pp. 2572-2575.
- Djatna, T., Hardhienata , . M. K. D. & Masruriyah , A. F. N., 2018. An intuitionistic fuzzy diagnosis analytics for stroke disease. *Journal of big data*, 5(1), pp. 35 -49.
- Dr Kavitha, R., Janani, M. S. & Dhanalakshimi, K., 2019. Detecting Glucose Level using IR Sensor. *International Journal of Computer Trends and Technology (IJCTT)*, 67(3), pp. 88-91.

- Duodo, Q., Panford, J. K. & Hayfron-Acquah, J. B., 2014. Designing Algorithm for Malaria Diagnosis using Fuzzy Logic for Treatment (AMDFLT) in Ghana. *International Journal of Computer Applications*, 91(17), pp. 22-27.
- Duodu, Q., Panford, J. K. & Hafron-Acquah, J. B., 2014. Designing Algorithm for Malaria Diagnosis using Fuzzy Logic for Treatment (AMDFLT) in Ghana. *International journal of computer applications*, 91(17), pp. 22-27.
- Dutta , A., Chandra, B. S. & Das, K., 2019. A non-invasive microcontroller based estimation of blood glucose concentration by using a modified capacitive sensor at low frequency. *AIP Advances*, 9(10), pp. 1-13.
- ebme, 2021. *Electromyography (EMG)*. [Online]
Available at: <https://www.ebme.co.uk/articles/clinical-engineering/electromyography-emg>
[Accessed 15 September 2021].
- Ehresh, M., Abatis, P. & Schlindwein, F. S., 2020. A portable electrocardiogram for real-time monitoring of cardiac. *SN Applied Sciences*, 2(8), pp. 1-12.
- Elaine, K. L., 2017. *Hot and Cold: Extreme Temperature Safety*. [Online]
Available at: <https://www.healthline.com/health/extreme-temperature-safety#extreme-cold-temperatures>
[Accessed 15 September 2021].
- Elena Muller, M., 2021. *The Difference Between Telehealth and Telemedicine, and Where Remote Patient Monitoring Fits In*. [Online]
Available at: <https://www.healthrecoveryolutions.com/blog/the-difference-between-telehealth-and-telemedicine>
[Accessed 14 September 2021].
- Erdal Kayacan, M. A. K., Kayacan, E. & Mojtaba, A. K., 2016. Fundamentals of Type-1 Fuzzy Logic Theory. In: *Fuzzy Neural Networks for Real Time Control Applications*. s.l.:Elsevier, pp. 13-24.
- Fang, J., Congcong, Z. & Ye, X., 2019. *Optimization of a Wearable Device for Core Body Temperature*. s.l., IOP Conference Series Materials Science and Engineering.
- Fang, X. F. & Kun, L. K., 2007. Uninterrupted ECG Mobile Monitoring. *International Journal of Bioelectromagnetism*, 9(1), pp. 33-34.
- G.F. Gensini, C. A. R. R. M. M. a. G. C., 2017. Value of Telemonitoring and Telemedicine in Heart Failure Management. *Cardiac failure review*, 3(2), pp. 116-121.
- Gangavarapu Sailasya, G. L. A. K., Sailasya, G. & Kumari, G. L. A., 2021. Analyzing the Performance of Stroke Prediction using ML Classification Algorithms. *International Journal of Advanced Computer Science and Applications*, 12(6), pp. 539-545.
- Gillis, A. S., 2021. *What is internet of things (IoT)?*. [Online]
Available at: <https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT>
[Accessed 13 September 2021].

- Grand View Research, 2019. *Internet of Things in Healthcare Market Size, Share & Trends Analysis Report By Component (Service, System & Software), By Connectivity Technology (Satellite, Cellular), By End Use (CRO, Hospital & Clinic), By Application, And Segment Forecasts, 2019 - 20*, s.l.: Grand View Research.
- Gusev, M., Poposka, L., Spasevski, G. & Kotoska, M., 2020. Noninvasive Glucose Measurement Using Machine Learning and. *Journal of sensors*, Volume 2020, pp. 1-14.
- Hameed, K. et al., 2020. An Intelligent IoT Based Healthcare System Using Fuzzy Neural Networks. *Journal of Big Data*, Volume 2020, pp. 1-15.
- Han, J., Pei, J. & Kamber, M., 2011. *Data mining concepts and techniques*. 3rd ed. Amsterdam: Elsevier.
- Haxha, S. & Jhoja, J., 2016. Optical Based Non-invasive Glucose Monitoring Sensor Prototype. *IEEE Photonics Journal* , 8(6), pp. 1-12.
- He, K., Zhang, X., Ren, S. & Sun, J., 2015. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9), pp. 1904-1916.
- Horta, E. T., Lopes, I. C., Rodrigues, J. J. P. C. & Misra, S., 2013. *Real time falls prevention and detection with biofeedback monitoring solution for mobile environments*. Lisbon, International Conference on e-health Networking, Applications and Services (HealthCom).
- Hulft, 2018. *IoT Architecture 101*. [Online]
Available at: <https://medium.com/enterprise-strategist/iot-architecture-basics-guide-ff4bcf8e6859>
[Accessed 15 September 2021].
- Hussain, A. et al., 2016. Personal Home Healthcare System for the Cardiac Patient of Smart City Using Fuzzy Logic. *Journal of Advances in Information Technology*, 7(1), pp. 58-64.
- IGI Global, 2021. *What is an Engineering Model?*. [Online]
Available at: <https://www.igi-global.com/dictionary/engineering-model/9875>
[Accessed 8 October 2021].
- Ionescu, M., 2018. *Measuring and detecting blood glucose by*. s.l., ECAI.
- Iriondo, R., 2018. *Machine Learning (ML) vs. Artificial Intelligence (AI) — Crucial Differences*. [Online]
Available at: <https://pub.towardsai.net/differences-between-ai-and-machine-learning-and-why-it-matters-1255b182fc6>
[Accessed 15 September 2021].
- Islam, F., 2018. *A fuzzy logic based predictive model for early detection of stroke*. s.l., ACM, pp. 1841-1844.
- J. Mathew, J. L. A. C. a. J. J., 2018. *Heart Failure in the Child and Young Adult*. illustrated ed. s.l.:Elsevier Science.
- Jang, J.-R., 1993. ANFIS: adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(3), pp. 665-685.

- Jiang, F. et al., 2017. Artificial intelligence in healthcare: past, present and future. *Stroke and Vascular*, 2(4), pp. 230-243.
- Jindal, N. et al., 2020. Fuzzy Logic Systems for Diagnosis of Renal Cancer. *Applied sciences*, 10(3464), pp. 1-20.
- Jindal, N. et al., 2020. Fuzzy Logic Systems for Diagnosis of Renal Cancer. *Applied Sciences*, 10(3464), pp. 1-20.
- John Hopkins Medicine, 2021. *Vital Signs (Body Temperature, Pulse Rate, Respiration Rate, Blood Pressure)*. [Online]
Available at: <https://www.hopkinsmedicine.org/health/conditions-and-diseases/vital-signs-body-temperature-pulse-rate-respiration-rate-blood-pressure>
[Accessed 15 September 2021].
- Kaelbling, L. P., Littman, M. L. & Moore, A. W., 1996. Reinforcement Learning: A Survey. *Journal of artificial intelligence research*, Volume 4, pp. 237-285.
- Kalogirou, S. A., 2014. Designing and Modeling Solar Energy Systems. In: *Solar Energy Engineering*. s.l.:Elsevier, pp. 583-699.
- Kasbe, T. & Pippal, R. S., 2017. *Design of heart disease diagnosis system using fuzzy logic*. s.l., ICECDS.
- KDnuggets, 2017. *IoT vs. Web-of-Things (WoT)*. [Online]
Available at: <https://www.kdnuggets.com/2017/02/iot-vs-related-concepts-terms.html>
[Accessed 13 September 2021].
- khaddar, M. a. e. & Boulmalf, M., 2017. Smartphone: The Ultimate IoT and IoE Device. In: N. Mohamudally, ed. *Smartphones from an Applied Research Perspective*. s.l.:Intech, pp. 138-152.
- Khan, M. A., 2020. An IoT Framework for Heart Disease Prediction Based on MDCNN Classifier. *IEEE Access*, Volume 8, pp. 34717-34727.
- Khanna, R., 2017. Internet of Things: Architectures, Protocols, and Applications. *Journal of electrical and computer engineering*, Volume 2017, p. 26.
- Kominos, A. & Stamou, S., 2006. *HealthPal: An Intelligent Personal Medical Assistant*. Orange County, Ubicomp.
- Kumar, S., Tiwari, P. & Zymbler, M., 2019. Internet of Things is a revolutionary approach for future technology enhancement: a review. *Journal of Big Data*, 1(115), pp. 21 -25.
- Linn, C.-T. & Lee, C. S. G., 1991. Neural-network-based fuzzy logic control and decision system. *IEEE*, 40(12), pp. 1320 - 1336.
- Maddison, R. & Mchurchu, C. N., 2009. Global positioning system: a new opportunity in physical activity measurement. *International Journal of Behavioral Nutrition and Physical Activity*, 6(73).
- Majumde, A. . J. A., ElSaadany, Y. A., Young, R. & Ucci, D. R., 2018. An Energy Efficient Wearable Smart IoT System to Predict Cardiac Arrest. *Advances in human-computer interaction*, 18(4), pp. 1160-1180.

- Majumder, A. J. A., ElSaadany, Y. A., Young, R. & Ucci, D., 2019. An Energy Efficient Wearable Smart IoT System to Predict Cardiac Arrest. *Advances in Human-Computer Interaction*, 2019(3), pp. 1-21.
- Manshahia, T. M. & Singh, M., 2018. Neuro Fuzzy Inference Approach : A Survey. *IJSRSET*, 4(7), pp. 505-519.
- Manurung, B. E., Munggaran, H. R., Ramadhan, G. F. & Koesoema, A. P., 2019. *Non-Invasive Blood Glucose Monitoring using Near-Infrared Spectroscopy based on Internet of Things using Machine Learning*. Indonesia, IEEE.
- Mathworks, 2021. *Mamdani and Sugeno Fuzzy Inference Systems*. [Online] Available at: <https://www.mathworks.com/help/fuzzy/types-of-fuzzy-inference-systems.html> [Accessed 16 September 2021].
- Mayo Clinic, 2018. *Symptoms Hypoxemia*. [Online] Available at: <https://www.mayoclinic.org/symptoms/hypoxemia/basics/definition/> [Accessed 16 September 2021].
- Mayoclinic, 2019. *Electromyography (EMG)*. [Online] Available at: <https://www.mayoclinic.org/tests-procedures/emg/about/pac-20393913> [Accessed 15 September 2021].
- Mete, B. et al., 2012. The Role of Invasive and Non-Invasive Procedures in Diagnosing Fever of Unknown Origin. *International journal of medical sciences*, 9(8), pp. 682-689.
- Mohammad, A. K. & Fahad, A., 2020. A Healthcare Monitoring System for the Diagnosis of Heart Disease in the IoMT Cloud Environment Using MSSO-ANFIS. *IEEE Access*, Volume 8, pp. 122259-122269.
- Mohammed, M. & Khan, M. B., 2017. *Machine learning algorithms and applications*. New York: CRC Press.
- Munir, M. W., Perälä, S. & Mäkelä, K., 2013. Utilization and Impacts of GPS Tracking in Healthcare: A Research Study for Elderly Care. *International journal of computer applications*, 45(11), pp. 35-37.
- Nazar, A. et al., 2017. Wearable arm band mobile ECG monitoring system. *International journal of innovative research in electrical, electronics , instrumentation and control engineering*, 5(4), pp. 173-179.
- Neeralagi, V. K. S. a. R. R., 2017. IoT based Health Monitoring using Fuzzy logic. *International Journal of Computational Intelligence Research* , 13(10), pp. 2419-2429.
- Nelles, O., 2001. Classical Approaches to Neural Networks and Fuzzy Models. In: *Nonlinear System Identification*. Berlin, Germany: Springer, pp. 137-155.
- Nichols, J. A., Chan, H. W. H. & Baker, M. A. B., 2018. Machine learning: applications of artificial intelligence to imaging and diagnosis. *Biophysical reviews*, 11(1), pp. 111-118.
- Oad, K. K., DeZhi, X. & Butt, P. K., 2014. A Fuzzy Rule based Approach to Predict Risk Level of Heart. *Global journey of computer science and technology: Software and data engineering*, 14(3), pp. 17-21.

- Oracle, 2021. *What is IoT?*. [Online]
Available at: <https://www.oracle.com/za/internet-of-things/what-is-iot/>
[Accessed 13 September 2021].
- Ozkan, H. et al., 2020. A PORTABLE WEARABLE TELE ECG MONITORING SYSTEM. *International Journal of Engineering Applied Sciences and Technology*, 4(12), pp. 683-690.
- P.Ortiz, K. J., Davalas, J. P. O. & Elora S. Eusebio, D. M. T., 2018. IOT: Electrocardiogram (ECG) monitoring system. *Indonesian journal of electrical engineering and computer science*, 10(2), pp. 480-489.
- Pacchierotti, C., 2018. A wearable haptic system for the health monitoring of elderly people in smart cities. *International journal of online biomedical engineering*, 14(8), pp. 52-66.
- Parmigiani, G., 2001. *International Encyclopedia of the Social & Behavioral Sciences*. 2nd ed. Amsterdam: Elsevier.
- Prawiroedjo, K. & Engelen, J., 2016. *Infrared-Based Glucose Level Measurement*. Indonesia, ASAIS.
- Price, D. D., 2010. *How to read an Electrocardiogram (ECG). Part One: Basic principles of the ECG. The normal ECG*. [Online]
Available at: <http://www.southsudanmedicaljournal.com/archive/may-2010/how-to-read-an-electrocardiogram-ecg.-part-one-basic-principles-of-the-ecg.-the-normal-ecg.html>
[Accessed 15 September 2021].
- Qin, Q., Zhou, X. & Jiang, Y., 2021. Prognosis Prediction of Stroke based on Machine Learning and Explanation Model. *Internation Journal of Computers Communications and Control*, 16(2), pp. 1-13.
- R. Mehtaa, a. J. S., 2018. Internet of Things: Vision, Applications and Challenges. *Procedia Computer Science* , Volume 132, pp. 1263-1269.
- Rao, S. P. C. et al., 2021. Binary chemical reaction optimization based feature selection techniques for machine learning classification problems. *Expert systems applications*, Volume 167.
- Rghioui, A., Lloret, J., Harane, M. & Oumnad, A., 2020. A smart glucose monitoring system for diabetic patient. *Electronics*, 9(678), pp. 1-18.
- Rian, M. F., Nasution, S. M. & Ratna, A. N., 2019. *Health monitoring application using fuzzy logic based on android*. s.l., IOP.
- Ribeiro, A. H. et al., 2020. Automatic diagnosis of the 12-lead ECG using a deep neural network. *Nature resources*, 11(1), pp. 1-9.
- Rodrigues, J. J. P. C., Lopes, I. M. C., Silva, B. M. C. & Torre, I. d. L., 2013. A new mobile ubiquitous computing application to control obesity: SapoFit. *Inform health soc care*, 38(1), pp. 37-53.
- Roihan, I., Iskandar, W. J. & Koestoer, R. A., 2019. *Prototype low-cost portable electrocardiogram (ECG) based on Arduino-Uno with Bluetooth feature*. Indonesia, International Symposium of Biomedical Engineering (ISBE).

- Ross P. Cafaro, D., 1960. Hypoxia: Its Causes and Symptoms. *J Am Dent Soc Anesthesiol*, 7(4), pp. 4-8.
- Ruiz,, M. R., Cajavilca, C. & Varon, J., 2008. Einthoven's String Galvanometer. *Texas heart institute journal*, 35(2), pp. 175-178.
- Sagar, D., 2020. *A Study On Non-Invasive Blood Glucose Meter Providing Glucose Measurements Painlessly, Without A Blood Sample Or Finger Pricks*. Bangalore, Computational Systems for Health and Sustainability(CSFHS).
- Sanfilippo , F. & Pettersen, K. Y., 2015. *A Sensor Fusion Wearable Health-Monitoring System with Haptic Feedback*. Dubai, IEEE.
- Santamaria, A. F., Pierfrancesco, R., Rango, F. D. & Serianni, A., 2016. *A two stages fuzzy logic approach for Internet of Things (IoT) wearable devices*. Italy, PIMRC.
- Santo, J. et al., 2016. An IoT-based Mobile Gateway for Intelligent. *Journal of Network and Computer Applications*, Volume 71, pp. 194-204.
- Sarfraz, Z., Sarfraz, A., Iftikar, H. M. & Akhund, R., 2021. Is COVID-19 pushing us to the Fifth Industrial Revolution (Society 5.0)?. *Pakistan journal of medical sciences*, 37(2), pp. 591-594.
- Sari, M. W. & Luthfi, M., 2016. *Design and analysis of non-invasive blood glucose levels monitoring*. Semarang, Indonesia, IEEE.
- Sarker, I. H., 2021. Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN computer science*, 2(160), pp. 1-28.
- Sarker, I. H., Kayes, A. S. M. & Watters, P., 2019. Effectiveness analysis of machine learning classification models for predicting personalized context-aware smartphone usage. *Journal of big data*, 6(1), pp. 1-28.
- SAS, 2021. *Machine Learning*. [Online]
Available at: https://www.sas.com/en_zh/insights/analytics/machine-learning.html
[Accessed 13 September 2021].
- Satyanarayana, K. et al., 2013. Transportation, GPS and GPRS Based Telemonitoring System for Emergency Patient. *Journal of medical engineering*, Volume 2013, pp. 1-9.
- Sattar, H., I. S. B., Shafi, O. & Farooq, U., 2019. An Intelligent Air Quality Sensing System for Open-Skin Wound Monitoring. *Electronics*, 8(1), pp. 801 - 826.
- Schrader, A., Carlson, D. & Rothenpieler, P., 2010. *SmartAssist - Wireless Sensor Networks for Unobtrusive Health Monitoring*. Karlsruhe, 5th BMI Workshop on Behaviour Monitoring and Interpretation.
- Section, 2020. *An Overview of Fuzzy Logic System*. [Online]
Available at: <https://www.section.io/engineering-education/an-overview-of-fuzzy-logic-system/>
[Accessed 16 September 2021].
- Shao, M. et al., 2020. A Wearable Electrocardiogram Telemonitoring System for Atrial Fibrillation Detection. *Sensors*, 20(3), pp. 1-16.

Simjanoska, M., OrclD, M. G., Gams, M. & Bogdanova, A. M., 2018. Non-Invasive Blood Pressure Estimation from ECG Using Machine Learning Techniques. *Sensors*, 18(4), pp. 1160 - 1180.

Softweb solutions, 2021. *Mobile as an IoT gateway - the next giant leap in IoT architecture*. [Online]

Available at: <https://www.softwebsolutions.com/resources/mobile-device-as-an-iot-gateway.html>

[Accessed 15 September 2021].

Stoppler, M. C., 2021. *Blood sugar levels in adults with type 1 or type 2 diabetes facts*. [Online] Available at:

https://www.medicinenet.com/normal_blood_sugar_levels_in_adults_with_diabetes/article.htm

[Accessed 15 September 2021].

Torres, N. A. & Juan, J., 2005. Fuzzy Logic in Medicine and Bioinformatics. *Journal of Biomedicine and Biotechnology*, Volume 2006, pp. 1-7.

Truong, V. A., Dinh, D. T., Tran, A. N. & Le, H. X., 2020. Non-invasive Glucose Monitoring System Utilizing Near-Infrared Technology. In: V. V. Toi, T. Q. Le, H. T. Ngo & T. Nguyen, eds. *7th International Conference on the Development of Biomedical Engineering in Vietnam*. Vietnam: Springer Singapore, pp. 401-405.

U.S. food and drug administration, 2021. *Pulse Oximeter Accuracy and Limitations: FDA Safety Communication*. [Online]

Available at: <https://www.fda.gov/medical-devices/safety-communications/pulse-oximeter-accuracy-and-limitations-fda-safety-communication>

[Accessed 15 September 2021].

Umar, U., Syarif, S. & Indrabayu, I. N. a., 2019. *A real time non-invasive cholesterol monitoring*. Indonesia, ICUDR.

Virone, G., Wood, A., Selavo, L. & Cao, Q., 2006. *An Advanced Wireless Sensor Network for Health Monitoring*. Arlington, Transdisciplinary conference on distributed diagnosis and home healthcare.

WebMD, 2020. *Hyponatremia*. [Online]

Available at: <https://www.webmd.com/a-to-z-guides/what-is-hyponatremia>

[Accessed 16 September 2021].

WebMD, 2020. *Hypoxia and Hypoxemia*. [Online]

Available at: <https://www.webmd.com/asthma/guide/hypoxia-hypoxemia>

[Accessed 16 September 2021].

WebMD, 2021. *What Is Normal Body Temperature?*. [Online]

Available at: <https://www.webmd.com/first-aid/normal-body-temperature>

[Accessed 15 September 2021].

World Health Organization, 2010. *Increasing access to health workers*, Switzerland: WHO Press.

World Health Organization, 2010. *Telemedicine opportunities and developments in member states*, Switzerland: WHO Press.

World Health Organization, 2021. *Cardiovascular diseases (CVDs)*. [Online]
Available at: [https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-\(cvds\)](https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds))
[Accessed 13 September 2021].

XU, G., 2020. IoT-Assisted ECG Monitoring Framework With Secure Data Transmission for Health Care Applications. *IEEE access*, Volume 8, pp. 74586 - 74594.

Yolanda Smith, B., 2021. *Types of Telemedicine*. [Online]
Available at: <https://www.news-medical.net/health/Types-of-Telemedicine.aspx>
[Accessed 14 September 2021].

Yun-Hsuan Chen, M. S., 2021. Trends and Challenges of Wearable Multimodal Technologies for Stroke Risk Prediction. *Sensors*, 21(2), p. 460.

Zanoguera, M. B. et al., 2019. *Portable ECG System Design Using the AD8232*. Mexico, Conference: 6th International Electronic Conference on Sensors and Applications.

Zhang, J. & Lu, Z., 2009. *The Mobile ECG Telemonitoring System Based on GPRS and GPS*. Wuhan, China, IEEE.

Appendices

Appendix A: Bill of Materials (BOM)

Equipment List	Illustration	Description	Dim (mm)	Function	Connection	Supplier	Price (R)	Total
Arduino Nano and micro USB cable		Microcontroller	18x45	Control the circuit	Connection to all other components and power source	Stelltron/Mantech	110.98	2
7.4 V 450mAh Lipo drone battery		Rechargeable battery	25x20x6	Power device	N/A	Mantech	190	2
2S,3S balance charger		Protective circuit charger for rechargeable battery	N/A	Charging battery	N/A	Mantech	355	1
Resistor Kit 600pcs: 30 Values x 20 pcs each	 30Value*20pcs=600pcs Metal Film Resistor 1%	Resistors	N/A	Voltage dividers for Bluetooth, resistors for sensors etc.	Connection with all components to reduce voltage drop	Stelltron	80	1

Bluetooth HM-10		Bluetooth	27×13×2.2	Wireless transmission	1) HC-05 RX to Arduino pin D3 (TX) via a voltage divider 2) Connect HM10 TX pin to Arduino RX pin 2) HM10 VCC to 5V on Arduino 3) HM10 to common ground	Stelltron	97.75	1
Vibration motor module Type 1		Haptic feedback vibration sensor	Not sure	Alert user for issues i.e. problematic health conditions	1) G to GND Arduino 2) V to 5V Arduino 3) S to digital pin Arduino	Make Electronics	27.5	1
Medium solderless breadboard or Vera board		Breadboard/Vera board solderless	N/A	To initially test components before soldering	Connects to all components	Stelltron	46	1
40pcs 10cm Male To Female Jumper Cable Dupont Wire		Jumper wires	100 length	For connection between sensors and breadboard	Connection of components to breadboard	Stelltron	50	1

AD8232		ECG sensor	36x28	Detect ECG readings	<ol style="list-style-type: none"> 1) Ground Arduino to ground sensor 2) 3.3V power supply Arduino to 3.3V sensor (use pull up resistor if using 5V supply) 3) Output sensor to A0 Arduino 4) LO sensor D2 Arduino 5) LO+ to D3 Arduino 	Sparkfun	200	1
N/A		Agcl 3D printed electrodes		Gel less micro contact with skin	N/A			1
LED 5mm (10 Pack)		LED Light	N/A	Flash user in the case of emergencies	<ol style="list-style-type: none"> 1) connect digital pin to resistor 2) connect resistor to Led 3) Connect LED to ground 	Stelltron	10	1

Soldering Iron 60 Watt Adjustable Temperature	 <p>60W 220V EU PLUG Adjustable Temperature 200-450 Deg C with 5 Tips</p>	Soldering iron	N/A	Solder components onto breadboard for final design	N/A			1
Digital Multimeter 830 Series		Digital Multimeter	N/A	Test voltage, current and resistor reading on circuit	N/A			1
L7805 Voltage Regulator		Voltage Regulator	Not sure	To regulate voltage across circuit (5V)		Stelltron	3	1
LD33CV Voltage regulator		Voltage Regulator	Not sure	To regulate voltage across circuit (3V)		Stelltron	3	1

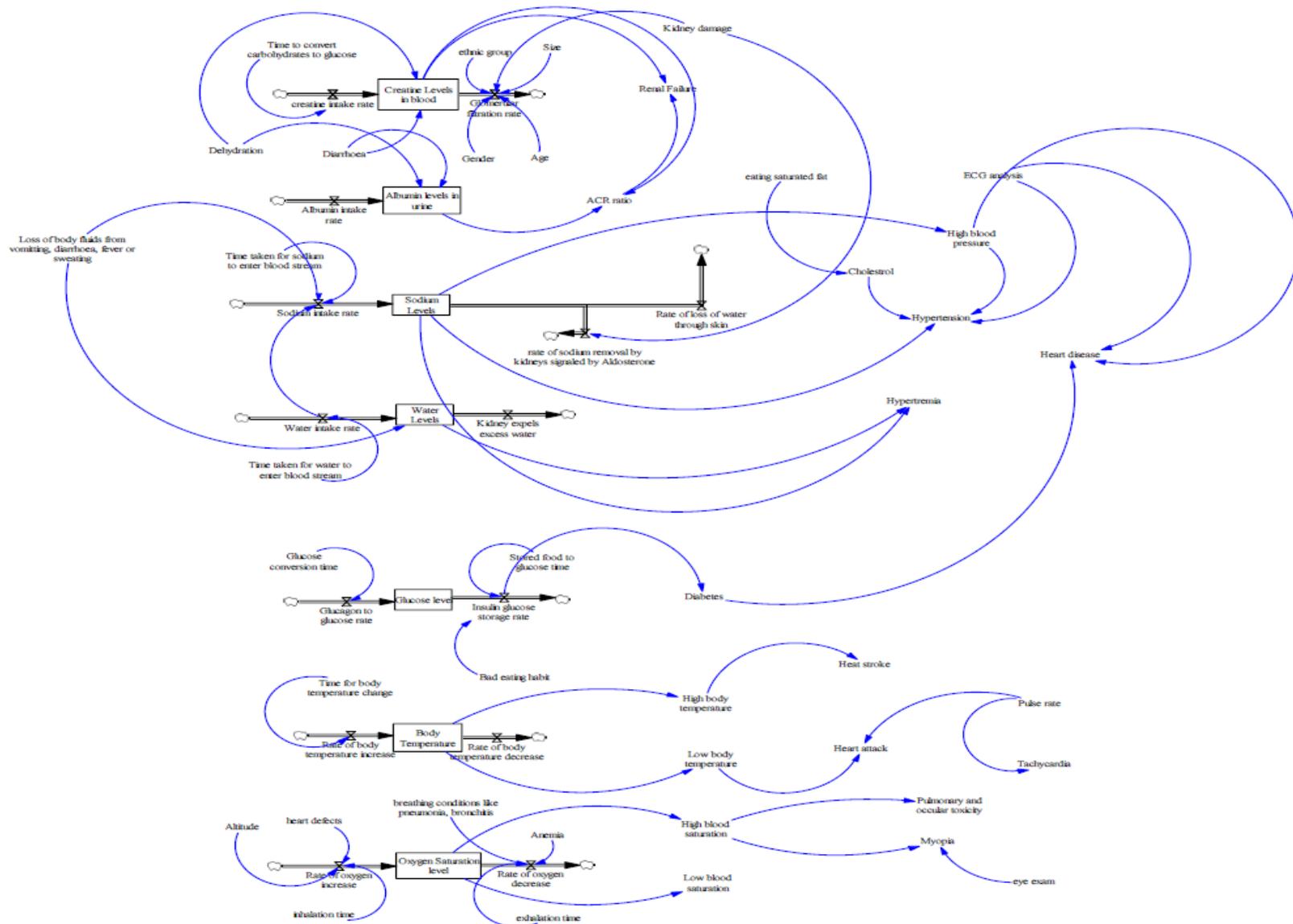
Appendix B: Project Planning

ID	Task Mode	Task Name	Duration	Start	Finish	Predecessors	January			March			May			July			September			November		
							B	E	M	B	E	M	B	E	M	B	E	M	B	E	M	B	E	M
1	✓	Coursework Review	16.5 days	Fri 1/1/21	Fri 1/15/21																			
2	✓	Mechanical Engineering Background	3 days	Fri 1/1/21	Sun 1/3/21																			
3	✓	Microcontroller Background	3 days	Sun 1/3/21	Tue 1/5/21																			
4	✓	Web and mobile app development	3 days	Thu 1/7/21	Sat 1/9/21																			
5	✓	Electronics and AI background	3 days	Sat 1/9/21	Mon 1/11/21																			
6	✓	Medivent biomedical engineering training	3 days	Mon 1/11/21	Wed 1/13/21																			
7	✓	IoT and sensor technology	3 days	Wed 1/13/21	Fri 1/15/21																			
8	✓	Literature Review	18 days	Sat 1/16/21	Sun 1/31/21																			
9	✓	Telemonitoring systems	3 days	Sat 1/16/21	Mon 1/18/21																			
10	✓	IoT telemonitoring systems	3 days	Tue 1/19/21	Thu 1/21/21																			
11	✓	ML and fuzzy logic systems	3 days	Thu 1/21/21	Sun 1/24/21																			
12	✓	Hybrid IoT systems	7 days	Mon 1/25/21	Sun 1/31/21																			
13	✓	Masters Approval	62 days	Tue 2/2/21	Mon 3/29/21																			
14	✓	Topic Proposal	1 day	Tue 2/2/21	Tue 2/2/21	8																		
15	✓	Topic approval	61 days	Wed 2/3/21	Mon 3/29/21																			
16	✓	Concept development and brainstorming	62 days	Tue 2/2/21	Mon 3/29/21	1355																		
17	✓	WBAN development	35 days	Tue 3/30/21	Fri 4/30/21																			
18	✓	Veroboard development	30 days	Tue 3/30/21	Sun 4/25/21																			
19	✓	Integration with wearable strap	5 days	Mon 4/26/21	Fri 4/30/21																			
20	✓	Cloud development	7.63 days	Sat 5/1/21	Fri 5/7/21																			
21	✓	Research and software selection	1 day	Sat 5/1/21	Sat 5/1/21																			
22	✓	Firebase Set-up	3 days	Sun 5/2/21	Tue 5/4/21																			
23	✓	MS Azure set-up	3 days	Wed 5/5/21	Fri 5/7/21																			
24	✓	Web app development	16.5 days	Sat 5/8/21	Sat 5/22/21																			
25	✓	Software research and selection	1 day	Sat 5/8/21	Sat 5/8/21																			
26	✓	Mock-ups	1 day	Sun 5/9/21	Sun 5/9/21																			
27	✓	Front end and back-end web design	9 days	Mon 5/10/21	Tue 5/18/21																			
28	✓	Integration with databases	2 days	Thu 5/20/21	Fri 5/21/21																			
29	✓	Web app testing	1 day	Sat 5/22/21	Sat 5/22/21																			
30	✓	Mobile app development	30.63 days	Sun 5/23/21	Sat 6/19/21																			
31	✓	Software and IDE research and selection	1 day	Sun 5/23/21	Sun 5/23/21																			
32	✓	Mock-ups	1 day	Mon 5/24/21	Mon 5/24/21																			
33	✓	Front end and back-end app design	25 days	Tue 5/25/21	Wed 6/16/21																			
34	✓	Integration with databases	2 days	Thu 6/17/21	Fri 6/18/21																			
35	✓	App testing	1 day	Sat 6/19/21	Sat 6/19/21																			
36	✓	Model development	30.63 days	Sun 6/20/21	Sat 7/17/21																			
37	✓	Model research and selection	1 day	Sun 6/20/21	Sun 6/20/21																			
38	✓	ML model	15 days	Mon 6/21/21	Sun 7/4/21	23																		
39	✓	Fuzzy logic model	12 days	Mon 7/5/21	Fri 7/16/21																			
40	✓	Mobile and web integration	1 day	Sat 7/17/21	Sat 7/17/21	30																		
41	✓	Pre-testing administration	4.25 days	Sun 7/18/21	Wed 7/21/21																			
42	✓	Ethical approval	1 day	Sun 7/18/21	Sun 7/18/21																			
43	✓	Design of testing procedures	2 days	Mon 7/19/21	Tue 7/20/21																			
44	✓	Appointments with volunteers	1 day	Wed 7/21/21	Wed 7/21/21																			
45	✓	Prototype system testing	2 days	Thu 7/22/21	Fri 7/23/21																			
46	✓	Wearability testing	2 days	Thu 7/22/21	Fri 7/23/21																			
47	✓	Sensor accuracy testing	2 days	Thu 7/22/21	Fri 7/23/21	4655																		
48	✓	Data analysis and minor adjustments	5.63 days	Sat 7/24/21	Wed 7/28/21																			
49	✓	Record and analyze data	3 days	Sat 7/24/21	Mon 7/26/21																			
50	✓	Adjust prototype	2 days	Tue 7/27/21	Wed 7/28/21																			
51	✓	Dissertation submission	183.5 days	Sun 6/20/21	Thu 12/2/21																			
52	✓	Thesis draft completion	129 days	Sun 6/20/21	Thu 10/14/21	3655																		
53	✓	Submission to supervisors	20 days	Fri 10/15/21	Mon 11/1/21																			
54	✓	Dissertation corrections and finalization	31 days	Tue 11/2/21	Mon 11/29/21	53																		
55	✓	Final dissertation submission	1 day	Tue 11/30/21	Tue 11/30/21																			
56	✓	Writing of journal entries	1 day	Wed 12/1/21	Wed 12/1/21																			
57	✓	Submission of journal entries	1 day	Thu 12/2/21	Thu 12/2/21	56																		

Appendix C: Mobile App UML diagram



Appendix F: Systems flow diagram for physiological parameters



Appendix H1: SQL Queries

//Creating Table MS Azure

```
private void CreateDBTable(String DBT) {
    String TheDBT = DBT;
    String TheDBT_ML = DBT + "ML";
    try {

        ConnectionAzure conStr = new ConnectionAzure();
        connect = conStr.CONN();
        if (connect == null) {

            System.out.println("connection goes wrong");

        } else {

            //long term data table
            String sql = "CREATE TABLE " + TheDBT +
                "(ID int IDENTITY(1,1) PRIMARY KEY, "+
                " ECGxValues NVARCHAR(255), " +
                " ECGyValues NVARCHAR(255)," +
                " EMGxValues NVARCHAR(255)," +
                " EMGyValues NVARCHAR(255)," +
                " HumxValues NVARCHAR(255)," +
                " HumyValues NVARCHAR(255)," +
                " TxValues NVARCHAR(255)," +
                " TyValues NVARCHAR(255)," +
                " PulsexValues NVARCHAR(255)," +
                " PulseyValues NVARCHAR(255)," +
                " IRxValues NVARCHAR(255)," +
                " IRyValues NVARCHAR(255))";

            //short term data table
            String sql_ML = "CREATE TABLE " + TheDBT_ML +
                "( ECGxValues NVARCHAR(255), " +
                " ECGyValues NVARCHAR(255)," +
                " EMGxValues NVARCHAR(255)," +
                " EMGyValues NVARCHAR(255)," +
                " HumxValues NVARCHAR(255)," +
                " HumyValues NVARCHAR(255)," +
                " TxValues NVARCHAR(255)," +
                " TyValues NVARCHAR(255)," +
                " PulsexValues NVARCHAR(255)," +
                " PulseyValues NVARCHAR(255)," +
                " IRxValues NVARCHAR(255)," +
                " IRyValues NVARCHAR(255))";

            Statement stat = null;

            try {
```

```

    stat = connect.createStatement();
} catch (SQLException throwables) {
    throwables.printStackTrace();
}
try {
    stat.executeUpdate(sql);
    stat.executeUpdate(sql_ML);

} catch (SQLException throwables) {
    throwables.printStackTrace();
}

System.out.println("success");
Delay1();

}
connect.close();

} catch (SQLException throwables) {
    throwables.printStackTrace();
}
}
//Uploading data to Azure SQL

```

@Override

```

protected String doInBackground(String... strings) {
    {
        String Name = FirebaseAuth.getInstance().getCurrentUser().getUid();
        try {

            ConnectionAzure conStr = new ConnectionAzure();
            connect = conStr.CONN();

            String Delete = "DELETE from "+ Name;
            Statement Stat2 = null;
            Stat2 = connect.createStatement();
            Stat2.executeUpdate(Delete);

            if (connect == null) {
                msg = "connection goes wrong";
                System.out.println(msg);
                String.setText(msg);
            } else {
                for (int j = 0; j < ECGxValues.size() && j < ECGyValues.size() && j < EMGxValues.size()
&& j < EMGyValues.size() && j < HumxValues.size() && j < HumyValues.size() && j < TxValues.size()
&& j < TyValues.size() && j < IRxValues.size() && j < IRyValues.size(); j++) {

                    String d = (String) ECGxValues.get(j);
                    String e = (String) ECGyValues.get(j);
                    String f = (String) EMGxValues.get(j);
                    String g = (String) EMGyValues.get(j);
                }
            }
        }
    }
}

```

```

String h = (String) HumxValues.get(j);
String i = (String) HumyValues.get(j);
String j2 = (String) TxValues.get(j);
String k = (String) TyValues.get(j);
String n = (String) PulsexValues.get(j);
String o = (String) PulseyValues.get(j);
String p = (String) IRxValues.get(j);
String q = (String) IRyValues.get(j);

```

```

String query0 = "INSERT INTO " + Name + "(ECGxValues, ECGyValues, EMGxValues,
EMGyValues, HUMxValues, HUMyValues, TxValues, TyValues, PulsexValues, PulseyValues,
IRxValues, IRyValues) VALUES (" + d + "," + e + "," + f + "," + g + "," + h + "," + i + "," + j2
+ "," + k + "," + n + "," + o + "," + p + "," + q + ")";

```

```

Statement stat3 = null;
try {
    stat3 = connect.createStatement();
} catch (SQLException throwables) {
    throwables.printStackTrace();
}
try {
    stat3.executeUpdate(query0);

} catch (SQLException throwables) {
    throwables.printStackTrace();
}
msg = "inserting successful";
System.out.println(msg);
String.setText(msg);
}
connect.close();
msg = "All data uploaded";
ECGxValues.removeAll(ECGxValues);
ECGyValues.removeAll(ECGyValues);
EMGxValues.removeAll(EMGxValues);
EMGyValues.removeAll(EMGyValues);
HumxValues.removeAll(HumxValues);
HumyValues.removeAll(HumyValues);
TxValues.removeAll(TxValues);
TyValues.removeAll(TyValues);
PulsexValues.removeAll(PulsexValues);
PulseyValues.removeAll(PulseyValues);
IRxValues.removeAll(IRxValues);
IRyValues.removeAll(IRyValues);

}

} catch (SQLException throwables) {
    throwables.printStackTrace();
}
}
return msg;
}
}

```

Appendix H2: SQLite Android Query code

```
//create table
```

```
@Override
```

```
public void onCreate(SQLiteDatabase sqLiteDatabase) {  
    String createTable = "create table myTable(ECGSensor REAL,ECGxValues REAL,ECGyValues  
REAL,EMGSensor REAL,EMGxValues REAL,EMGyValues REAL,HumSensor REAL,HumxValues  
REAL,HumyValues REAL,IRSensor REAL,IRxValues REAL,IRyValues REAL,TSensor  
REAL,TxValues REAL,TyValues REAL,PulseSensor REAL,PulsexValues REAL,PulseyValues  
REAL);";  
    sqLiteDatabase.execSQL(createTable);  
}
```

```
//Resize table
```

```
public void ResizeSQL() {  
    SQLiteDatabase sqLiteDatabase = this.getWritableDatabase();  
    sqLiteDatabase.execSQL("DELETE FROM MyTable WHERE rowid < (SELECT + MAX(rowid)  
FROM MyTable)-5000");  
}
```

```
//Retirieving data from SQLite
```

```
public Cursor alldata() {  
    SQLiteDatabase sqLiteDatabase = this.getWritableDatabase();  
    Cursor cursor = sqLiteDatabase.rawQuery("select * from myTable", null);  
    return cursor;  
}
```

```
public Cursor somedata() {  
    SQLiteDatabase sqLiteDatabase = this.getWritableDatabase();  
    Cursor cursor = sqLiteDatabase.rawQuery("SELECT * FROM myTable ORDER BY ROWID DESC  
LIMIT 1",null);  
    return cursor;  
}
```

```
public Boolean insertData(String ECGSensor, String ECGValueX, String ECGValueY, String  
EMGSensor, String EMGValueX, String EMGValueY,String HumSensor,String HumValueX,String  
HumValueY,String IRSensor,String IRValueX,String IRValueY,String TSensor,String TValueX, String  
TValueY,String PulseSensor,String PulseValueX,String PulseValueY) {  
    SQLiteDatabase sqLiteDatabase = this.getWritableDatabase();  
    ContentValues contentValues = new ContentValues();
```

```
    //different columns for X and y values of each graph  
    contentValues.put("ECGSensor", ECGSensor);  
    contentValues.put("ECGxValues", ECGValueX); //under the x column of table for sql database put x  
value passed into function from mainactivity  
    contentValues.put("ECGyValues", ECGValueY); // as above for y values  
    contentValues.put("EMGSensor", EMGSensor);
```

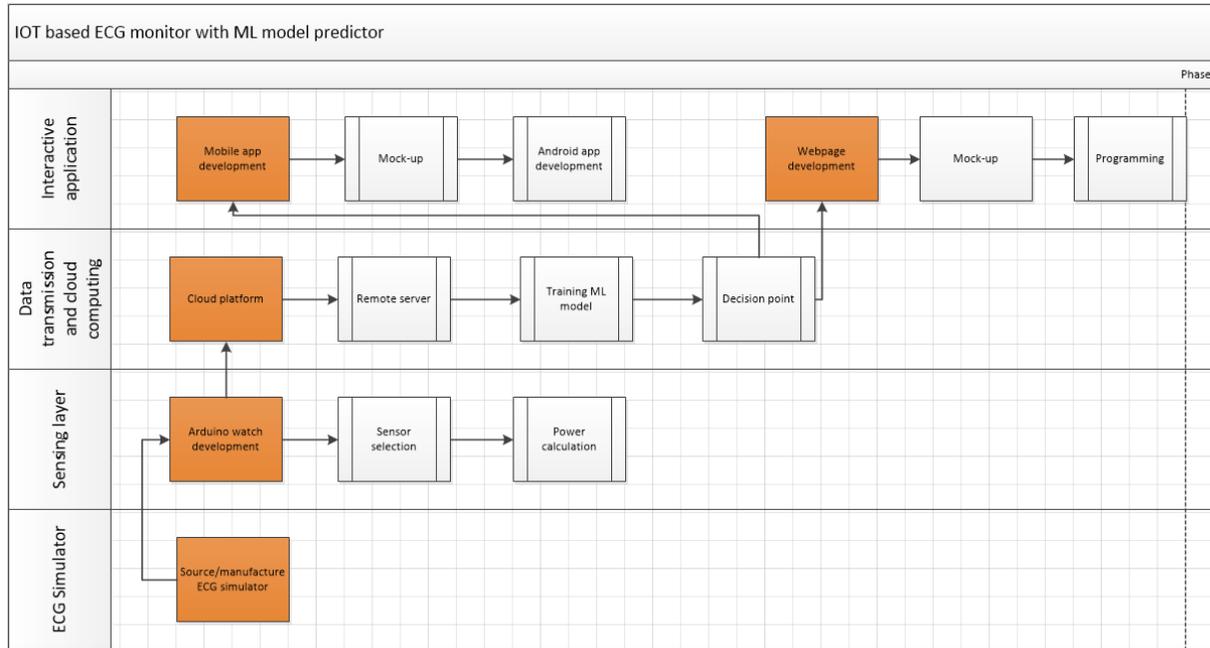
```
contentValues.put("EMGxValues", EMGValueX);
contentValues.put("EMGyValues", EMGValueY);
contentValues.put("HumSensor", HumSensor);
contentValues.put("HumxValues", HumValueX);
contentValues.put("HumyValues", HumValueY);
contentValues.put("TSensor", TSensor);
contentValues.put("TxValues", TValueX);
contentValues.put("TyValues", TValueY);
contentValues.put("PulseSensor", PulseSensor);
contentValues.put("PulsexValues", PulseValueX);
contentValues.put("PulseyValues", PulseValueY);
contentValues.put("IRSensor", IRSensor);
contentValues.put("IRxValues", IRValueX);
contentValues.put("IRyValues", IRValueY);

sqliteDatabase.insert("myTable", null, contentValues);
return true;
}
```

Appendix I: Battery Sizing

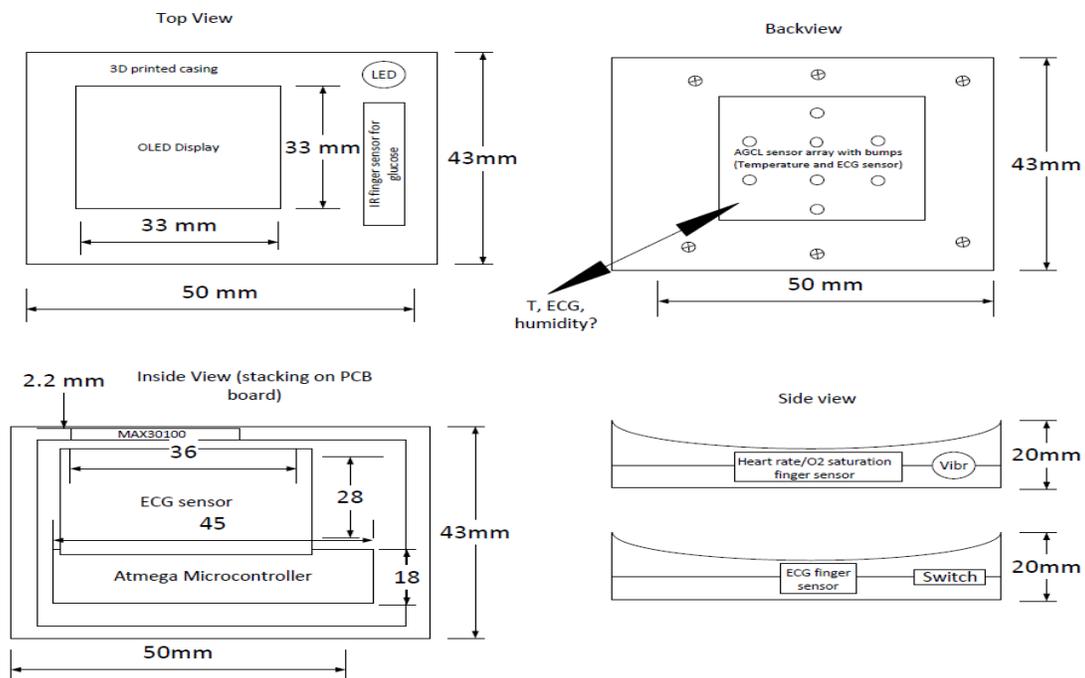
Component	Power (mA)
Arduino Nano	19
Bluetooth	50
HIH-4000	0.2
Myoware EMG sensor	9
AD8232 ECG sensor	0.17
LED	15
IR transmitter	20
IR receiver	0.04
LM35 temperature sensor	0.06
MAX30100 sensor	1.2
MAX30205 sensor	0.6
Vibration Module	1
Total	116.27
Total *1.5	174.405

Appendix J: High Level Design approach



Appendix K: Sensor Positioning Matrix and Sketches

Part	Top	Bottom	Side 1	Side 2	Watch buckle	Drawn	Additional notes
Myoware EMG sensor		2* sensor pads				YES	* need to check if require GND lead * sensor pad size fixed
AD8232 ECG sensor	1*sensor pad	2* sensor pads					* Sensor pads to be 3D printed so sensors can be made to accommodate size requirements (e.g. strips, circles etc.)
MAX30205 body temperature sensor	X		X	X			
MAX30100 Pulse Oximeter sensor	X		X	X		YES	
HIH-4000 humidity sensor	X		X	X	X	YES	
LM35 Temperature sensor	X		X	X	X	YES	
PCB Board							Can be positioned anywhere (even middle)
> Microcontroller							Can be positioned anywhere (even middle)
> Regulators							Can be positioned anywhere (even middle)
> Resistors							Can be positioned anywhere (even middle)
LED light	X						
Vibration motor							> Can be positioned anywhere (even middle) > needs to be properly secured so that it doesn't loosen contacts between components
Bluetooth							Can be positioned anywhere (even middle)
IR sensor		X					
IR LED					X		
Battery							> Battery needs to be removable



Appendix L: Rules Bases for FL Models

Model 1:

Rules	Output
IF Cold Temp AND Low Pulse AND Low Glucose AND Adolescent Age	Critical Health
IF Cold Temp AND Low Pulse AND Low Glucose AND Adult Age	Critical Health
IF Cold Temp AND Low Pulse AND Low Glucose AND Senior Adult Age	Critical Health
IF Cold Temp AND Low Pulse AND Normal Glucose AND Adolescent Age	Unwell
IF Cold Temp AND Low Pulse AND Normal Glucose AND Adult Age	Unwell
IF Cold Temp AND Low Pulse AND Normal Glucose AND Senior Adult Age	Critical Health
IF Cold Temp AND Low Pulse AND High Glucose AND Adolescent Age	Critical Health
IF Cold Temp AND Low Pulse AND High Glucose AND Adult Age	Critical Health
IF Cold Temp AND Low Pulse AND High Glucose AND Senior Adult Age	Critical Health
IF Cold Temp AND Normal Pulse AND Low Glucose AND Adolescent Age	Unwell
IF Cold Temp AND Normal Pulse AND Low Glucose AND Adult Age	Unwell
IF Cold Temp AND Normal Pulse AND Low Glucose AND Senior Adult Age	Critical Health
IF Cold Temp AND Normal Pulse AND Normal Glucose AND Adolescent Age	Healthy
IF Cold Temp AND Normal Pulse AND Normal Glucose AND Adult Age	Healthy
IF Cold Temp AND Normal Pulse AND Normal Glucose AND Senior Adult Age	Healthy
IF Cold Temp AND Normal Pulse AND High Glucose AND Adolescent Age	Unwell
IF Cold Temp AND Normal Pulse AND High Glucose AND Adult Age	Unwell
IF Cold Temp AND Normal Pulse AND High Glucose AND Senior Adult Age	Critical Health
IF Cold Temp AND High Pulse AND Low Glucose AND Adolescent Age	Critical Health
IF Cold Temp AND High Pulse AND Low Glucose AND Adult Age	Critical Health
IF Cold Temp AND High Pulse AND Low Glucose AND Senior Adult Age	Critical Health
IF Cold Temp AND High Pulse AND Normal Glucose AND Adolescent Age	Healthy
IF Cold Temp AND High Pulse AND Normal Glucose AND Adult Age	Healthy
IF Cold Temp AND High Pulse AND Normal Glucose AND Senior Adult Age	Critical Health
IF Cold Temp AND High Pulse AND High Glucose AND Adolescent Age	Critical Health
IF Cold Temp AND High Pulse AND High Glucose AND Adult Age	Critical Health
IF Cold Temp AND High Pulse AND High Glucose AND Senior Adult Age	Critical Health
IF Normal Temp AND Low Pulse AND Low Glucose AND Adolescent Age	Unwell
IF Normal Temp AND Low Pulse AND Low Glucose AND Adult Age	Unwell
IF Normal Temp AND Low Pulse AND Low Glucose AND Senior Adult Age	Critical Health
IF Normal Temp AND Low Pulse AND Normal Glucose AND Adolescent Age	Unwell
IF Normal Temp AND Low Pulse AND Normal Glucose AND Adult Age	Unwell
IF Normal Temp AND Low Pulse AND Normal Glucose AND Senior Adult Age	Critical Health
IF Normal Temp AND Low Pulse AND High Glucose AND Adolescent Age	Unwell
IF Normal Temp AND Low Pulse AND High Glucose AND Adult Age	Unwell
IF Normal Temp AND Low Pulse AND High Glucose AND Senior Adult Age	Critical Health
IF Normal Temp AND Normal Pulse AND Low Glucose AND Adolescent Age	Unwell
IF Normal Temp AND Normal Pulse AND Low Glucose AND Adult Age	Unwell
IF Normal Temp AND Normal Pulse AND Low Glucose AND Senior Adult Age	Critical Health

IF Normal Temp AND Normal Pulse AND Normal Glucose AND Adolescent Age	Healthy
IF Normal Temp AND Normal Pulse AND Normal Glucose AND Adult Age	Healthy
IF Normal Temp AND Normal Pulse AND Normal Glucose AND Senior Adult Age	Unwell
IF Normal Temp AND Normal Pulse AND High Glucose AND Adolescent Age	Unwell
IF Normal Temp AND Normal Pulse AND High Glucose AND Adult Age	Unwell
IF Normal Temp AND Normal Pulse AND High Glucose AND Senior Adult Age	Critical Health
IF Normal Temp AND High Pulse AND Low Glucose AND Adolescent Age	Unwell
IF Normal Temp AND High Pulse AND Low Glucose AND Adult Age	Unwell
IF Normal Temp AND High Pulse AND Low Glucose AND Senior Adult Age	Critical Health
IF Normal Temp AND High Pulse AND Normal Glucose AND Adolescent Age	Healthy
IF Normal Temp AND High Pulse AND Normal Glucose AND Adult Age	Healthy
IF Normal Temp AND High Pulse AND Normal Glucose AND Senior Adult Age	Critical Health
IF Normal Temp AND High Pulse AND High Glucose AND Adolescent Age	Unwell
IF Normal Temp AND High Pulse AND High Glucose AND Adult Age	Unwell
IF Normal Temp AND High Pulse AND High Glucose AND Senior Adult Age	Critical Health
IF Hot Temp AND Low Pulse AND Low Glucose AND Adolescent Age	Critical Health
IF Hot Temp AND Low Pulse AND Low Glucose AND Adult Age	Critical Health
IF Hot Temp AND Low Pulse AND Low Glucose AND Senior Adult Age	Critical Health
IF Hot Temp AND Low Pulse AND Normal Glucose AND Adolescent Age	Unwell
IF Hot Temp AND Low Pulse AND Normal Glucose AND Adult Age	Unwell
IF Hot Temp AND Low Pulse AND Normal Glucose AND Senior Adult Age	Critical Health
IF Hot Temp AND Low Pulse AND High Glucose AND Adolescent Age	Critical Health
IF Hot Temp AND Low Pulse AND High Glucose AND Adult Age	Critical Health
IF Hot Temp AND Low Pulse AND High Glucose AND Senior Adult Age	Critical Health
IF Hot Temp AND Normal Pulse AND Low Glucose AND Adolescent Age	Unwell
IF Hot Temp AND Normal Pulse AND Low Glucose AND Adult Age	Unwell
IF Hot Temp AND Normal Pulse AND Low Glucose AND Senior Adult Age	Critical Health
IF Hot Temp AND Normal Pulse AND Normal Glucose AND Adolescent Age	Unwell
IF Hot Temp AND Normal Pulse AND Normal Glucose AND Adult Age	Unwell
IF Hot Temp AND Normal Pulse AND Normal Glucose AND Senior Adult Age	Critical Health
IF Hot Temp AND Normal Pulse AND High Glucose AND Adolescent Age	Unwell
IF Hot Temp AND Normal Pulse AND High Glucose AND Adult Age	Unwell
IF Hot Temp AND Normal Pulse AND High Glucose AND Senior Adult Age	Critical Health
IF Hot Temp AND High Pulse AND Low Glucose AND Adolescent Age	Critical Health
IF Hot Temp AND High Pulse AND Low Glucose AND Adult Age	Critical Health
IF Hot Temp AND High Pulse AND Low Glucose AND Senior Adult Age	Critical Health
IF Hot Temp AND High Pulse AND Normal Glucose AND Adolescent Age	Unwell
IF Hot Temp AND High Pulse AND Normal Glucose AND Adult Age	Unwell
IF Hot Temp AND High Pulse AND Normal Glucose AND Senior Adult Age	Critical Health
IF Hot Temp AND High Pulse AND High Glucose AND Adolescent Age	Critical Health
IF Hot Temp AND High Pulse AND High Glucose AND Adult Age	Critical Health
IF Hot Temp AND High Pulse AND High Glucose AND Senior Adult Age	Critical Health

Model 2:

Rules	Output
IF Low Humidity AND Adolescent Age AND Cold Body Temperature	Risky
IF Low Humidity AND Adolescent Age AND Normal Body Temperature	Low Risk
IF Low Humidity AND Adolescent Age AND Hot Body Temperature	Risky
IF Low Humidity AND Adult Age AND Cold Body Temperature	Risky
IF Low Humidity AND Adult Age AND Normal Body Temperature	Low Risk
IF Low Humidity AND Adult Age AND Hot Body Temperature	High Risk
IF Low Humidity AND Senior Adult Age AND Cold Body Temperature	High Risk
IF Low Humidity AND Senior Adult Age AND Normal Body Temperature	Low Risk
IF Low Humidity AND Senior Adult Age AND Hot Body Temperature	High Risk
IF Normal Humidity AND Adolescent Age AND Cold Body Temperature	Risky
IF Normal Humidity AND Adolescent Age AND Normal Body Temperature	Low Risk
IF Normal Humidity AND Adolescent Age AND Hot Body Temperature	Risky
IF Normal Humidity AND Adult Age AND Cold Body Temperature	Risky
IF Normal Humidity AND Adult Age AND Normal Body Temperature	Low Risk
IF Normal Humidity AND Adult Age AND Hot Body Temperature	Risky
IF Normal Humidity AND Senior Adult Age AND Cold Body Temperature	Risky
IF Normal Humidity AND Senior Adult Age AND Normal Body Temperature	Low Risk
IF Normal Humidity AND Senior Adult Age AND Hot Body Temperature	High Risk
IF High Humidity AND Adolescent Age AND Cold Body Temperature	High Risk
IF High Humidity AND Adolescent Age AND Normal Body Temperature	Low Risk
IF High Humidity AND Adolescent Age AND Hot Body Temperature	High Risk
IF High Humidity AND Adult Age AND Cold Body Temperature	High Risk
IF High Humidity AND Adult Age AND Normal Body Temperature	Low Risk
IF High Humidity AND Adult Age AND Hot Body Temperature	High Risk
IF High Humidity AND Senior Adult Age AND Cold Body Temperature	High Risk
IF High Humidity AND Senior Adult Age AND Normal Body Temperature	Low Risk
IF High Humidity AND Senior Adult Age AND Hot Body Temperature	High Risk

Appendix M: MATLAB Code

```
//Surface Plots

% define the co-ordinates along the x and y axis

x=[46.8 77.4 43.2 36.0 77.4 37.8 30.6 46.8 32.4];
y=[50.4 54.0 61.2 64.8 68.4 57.6 61.2 52.2 59.4];
z=[106.2 99.0 88.2 117.0 70.2 104.4 66.6 50.4 66.6];

Figure
scatter3(x,y,z, 'filled', 'MarkerEdgeColor', 'k');
grid on;
xlabel('Before Eating')
ylabel('During Day')
zlabel('After Eating')

box on

% define the co-ordinates along the x and y axis

x=[55.0 66.7 66.7 22.2 55.6 0.0 0.0 0.0 0.0 0.0 11.1 44.4 22.2 22.2 33.3];
y=[86.0 90.0 75.0 40.0 45.0 10.0 15.0 25.0 20.0 10.0 10.0 60.0 55.0 45.0 40.0];
z=[80.0 80.0 72.5 46.8 72.4 32.1 45.8 32.5 31.3 25.0 33.3 60.0 60.0 46.8 60.0];

[X,Y]= meshgrid(x,y);
Z=meshgrid(z);

surf(X,Y,Z);
xlabel('MEWS Results')
ylabel('Physician Results')
zlabel('FL model Results')
```