Informatics Institute of Technology
In Collaboration With
University of Westminster, UK

University of Westminster, Coat of Arms

Health Status Assessment in Remote Patient Monitoring Systems using Hybrid Machine Learning

Project Specifications & Project Design by
Mr. Hammadh Yusuf Mohamed Arquil
w17617808 / 2018128

Supervised by
Ms. Divya Premanantha

February 2023

This document is submitted in partial fulfilment of the requirements for
the BSc (Hons) Computer Science degree at
the University of Westminster.

ABSTRACT

In the healthcare sector, the field of remote patient monitoring is expanding quickly and giving healthcare professionals a tool to remotely monitor patient health data. The models utilized are inefficient and inaccurate, which are two drawbacks of the existing approaches used for assessing patient health status in remote patient monitoring systems.

In the past, remote patient monitoring systems that monitor vital signs have not frequently used hybrid machine learning models for health status assessments. To perform health status assessments on the patient’s vital signs, the author has decided to implement a novel hybrid machine learning model because hybrid models have an excellent track record of handling similar challenges.

After the development of the prototype model, a classification accuracy of 89.3% was gained utilizing the prototype model for hybrid health status assessment. The precision and recall scores, which are 0.91 and 0.95 respectively with an F1 score of 0.92 is quite adequate. Although the results of this prototype are promising, it is crucial to keep in mind that more analysis and testing are still needed to fully confirm the viability of this hybrid model for remote patient monitoring systems.

Keywords: Remote Patient Monitoring Systems, Hybrid Machine Learning, Automated Health Assessments, Vital Signs, Healthcare

Subject Descriptors:

- Computing methodologies — Machine learning
- Applied computing — Life and medical sciences — Health informatics.
- Computing methodologies — Machine learning — Machine learning algorithms — Ensemble methods
- Applied computing — Life and medical sciences — Consumer health.

CONTENTS

List of Tables

List of Figures

Table of Acronyms

Acronym	Description
RPM	Remote Patient Monitoring
ML	Machine Learning
DNN	Deep Neural Network
MIT	Massachusetts Institute of Technology
PHC	Portable Healthcare System
RNN:	Recurrent neural network
LSTM	Long Short-Term Memory
OOADM	Object-oriented Analysis and Design Methodology

Acronym	Description
RF	Random Forest
DT	Decision Trees
GB	Gradient Boosting
SSADM	Structured Systems Analysis and Design Methodology

CHAPTER 1: INTRODUCTION

1.1 Chapter Overview

In this chapter, the author provides an overview of the current state of remote patient monitoring solutions in healthcare delivery proposing a hybrid machine learning model to automate the health status/acuity risk assessment process of a patient to alert the healthcare providers. Also defined in this chapter by the author is the problems found, the aims and objectives of the project, challenges and the proposed contribution to the bodies of knowledge.

1.2 Problem Background

As the monitoring of diseases and treatment require people to go frequently to hospitals, the healthcare industry dramatically changes and constantly advances towards a more technological transition, remote monitoring and remote healthcare are rapidly growing fields of study and research (Malasinghe, Ramzan and Dahal, 2019). Remote healthcare which could be divided into several domains such as telehealth, mobile health, etc. refers to the use of techniques and technologies to monitor patients situated outside the hospital environment. Remote Patient Monitoring allows healthcare providers and caretakers to view and monitor patient health information remotely from a distance and allow them to take the necessary steps to prevent the patient from deteriorating (El-Rashidy et al., 2021). The usage of these monitoring systems contributes to reduced healthcare costs by further decreasing the number of hospital inpatient admissions and usage of facilities (Dhinakaran et al., 2022). Also, with the usage of the remote patient monitoring solutions, these resources can further improve the quality of life of its users (Salehi et al., 2020).

In the current age of healthcare delivery, there is a huge importance and need for analyzing and assessing medical data with better performance and greater accuracy of medical assessments by providing diagnoses and predictions done with the usage of machine learning models especially when humans cannot directly recognize certain abnormalities (Jayatilake and Ganegoda, 2021). Assessing a patient’s health status ahead of time would heavily improve a patient’s chances of recovery as a late diagnosis could lead to several serious complications on the patients’ health (Nallakaruppan and Kumaran, 2020). It is therefore clear that the neglect of accurate or late diagnosis often places the patient at the risk of poor consequences, so early detection of such deterioration enables immediate involvement from medical personnel (Elliott and Endacott, 2022).

1.3 Problem Definition

Traditional healthcare delivery systems require doctors to view the patient, diagnose the disease, and advise the future course of action for the patient (Vinutha, Kavyashree and Raju, 2022) although these methods pose some difficulty in terms of patient's mobility and daily workflow (Malasinghe, Ramzan and Dahal, 2019). Traditional remote patient monitoring further assists in that data transaction in healthcare delivery allowing doctors to assess the data and make decisions for the patient from afar while allowing patients to continue with their daily workflow. These systems too have their own limitations, one of them being due its remote nature certain emergencies cannot be accurately predicted or diagnosed prior. With the usage of machine learning algorithms, this limitation can be resolved to allow us to automate this process of analyzing the data by detecting abnormalities in the patient data early on in time and alerting the relevant healthcare providers of the patient's condition (Lata Sahu et al., 2021).

Due to the nature of the data, available machine learning model integrations with RPM systems point out a number of performance concerns. (Tabassum et al., 2020) proposed a hybrid encoding technique which they used in conjunction with the Random Forest algorithm, although they had received sufficient accuracy, they suffered from overall deficient performance in their solution for large datasets due to the large covariance between the techniques used. (Gontarska et al., 2021) developed a deep neural network model to calculate risk scores for cardiovascular diseases using vital signs collected however they also suffered from mediocre performance due to the time series nature of the data used. Having said that, we understand that one of the root issues that most of the available solutions for integrating machine learning into RPM suffer from the performance of machine learning models leading to a lowered efficiency of such systems.

1.3.1 Problem Statement

Existing solutions for risk assessments utilizing machine learning models that are in remote patient monitoring systems have a series of model performance difficulties resulting in lower accuracy and poor efficiency of such systems.

1.4 Aims & Objectives

1.4.1 Research Questions

- **RQ1**: What improvements in hybrid machine learning can be taken into consideration to maximize RPM performance?
- **RQ2**: What benefits can the adopted hybrid model bring to RPM systems?
- **RQ3**: How can the proposed system assist medical professionals in delivering better healthcare?

1.4.2 Research Aims

This research aims to conduct the design, development, and evaluation of a hybrid machine-learning model to be used for acuity risk assessment and predict the future state of the health status of a patient using remote patient monitoring.

By utilizing the novel hybrid machine learning model that has been proposed, the research primarily intends to improve the performance and accuracy of health risk status assessments in remote patient monitoring systems.

1.4.3 Research Objectives

Table 1 - Research Objectives

Objective	Description	Learning Outcomes	Research Questions
Literature Review	Gather and review relevant information by reading, comprehending, and assessing previously published work. **RO1**: Conduct investigations and studies done on existing remote patient monitoring systems. **RO2**: Investigate the need for specific prediction models in RPM systems. **RO3**: Obtain and review insights on the architecture of systems in remote patient monitoring	LO2, LO4, LO5	RQ1
Data Gathering and Analysis	Collect and analyze the requirements of the project using relevant tools and techniques available. **RO4**: Gather requirements from the architecture of RPM systems. **RO5**: Collect available data on patient medical data and vital signs. **RO6**: Obtain views and recommendations from specialists in the respective field and technologies	LO1, LO2, LO3	RQ1, RQ2, RQ3

Research Design	Design a system that can successfully handle the issues that have been noted. **RO7**: Design an optimized hybrid framework capable of both diagnosis and prognosis for patient health status. **RO8**: Design a model with support for prognosis and diagnosis inputs.	LO1	RQ2, RQ3
Implementation	Implement the system that has been proposed to fill in the available research gaps. **RO9**: Implement a novel hybrid machine learning model capable of assessing patient health status efficiently with fewer false positives. **RO10**: Implement a remote patient monitoring system and integrate the developed model.	LO1, LO5, LO6, LO7	RQ1, RQ2
Testing and Evaluation	Effectively evaluate the implemented model and the RPM system using recommended techniques. **RO11**: Create a test plan with test cases, unit testing, performance testing and integration testing. **RO12**: Test and evaluate the model against several benchmarking	LO4	RQ2, RQ3
Documentation	Document the progression of the research project.	LO6, LO8	RQ1

1.5 Research Novelty

1.5.1 Problem Novelty

This project addresses a novel problem in remote patient monitoring systems where there is a need for an automated method that is optimized for assessing patients' health state without relying on threshold limits set by doctors.

1.5.2 Solution Novelty

The problem outlined above is addressed in this research by introducing a novel hybrid machine learning model architecture that would diagnose the patient's vital signs to show the current status level, predict the patient's health's future condition, and alert medical staff in advance to prevent further deterioration. The suggested hybrid model is new in its methodology because it has not been widely used in remote patient monitoring systems and is anticipated to outperform earlier research.

1.6 Research Gap

Based on existing research, there is a need for an optimized and automated method of evaluating patients' health status in remote patient monitoring systems leveraging machine learning models without relying on threshold limitations defined by doctors. (Lata Sahu et al., 2021). Due to the characteristics of the dataset used, some of the available studies have shown that their models did not give the desired performance. (Gontarska et al., 2021).

However, hybrid models in particular have not previously been extensively employed in remote patient monitoring systems; as a result, in this research, the author proposes a novel hybrid machine learning model that would diagnose the patient's vital signs to show the present status level, forecast the future condition of the patient’s health that it is progressing towards and utilize RPM systems to inform the healthcare personnel in advance to stop further patient deterioration.

1.7 Project Contributions

1.7.1 Problem Domain Contribution

The author proposes a comprehensive remote patient monitoring system capable of assessing the current health risk status of a patient based on his/her vital signs utilizing machine learning techniques. The system would also be capable of providing a prognosis of patient’s health status and alert the relevant healthcare providers to take appropriate action to prevent future deterioration.

1.7.1 Research Domain Contribution

This project proposes an implementation of a novel hybrid machine-learning model which will be developed to perform a health status risk assessment of a patient using vital signs collected in a remote patient monitoring system. Given the nature of the medical data employed, it is expected that this novel hybrid model will perform more effectively than previous works that have offered similar outputs.

1.8 Research Challenge

The author faces a few obstacles during the project's timeline, some of which are listed below. Given the lack of literature currently available on health status risk assessment using hybrid machine learning models in remote patient monitoring systems, the following challenges are some of those faced by the author:

- Receiving access to a large critical healthcare dataset and undergoing required training to handle the data provided.
- Identifying the limitations of available literature to develop a novel framework that would provide an optimized performance during diagnosis and prognosis.
- Analyzing and utilizing appropriate tools, techniques, technologies and libraries to develop the hybrid machine learning model which will accept patient vital signs.

1.9 Chapter Summary

This chapter presented the problem, the gap in the available literature, the aim and objectives of the project where its learning outcomes has been mapped to the learning outcomes of the university module. the research challenges the author faces, and the contribution to the bodies of knowledge that the author would follow over the next several months to present the deliverables.

CHAPTER 2: SYSTEM REQUIREMENTS SPECIFICATION

2.1 Chapter Overview

This chapter discusses the system stakeholders, requirement elicitation techniques, functional and non-functional requirements based on priority level, use case diagrams, and use case descriptions of the system.

2.2 Rich Picture Diagram

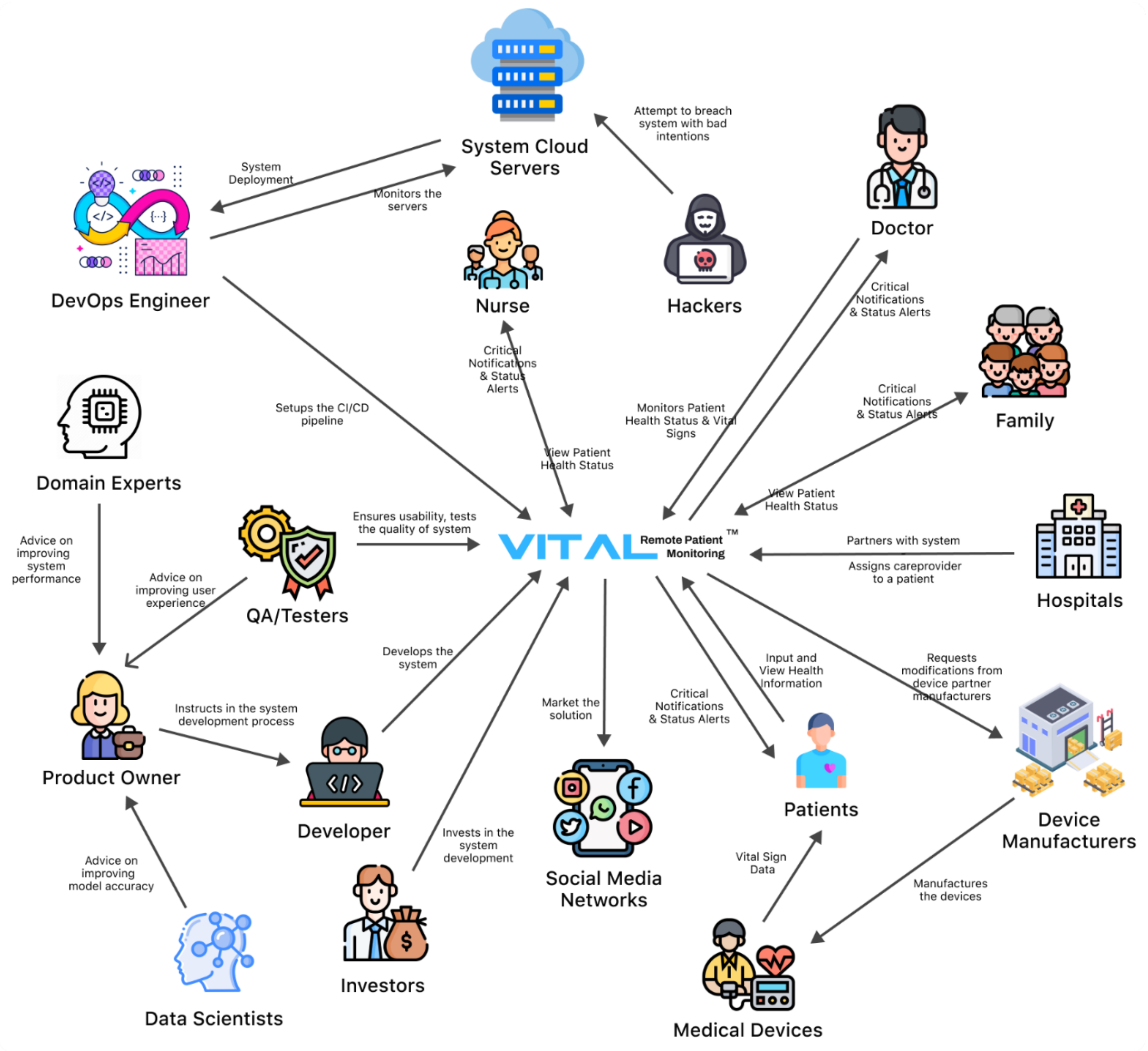

System Cloud Servers
Attempt to breach system with bad intentions
Doctor
System Deployment
Monitors the servers
DevOps Engineer
Nurse
Hackers
Critical Notifications & Status Alerts
Family
Critical Notifications & Status Alerts
Setups the CI/CD pipeline
Monitors Patient Health Status & Vital Signs
Critical Notifications & Status Alerts
Domain Experts
View Patient Health Status
View Patient Health Status
Ensures usability, tests the quality of system
VITAL Remote Patient Monitoring ™
Partners with system
Assigns careprovider to a patient
Hospitals
Advice on improving system performance
Advice on improving user experience
QA/Testers
Develops the system
Input and View Health Information
Requests modifications from device partner manufacturers
Instructs in the system development process
Market the solution
Critical Notifications & Status Alerts
Product Owner
Developer
Patients
Device Manufacturers
Social Media Networks
Invests in the system development
Vital Sign Data
Advice on improving model accuracy
Manufactures the devices
Investors
Data Scientists
Medical Devices

Figure 1 - Rich Picture Diagram *(Self-Composed)*

The rich picture diagram above showcases the flow of interaction within the system which would be used to understand who and what interacts with the system to make it functional.

2.3 Stakeholder Analysis

The onion model below showcases all the major stakeholders who can be associated with the system with a description of their involvement in the stakeholder viewpoints.

2.3.1 Stakeholder Onion Model

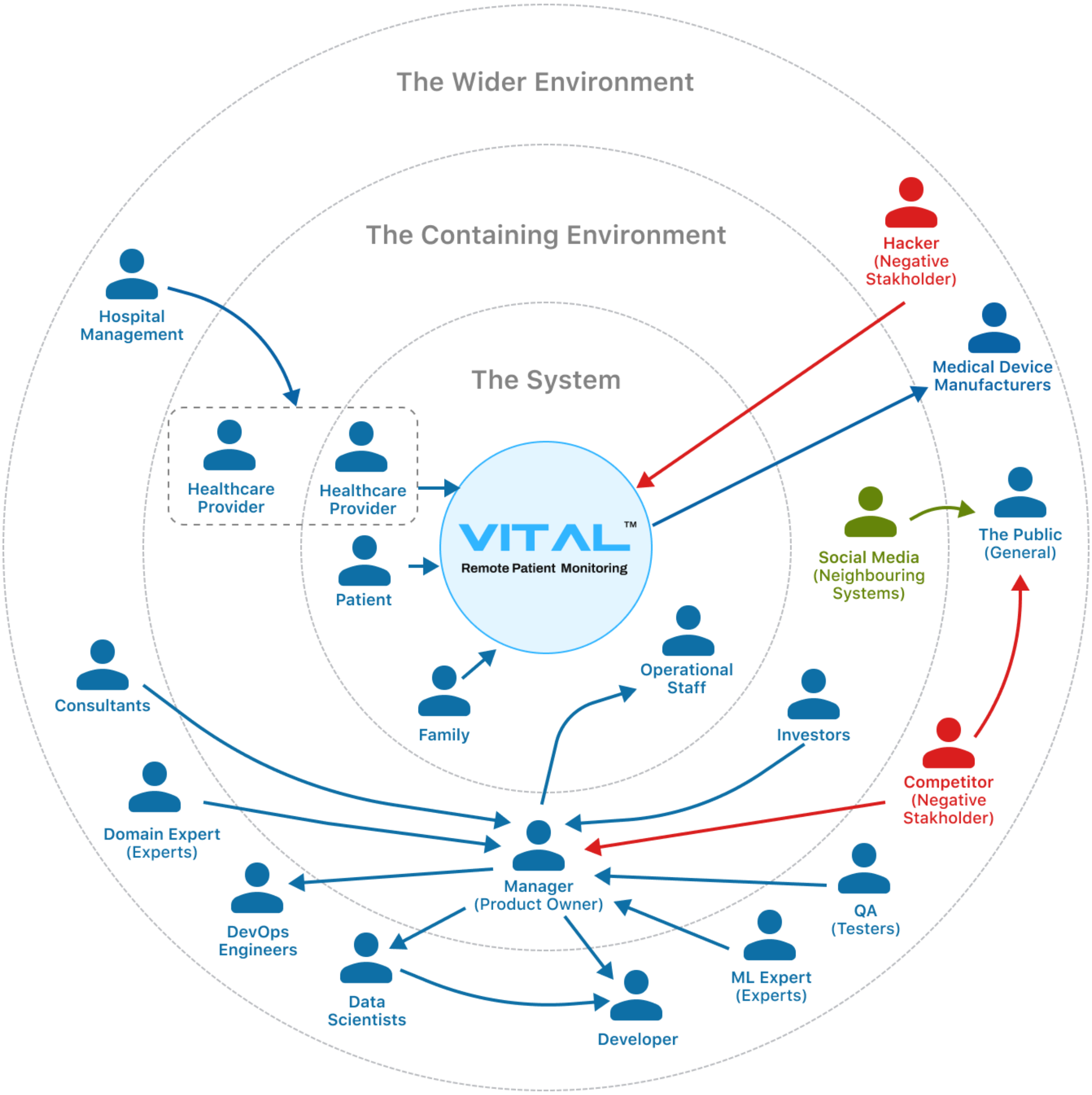


Figure 2 - Stakeholder Onion Model *(Self-Composed)*

The above figure represents the onion model of the VitalHealth system. Detailed description of all the system stakeholders is provided below.

2.3.2 Stakeholder Viewpoints

The table below briefly describes each of the stakeholders and their roles in the system.

Table 2 - Stakeholder Viewpoints

Stakeholder	Role	Description
Patient	Functional Beneficiary	It becomes easier for patients to receive medical monitoring and assessments from the comfort of their home, reduces financial burdens and ease of access to their healthcare providers
Family/Guardian	Functional Beneficiary	It becomes easier for family members to monitor the health of the patient while away and receives alerts during emergencies.
Healthcare Provider	Functional Beneficiary - Financial Beneficiary	it allows doctors to monitor patients from a distance and get alerted when there are irregularities allowing them to manage and monitor more patients tending to those in need of immediate services.
Developer	Financial Beneficiary, Maintenance	Develops the system and works on new features/functionalities
Manager/Product Owner	Financial Beneficiary, Administration	Oversees the team, providing direction to the programmers and support employees.
Domain & ML Experts	Expert & Quality Regulator	Offers expertise, guidance and insights into the domain and technologies to enhance system performance.
Data Scientists	Expert & Quality Regulator	Improves the performance of the system's models and algorithms.
DevOps Engineers	Operational, Deployment & Upkeep of the System	Ensures that the system is fully operational in the cloud and is serving users without being overloaded.

Stakeholder	**Role**	**Description**
Testers	Operational, Quality Assurance	Tests the system and assesses whether it is acceptable for deployment in production
Operational Staff	Operational, System Support	Maintains system functionality and responds to user concerns and requests.
Medical Device Manufacturers	Financial Beneficiary	Creates the medical equipment that patients use. works closely with the system and adjusts device output to suit system requirements
Competitor	Negative Stakeholder	May study the product and build competing applications that can outperform the system
Hackers	Negative Stakeholder	If proper security measures aren't met, it allows hackers to manipulate and access system information, highly confidential patient information with malicious intentions
Investors	Financial Beneficiary	Invests in the system development and benefits off the profit received.
Hospital Management	Operational, Administration	Works together with the system to add healthcare providers, and assign patients to the doctors

2.4 Requirements Gathering

2.4.1 Techniques for Gathering Requirements

Table 3 - Techniques for Gathering Requirements

Method 1: Literature Reviews
During the initial stages of the project, the author has performed thorough analysis of the literature available to identify the available research gaps in the chosen domain where similar systems and technologies that could be integrated into such systems were studied
Method 2: Survey Questionnaire

The author created a questionnaire to gather needs and opinions from potential users of the proposed system. This questionnaire would help the author better understand the target users and determine whether or not such a system would be beneficial to them. It would also help the author to determine what features and functionalities would be useful to them.
Method 4: Prototyping
Since the author had decided to use Agile development, prototyping would allow him to test and evaluate the system while highlighting any areas that needed improvement.

2.5 Analysis of Gathered Results

2.5.1 Findings from Literature Review

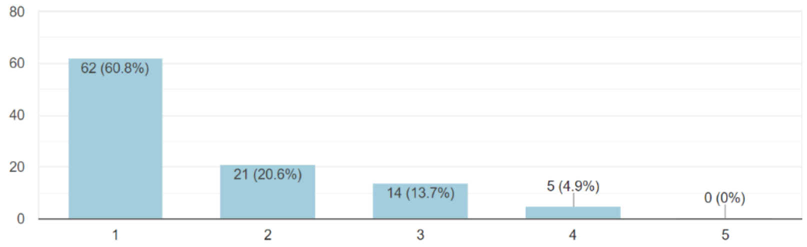
Table 4 - Findings from Literature Review

Finding	**Citation**
After reviewing the literature, the author had identified that automated health status alerts generated in cloud based remote patient monitoring system would not only heavily benefit the patients but the care providers aswell by reducing costs on both sides while keeping a constant monitor on the patient vital signs.	(Lata Sahu et al., 2021; El-Rashidy, N. et al., 2021)
It was identified while exploring different technologies that could be applied to the domain that available machine learning models that were used in remote patient monitoring systems came back with several performance issues.	(Gontarska et al., 2021; Chang et al., 2019; Vinutha et al., 2022)
The author identified that the vital sign predictors that were often used in health status assessments of patients were the heart rate, temperature, respiratory rate, oxygen saturation, systolic and diastolic blood pressure were essential to the patient monitoring process.	(Naemi, A. et al., 2021)
Allowing the usage of physician set thresholds for the vital signs allows the models to be more patient specific for the prediction model.	(Dhinakaran, M et al., 2022)

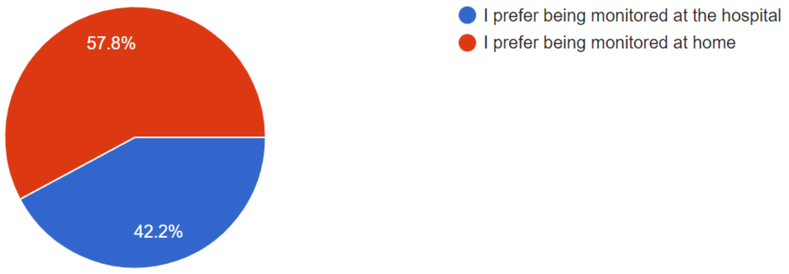
2.5.2 Findings from Survey

Table 5 - Findings from Survey

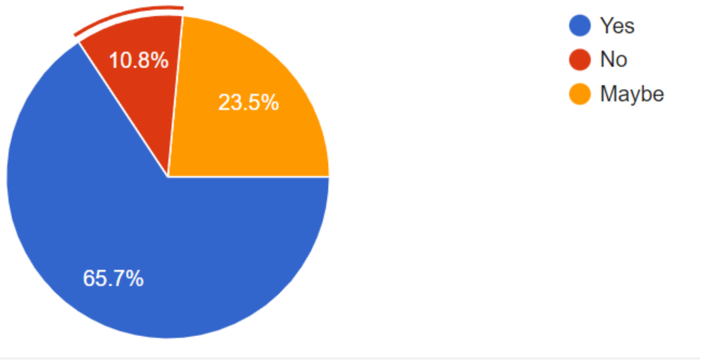
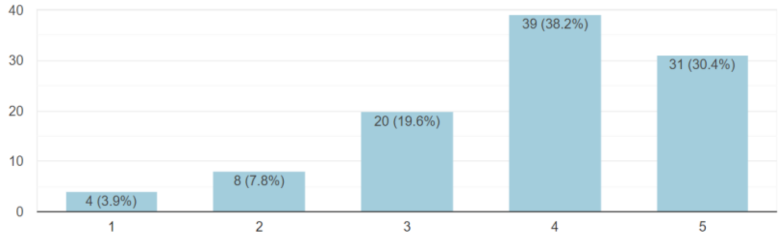
<table>
<tr><td><b>Question</b></td><td><b>How often do you visit the hospital for medical checkups/assessments?</b></td></tr>
<tr><td><b>Aim of Question</b></td><td>To understand how often a potential user might go to the hospital for assessments.</td></tr>
<tr><td colspan="2"><b>Findings & Conclusions</b>



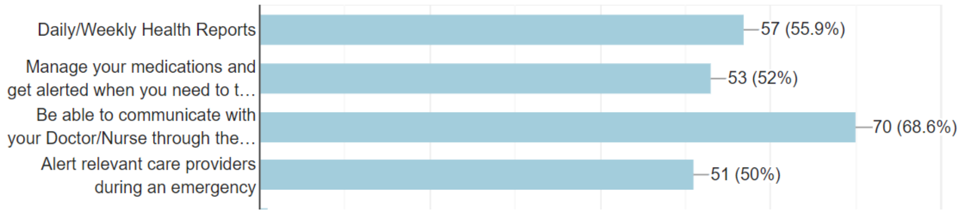
Based on the responses, it is clear that most of the participants don't visit the hospital frequently; the data revealed that only 5% of them would do so often, while more than 60% visited the hospital just sometimes or hardly at all.</td></tr>
<tr><td><b>Question</b></td><td><b>If given the option, would you prefer your health being monitored at home instead?</b></td></tr>
<tr><td><b>Aim of Question</b></td><td>To understand the preference of a potential user of whether they prefer being monitored at the hospital over being monitored at home.</td></tr>
<tr><td colspan="2"><b>Findings & Conclusions</b>



We might conclude from these findings that some of the participants are still somewhat hesitant about adopting home-based remote monitoring systems. Where 42.2% of individuals preferred being monitored at the hospital, only 57.8% of participants preferred being monitored at home.</td></tr>
<tr><td><b>Question</b></td><td><b>Would you prefer an automated system where your doctors will be alerted when there are irregularities in your health?</b></td></tr>
</table>

Aim of Question	To understand if such a system was available, would the participants prefer it where their doctor would be alerted during irregularities in their health.
Findings & Conclusions  While 23.5% of respondents replied Maybe, which could indicate that they are unsure at the time of this study, 65.7% of participants stated they preferred having an automated system where their doctors will be alerted when there are deviations in their health. 10.8% of the participants thought that having a system like this was not something they preferred. Overall, it's encouraging to see that the majority of participants are eager to use such a system!	
Question	**How beneficial would a system like this be for you?**
Aim of Question	To understand how beneficial the proposed system would be to them?
Findings & Conclusions  The majority of participants believe that the suggested system would be highly helpful to them, as shown by the histogram of the replies gathered from the participants. It is incredibly exciting to receive such favorable comments because they are the primary target audience.	
Question	**What additional features would you like to see in such systems?**
Aim of Question	To identify any additional features that may be added to the system to improve usability and user experience.

Findings & Conclusions



Majority of the participants felt that being able to communication directly with their care provider through the system would help improve their user experience, where others suggested daily/weekly health reports, a medication management and reminder system and being able to alert care providers manually during an emergency.

2.5.3 Findings from Prototyping

The findings from the prototyping of the Hybrid Health Status Assessment machine learning model revealed the effectiveness of the model in accurately predicting the health status of patients based on their vital signs. During testing, the model was able to successfully take in inputs of heart rate, temperature, respiratory rate, oxygen saturation, systolic and diastolic blood pressure and produce the patient's acuity level which is used as the health status level. The results of the prototyping confirmed the viability of the model and its potential to be integrated into a remote patient monitoring system for the minimum viable product.

2.6 Summary of Findings

Table 6 - Summary of Findings

ID	Finding	Literature Review	Survey	Prototyping
1	The proposed system would heavily benefit the users which includes both the patients and care providers	✓	✓	
2	Machine learning models used in remote patient monitoring systems have performance issues	✓		
3	Essential vital signs that are easily accessible to the public which are used in health status assessments (heart rate, temperature, respiratory rate, oxygen	✓	✓	✓

	saturation, systolic, and diastolic blood pressure)			
4	Having physician-set thresholds can make prediction models more patient specific	✓		
5	The system's performance would depend on having a sufficient amount of well-cleaned and pre-processed data.			✓
6	Not many people are aware of remote patient monitoring solutions, and many are still reluctant to use the available solutions		✓	
7	Some patients need to segment based on different diseases categories and vital tracking based on that		✓	

2.7 Context Diagram

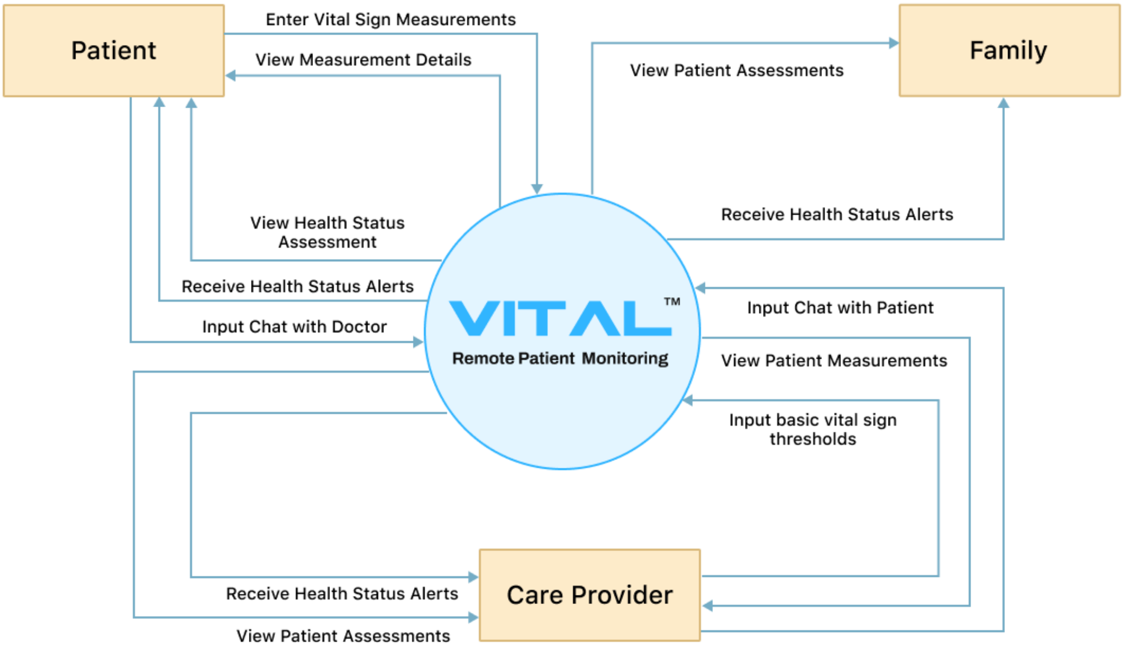

Patient
Enter Vital Sign Measurements
View Measurement Details
View Patient Assessments
Family
View Health Status Assessment
Receive Health Status Alerts
Receive Health Status Alerts
Input Chat with Doctor
VITAL™
Remote Patient Monitoring
Input Chat with Patient
View Patient Measurements
Input basic vital sign thresholds
Receive Health Status Alerts
Care Provider
View Patient Assessments

Figure 3 - Context Diagram *(Self-Composed)*

2.8 Use Case Diagram

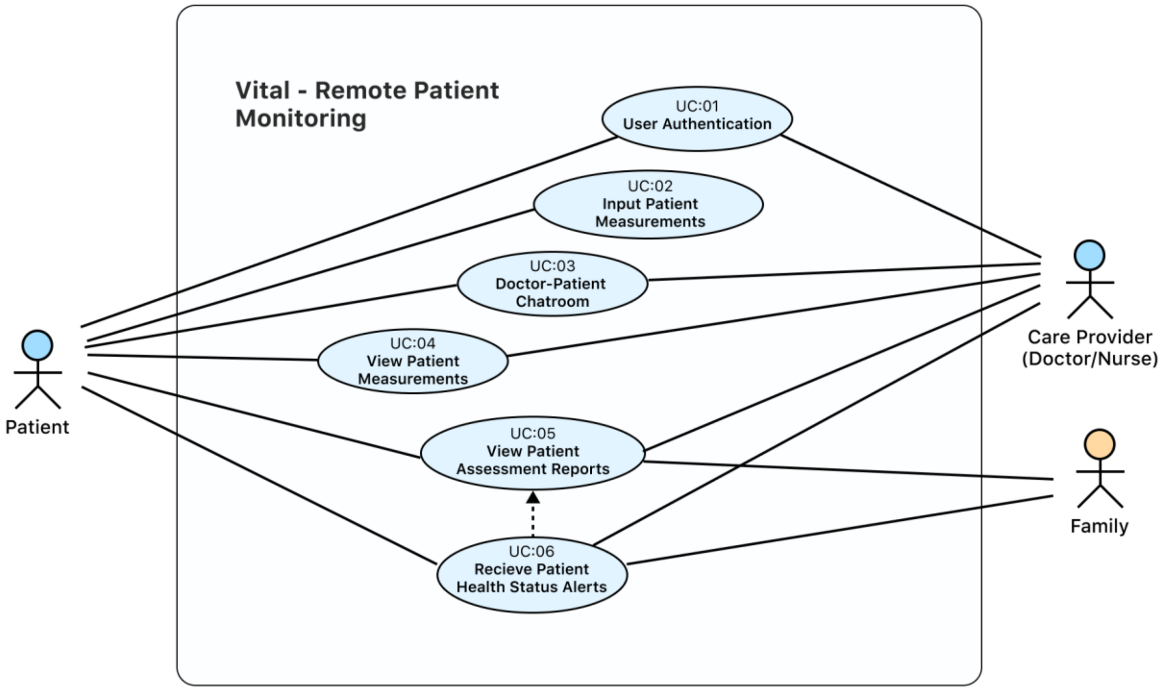

Vital - Remote Patient Monitoring
UC:01
User Authentication
UC:02
Input Patient Measurements
UC:03
Doctor-Patient Chatroom
UC:04
View Patient Measurements
UC:05
View Patient Assessment Reports
UC:06
Recieve Patient Health Status Alerts
Patient
Care Provider (Doctor/Nurse)
Family

Figure 4 – Use case Diagram. *(Self-Composed)*

2.9 Use Case Description

The core use case descriptions that were identified are provided below.

2.9.1 Input Patient Measurements

Table 7 – Use case Description - Input Patient Measurements

Use Case	Input Patient Measurements
ID	UC:02
Description	The patient can input their vital sign measurements into the system which are required.
Primary Actor	Patient
Supporting Actors (if any)	None
Stakeholders (if any)	Doctor, Family
Pre-Conditions	Patient should be logged in to the correct account.
Post Conditions	Success scenario: The patient will be presented with a basic overview of the vital measurements added to the system.
Main Scenario	- Patient selects the add measurement option.

	- Patient selects measurement type and inputs the required fields based on that measurement. - The measurement is submitted and processed. - The system then navigates the patient to the vital measurement page containing a list of all the measurements added for that particular vital sign.
Alternative Scenarios	The patient can be navigated to the health status assessment screen with an overall assessment of the health status and vital signs used for it. However, this would require the patient to have added other required vital sign measurements previously.

2.9.2 View Patient Assessment Reports

Table 8 - Use case Description - View Patient Assessment Reports

Use Case	View Patient Assessment Reports
ID	UC:05
Description	Displays a report of the patient's health status generated automatically from the vital sign measurements added by the user.
Primary Actor	Patient
Supporting Actors (if any)	Doctor, Family
Stakeholders (if any)	None
Pre-Conditions	All measurements would need to be added previously. At least one health status assessment report will have needed to have been generated.
Post Conditions	Success scenario: The patient is presented with the health status assessment reports generated
Main Scenario	- The health status report is automatically generated once patient enters all required vital signs. - The patient selects the reports screen option on the navigation bar. - The system loads all available health status assessments generated by the system.

	- User selects the report he wishes to view. - The system opens the report in a new screen showing all the details on the report mentioning the status, the vital signs used to come to that conclusion and any alerts related to it.
Alternative Scenarios	If no previous health status report is generated, the user is presented with a screen containing all the pending measurements required to generate a report.

2.10 Requirements

2.10.1 Functional Requirements

Based on their significance, the MoSCoW approach was used to rank the system demands in order of priority.

Table 9 - Priority Descriptions - MoSCoW approach

Priority Level	**Description**
Must have (M)	The demand at this level is the primary functional requirement for a prototype, and it must be carried out.
Should have (S)	While not strictly necessary for the desired prototype to function, these requirements do provide a lot of value.
Could have (C)	Requirements that are deemed as desirable and optional but aren't essential to the system
Will not have (W)	Requirements that will not be implemented due to certain constraints but could be integrated as future work.

Table 10 - Functional Requirements

FR ID	**Requirement**	**Priority**	**Use Case**
FR1	Users must be able to add vital sign measurements into the system	M	UC:02
FR2	Doctors should be able to monitor patient vital signs and status through the system	S	UC:04
FR3	The system should be able to fetch the patient vital signs collected over a period of time and make a prognosis	M	UC:05

FR4	The system should take the patient vital signs collected and provide an immediate diagnosis/status check	M	UC:05
FR5	The system should be able to generate daily/weekly health reports based on data collected/generated	C	UC:05
FR6	The system should alert healthcare providers when there is an irregularity in patient vital signs and health status level	S	UC:06
FR7	Integrating IOT medical devices into the system to automate the data collection process for continuous tracking	W	UC:02
FR8	Allow family members to view details on the patients' health vital signs/health status and get alerted when certain conditions are met.	C	UC:05, UC:06
FR9	Having a chatroom system, which would allow the patient and doctor to communicate directly	C	UC:03
FR10	Include multi-language translation support to the system to allow support for a wider variety of people	C	-

2.10.2 Non-functional Requirements

Table 11- Non-Functional Requirements

NFR ID	Requirement	Description	Priority Level
NFR1	Performance	The system should be capable of providing the assessment classifications as soon as possible once the required data is available.	Important
NFR2	User Experience	The usability and user experience of the system is one of the priorities of the system to ensure it is makes it easier for the stakeholders and that it is simple to use.	Desirable
NFR3	System Accuracy	It is very important to make sure the accuracy provided by the system is high.	Important
NFR4	Security	The system should make use of proper techniques to prevent any attackers from accessing the patient data.	Important

NFR5	Scalability	The system should be scalable to allow future growth within the system.	Desirable

2.11 Chapter Summary

This chapter discussed the various system stakeholders, different techniques for gathering requirements, the distinction between functional and non-functional requirements based on their priority level, and an overview of the system's use case diagrams and descriptions. The chapter's goal was to provide a thorough understanding of the process of gathering and organizing system requirements.

CHAPTER 3: DESIGN

3.1 Chapter Overview

This chapter describes the system's design, which includes a tiered architecture, a UML component diagram, level 1 and level 2 data flow diagrams, system sequence diagrams, the user interface design, and a diagram of the system's process flow.

3.2 Design Goals

Table 12 - Design Goals

Design Goal	Description
Performance	To ensure that the patient's health state is not critical, the system must be able to produce results as fast as possible, making system performance a top priority.
Quality & Correctness	A late or inaccurate assessment of the patient's health status could have fatal consequences; as a result, the system's accuracy and quality should be at the highest level possible with the data available at the moment.
Scalability	Due to the nature of the system, it must be easily expandable to accommodate future development and be able to manage hundreds or thousands of users so that inputs and generation can occur in the future with the least amount of work.
Usability	The primary goal of this system is to make it easier for patients to monitor their vital signs from the comfort of their own homes, hence it is crucial that it be simple to comprehend and use.
Adaptability	The system should enable the models used to be quickly swappable for improved performance and adaptation to the newer requirements based on the data currently available and anticipated requirements in the future. By ensuring the system can adapt to such changes, we can also ensure that it will function normally during the upgrading process.

3.3 High-Level Design

3.3.1 Three-Tiered Architecture

The system’s high level architecture design is showcased in the diagram below utilizing the tiered architecture containing 3 distinct tiers – The frontend (Presentation tier), The backend (Logic tier) and the Data tier.

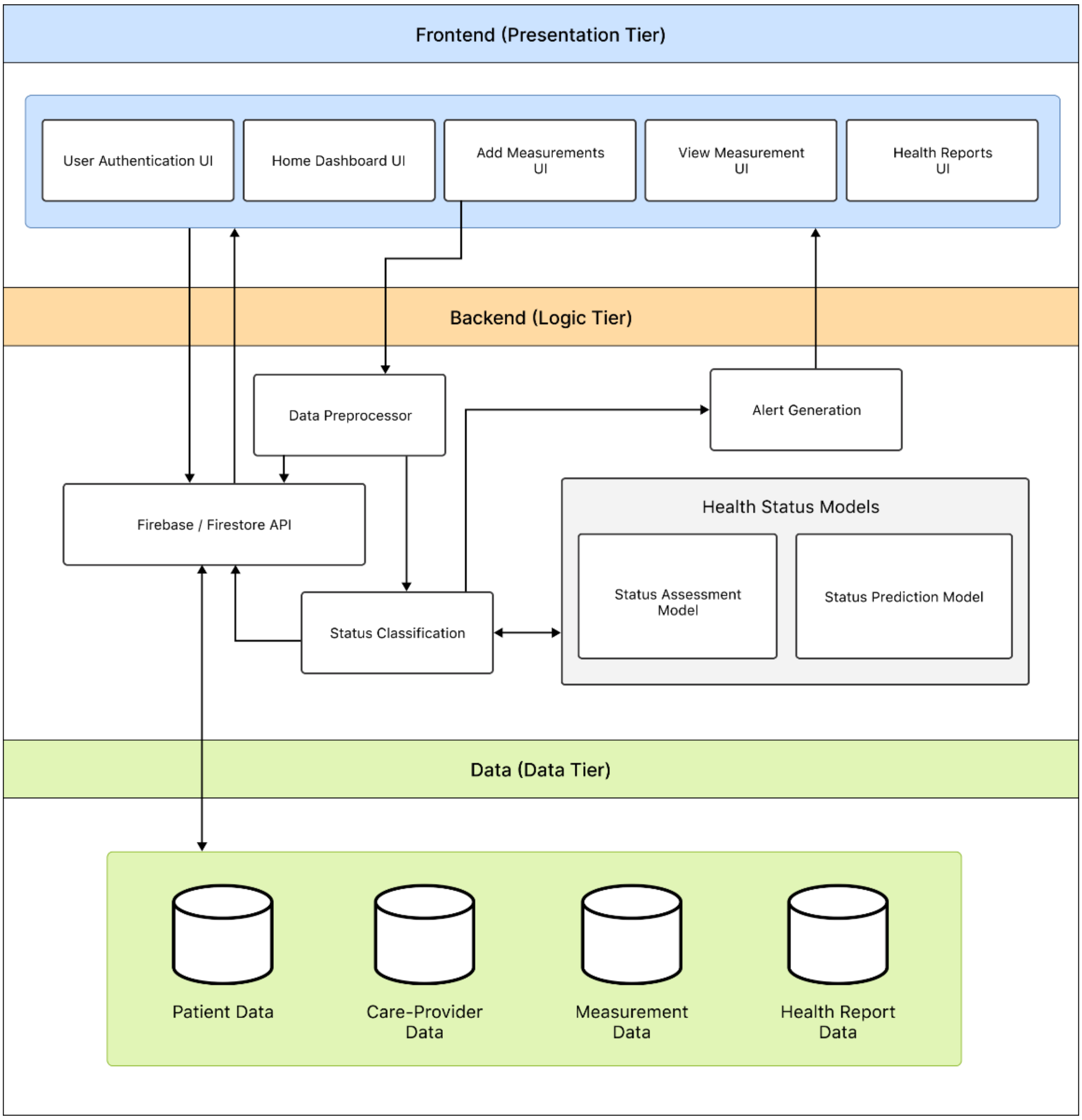


Figure 5 - High Level Architecture *(Self-Composed)*

3.3.2 Discussion of tiers

3.3.2.1 Frontend (Presentation Tier)

1. User Authentication UI – The user interface which allows users to login/register into the system.
2. Home Dashboard UI – The main dashboard homepage user interface which displays the patient’s last recorded measurements, any pending tasks etc.
3. Add Measurements UI – This interface is very adaptive to the measurement types; it allows the patient to input the measurement and relevant information to the vital sign.
4. View Measurement UI – This UI allows the patient to view all statistics and information regarding the measurement added.

5. Health Reports UI – This user interface allows the patient to view a list of previous health status assessments including classifications and predictions done by the system.

3.3.2.2 Backend (Logic Tier)

1. Data Preprocessor – To ensure that all data received from the front-end user interfaces is suitable for the models and database by preprocessing it before it is used within the system.
2. Firebase/Fire-store API – The primary API would be utilized to establish a connection to the database for the purpose of reading and writing data.
3. Status Classifier – the main module that is activated by the system to utilize the models whenever an assessment is necessary.
4. Health Status Models – The models that the system uses to produce health status assessment classification and health status predictions for the patient.
5. Alert Generation - This module deals with the system's alert creation when an evaluation is available and there is a need to inform the patient and a healthcare professional immediately.

3.3.2.3 Data (Data Tier)

1. Patient Data – The data gathered and stored on the patient.
2. Care Provider Data – A collection of the care providers' data required by the system to function.
3. Measurement Data – Data which is retrieved from the patient when they input it into the system to be monitored.
4. Health Report Data – A collection of data containing health status assessment reports of both classification and prediction assessments of the patient's health status.

3.4 System Design

3.4.1 Choice of Design Paradigm

The author had decided to switch the design methodology from Object Oriented Analysis and Design Methodology (OOADM) which was mentioned initially during the project proposal to Structured Systems Analysis and Design Methodology (SSADM) naturally based on the factors that the main core machine learning components do not specifically gain any benefit from an object-oriented design, also for the ease of implementation of the minimum viable product.

3.5 Design Diagrams

3.5.1 Data Flow Diagrams

3.5.1.1 Data Flow Diagram – Level 1

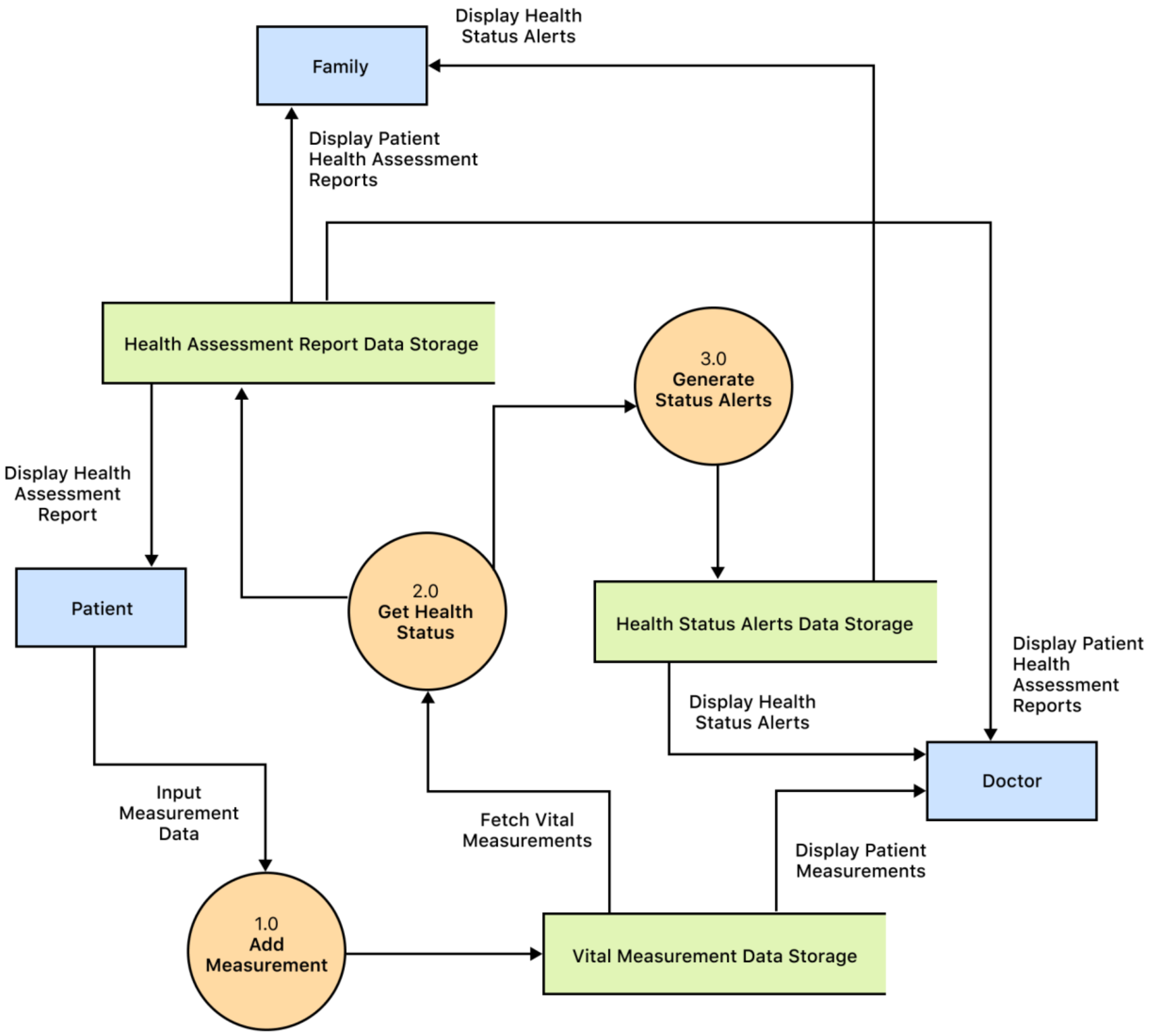

Display Health Status Alerts
Family
Display Patient Health Assessment Reports
Health Assessment Report Data Storage
3.0 Generate Status Alerts
Display Health Assessment Report
Patient
2.0 Get Health Status
Health Status Alerts Data Storage
Display Patient Health Assessment Reports
Display Health Status Alerts
Doctor
Input Measurement Data
Fetch Vital Measurements
Display Patient Measurements
1.0 Add Measurement
Vital Measurement Data Storage

Figure 6 - Data Flow Diagram - Level 1 *(Self-Composed)*

3.5.1.2 Data Flow Diagram – Level 2

3.5.1.2.1 Add Measurements

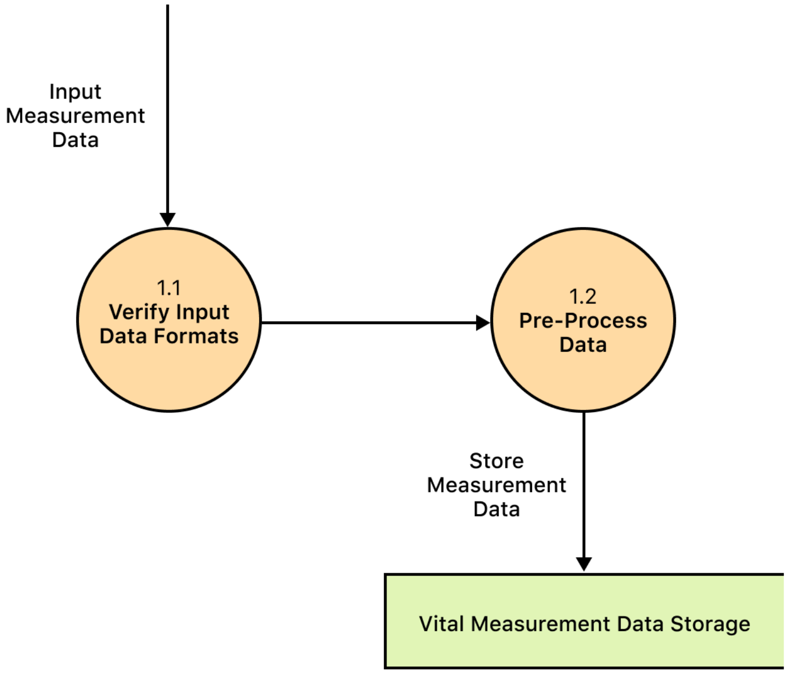


Figure 7 - Data Flow Diagram - Level 2 - Add Measurements *(Self-Composed)*

3.5.1.2.2 Get Health Status

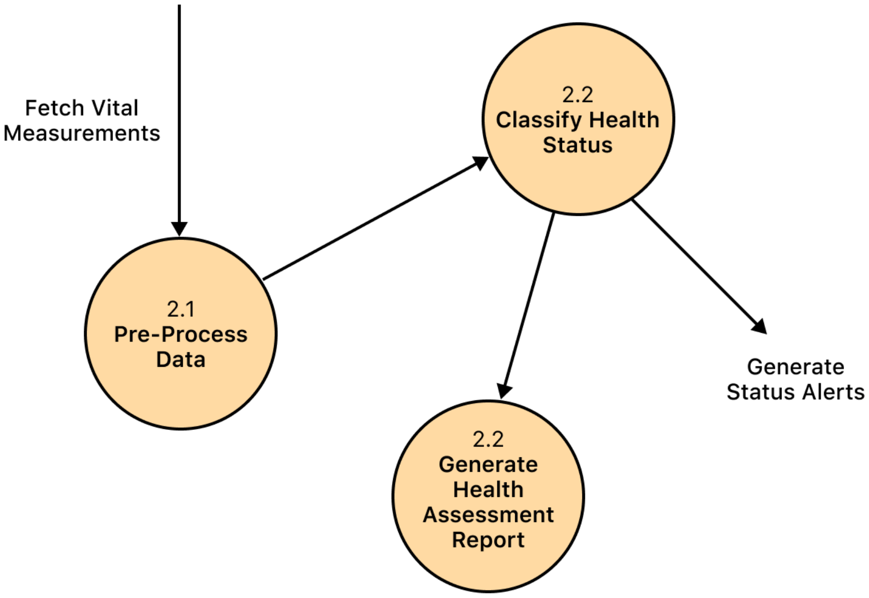


Figure 8 - Data Flow Diagram - Level 2 - Get Health Status *(Self-Composed)*

3.5.2 Sequence Diagram

3.5.2.1 Input Measurement

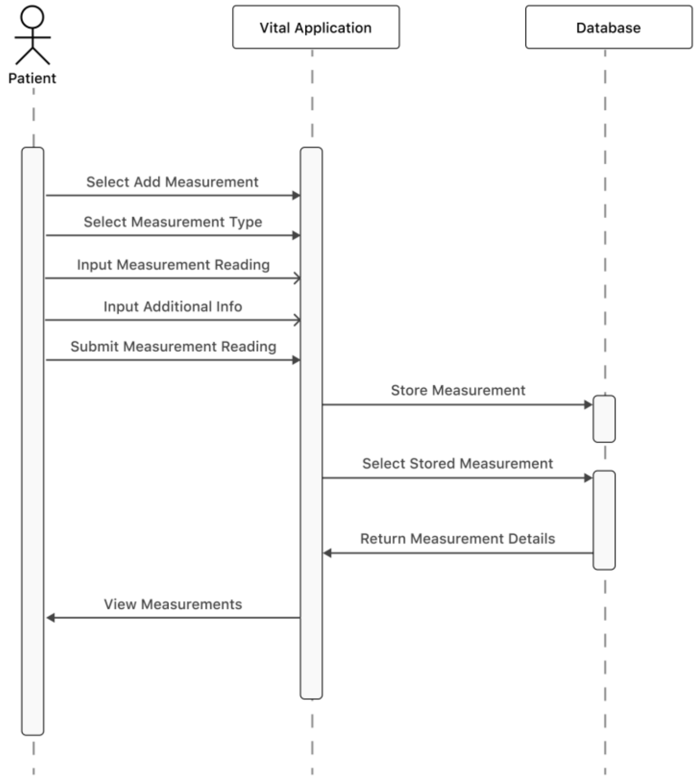


Figure 9 - Sequence Diagram - Input Measurements *(Self-Composed)*

3.5.2.2 View Patient Assessment Report

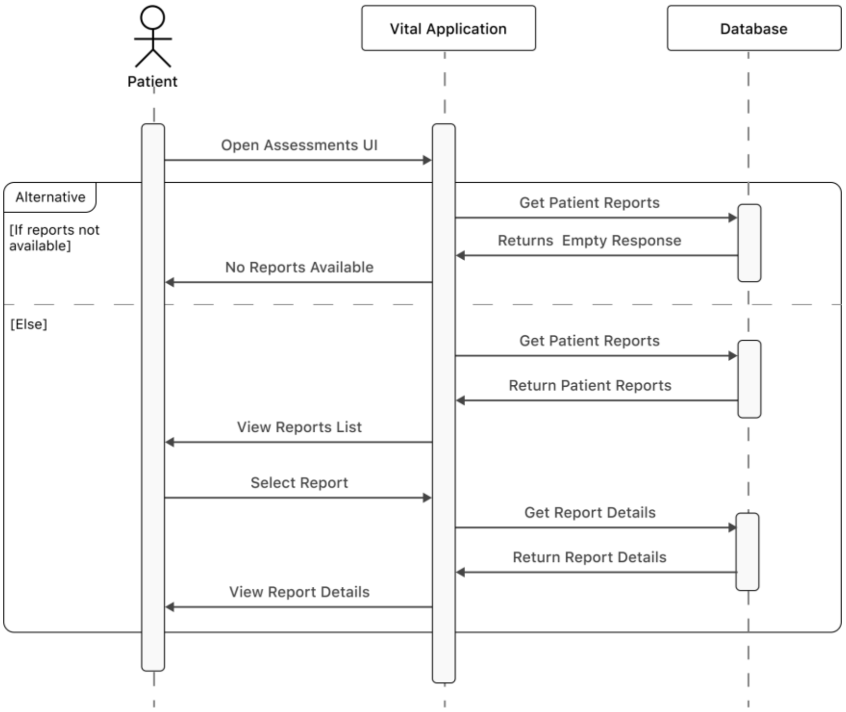
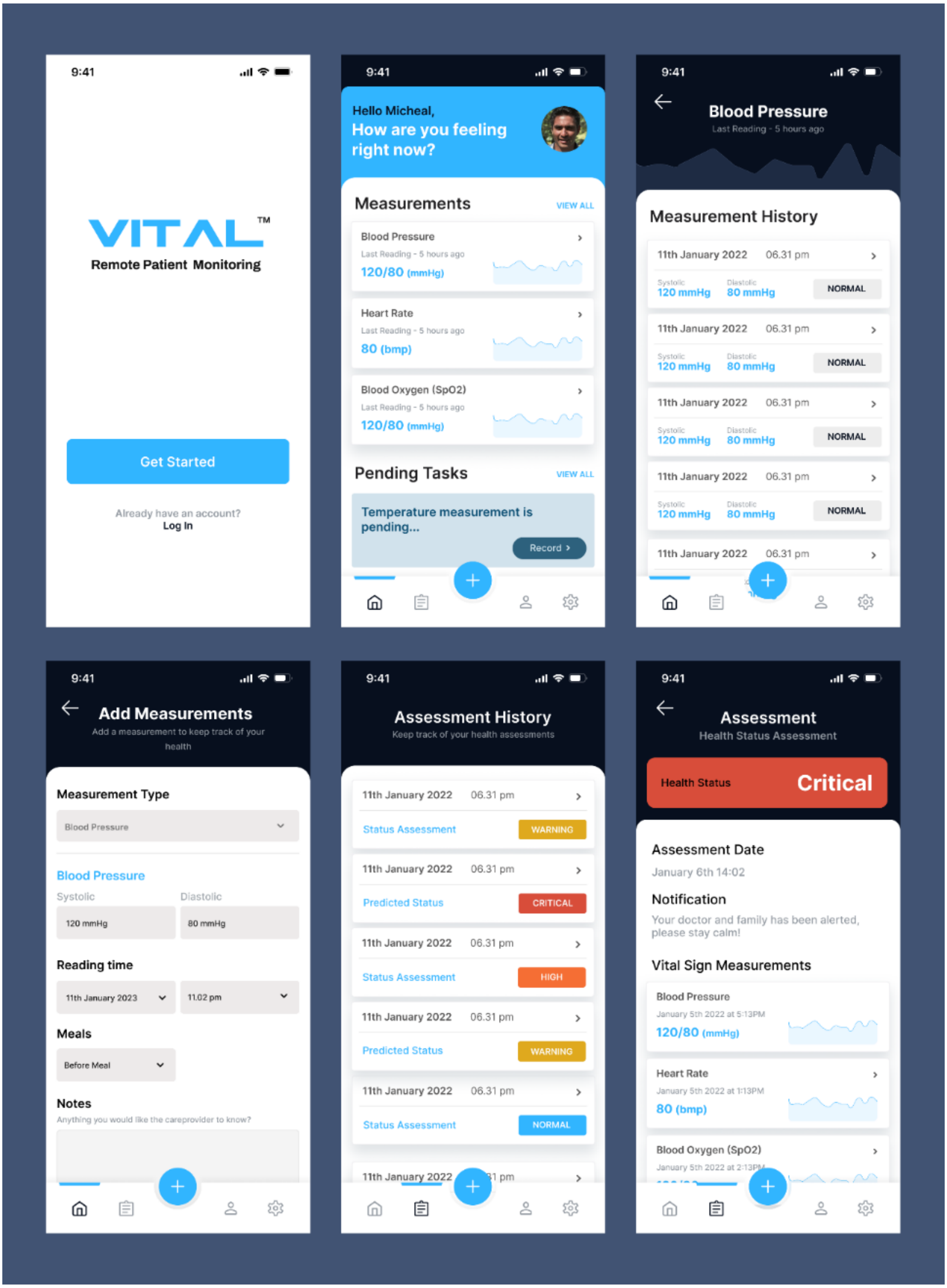


Figure 10 - Sequence Diagram - View Patient Assessment Report *(Self-Composed)*

3.5.3 UI Design

\



9:41

VITAL™

Remote Patient Monitoring

Already have an account?
Log In

9:41

Hello Micheal,
How are you feeling right now?

Measurements

Blood Pressure
Last Reading - 5 hours ago
120/80 (mmHg)

Heart Rate
Last Reading - 5 hours ago
80 (bmp)

Blood Oxygen (SpO2)
Last Reading - 5 hours ago
120/80 (mmHg)

Pending Tasks

Temperature measurement is pending...

9:41

Blood Pressure
Last Reading - 5 hours ago

Measurement History

11th January 2022 06.31 pm
Systolic 120 mmHg Diastolic 80 mmHg NORMAL

11th January 2022 06.31 pm
Systolic 120 mmHg Diastolic 80 mmHg NORMAL

11th January 2022 06.31 pm
Systolic 120 mmHg Diastolic 80 mmHg NORMAL

11th January 2022 06.31 pm
Systolic 120 mmHg Diastolic 80 mmHg NORMAL

11th January 2022 06.31 pm

9:41

Add Measurements
Add a measurement to keep track of your health

Measurement Type
Blood Pressure

Blood Pressure
Systolic: 120 mmHg
Diastolic: 80 mmHg

Reading time
11th January 2023
11.02 pm

Meals
Before Meal

Notes
Anything you would like the careprovider to know?

9:41

Assessment History
Keep track of your health assessments

11th January 2022 06.31 pm
Status Assessment WARNING

11th January 2022 06.31 pm
Predicted Status CRITICAL

11th January 2022 06.31 pm
Status Assessment HIGH

11th January 2022 06.31 pm
Predicted Status WARNING

11th January 2022 06.31 pm
Status Assessment NORMAL

11th January 2022 06.31 pm

9:41

Assessment
Health Status Assessment

Health Status Critical

Assessment Date
January 6th 14:02

Notification
Your doctor and family has been alerted, please stay calm!

Vital Sign Measurements

Blood Pressure
January 5th 2022 at 5:13PM
120/80 (mmHg)

Heart Rate
January 5th 2022 at 1:13PM
80 (bmp)

Blood Oxygen (SpO2)
January 5th 2022 at 2:13PM

Figure 11 - User Interface Design *(Self-Composed)*

3.5.4 System Process Flow Diagram

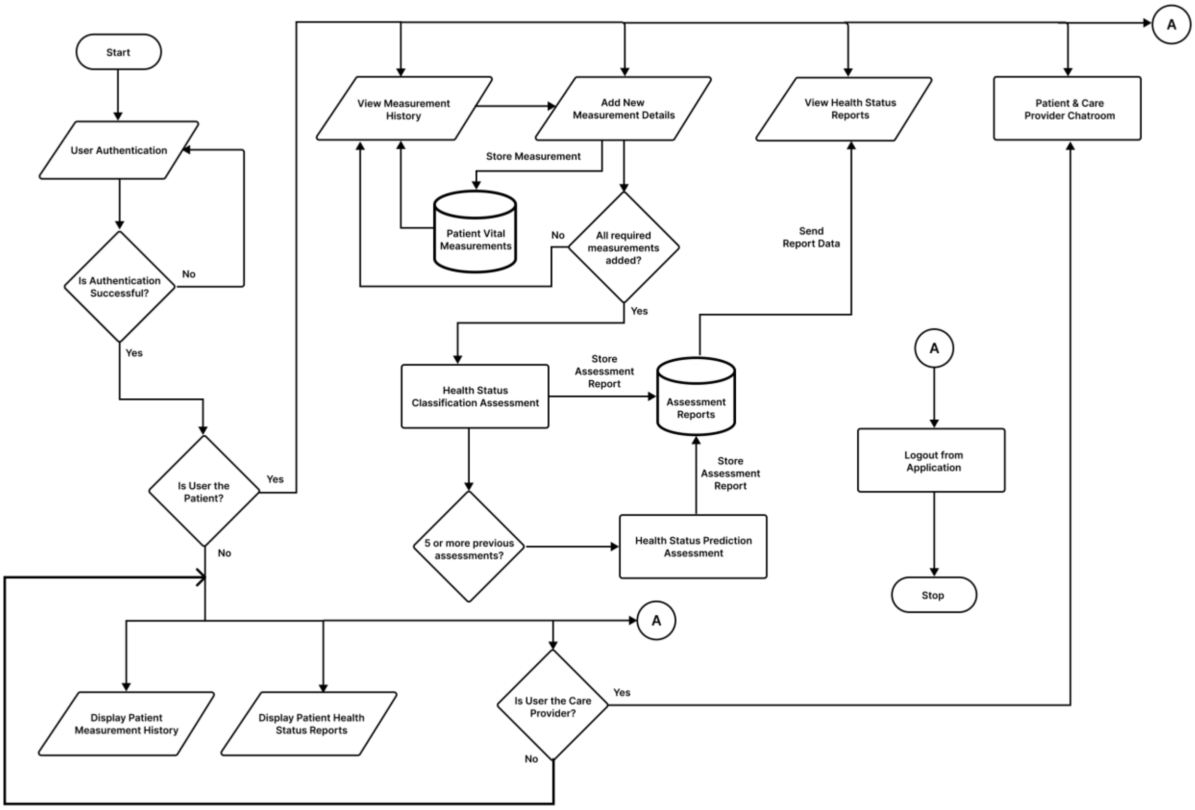

Start
User Authentication
Is Authentication Successful?
No
Yes
Is User the Patient?
Yes
No
View Measurement History
Add New Measurement Details
Store Measurement
Patient Vital Measurements
No
All required measurements added?
Yes
Health Status Classification Assessment
Store Assessment Report
Assessment Reports
Send Report Data
View Health Status Reports
Patient & Care Provider Chatroom
A
5 or more previous assessments?
Health Status Prediction Assessment
Store Assessment Report
A
Logout from Application
Stop
A
Display Patient Measurement History
Display Patient Health Status Reports
Is User the Care Provider?
Yes
No

Figure 12 - System Process Flow Diagram *(Self-Composed)*

3.6 Chapter Summary

This chapter presented the design of the system, which includes a tiered architecture, level 1 and level 2 data flow diagrams, system sequence diagrams, and a detailed description of the user interface design. The chapter also included a diagram showing the process flow of the system. This chapter provided a comprehensive understanding of the system design and how the different components and processes fit together to form the system as a whole.

CHAPTER 4: INITIAL IMPLEMENTATION

4.1 Chapter Overview

This chapter provides an overview of the core implementation of the research project, including the technologies, languages, and supporting tools used, and explains the rationale behind the author’s choices.

4.2 Technology Selection

4.2.1 Technology Stack

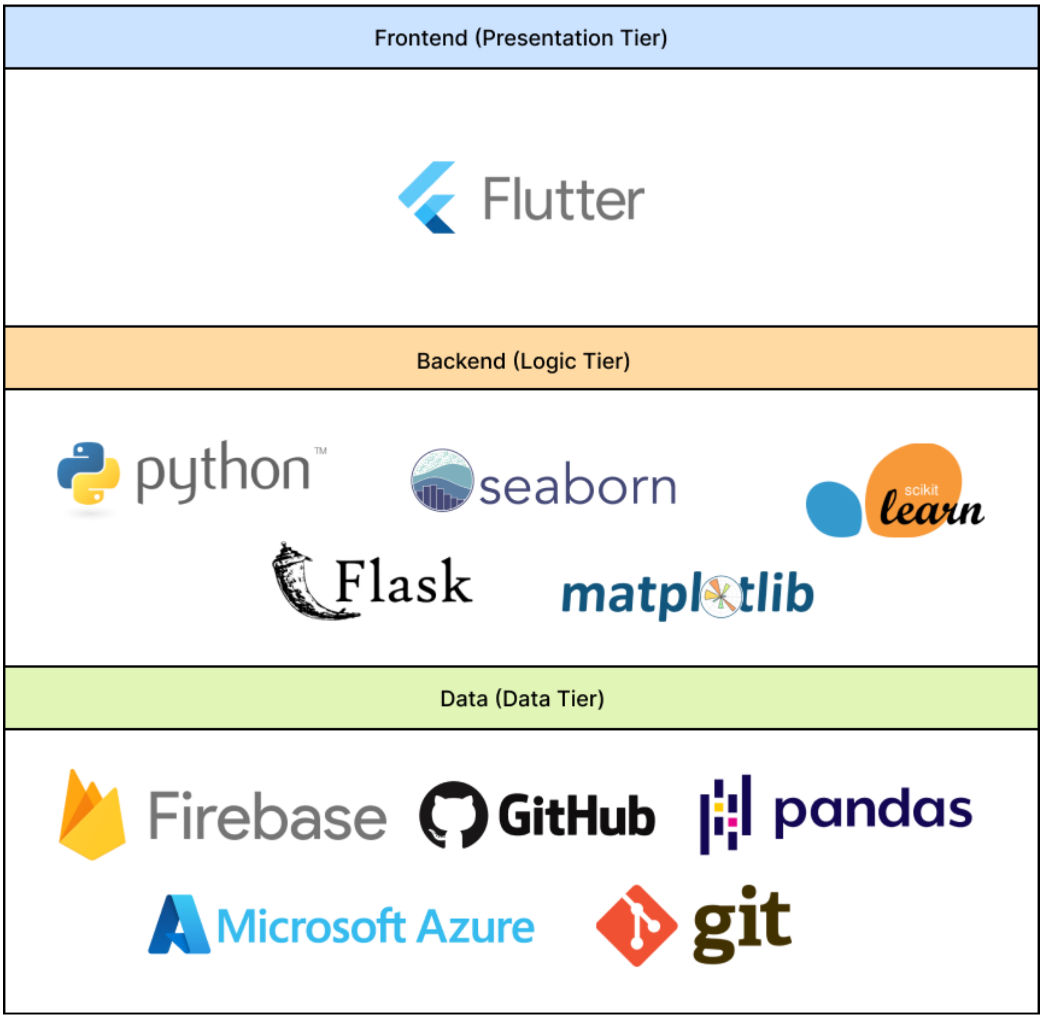


Figure 13 - Technology Stack *(Self-Composed)*

Windows Operating System will be utilized as the default operating system because all necessary tools are readily available.

4.2.2 Dataset Selection

Due to this being a data science project at its core, the author had to make sure that enough data were collected to be sufficient for the assessments alongside it being a widely trusted source for medical data.

The dataset needed to include triage assessments of a large number of patients for the system to be sufficiently trained. The assessments would include details of the patient's vital signs, as well as the acuity assessment by the doctor. The acuity was calculated using the Emergency Severity Index (ESI), a five-level triage algorithm which classifies the patient into five groups, ranging from 1 being most urgent to 5 being least urgent. This allows the risk level of the patient to be identified and care to be prioritized with limited resources.

The patient vital sign triage dataset used in this project is the **Medical Information Mart for Intensive Care IV (MIMIC IV) ED Dataset** (Johnson et al., 2022) which was provided by the Massachusetts Institute of Technology. To receive access to the dataset, the author was required to undergo Data or Specimens Only Research training course certification for human subjects' research and was also required to have credentialed access to the PhysioNet platform.

4.2.3 Development Frameworks Used

Table 13 - Development Frameworks Used

Framework	**Usage Justification**
Flutter	Used to create the user interface of the system, Flutter is an open-source framework developed by Google for building multi-platform applications using a single codebase and has the capability to create beautiful interfaces with its vast number of packages and widgets from the community.

4.2.4 Programming Languages Used

Table 14 - Programming Languages Used

Language	**Usage Justification**
Python	Used in the development of the core data science models. It is a general-purpose language however it is heavily used in many data science projects with a large collection of libraries capable of supporting the project development.

Dart	Used in front-end development for the flutter framework. It is a client-optimized language allowing people to create fast apps for any platform.

4.2.5 Libraries Used

Table 15 - Libraries Used

Library	**Usage Justification**
Pandas	Used during the model development mainly for data reading and manipulating. This library has a large number of functionalities that could be used for data cleaning, filtering and manipulating the data for data analysis
Scikit-learn	A python library, used during the model development mainly for preprocessing, training and evaluation of the model
Matplotlib	Used for data visualization and is highly customizable for data analysis
Seaborn	Used for data visualization, it’s an extension of matplotlib library that also allows to plot various graphs from pandas library and NumPy library
NumPy	It is library that has a large collection of mathematical functions that can be used on large arrays/matrices
Joblib	Used for the serialization of the models developed to save the trained models making it easy to be reused for evaluation

4.2.6 IDEs Used

Table 16 - IDEs Used

IDE	**Usage Justification**
Jupyter Notebook	Used for the data science model development

VSCode	Used during application development. it is very lightweight and has tons of features that assist during the development.
Android Studio	Used during application development. Contains inbuilt mobile emulators and works well with the flutter framework.

4.2.7 Summary of Technology Selection

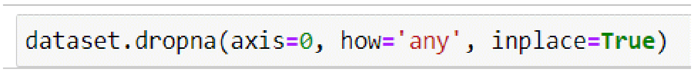
Table 17 - Summary of Technology Selection

Component	Tools Used
Development Frameworks	Flutter
Programming Languages	Dart, Python
Libraries	Pandas, Scikit-learn, Matplotlib, Seaborn, NumPy
IDEs	Android Studio, Jupyter Notebook, VSCode
Version Control	Git, GitHub
Database & Hosting	Firebase, Azure

4.3 Implementation Code

Based on the documentation provided for the dataset mentioned above, the author was required to do some data preprocessing in order to make it usable for the system.

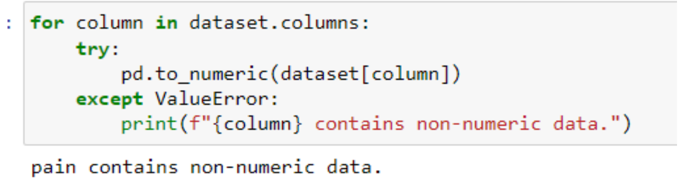
The author first dropped rows of data which were incomplete and had null variables using the ‘dropna’ command on the pandas data frame.



```
dataset.dropna(axis=0, how='any', inplace=True)
```

Figure 14 - Implementation Code Segment - Drop Null Values

In the below code segment, the author then did a check for text values in any of the columns as the values are meant to be numerical for the classification.

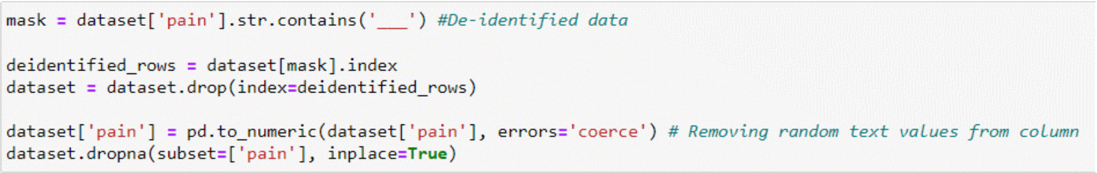


```
for column in dataset.columns:
    try:
        pd.to_numeric(dataset[column])
    except ValueError:
        print(f"{column} contains non-numeric data.")
```

```
pain contains non-numeric data.
```

Figure 15 - Implementation Code Segment -Check for Non-numeric Data

Since the pain column within the dataset had some text values, the author decided to clean the column. The dataset documentation had mentioned that some of the variables within the column had been converted to the format of three underscores for deidentification purposed. Therefore, it was best for the author to find those deidentified rows and drop any data which will not help with the classification assessment. Below is the code segment which was used for its implementation.



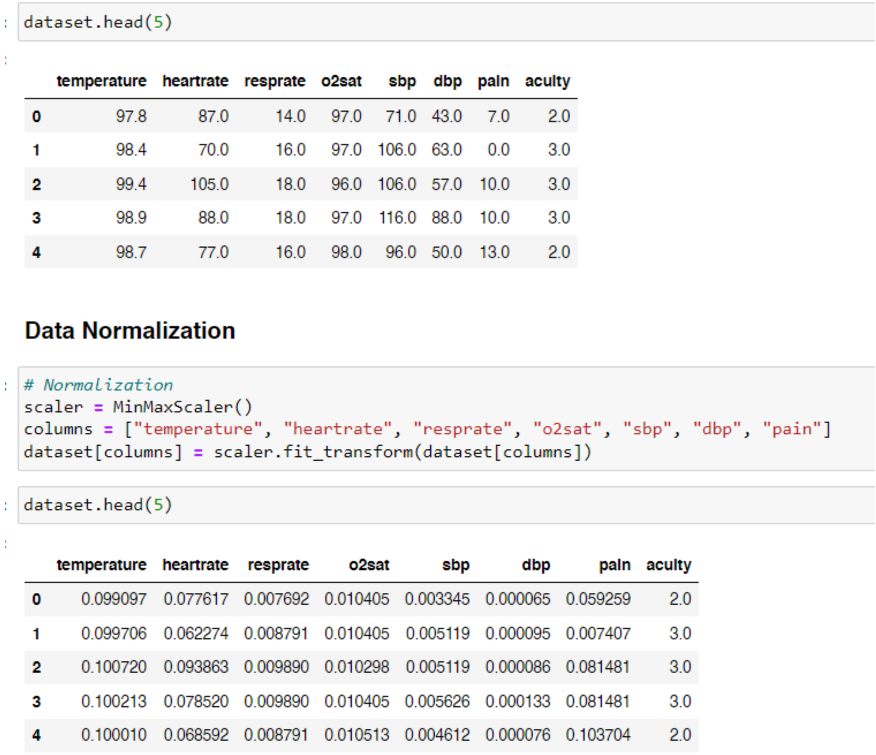
```
mask = dataset['pain'].str.contains('___') #De-identified data

deidentified_rows = dataset[mask].index
dataset = dataset.drop(index=deidentified_rows)

dataset['pain'] = pd.to_numeric(dataset['pain'], errors='coerce') # Removing random text values from column
dataset.dropna(subset=['pain'], inplace=True)
```

Figure 16 - Implementation Code Segment - Drop De-identified Rows

Since the vital sign columns are each between different ranges, the author decided to normalize the data between one specific range to make it easier to train the machine learning model to identify differences in the data. The author used the MinMaxScaler from the ‘sklearn’ library to scale the data so that it is within a specific range of zeros to one and then transforms each feature by taking the maximum and minimum value as the range. The below code segment shows its implementation.



```
dataset.head(5)
```

	temperature	heartrate	resprate	o2sat	sbp	dbp	pain	acuity
0	97.8	87.0	14.0	97.0	71.0	43.0	7.0	2.0
1	98.4	70.0	16.0	97.0	106.0	63.0	0.0	3.0
2	99.4	105.0	18.0	96.0	106.0	57.0	10.0	3.0
3	98.9	88.0	18.0	97.0	116.0	88.0	10.0	3.0
4	98.7	77.0	16.0	98.0	96.0	50.0	13.0	2.0

Data Normalization

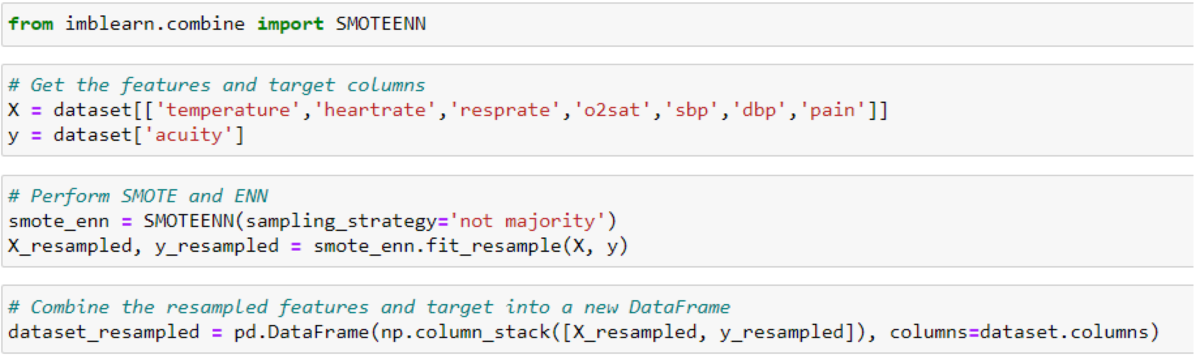
```
# Normalization
scaler = MinMaxScaler()
columns = ["temperature", "heartrate", "resprate", "o2sat", "sbp", "dbp", "pain"]
dataset[columns] = scaler.fit_transform(dataset[columns])
```

```
dataset.head(5)
```

	temperature	heartrate	resprate	o2sat	sbp	dbp	pain	acuity
0	0.099097	0.077617	0.007692	0.010405	0.003345	0.000065	0.059259	2.0
1	0.099706	0.062274	0.008791	0.010405	0.005119	0.000095	0.007407	3.0
2	0.100720	0.093863	0.009890	0.010298	0.005119	0.000086	0.081481	3.0
3	0.100213	0.078520	0.009890	0.010405	0.005626	0.000133	0.081481	3.0
4	0.100010	0.068592	0.008791	0.010513	0.004612	0.000076	0.103704	2.0

Figure 17 - Implementation Code Segment - Data Normalization

After data normalization, the author checked into the balance of the dataset and noticed it was heavily unbalanced. However, when under sampling or oversampling this dataset, it would provide even worser accuracy and performance. Therefore, the author turned to a combination of both for the prototype implementation. The author utilized the SMOTEENN functionality from the 'imblearn.combine' library, which is a combination of SMOTE and ENN, allowing the author to generate synthetic samples of the minority classes and prevent noise and over generalization. This overall increased the performance of the dataset. The below code segment shows the combination sampling implementation.



```
from imblearn.combine import SMOTEENN
```

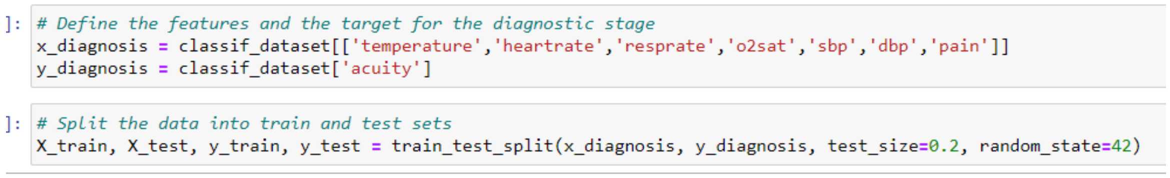
```
# Get the features and target columns
X = dataset[['temperature','heartrate','resprate','o2sat','sbp','dbp','pain']]
y = dataset['acuity']
```

```
# Perform SMOTE and ENN
smote_enn = SMOTEENN(sampling_strategy='not majority')
X_resampled, y_resampled = smote_enn.fit_resample(X, y)
```

```
# Combine the resampled features and target into a new DataFrame
dataset_resampled = pd.DataFrame(np.column_stack([X_resampled, y_resampled]), columns=dataset.columns)
```

Figure 18 - Implementation Code Segment - Combination Sampling

After data preprocessing, the author performed train and test splitting where 80% was used from training while 20% was used for testing. The below code segment shows its implementation.

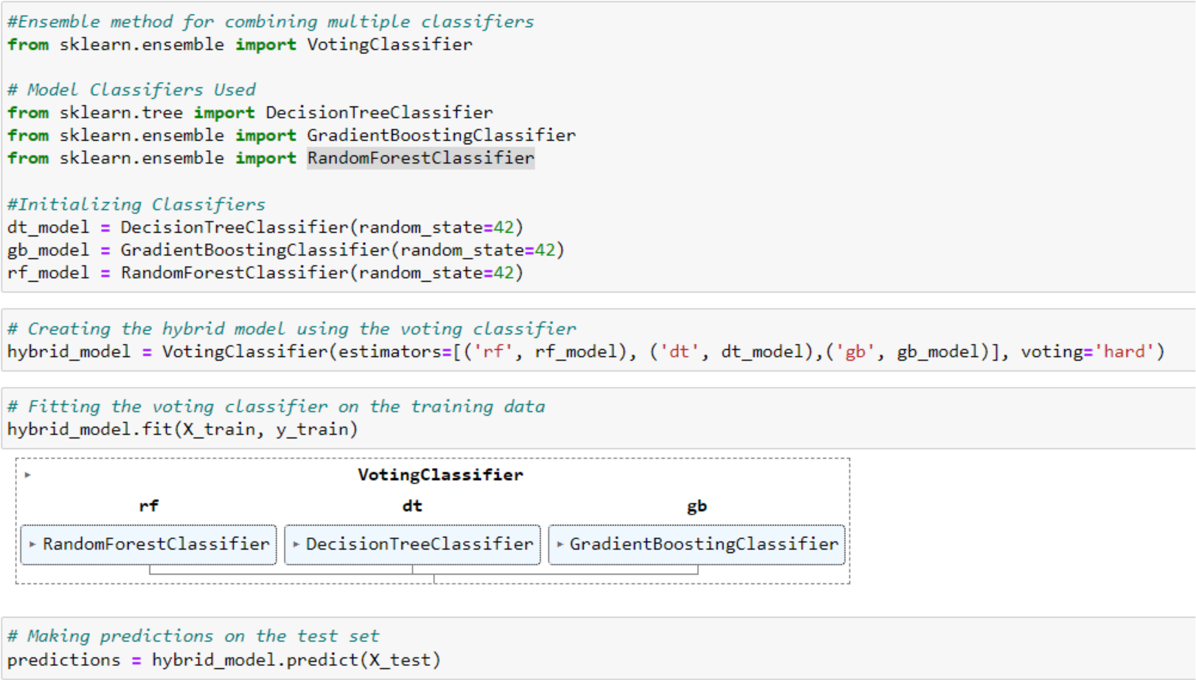


```
]: # Define the features and the target for the diagnostic stage
   x_diagnosis = classif_dataset[['temperature','heartrate','resprate','o2sat','sbp','dbp','pain']]
   y_diagnosis = classif_dataset['acuity']
```

```
]: # Split the data into train and test sets
   X_train, X_test, y_train, y_test = train_test_split(x_diagnosis, y_diagnosis, test_size=0.2, random_state=42)
```

Figure 19 - Implementation Code Segment - Train & Test Split

After splitting the data, the author implemented a voting classifier which is an ensemble model from the 'sklearn' library. The voting classifier combines the predictions of other classifiers to get a prediction that is more accurate. With hard voting, the majority class's prediction is used as the outcome. The author then implemented the RandomForestClassifier, GradientBoostingClassifier and DecisionTreeClassifier to be used within the voting classifier as can be seen from the code segment below.



```
#Ensemble method for combining multiple classifiers
from sklearn.ensemble import VotingClassifier

# Model Classifiers Used
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier

#Initializing Classifiers
dt_model = DecisionTreeClassifier(random_state=42)
gb_model = GradientBoostingClassifier(random_state=42)
rf_model = RandomForestClassifier(random_state=42)
```

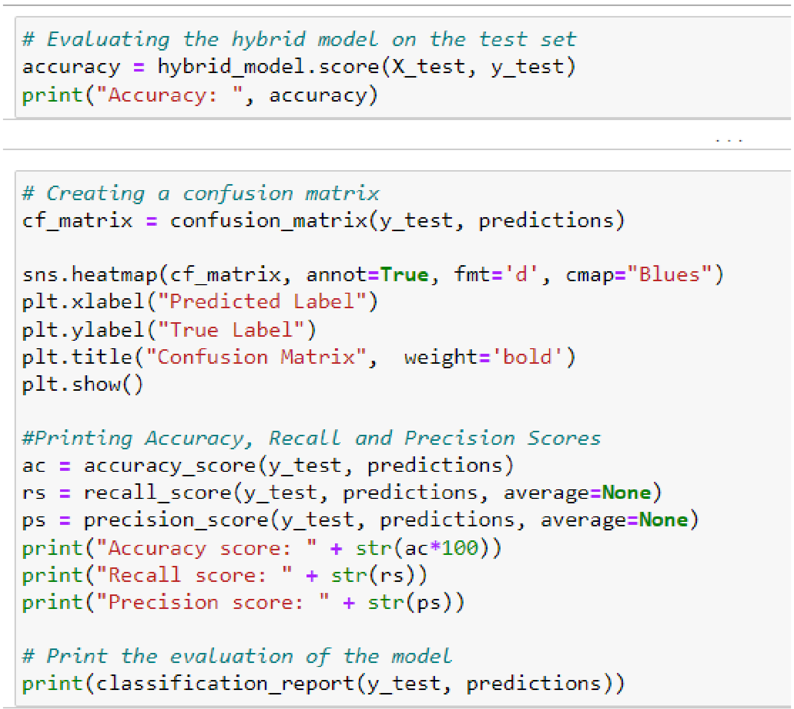
```
# Creating the hybrid model using the voting classifier
hybrid_model = VotingClassifier(estimators=[('rf', rf_model), ('dt', dt_model),('gb', gb_model)], voting='hard')
```

```
# Fitting the voting classifier on the training data
hybrid_model.fit(X_train, y_train)
```

```
# Making predictions on the test set
predictions = hybrid_model.predict(X_test)
```

Figure 20 - Implementation Code Segment - Hybrid Voting Classifier Model

The code segment below is what the author used to generate the accuracy level, classification report and confusion matrix.



```
# Evaluating the hybrid model on the test set
accuracy = hybrid_model.score(X_test, y_test)
print("Accuracy: ", accuracy)
```

```
# Creating a confusion matrix
cf_matrix = confusion_matrix(y_test, predictions)

sns.heatmap(cf_matrix, annot=True, fmt='d', cmap="Blues")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix",  weight='bold')
plt.show()

#Printing Accuracy, Recall and Precision Scores
ac = accuracy_score(y_test, predictions)
rs = recall_score(y_test, predictions, average=None)
ps = precision_score(y_test, predictions, average=None)
print("Accuracy score: " + str(ac*100))
print("Recall score: " + str(rs))
print("Precision score: " + str(ps))

# Print the evaluation of the model
print(classification_report(y_test, predictions))
```

Figure 21 - Implementations Code Segment - Model Evaluation

4.1 Chapter Summary

In this chapter, the author showcased the implementation of proposed project, highlighting the dataset that was selected for usage, the development frameworks, programming languages, libraries, and IDEs that were utilized. The chapter provided a justification for the usage of each technology and presented the initial core implementation code of the project, which was used during the development of the model. This chapter offered insight into the technical aspects of the project and how they were executed to implement the prototype.

CHAPTER 5: CONCLUSION

5.1 Chapter Overview

The project specification and design document come to an end with this chapter. In this section, any changes from the original proposal are recorded. The prototype's initial testing findings are reported, along with a review of any improvements that could be required for the minimum viable product. The author describes how they intend to deal with these issues in the upcoming weeks. For reference, a hyperlink to a video showing the prototype is also provided.

5.2 Deviations

5.2.1 Scope Related Deviations

There weren’t any deviations performed to the initial project scope presented during the project proposal.

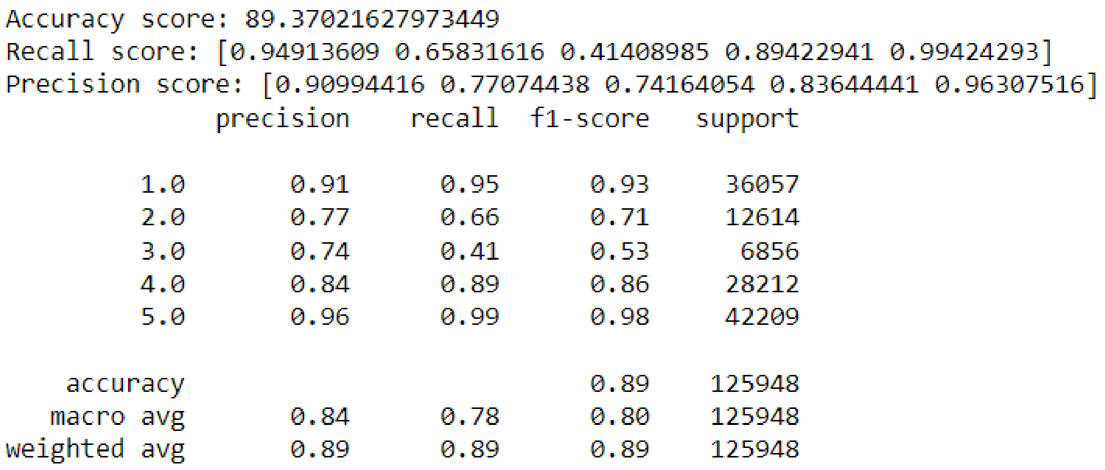
5.2.2 Schedule Related Deviations

The author has revised the initial schedule due to unforeseen delays in the implementation of the initial prototype and core components. Despite these challenges, the author was able to successfully prepare the prototype for this project. However, further development is still required, including the enhancement of the front-end interface, integration with a database, API development for the machine learning models, and thorough testing and evaluation. The author intends to address these outstanding tasks in the remaining time frame of the schedule.

5.3 Initial Test Results

The initial model demonstrated a classification accuracy of 89.3%, indicating that it is capable of accurately determining the patient's health status 89.3% of the time. Despite this promising performance, there is room for improvement. The model's F1-score of 0.93 reflects a satisfactory balance between precision and recall, with precision and recall scores of 0.91 and 0.95, respectively. This suggests that the model can accurately identify 95% of patients' acuity levels. In conclusion, while the prototype model presents acceptable accuracy and robustness, it remains subject to further optimization.

5.3.1 Model Classification Report



```
Accuracy score: 89.37021627973449
Recall score: [0.94913609 0.65831616 0.41408985 0.89422941 0.99424293]
Precision score: [0.90994416 0.77074438 0.74164054 0.83644441 0.96307516]
              precision    recall  f1-score   support

         1.0       0.91      0.95      0.93     36057
         2.0       0.77      0.66      0.71     12614
         3.0       0.74      0.41      0.53      6856
         4.0       0.84      0.89      0.86     28212
         5.0       0.96      0.99      0.98     42209

    accuracy                           0.89    125948
   macro avg       0.84      0.78      0.80    125948
weighted avg       0.89      0.89      0.89    125948
```

Figure 22 – Prototype Classification Report

5.3.2 Confusion Matrix

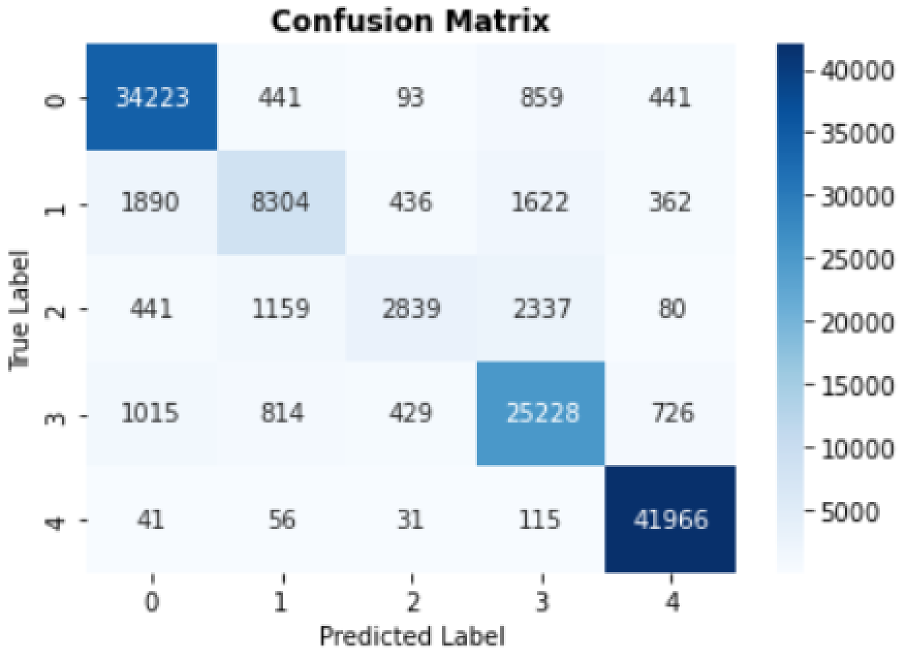


Figure 23 - Confusion Matrix

5.4 Required Improvements

The author has identified several improvements that are required for the minimum viable product namely:

- Improved Performance and Accuracy of Models: The evaluation of alternative methods to increase the performance and accuracy of the machine learning models utilized in the system.

- Model Hosting: The hosting of the machine learning models to make them accessible and usable by the system.
- Frontend Development: The implementation of the frontend client-side of the application as designed, incorporating the hosted models to display the responses.
- Usability Enhancements: The enhancement of the overall user experience through the implementation of a more intuitive and user-friendly interface.
- Data Security Measures: the implementation of the required security measures to guarantee the confidentiality and privacy of user data.

5.5 Video Demo

The link to the video demo where the author showcases the prototype implementation can be found here: https://youtu.be/7UwQ3KSY7K8

5.6 Implementation Code

The link to the repository which the author uses for the implementation is available here: https://github.com/hammvdh/FYP_RPM

5.7 Chapter Summary

This chapter provided the overall conclusion of this document, including any deviations from the initial proposal. The initial test results of the prototype were presented, along with any necessary improvements for the minimum viable product. The chapter also included a link to the video demo of the prototype. This chapter served as a summary of the project's progress and a look ahead to future work that needs to be done.

REFERENCES

Johnson, A., Bulgarelli, L., Pollard, T., Horng, S., Celi, L. A., and Mark, R. (2022) 'MIMIC-IV' (version 2.0), PhysioNet. Available at: https://doi.org/10.13026/7vcr-e114.

Saunders, M., Lewis, P. and Thornhill, A. (2003), 'Research methods for business students, Essex: Prentice Hall: Financial Times. Available from https://toc.library.ethz.ch/objects/pdf_ead50/4/E57_7072090_TB-Index_005013522.pdf

Malasinghe, L.P., Ramzan, N. and Dahal, K. (2019). Remote patient monitoring: a comprehensive study. *Journal of Ambient Intelligence and Humanized Computing*, 10 (1), 57–76. Available from https://doi.org/10.1007/S12652-017-0598-X/TABLES/6.

El-Rashidy, N. et al. (2021). Mobile health in remote patient monitoring for chronic diseases: Principles, trends, and challenges. *Diagnostics*, 11 (4). Available from https://doi.org/10.3390/diagnostics11040607.

Dhinakaran, M. et al. (2022). A System of Remote Patients' Monitoring and Alerting Using the Machine Learning Technique. Available from https://doi.org/10.1155/2022/6274092.

Salehi, S. et al. (2020). Assessment of remote patient monitoring (RPM) systems for patients with type 2 diabetes: a systematic review and meta-analysis. Journal of Diabetes and Metabolic Disorders, 19 (1), 115–127. Available from https://doi.org/10.1007/S40200-019-00482-3/TABLES/4.

Jayatilake, S.M.D.A.C. and Ganegoda, G.U. (2021). Involvement of Machine Learning Tools in Healthcare Decision Making. Journal of Healthcare Engineering, 2021. Available from https://doi.org/10.1155/2021/6679512.

Nallakaruppan, M.K. and Kumaran, U.S. (2020). Hybrid machine learning model for healthcare monitoring systems. *International Journal of Internet Technology and Secured Transactions*, 10 (5), 538–551. Available from https://doi.org/10.1504/IJITST.2020.109532.

Elliott, M. and Endacott, R. (2022). The clinical neglect of vital signs' assessment: an emerging patient safety issue? Contemporary nurse. Available from https://doi.org/10.1080/10376178.2022.2109494

Vinutha, D.C., Kavyashree and Raju, G.T. (2022). Machine Learning–Assisted Remote Patient Monitoring with Data Analytics. Tele-Healthcare, 1–26. Available from https://doi.org/10.1002/9781119841937.CH1 [Accessed 10 October 2022].

Lata Sahu, M. et al. (2021). Cloud-Based Remote Patient Monitoring System with Abnormality Detection and Alert Notification. Mobile Networks and Applications, 1, 16. Available from https://doi.org/10.1007/s11036-022-01960-4 [Accessed 4 October 2022].

Tabassum, S. et al. (2020). A data enhancement approach to improve machine learning performance for predicting health status using remote healthcare data. *2020 2nd International Conference on Advanced Information and Communication Technology, ICAICT 2020*, 308–312. Available from https://doi.org/10.1109/ICAICT51780.2020.9333506 [Accessed 19 September 2022].

Gontarska, K. et al. (2021). Predicting Medical Interventions from Vital Parameters: Towards a Decision Support System for Remote Patient Monitoring. Available from http://arxiv.org/abs/2104.10085.

Chang, Daniel, Chang, David and Pourhomayoun, M. (2019). Risk prediction of critical vital signs for ICU patients using recurrent neural network. Proceedings - 6th Annual Conference on Computational Science and Computational Intelligence, CSCI 2019. 1 December 2019. Institute of Electrical and Electronics Engineers Inc., 1003–1006. Available from https://doi.org/10.1109/CSCI49370.2019.00191.

Naemi A, Schmidt T, Mansourvar M, et al (2021). Machine learning techniques for mortality prediction in emergency departments: a systematic review. BMJ Open 2021. Available from https://doi.org/10.1136/bmjopen-2021-052663