UNIVERSITY OF SIALKOT
A CHARTERED UNIVERSITY

Faculty of Computing & IT

University of Sialkot

CardioPulse: Cardiovascular Disease Prediction through an Integrated Machine Learning Framework

Session: BS-SE Fall 2020-2024

Project Advisor: Mr. Attique Ur Rehman

Submitted By

Ali Aoun	*20101001-038*
Muneeba Javed	*20101001-022*
Fariha	*20101001-150*
Hammad	*20101001-166*

Faculty of Computing & IT
University of Sialkot

STATEMENT OF SUBMISSION

This is certified that Ali Aoun Roll No. 20101001-038 and Hammad Roll No. 20101001-166 and Fariha Roll No. 20101001-150 and Muneeba Javed Roll No. 20101001-022 have successfully completed the final year project named as CardioPulse: Cardiovascular Risk Prediction through an Integrated Machine Learning Framework at the Department of Software Engineering, University of Sialkot, to fulfill the requirement of the degree of **Bachelor's in Software Engineering**.

_____

Project Supervisor
Lecturer
Mr. Attique Ur Rehman
Department of Software Engineering
Faculty of Computing & Information Technology
University of Sialkot, Punjab, Pakistan.

_____

Project Management Office
Faculty of C&IT -USKT

_____

Head of the Department
Assistant Professor
Dr. Sabeen Javaid
Department of Software Engineering
Faculty of Computing & Information Technology
University of Sialkot, Punjab, Pakistan

Acknowledgement

We truly acknowledge the cooperation and guidance by our supervisor Mr. Attique Ur Rehman Department of Software Engineering, University of Sialkot. He has been a constant source of guidance throughout the course of this project. Also, we are truly grateful to our parents whose effortless care, prayers and silent support led us to complete our project.

Date: March 28, 2024

Abstract

The era we live in is called the modern era due to our increasing reliance on technology. From simple to complex tasks, our societies prefer to utilize software and machines, driving a significant rise in the demand for machine learning applications. CardioPulse is a mobile application that leverages machine learning to predict cardiovascular disease, addressing the global health concern of heart diseases. Positioned at the nexus of technological innovation and global health priorities, CardioPulse stands out as a beacon, predicting and addressing cardiovascular diseases. By analyzing a range of health data, including lifestyle, medical history, and physiological measurements, CardioPulse provides users with personalized disease assessments in a user-friendly and cost-effective manner. Much like a modern stethoscope, this application empowers individuals to proactively manage their cardiovascular health by offering informative reports and actionable insights, ultimately contributing to early disease detection, healthier hearts, and improved public health. CardioPulse employs advanced machine learning algorithms to analyze diverse health metrics, offering a robust platform for early cardiovascular disease detection. The application integrates data from user inputs, to ensure comprehensive health monitoring. By utilizing predictive analytics, CardioPulse not only identifies potential risks but also suggests preventive measures tailored to individual users. This proactive approach aids in reducing the incidence of severe cardiovascular events and promotes a culture of health awareness and preventive care. In the development of CardioPulse, significant attention has been paid to creating an intuitive user interface that facilitates easy navigation and accessibility for users of all ages. The application features a seamless integration with healthcare systems, enabling users to share their health data with medical professionals securely. Additionally, the application includes educational resources to inform users about cardiovascular health, risk factors, and lifestyle changes that can mitigate the risk of heart disease. CardioPulse's vision is to revolutionize public health by making advanced healthcare tools accessible to everyone, regardless of their geographical or socio-economic status. By democratizing access to cutting-edge technology, CardioPulse aims to bridge the gap in healthcare disparities and contribute to a healthier global community. The application underscores the potential of machine learning in transforming healthcare delivery and enhancing the quality of life through early intervention and continuous health monitoring.

Table of Contents

List of Figures

List of Tables

Chapter 1:

Project Feasibility Report

1.1. Introduction

In our current "modern era," technology plays an increasingly vital role in our daily lives, driving our reliance on software and machines to manage a wide array of tasks, both simple and complex. Within this technological landscape, the field of artificial intelligence, specifically machine learning, and its subset deep learning, have emerged as pivotal components. However, as we navigate this digital age, we also encounter a significant challenge within the realm of healthcare — the accurate identification of cardiovascular health risks. Cardiovascular diseases affect a substantial portion of the population globally, with early detection serving as a critical factor in prevention and management. Traditional risk assessment methods, reliant on basic health data, may not always provide the precision required to protect individuals from the perils of heart-related issues. This problem transcends the healthcare sector, affecting individuals, families, and communities at large. Inaccurate diagnoses or delays in providing necessary support can exacerbate health problems, diminish the quality of life, and add to societal burdens. To address this issue, our project represents a harmonious convergence of healthcare, technology, and research. We aim to leverage modern technology, including data from a comprehensive set of health parameters and cutting-edge computing, to revolutionize cardiovascular risk assessment. Our objective is to equip healthcare professionals with an advanced tool that delivers more accurate, timely, and actionable insights into an individual's heart health. Through CardioPulse, we strive to enhance both the quality of healthcare and the overall well-being of individuals, ensuring that the digital age becomes a powerful ally in safeguarding cardiovascular health.

1.2. Problem Statement

We're confronted with a significant issue: the current methods for assessing cardiovascular health are often insufficient. Presently, healthcare providers predominantly rely on basic health data and subjective patient reports for evaluating heart disease risk, leading to inconsistent and inconclusive risk assessments. This issue carries substantial consequences, impacting individuals, the healthcare system, and society on a broader scale. Inaccurate risk assessments or delayed interventions can lead to worsening health outcomes for those at risk of cardiovascular diseases, both in terms of individual well-being and economic efficiency. To provide context, an estimated 17.9 million people worldwide die from cardiovascular diseases each year, representing a staggering 31% of global deaths. Early, accurate risk detection is essential to reduce this burden. Quantitatively, approximately 524 million individuals worldwide suffer from high blood pressure, a significant risk factor for heart disease, with many unaware of their condition. Furthermore, over 20 million people endure coronary heart disease, and 17.9 million individuals experience strokes annually. These conditions not only bring personal suffering but also incur enormous costs for healthcare systems. Our project aims to address this challenge. We intend to employ a diverse range of health-related data and advanced machine learning techniques to develop CardioPulse, a solution that offers improved and more accurate cardiovascular risk assessments. This solution is designed to

benefit individuals, healthcare providers, and organizations by enhancing early detection and management of heart disease, ultimately reducing its impact on both personal and societal well-being.

1.3. Objectives

We're developing a system focused on accurately predicting cardiovascular diseases. Using machine learning algorithms, we enhance the accuracy of our predictions. Early detection is crucial for timely intervention, allowing individuals to seek confirmation or further evaluation from healthcare professionals. Our system concentrates on prediction, providing valuable insights into potential cardiovascular issues.

1.4. Project Motivation

We want to help people catch heart problems early. Heart disease is a big health issue that can be really serious if not spotted in time. We believe that using smart technology, like machine learning, we can create a system that finds signs of heart disease early. Our main goal is to make a tool that can tell if someone might have heart trouble when it's just starting. If we can do this, it means people can get help and treatment early on, which can stop the heart disease from getting worse. We want to make it easier for everyone to look after their heart health. Our motivation is to use technology to make a positive impact on people's lives by preventing heart problems. We hope that with our project, we can help more people live healthier lives by catching heart disease before it becomes a big problem.

1.5. Project/Product Feasibility Report

Our CardioPulse app is feasible as we will be tracking the feasibility areas that are required to be fulfilled for our overall project feasibility check. The success rate of our project can be determined by the following types of feasibility attributes given below: There are many types of feasibilities:

- Technical
- Operational
- Economic
- Schedule
- Specification
- Information
- Motivational
- Legal and Ethical

1.5.1. Technical Feasibility

Our project holds technical feasibility and it is very important for the successful development of our CardioPulse application. Our project is feasible regarding its scope and can be developed with required technical skills. The group members will make sure to gain all the skill sets required to develop our application CardioPulse. The tools and technology required for the development of our project are available and accessible also the group members have the expertise of using those technologies.

1.5.2. Operational Feasibility

As we know that Operational feasibility is to measure how an application can handle problems and how it solves it. So, we are going to make our CardioPulse app easy to use for the heart disease prediction. Thus, we have tried our best to develop the app that our users can easily operate without facing any issues.

1.5.3. Economic Feasibility

As we know that the economic feasibility analysis is the most used method for determining the efficiency of a new project. It helps in identifying profit against investment expected from a project. Our CardioPulse app is totally free of cost to be used by the users and is feasible economically.

1.5.4. Schedule Feasibility

The schedule feasibility holds great importance to determine whether the project would be completed in time. For this we used Gantt chart to estimate the total days for the completion of each deliverable and overall project. Every member of the team will ensure that the task will be completed on time in accordance with the Gantt chart.

1.5.5. Specification Feasibility

All the requirements are clear and concise. There is no ambiguity in the requirements. We also kept the user demands and needs in check. All the requirements can be found in the second chapter of this documentation. We have mentioned constraints and conditions for the project, and they are easy to understand. The phase for requirement specification and analysis will be given high prominence, requirement, and persistent elicitation.

1.5.6. Market Analysis/Preliminary Record

There is a lower competition and high demand of medical systems. The size of market in automated system has grown over the time. Therefore, the proposed project has the advantage of exposed market over the globe.

1.5.7. Motivational Feasibility

Motivation is necessary for all team members for the project to be successful. Our supervisor and team members motivate each other on daily basis to keep team members motivated. The entire team is compassionate and corporative. Every member of the team is motivated to work on this project. The project team is motivated to achieve the objectives of the project. Due to proper planning of the project, tasks have been assigned to the related team member and flexible yet definite schedule provides the proper environment for the motivation to complete the project.

1.5.8. Legal & Ethical Feasibility

Our project purpose is to assist individuals with cardiovascular health concerns, providing services not available in existing systems. it is totally legal and the services offered by our CardioPulse application are not available in the existing systems. The project follows the rules and laws about healthcare and privacy to make sure everything is legal and right. So, this project is legally and ethically feasible.

1.6. Project/Product Scope

Our primary motivation is to make a substantial impact in the realm of cardiovascular health. Similar to the challenge of mental health diagnosis, heart disease represents a significant global health issue, often leading to severe consequences when not detected

early. We firmly believe that by harnessing advanced technology, such as machine learning, we can create a system that identifies early warning signs of heart disease. Our central objective is to develop a user-friendly tool that can effectively alert individuals to potential heart issues in their early stages. Achieving this goal holds the promise of providing timely support and treatment to those at risk, potentially preventing the progression of heart conditions to critical stages.

Our overarching mission is to simplify the process of monitoring and maintaining heart health, making it accessible and beneficial to a broad audience. Our motivation is rooted in the aspiration to utilize technology as a force for positive change in people's lives. We are dedicated to the prevention of heart-related issues and the enhancement of overall well-being for individuals through our project. We aim to help a greater number of individuals lead healthier lives by identifying cardiovascular risks before they become significant problems, thereby contributing to an overall improvement in public health.

1.7. Project/Product Costing

For our project, the project costing is used to help us recognize project costs and its benefit that we will get. Project costing helps in understanding that whether we will be able to complete our project under our budget or not.

1.7.1. Project Cost Estimation by Function Point Analysis

Function Points (FP) is a metric used in software development to quantify the functionality provided by a software application from the user's perspective. It serves as a means to estimate the effort, resources, and cost required for software development early in the project life cycle. The concept of Function Points was introduced by Allan Albrecht in the late 1970s.

Collect Data for Function Points:
Number of User Inputs (UI): 20
Number of User Outputs (UO): 15
Number of User Inquiries (UI): 5
Number of Files (FI): 5
Number of External Interfaces (EI): 2

Calculate Unadjusted Function Points (UFP):
UFP = UI * (4) + UO * (4) + UI * (5) + FI * (3) + EI * (3)
UFP = 20 * 4 + 15 * 4 + 5 * 5 + 5 * 3 + 2 * 3
UFP = 80 + 60 + 25 + 15 + 6
UFP = 186

Calculate Adjusted Function Points (AFP):
Assuming VAF of 1.15.

AFP = UFP * VAF
AFP = 186 * 1.15
AFP = 213.9 (Approximately 214)

Calculate Total Project Cost:
Assuming labor rate of $60 per hour and a productivity parameter of 2.

Cost / FP = $60 / 2 = $30 per FP
Total Project Cost = AFP * Cost / FP
Total Project Cost = 214 * $30
Total Project Cost = $6,420

1.7.2. Project Cost Estimation by using COCOMO'81 (Constructive Cost Model)

We have used the COCOMO Basic Model in order to estimate the cost for our app CardioPulse development in terms of effort resources required to complete the project work and time required to complete the project work based on the size of our FYP project.

1.7.1.1 COCOMO Basic model

The COCOMO basic model is as follows:
KLOC = 10.000
SLOC = 10000
Mode = Organic
a = 3.694880219919591
b = 1.05
c = 2.5
d = 0.38
Formula used:
effort = a*KLOCb , in person-months with KLOC - lines of code (in thousands)
staffing = effort / duration

1.7.1.2 COCOMO Results for CardioPulse

Table 1.1: Cocomo Result

MODE	Organic
"A" Variable	2.4
"B" Variable	1.05
"C" Variable	2.5
"D" Variable	0.38
KLOC	7.000
Efforts (in person-months)	18.517
Duration (in months)	7.579
Staffing	2.443

Product Attributes	
Required Reliability	1.00 (N)
Database Size	1.00 (N)
Product Complexity	1.00 (N)

Computer Attributes	
Execution Time Constraint	1.00 (N)
Main Storage Constraint	1.00 (N)
Platform Volatility	1.00 (N)

Computer Turnaround Time	1.00 (N)

Personnel Attributes

Analyst Capability	1.00 (N)
Applications Experience	1.00 (N)
Programmer Capability	1.00 (N)
Platform Experience	1.00 (N)
Programming Language and Tool Experience	1.00 (N)

Project Attributes

Modern Programming Practices	1.00 (N)
Use of Software Tools	1.00 (N)
Required Development Schedule	1.00 (N)

1.7.3. Activity Based Costing

Activity-Based Costing (ABC) is a pivotal methodology within the CardioPulse project, enabling the meticulous measurement and optimization of costs associated with various activities, resources, and overall cost objects. By aligning resources with specific activities and connecting them to relevant cost objects, ABC offers a granular understanding of cost distribution.

Objectives:

1. **Measuring Process Performance:** ABC systematically measures the performance of each CardioPulse business process, dissecting the cost dynamics activity by activity for enhanced insights.
2. **Estimating Output Costs:** The methodology allows CardioPulse to estimate the cost of producing outputs associated with different business processes by delving into the resources consumed in each activity.
3. **Identifying Improvement Opportunities:** ABC empowers CardioPulse to pinpoint areas for process optimization by bringing to light unusually high or low costs linked to specific activities.

Cost Drivers:

1. **Resources:** CardioPulse identifies the business workers and entities participating in each activity, incurring associated costs.
2. **Cost Rate:** Each instance of a business worker or entity is allocated a cost per unit of time in use.
3. **Duration:** Activities in CardioPulse have defined durations, influencing the allocation of costs.
4. **Overhead:** Fixed costs incurred with the initiation of workflows or activities are considered in overall cost calculations.

Activity-Based Management (ABM):

ABC forms the cornerstone of Cardio Pulse's Activity-Based Management (ABM) practices, concentrating on optimizing processes for heightened efficiency and effectiveness. Embracing Activity-Based Costing, CardioPulse aims to attain a nuanced understanding of its cost structure, identify avenues for cost reduction, and elevate overall project management efficiency.

1.8. Task Dependency Table

The following are the steps to develop a task dependency table:

Table 1.2: Task Dependency Table

Task Id	Activity Name	Dependency
T1	Proposal	-
T2	Feasibility Report	T1
T3	Research	T1
T4	Requirements Specification	T1, T2
T5	Design Document	T2, T4
T6	Interface Design	T5
T7	Backend Development	T5, T6
T8	App Implementation	T7
T9	Testing	T8
T10	Deployment	T9

1.9. CPM - Critical Path Method

We have used the CPM in order to provide a graphical view of our project, predict the time required to complete the project and to show which activities are critical to maintaining the schedule and which are not.

1.9.1 Specify the Individual Activities

For our CardioPulse Application, we divided our application into following activities:

- Proposal
- Feasibility Report
- Research
- Requirements Specification
- Design Document
- Interface Design
- Backend Development
- App Implementation
- Testing
- Deployment

1.9.2 Determine the Sequence of the Activities

The sequence of activities is to identify the relationships among the project activities. Following activities, their predecessor and duration are shown in the table below:

Table 1.3: Activities Sequence and Duration

Activity Id	Activity Name	Duration(days)	Predecessor
T1	Proposal	20	-
T2	Feasibility Report	20	T1
T3	Research	169	T1
T4	Requirements Specification	17	T1, T2
T5	Design Document	48	T2, T4
T6	Interface Design	42	T5
T7	Backend Development	56	T5, T6
T8	App Implementation	64	T7
T9	Testing	27	T8
T10	Deployment	3	T9

1.9.3 Activity Network Diagram

In order to show the sequential relationships of activities using arrows and nodes we have used the activity network diagram. Following is the activity network diagram of our project:

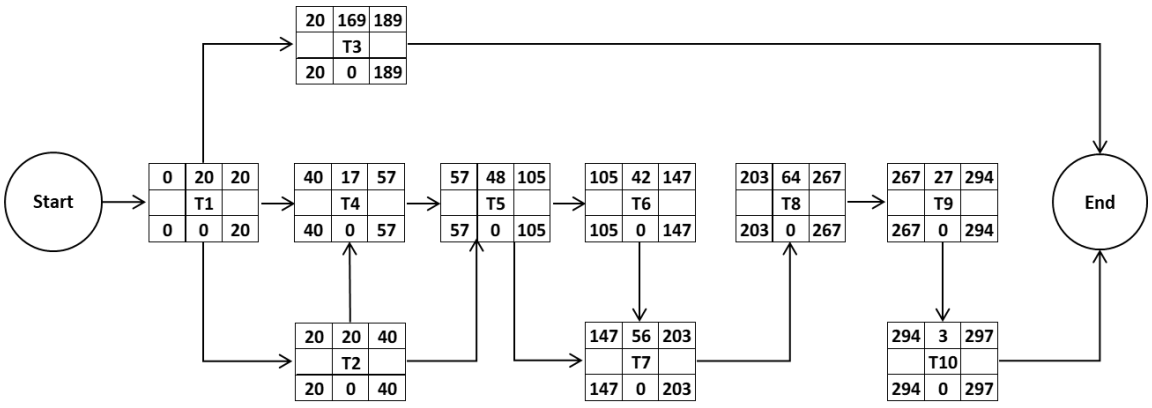

Figure 1.1: Network Diagram

1.9.4 Estimating Activity Completion Time and Slack Time

Following table displays the slack time calculated for each activity:

Table 1.4: Slack Time

Activity	Duration(days)	Predecessor	ES	EF	LS	LF	Slack time
T1	20	-	0	20	0	20	0
T2	20	T1	20	40	20	40	0

T3	169	T1	20	189	20	189	0
T4	17	T1, T2	40	57	40	57	0
T5	48	T2, T4	57	105	57	105	0
T6	42	T5	105	147	105	147	0
T7	56	T5, T6	147	203	147	203	0
T8	64	T7	203	267	203	267	0
T9	27	T8	267	294	267	294	0
T10	3	T9	294	297	294	297	0

1.9.5 Identifying the Critical Path

From the above table of slack time, we can see the critical path for our project is:
T1 -> T2 -> T3 -> T4 -> T5 -> T6 -> T7 -> T8 -> T9 -> T10 = 297 days

1.10. Gantt chart

To graphically represent the tasks to be performed with their estimated durations during this journey of final year project completion, we have used the Gantt chart to fulfill this purpose. Following is the table used to develop the Gantt chart:

ID	Name	Start	Finish	Duration
1	Proposal	9/29/2023	10/4/2023	4 day
2	Feasibility Report	10/5/2023	10/18/2023	10 day
3	Research	10/19/2023	11/8/2023	15 day
4	Requirements Specification	11/9/2023	11/22/2023	10 day
5	Design Document	11/23/2023	12/20/2023	20 day
6	Interface Design	12/21/2023	1/31/2024	30 day
7	Backend Development	2/1/2024	2/21/2024	15 day
8	App Implementation	2/22/2024	3/20/2024	20 day
9	Testing	3/21/2024	4/24/2024	25 day
11	Deployment	5/23/2024	6/19/2024	20 day

Figure 1.2: Gantt Chart

1.11. Introduction to Team member and their skill set

We are a team of four group members, each member has skills in their field of interest respectively.

Group members:
Following are the group members of our team with assigned Id.

Table 1.5: Team Members

Name	Roll Number	ID
Ali Aoun	20101001-038	M1
Hammad	20101001-166	M2
Fariha	20101001-150	M3
Muneeba Javed	20101001-022	M4

The skill set of each member are mentioned below:

Table 1.6: Skill Set

Team Member	Skill Set
Ali Aoun	**• Feasibility Study • Research Work • Software Requirements Specification • Software Design and Architecture • User Interface Designing • Database design and implementation • Backend development • Testing**
Hammad	**• Feasibility Study • Research Work • Software Requirements Specification • Software Design and Architecture • User Interface Designing • Database design and implementation • Backend development • Testing**
Fariha	**• Feasibility Study • Research Work • Software Requirements Specification • Software Design and Architecture • User Interface Designing • Database design and implementation • Backend development • Testing**
Muneeba Javed	**• Feasibility Study • Research Work • Software Requirements Specification • Software Design and Architecture • User Interface Designing • Database design and implementation • Backend development • Testing**

1.12. Task and Member Assignment Table

The task and member assignment table are given as follows:

Table 1.7: Task and Member Assignment Table

Task	Duration (days)	Dependencies	Assigned
T1	20	-	M1, M2, M3, M4
T2	20	T1	M1
T3	169	T1	M1
T4	17	T1, T2	M1
T5	48	T2, T4	M1, M2, M3, M4
T6	42	T5	M1, M2, M3, M4
T7	56	T5, T6	M2, M3
T8	64	T7	M2, M3, M4
T9	27	T8	M1
T10	3	T9	M1, M2, M3, M4

The task and member allocation table are given as follows:

Table 1.8: Task Allocation

Task	Engineer
T1	• Ali Aoun • Hammad • Fariha • Muneeba Javed
T2	Ali Aoun
T3	Ali Aoun
T4	Ali Aoun
T5	• Ali Aoun • Hammad • Fariha • Muneeba Javed

T6	• Ali Aoun • Hammad • Fariha • Muneeba Javed
T7	• Hammad • Fariha
T8	• Hammad • Fariha • Muneeba Javed
T9	Ali Aoun
T10	• Ali Aoun • Hammad • Fariha • Muneeba Javed

1.13. Tools and Technology with reasoning

The languages, framework and application tools, which are to be used on the front and back end of the system to be developed, are listed here. The reasons for these tools have also been described as follows:

Frontend Development:

- **Android Studio:**
  Android Studio is the official IDE for Android app development, providing a comprehensive environment with visual design tools and Kotlin support.

Backend Development:

- **Python:**
  Python will be used for backend development and machine learning model creation due to its popularity and extensive libraries for machine learning.

- **Flask (or Django) for API Development:**
  Flask is chosen for its simplicity and API development suitability. Django, a more robust framework, can be considered for larger applications.

- **SQLite (or PostgreSQL) for Database:**
  SQLite is suitable for smaller applications; PostgreSQL is a robust option for more complex queries and scalability.

Machine Learning:

- **scikit-learn:**
  scikit-learn is a user-friendly, Python-based library suitable for machine learning tasks with smaller to medium-sized datasets.

- **TensorFlow Lite (for Integration with Android):**
  TensorFlow Lite is designed for mobile devices, ensuring efficient integration of machine learning models with Android.

Version Control:

- **Git (GitHub or GitLab):**
  Git is employed for version control, and the repository can be hosted on GitHub or GitLab for collaboration.

Development Process and Collaboration:

- **Agile Board (e.g., Jira, Trello):**
  An Agile board (Jira, Trello) aids in managing tasks and sprints for an iterative development process.

IDE and Text Editors:

- **PyCharm (for Python development):**
  PyCharm is a robust Python IDE offering features like code completion and debugging.

1.14. Vision Document

CardioPulse is a pioneering Android mobile application engineered to transform cardiovascular health management through predictive risk assessment. Utilizing machine learning algorithms, the app enables users to proactively monitor their heart health. The primary objectives include developing a user-friendly interface for seamless interaction on Android devices, empowering users to input heart health data, receive predictive results, and conveniently track their progress. The app integrates five diverse datasets for comprehensive risk assessment and provides voice-assisted functionalities to enhance accessibility. The scope is exclusive to Android, ensuring a tailored and accessible experience. Periodic checkpoints will validate key aspects, adapting to evolving requirements and technological advancements.

Main Features:

- User and Guardian Login/Sign Up for personalized experiences.
- Integration of five diverse datasets for comprehensive risk assessment.
- Predictive Modeling for Heart Disease using an integrated machine learning framework.
- Access to historical reports and tracking progress over time.

Assumptions:

- CardioPulse will be accessible on Android devices, requiring internet connectivity for real-time data processing.
- Laptop or pc having HDD and SDD required.
- Internet availability is essential.
- Proper tools for specified purpose must be selected properly.

1.15. Risk List

Considering our CardioPulse app, the possibility of suffering harm or loss in terms of danger is mentioned in this section of risk list. The Risk List is designed to capture the perceived risks to the success of the project. Throughout the project the risk list is maintained that is created early in the Inception phase and is continually updated as new risks are uncovered and existing risks are mitigated or retired. Some of the risks that we

have identified in our project are:

• Unavailability of resources
• Highly expensive out of project budget APIs or tools
• Fail to achieve quality required
• Requirements contradictions
• Fail to integrate all the modules together
• Fail to deal with unknown anomalies
• Natural disaster
• Accidental loss of project files
• Fail to meet the project completion duration

1.16. Product Features/ Product Decomposition

In CardioPulse, the product features and decomposition align with the functional requirements that drive the intended behavior of the system, ensuring a comprehensive and user-centric cardiovascular health management platform.

User Registration and Login:
Enable seamless onboarding for users and healthcare professionals.
Provide secure login authentication for data privacy.

Health Data Collection and Storage:
Facilitate input and secure storage of user cardiovascular data.
Employ advanced encryption techniques to ensure data integrity and confidentiality.

Cardiovascular Risk Prediction:
Utilize machine learning algorithms for real-time risk assessments.
Empower users with personalized insights for proactive health management.

Progress Monitoring and Reporting:
Enable users to track their cardiovascular health journey over time.
Generate detailed reports highlighting progress and potential areas of concern.

Chapter 2:

Software Requirement Specification

2.1 Introduction:

The purpose of the document is to specify the requirements that have to be fulfilled by the cardiovascular diseases prediction which will be developed using machine learning. The requirements describe functional and qualitative necessities of our project. The description of requirements is intended to help users express their needs. The specification of requirements facilitates the development of the system by providing measurable objectives. Therefore, the specification is formulated in a way that allows the assessment of the fulfillment of requirements specifying the quality at runtime (e.g., performance) which can be evaluated.

2.1.1 Systems Specifications

Following are the system specifications mentioned in detail for the CardioPulse:

Introduction

Our final year project CardioPulse is centered around addressing the needs of individuals facing cardiovascular health challenges through innovative software technology. CardioPulse aims to provide a comprehensive solution with key features designed to empower users in managing their cardiovascular well-being. These features include Cardiovascular Risk Prediction, Health Data Collection and Storage, Progress Monitoring and Reporting, and User Registration and Login. Our focus is not only on functionality but also on creating a user-friendly, efficient, and secure application. CardioPulse strives to revolutionize how individuals approach cardiovascular health, offering a proactive and personalized experience.

Existing System

In recent years, there has been a surge of interest and research endeavors aimed at leveraging machine learning techniques for the early prediction and detection of cardiovascular diseases (CVDs). These diseases, encompassing a range of heart and blood vessel conditions, represent a significant global health concern. Notably, various risk factors such as high blood pressure, smoking, diabetes, and familial history have been identified as pivotal contributors to the onset of CVDs. As a result, researchers have delved into the development of predictive models utilizing diverse datasets and machine learning algorithms. The Cleveland, Framingham, and heart disease datasets have emerged as prominent focal points for investigations in this domain. While substantial progress has been made, there remains a need to critically evaluate and compare the efficacy of different machine learning algorithms for accurate and timely CVD prediction. This study builds upon prior research by conducting a comprehensive analysis of four key machine learning methods — Decision Tree, MARS, Random Forest, and TMGA — employing the Cleveland dataset. By examining both accuracy and computational efficiency, we aim to identify the most proficient algorithm in advancing early CVD prediction.

Organizational Chart

The organizational chart presents a pictorial view of the organizational structure. It illustrates the members or components of the organization with respect to their roles and responsibilities. Following is the organizational chart of both the internal team structure and external business organizational chart.

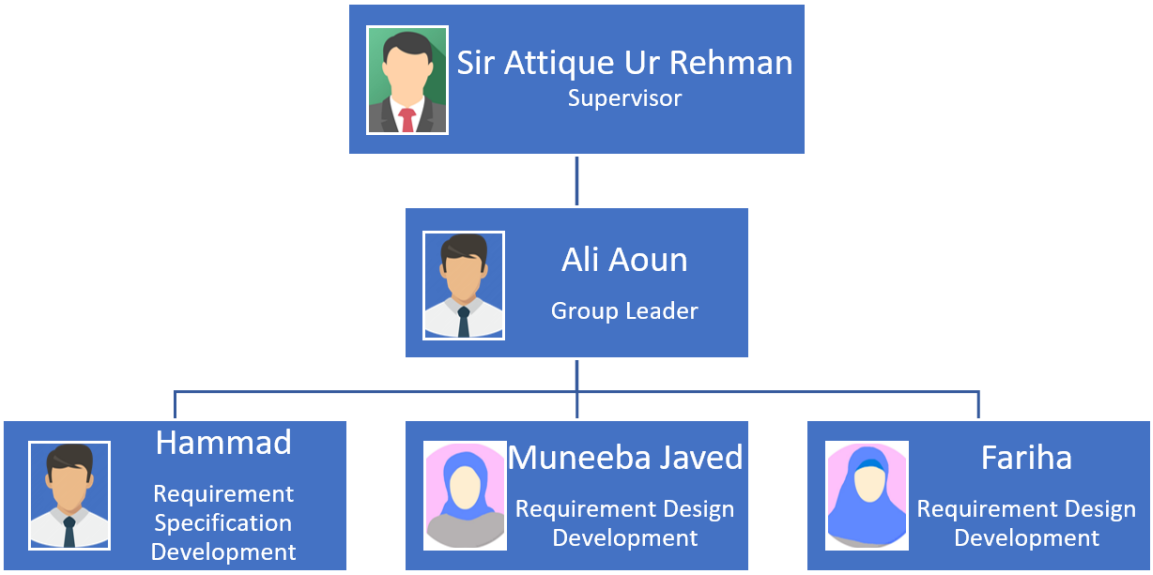

Figure 2.1: Team Organization Chart

Scope of the System

We propose prediction of cardiovascular disease provided the symptoms considering likelihood of having cardiovascular disease in patients having other chronic diseases like diabetes. Our primary motivation is to make a substantial impact in the realm of cardiovascular health. Similar to the challenge of mental health diagnosis, heart disease represents a significant global health issue, often leading to severe consequences when not detected early. We firmly believe that by harnessing advanced technology, such as machine learning, we can create a system that identifies early warning signs of heart disease. Our central objective is to develop a user-friendly tool that can effectively alert individuals to potential heart issues in their early stages. Achieving this goal holds the promise of providing timely support and treatment to those at risk, potentially preventing the progression of heart conditions to critical stages.

Summary of Requirements: (Initial Requirements)

The summary of our project CardioPulse is described by dividing it into functional and non-functional requirements as per the need of our system.

Functional Requirements:

Following are the functional requirements for our CardioPulse App:

- The system shall provide users with a comprehensive Cardiovascular Risk Prediction feature.
- Users shall be able to seamlessly register and log in to the CardioPulse application.
- Health Data Collection and Storage functionalities shall enable users to manage and store their cardiovascular health data securely.
- CardioPulse shall employ advanced machine learning algorithms to predict cardiovascular risks based on user data.
- Progress Monitoring and Reporting features shall empower users to track and visualize their cardiovascular health progress over time.

Non-Functional Requirements:

Following are the non-functional requirements of our CardioPulse app:

- **Reliability:** Considering reliability, the system should be able to tackle any exception. It should be full proof and able to handle undesired events and it should give accurate results.
- **Safety Requirement:** The database must be located in a special secure server and must be backups to other servers directly at specific times to avoid loss and damage data.
- **Performance Requirement:** The system must be on very high standard servers to accommodate the huge amount of data. The system should be efficient and responsive.
- **Usability Requirement:** The user interface must be familiar to user, so that the user can easily do his tasks without any training or help. The system should be easy to understand and easy to use for user.
- **Flexibility Requirements:** Our app must be able to provide add or delete features.
- **Availability Requirements:** Our app should be available at all times, meaning the user can access it using a device application.
- **Maintainability Requirements:** Our app should be easy to maintain regarding all its functionalities.

2.1.2. Identifying External Entities

The external entities of our proposed system have been identified on the basis of our business model. The dependencies include stakeholders such as patients, cardiologist and other health experts. Other than health domain, system administration and social societies are also included in the stakeholders. A healthcare facility (mainly related to the Cardiology Department) is main external dependency of the proposed project.

2.1.3. Context Level Data Flow Diagram:

The outline of external elements with software can be visually represented by the context level data flow diagram. The flow of proposed system has been represented in the following data flow diagram level 0. It contains one process named cardiovascular disease prediction which is representative of the system. All the other entities are shown with the data flow from user. Context level data flow diagram represents how external entities will use the proposed software through a series of processes. The data flow is represented by the arrows which are directed to a single process. The arrows out of the system towards the entities shows what process will return to the individual entity as a response to the provokes action. The context level data flow diagram has presented a high-level view of data flow of the system through the proposed project. The illustration of how user can interact with the proposed system is given with the type of processes that can be used by a particular user. A user can perform activities such as use account settings, registration, login and input symptoms to retrieve results. The user can view the report and result as a response from the system. The context level diagram of the proposed system is shown in the following Figure.

2.1.4. Capture "shall" Statements:

All the functional requirements are captured as “Shall” statements. The functional requirements of the system are identified and captured in “shall” statements. The “shall”

requirements of the proposed system are given below.

Table 2.1: "shall Statement"

N#	**Initial Requirements**
1.0	The system shall allow user to sign up to the system using email and password.
2.0	The system shall allow user to sign up as patient or doctor.
3.0	The system shall allow admin to manage users.
4.0	User shall login to the system using his/her username and password.
5.0	The system shall allow verified users to log in.
6.0	The system shall allow the user to change password.
7.0	The system shall allow the user to logout.
8.0	The system shall allow visitors to access only view mode of application.
9.0	The system shall allow user to retrieve data by giving input of patient's ID.
10.0	The system shall allow user to view history.
11.0	Admin shall be able to add data.
12.0	Admin shall be able to delete account.
13.0	The system shall allow admin to login.
14.0	The system shall allow admin to logout.
15.0	The system shall display dataset for user to choose from.
16.0	The system shall allow user to input parameters/fill form.
17.0	The system shall utilize database to identify parameters.
18.0	The system shall allow the user to view result.
19.0	The system shall display predicted result.
20.0	The system shall allow the user to generate report.
21.0	The system shall store the user's history in the database.
22.0	The system shall allow the user to update account settings
23.0	The system shall allow the Users to view account details.
24.0	The system shall allow the Users to update profile.
25.0	The system shall allow the Users to view Doctors.
26.0	The system shall provide the function to reset passwords.
27.0	The system "Shall" allow the Patient to ask query.
28.0	System shall be able to track and analyse system performance, patient engagement, and diagnostic numeracy.
29.0	System shall ensure the confidentiality and security of doctor data.
30.0	System shall search any syste details.
31.0	The system shall handle a large volume of data for disease detection.
32.0	Patient shall receive timely and clear results on the app.
33.0	System shall ensure the confidentiality and security of patient data.
34.0	System shall search any patient details.
35.0	Patient shall find the app intuitive and easy to use, facilitating seamless interaction with the disease detection features.
36.0	Patient shall benefit from rapid and accurate disease diagnoses, enabling prompt actions to protect their health.
37.0	The system shall have privileged access to manage and monitor doctor accounts and system functionalities.

38.0	Patient shall communicate with doctor
39.0	System shall Publish the content

2.1.5. Allocate Requirements:

The allocation of the requirements to the use cases takes place in this step. The functional requirements defined in the "Shall" statements are assigned a Use case name to identify the related use case of the requirement specification. The description of the requirements along with the related use case name has been defined. The requirements have been allocated to the use cases as below.

Table 2.2: Allocate Requirements

N#	Initial Requirements	Use case name
1.0	The system shall allow user to sign up to the system using email and password.	UC_SignUp
2.0	The system shall allow user to sign up as doctor.	UC_SignUp_Doctor
3.0	The system shall allow user to sign up as patient.	UC_SignUp_ patient.
4.0	The system shall allow admin to manage users.	UC_Manage
5.0	User shall login to the system using his/her username and password.	UC_Login
6.0	The system shall allow verified users to log in.	UC_Verify
7.0	The system shall allow the user to change password.	UC_Change_Password
8.0	The system shall allow the user to logout.	UC_Logout
9.0	The system shall allow visitors to access only view mode of application.	UC_View Application
10.0	The system shall allow user to retrieve data by giving input of patient's ID.	UC_ Retrieve_Data
11.0	The system shall allow user to view history.	UC_History
12.0	Admin shall be able to add data.	UC_Add_Data
13.0	Admin shall be able to delete account.	UC_Delete_Account
14.0	The system shall allow admin to login.	UC_Admin_login
15.0	The system shall allow admin to logout.	UC_Admin_Logout
16.0	The system shall display dataset for user to choose from.	UC_Display_Dataset
17.0	The system shall allow user to input parameters/fill form.	UC_Fillform
18.0	The system shall utilize database to identify parameters.	UC_Identify
19.0	The system shall allow the user to view result.	UC_View_Result
20.0	The system shall display predicted result.	UC_Predicted_Result
21.0	The system shall allow the user to generate report.	UC_Genrate_Report
22.0	The system shall store the user's history in the database.	UC_Store_History
23.0	The system shall allow the user to update account settings	UC_Update_Account
24.0	The system shall allow the Users to view account details.	UC_View_Account
25.0	The system shall allow the Users to update profile.	UC_Update_Profile
26.0	The system shall allow the Users to view Doctors.	UC_View_Doctor

27.0	The system shall provide the function to reset passwords.	UC_Reset_Password
28.0	The system "Shall" allow the Patient to ask query.	UC_Query
29.0	System shall be able to track and analyse system performance, patient engagement, and diagnostic numeracy.	UC_Performance
30.0	System shall ensure the confidentiality and security of doctor data.	UC_Security
31.0	System shall search any doctor details.	UC_Search
32.0	The system shall handle a large volume of data for disease detection.	UC_Handle
33.0	Patient shall receive timely and clear results on the app.	UC_Timely
34.0	System shall ensure the confidentiality and security of patient data.	UC_Confidential
35.0	System shall search any patient details.	UC_Seach_patient
36.0	Patient shall find the app intuitive and easy to use, facilitating seamless interaction with the disease detection features.	UC_UI
37.0	Patient shall benefit from rapid and accurate disease diagnoses, enabling prompt actions to protect their health.	UC_ Diagnose
38.0	The system shall have privileged access to manage and monitor doctor accounts and system functionalities.	UC_ Communicate
39.0	System shall Publish the content	UC_Publish

2.1.6. Prioritize Requirements:

The priority ranks of the requirements have been assigned as follows.

Table 2.3: Prioritize Requirements

N#	Rank	Initial Requirements	Use Case ID	Use case name
1.0	Medium	The system shall allow user to sign up to the system using email and password.	UC_1	UC_SignUp
2.0	Highest	The system shall allow user to sign up as doctor.	UC_2	UC_SignUp_Doctor
3.0	Lowest	The system shall allow user to sign up as patient.	UC_3	UC_SignUp_ patient.
4.0	Highest	The system shall allow admin to manage users.	UC_4	UC_Manage
5.0	Highest	User shall login to the system using his/her username and password.	UC_5	UC_Login

6.0	Highest	The system shall allow verified users to log in.	UC_6	UC_Verify
7.0	Medium	The system shall allow the user to change password.	UC_7	UC_Change_Password
8.0	Medium	The system shall allow the user to logout.	UC_8	UC_Logout
9.0	Lowest	The system shall allow visitors to access only view mode of application.	UC_9	UC_View Application
10.0	Lowest	The system shall allow user to retrieve data by giving input of patient's ID.	UC_10	UC_ Retrieve_Data
11.0	Medium	The system shall allow user to view history.	UC_11	UC_History
12.0	Highest	Admin shall be able to add data.	UC_12	UC_Add_Data
13.0	Medium	Admin shall be able to delete account.	UC_13	UC_Delete_Account
14.0	Medium	The system shall allow admin to login.	UC_14	UC_Admin_login
15.0	Medium	The system shall allow admin to logout.	UC_15	UC_Admin_Logout
16.0	Highest	The system shall display dataset for user to choose from.	UC_16	UC_Display_Dataset
17.0	Highest	The system shall allow user to input parameters/fill form.	UC_17	UC_Fillform
18.0	Highest	The system shall utilize database to identify parameters.	UC_18	UC_Identify
19.0	Medium	The system shall allow the user to view result.	UC_19	UC_View_Result
20.0	Highest	The system shall display predicted result.	UC_20	UC_Predicted_Result
21.0	Medium	The system shall allow the user to generate report.	UC_21	UC_Genrate_Report
22.0	Medium	The system shall store the user's history in the database.	UC_22	UC_Store_History
23.0	Lowest	The system shall allow the user to update account settings	UC_23	UC_Update_Account
24.0	Highest	The system shall allow the Users to view account details.	UC_24	UC_View_Account
25.0	Medium	The system shall allow the Users to update profile.	UC_25	UC_Update_Profile
26.0	Lowest	The system shall allow the Users to view Doctors.	UC_26	UC_View_Doctor
27.0	Medium	The system shall provide the function to reset passwords.	UC_27	UC_Reset_Password

28.0	Medium	The system Shall allow the Patient to ask query.	UC_28	UC_Query
29.0	Medium	System shall be able to track and analyse system performance, patient engagement, and diagnostic numeracy.	UC_29	UC_Performance
30.0	Highest	System shall ensure the confidentiality and security of doctor data.	UC_30	UC_Security
31.0	Medium	System shall search any doctor details.	UC_31	UC_Search
32.0	Lowest	The system shall handle a large volume of data for disease detection.	UC_32	UC_Handle
33.0	Highest	Patient shall receive timely and clear results on the app.	UC_33	UC_Timely
34.0	Medium	System shall ensure the confidentiality and security of patient data.	UC_34	UC_Confidential
35.0	Lowest	System shall search any patient details.	UC_35	UC_Seach_patient
36.0	Medium	Patient shall find the app intractive and easy to use, facilitating seamless interaction with the disease detection features.	UC_36	UC_UI
37.0	Medium	Patient shall benefit from rapid and accurate disease diagnoses, enabling prompt actions to protect their health.	UC_37	UC_ Diagnose
38.0	Medium	Patient shall communicate with doctor	UC_38	UC_ Communicate
39.0	Highest	System shall Publish the content	UC_39	UC_Publish

2.1.7. Requirements Trace-ability Matrix:

All the requirements specified are categorized and allocated to the respective use case. The description of the requirements is also given in order to form a traceability matrix to map the requirements. The parameters of the requirements trace ability matrix are build, paragraph, description, use case and category. These parameters are defined to map the specified requirements accordingly. Build number is assigned to the requirements according to the business requirement of each statement. The paragraph number defines the paragraph where the requirement has been summarized in the document. The requirement traceability matrix for the system is given below.

Table 2.4: Requirements Trace-ability

N#	Paragraph	System Specification text	Build	Use case name	Category
1.	P1	The system shall allow user to sign up to the system using email and password.	B1	UC_SignUp	Business
2.	P1	The system shall allow user to sign up as doctor.	B1	UC_SignUp_Doctor	Business
3.	P1	The system shall allow user to sign up as patient.	B1	UC_SignUp_ patient.	Business
4.	P2	The system shall allow admin to manage users.	B1	UC_Manage	Business
5.	P3	User shall login to the system using his/her username and password.	B1	UC_Login	Business
6.	P2	The system shall allow verified users to log in.	B1	UC_Verify	Business
7.	P1	The system shall allow the user to change password.	B1	UC_Change_Password	Business
8.	P2	The system shall allow the user to logout.	B1	UC_Logout	Business
9.	P2	The system shall allow visitors to access only view mode of application.	B1	UC_View Application	Business

10.	P3	The system shall allow user to retrieve data by giving input of patient's ID.	B1	UC_ Retrieve_Data	Business
11.	P1	The system shall allow user to view history.	B1	UC_History	Business
12.	P3	Admin shall be able to add data.	B1	UC_Add_Data	Business
13.	P1	Admin shall be able to delete account.	B1	UC_Delete_Account	Business
14.	P1	The system shall allow admin to login.	B1	UC_Admin_login	Business
15.	P2	The system shall allow admin to logout.	B1	UC_Admin_Logout	Business
16.	P3	The system shall display dataset for user to choose from.	B1	UC_Display_Dataset	Business
17.	P2	The system shall allow user to input parameters/fill form.	B1	UC_Fillform	Business
18.	P1	The system shall utilize database to identify parameters.	B1	UC_Identify	Business
19.	P2	The system shall allow the user to view result.	B1	UC_View_Result	Business
20.	P1	The system shall display predicted result.	B1	UC_Predicted_Result	Business
21.	P3	The system shall allow the user to generate report.	B1	UC_Genrate_Report	Business
22.	P2	The system shall store the user's history in the database.	B1	UC_Store_History	Business

23.	P1	The system shall allow the user to update account settings	B1	UC_Update_Account	Business
24.	P1	The system shall allow the Users to view account details.	B1	UC_View_Account	Business
25.	P2	The system shall allow the Users to update profile.	B1	UC_Update_Profile	Business
26.	P3	The system shall allow the Users to view Doctors.	B1	UC_View_Doctor	Business
27.	P2	The system shall provide the function to reset passwords.	B1	UC_Reset_Password	Business
28.	P2	The system "Shall" allow the Patient to ask query.	B1	UC_Query	Business
29.	P3	System shall be able to track and analyse system performance, patient engagement, and diagnostic numeracy.	B1	UC_Performance	Business
30.	P2	System shall ensure the confidentiality and security of doctor data.	B1	UC_Security	Business
31.	P1	System shall search any doctor details.	B1	UC_Search	Business
32.	P1	The system shall handle a large volume of data for disease detection.	B1	UC_Handle	Business
33.	P2	Patient shall receive timely and clear results on the app.	B1	UC_Timely	Business

34.	P3	System shall ensure the confidentiality and security of patient data.	B1	UC_Confidential	Business
35.	P2	System shall search any patient details.	B1	UC_Seach_patient	Business
36.	P2	Patient shall find the app intractive and easy to use, facilitating seamless interaction with the disease detection features.	B1	UC_UI	Business
37.	P3	Patient shall benefit from rapid and accurate disease diagnoses, enabling prompt actions to protect their health.	B1	UC_ Diagnose	Business
38.	P2	Patient shall communicate with doctor	B1	UC_ Communicate	Business
39.	P1	System shall Publish the content	B1	UC_Publish	Business

2.1.8. High Level Use Case Diagram:

The high-level use case describes the relation between actors, system and its defined use cases. The proposed system has three main actors that interact with the system. The high-level use case diagram presents a brief summary of the proposed system in a simple visual representation. There is an actor that will interact with our system which are given user and system. The high-level use case diagram of cardiovascular disease prediction is as follows.

Figure 2.2: High level use case diagram

2.1.9. Analysis Level Use Case Diagram:

Analysis level diagram describes the system in more detail than the high-level diagram. It provides extensive functionalities of the system using extend and include relationship. The analysis level use case diagram of the proposed system is as follows.

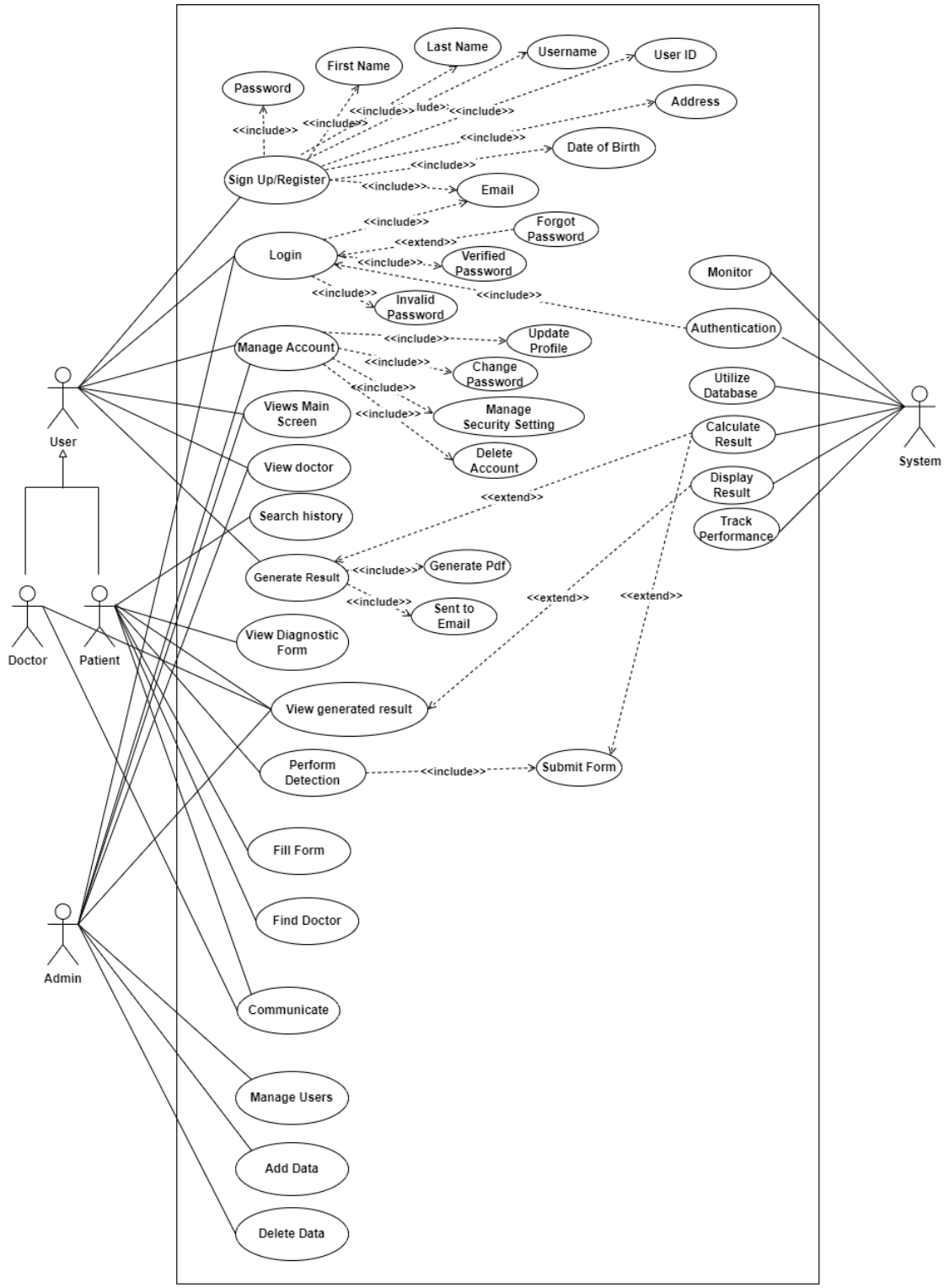

Figure 2.3: Analysis level use case diagram

2.1.10. Use case Description

The use case description of the use case diagram is given below. The description of Login use case is in the following table.

Table 2.5: Login

Use case ID	UC_2
Name	Login
Brief description	User login to the system to access the functionality of the system.
Actors	User
Preconditions	User must be connected to the system and have an account
Basic flow	If login information is correct, the user is logged in. After login, the system shows home page.
Alternate flows	An error message is displayed. The actor can input a new name or password or choose to cancel the operation.
Exception flows	If network is not connected, system shows error.
Post conditions	After successful login, actor is taken to the home page.
Include use cases	Identification and authentication of user.

The description of Delete account use case is in the following table.

Table 2.6: Delete

Use case ID	UC_5
Name	Delete Account
Brief description	User can delete account.
Actors	System, user
Preconditions	User must be logged in. User sends account deletion request.
Basic flow	User sends a deletion request or administrator clicks on “Delete” button. The system deletes the account.
Alternate flows	Administrator or user cancels the deletion request.
Exception flows	Incorrect credentials
Post conditions	The account has been successfully removed.
Include use cases	None

The description of search use case is in the following table.

Table 2.7: Search

Use case ID	UC_12

Name	Search
Brief description	Registered User can search history record.
Actors	System, user
Preconditions	The User views the home page and enters the patient's ID in search bar.
Basic flow	The System executes the query retrieved from the User. The System retrieves the matching record. The System displays the search results
Alternate flows	User clicks on "View History".
Exception flows	Patient's ID does not exist or not found.
Post conditions	The System retrieves the search result and displays to the User
Include use cases	Search history.

The description of input parameters use case is in the following table.

Table 2.8: Enter Parameters

Use case ID	UC_13
Name	Enter Parameters
Brief description	Users can input parameters and provide symptoms..
Actors	System, user
Preconditions	None.
Basic flow	User provides symptoms. The system processes the input and predicts disease if any.
Alternate flows	N/A
Exception flows	N/A
Post conditions	System displays result.
Include use cases	Analyse data.

The description of manage user account use case is in the following table.

Table 2.9: Manage User Account

Use case ID	UC_7
Name	Manage User Account
Brief description	Users can register their accounts to get additional services such as request data by providing patient's ID.
Actors	System, user
Preconditions	User enters information on the registration form
Basic flow	User enters necessary information. If the retrieved information is correct the system registers the user account. Otherwise, the system displays an error message.
Alternate flows	If the retrieved information is incorrect the system displays an error message.

Exception flows	If the given information is fake, the system notifies then administration.
Post conditions	User can access account settings to update or edit information.
Include use cases	None

The description of Logout use case is in the following table.

Table 2.10: Log Out

Use case ID	UC_3
Name	Log Out
Brief description	The user clicks on “Logout” and the session is terminated.
Actors	User
Preconditions	User is logged in. User no longer wants to be logged in.
Basic flow	User clicks on the logout button. The user is logged out of the system. The system redirects to the default logout page.
Alternate flows	N/A
Exception flows	Log out is not successful due to failure in connection.
Post conditions	The user is logged out
Include use cases	Login to the system

The description of Sign-up use case is in the following table.

Table 2.11: Sign Up

Use case ID	UC_1
Name	Sign Up
Brief description	User is registered to the system by providing required information.
Actors	User
Preconditions	User must be connected to the system.
Basic flow	If all required fields are filled, the user is registered otherwise it shows an error. After Signing Up, user can login from registered account.
Alternate flows	If the system cannot find the name or the password is invalid, an error message is displayed. User can type in a new name or password or choose to cancel the operation.
Exception flows	If network is not connected, system shows error.
Post conditions	After successful Signup, user is taken to the login page and then to homepage.
Include use cases	User Registration

The description of generate result use case is in the following table

Table 2.12: Generate Result

Use case ID	UC_10
Name	Generate Result
Brief description	User can generate result after providing parameters. The result includes disease prediction.
Actors	User
Preconditions	User must provide symptoms to generate result.
Basic flow	User clicks on “Result” button. The system processes provided symptoms. The result is generated by the system.
Alternate flows	N/A
Exception flows	N/A
Post conditions	The system generates result.
Include use cases	None

The description of generate report use case is in the following table.

Table 2.13: Generate Reports

Use case ID	UC_11
Name	Generate Report
Brief description	User can generate report after result generation.
Actors	User
Preconditions	User must have predicted result generated.
Basic flow	User clicks on “Report” or “Download” button. The system downloads the report.
Alternate flows	N/A
Exception flows	N/A
Post conditions	The system generates report of result.
Include use cases	None

The description of Delete Account use case is in the following table.

Table 2.14: Delete Account

Use case ID	UC_12
Name	Delete Account
Brief description	Admin can user account
Actors	Admin
Preconditions	Admin must have Log in
Basic flow	Admin clicks on “Delete” or “button. The system Delete the user account
Alternate flows	N/A

Exception flows	N/A
Post conditions	The system generates report of result.
Include use cases	None

Chapter 3:

Software Architecture and Design

3.1. Introduction:

Software design of the proposed system is based on the requirements specified in the requirement specification document. We have all the requirements specified and now we will design the system to be developed according to the requirements. We will present the software design using object-oriented approach. Following artifacts are included in the software design document

1. Domain Model
2. System Sequence Diagram
3. Sequence Diagram
4. Collaboration Diagram
5. Operation Contracts
6. Design Class Diagram
7. State Transition Diagram
8. Data Model

Now we discuss these artifacts one by one as follows:

3.2. Domain Model

The domain model for CardioPulse represents a comprehensive overview of the essential entities and their interrelationships within the cardiovascular health monitoring system. The scope of the domain encompasses various aspects critical to cardiovascular health management. The primary components include:

- **User:**
  Attributes: UserID, Username, Password.
  Description: Represents individuals using the CardioPulse app to monitor their cardiovascular health.

- **Health Data:**
  Attributes: DataID, UserID, Timestamp, HeartRate, BloodPressure, CholesterolLevel etc.
  Description: Captures essential health metrics recorded by users, such as heart rate, blood pressure, and cholesterol levels etc.

- **Risk Prediction:**
  Attributes: PredictionID, UserID, RiskLevel.
  Description: Provides predictions related to the user's cardiovascular risk level based on recorded health data.

- **Progress Report:**
  Attributes: ReportID, UserID, Date, Description.
  Description: Documents users' progress in managing cardiovascular health, including relevant descriptions and dates.

The domain model demonstrates associations between these entities, illustrating the relationships essential for the CardioPulse system. The User entity serves as the central point, connected to Health Data, Risk Prediction, and Progress Report entities. This

interconnected structure allows for a holistic representation of the cardiovascular health monitoring domain.

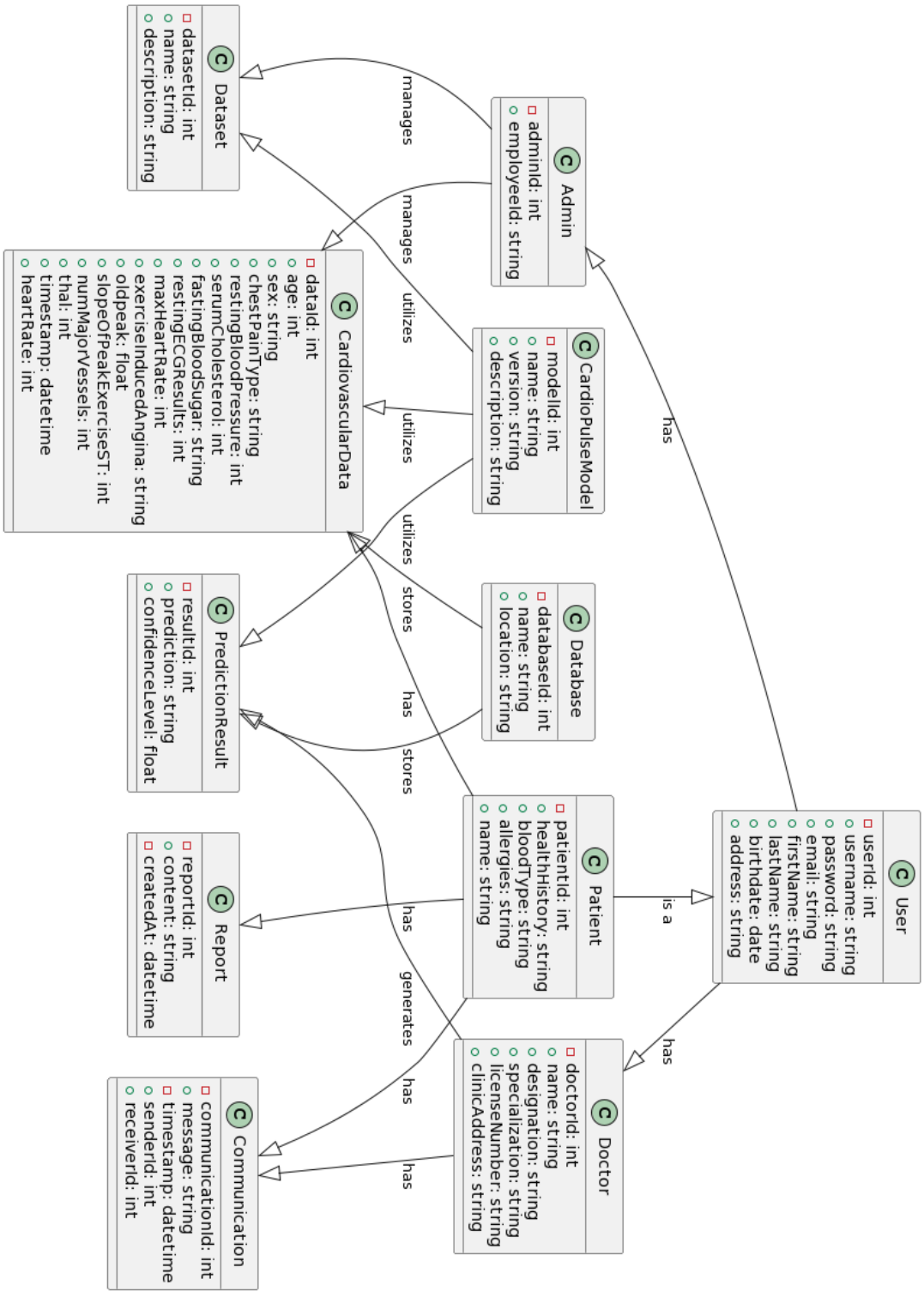

Figure 3.1: Domain Model Diagram

3.3. Architecture/Design Pattern

The design pattern for the CardioPulse software system describes the complexity level of

development stages and the approach to be followed throughout the system construction. We have proposed the design pattern for cardiovascular disease prediction system with a focus on reusability and reduced complexity. The design pattern used for the system is described as follows.

3.3.1. Factory Method

Within the CardioPulse system, the Factory Method architectural pattern is employed to define an interface for creating various objects, allowing subclasses to determine the specific class instantiation. For instance, the CardiovascularDataFactory interface enables subclasses to decide the creation of different cardiovascular data objects based on specific needs, contributing to a flexible and extensible design.

3.3.2. Adapter

The Adapter architectural pattern is utilized in CardioPulse to convert the interface of classes into formats expected by clients. For instance, the CardiovascularDataAdapter transforms diverse data interfaces related to cardiovascular health into a unified format, enabling seamless collaboration between classes with otherwise incompatible interfaces. This adaptation enhances interoperability and data consistency within the system.

3.3.3. Façade

In CardioPulse, the Façade architectural pattern is implemented to provide a unified and simplified interface to a set of subsystem interfaces. The CardiovascularSubsystemFacade defines a higher-level interface, making interactions with the cardiovascular subsystem more user-friendly. This Façade pattern enhances the overall usability and comprehensibility of the cardiovascular features within the system.

3.3.4. MVC

This pattern is commonly used in most mobile applications due to its high reusability and the ability to reduce system complexity. The MVC design pattern comprises the following layers:

3.3.4.1. Model Responsibilities

The Model in the MVC design pattern represents the logical layer of the system or operations to be performed. The logical layer is encapsulated in the model. The proposed CardioPulse system has the following models in the model layer:

- User
- CardiovascularData
- RiskPrediction
- Report

These models control the modules related to each actor of the system. The user accounts and cardiovascular data must be handled following the design pattern. Forms for different actors or users of the system are designed according to the required view. Strongly typed views are listed in ViewModel types containing the data to be displayed, and the associated controller populates these ViewModel objects from the model layer.

3.3.4.2. View Responsibilities

Views in the MVC design pattern are used to present content through the user interface. Major views of the CardioPulse system include:

- User Dashboard
- Cardiovascular Data View
- Risk Prediction View
- Report View

These views represent the mobile pages displayed to users. Each view is designed to meet the specific needs of the corresponding user or actor.

3.3.4.3. Controller Responsibilities

Controllers in the MVC design pattern are component handlers that control user interaction, model, and view. They respond to user interaction with the system. The required controllers for the CardioPulse system are:

- UserController
- CardiovascularDataController
- RiskPredictionController
- ReportController

Each controller receives user requests, fetches data from the related model, and interacts with the associated view. The Model-View-Controller design pattern is suitable for the CardioPulse mobile application. It reduces design complexity by dividing responsibilities into three layers.

3.4. Algorithm Flow Chart

A flowchart is a graphical representation of an algorithm. Programmers often use it as a program-planning tool to solve a problem. It makes use of symbols which are connected among them to indicate the flow of information and processing. The process of drawing a flowchart for an algorithm is known as “flowcharting”.

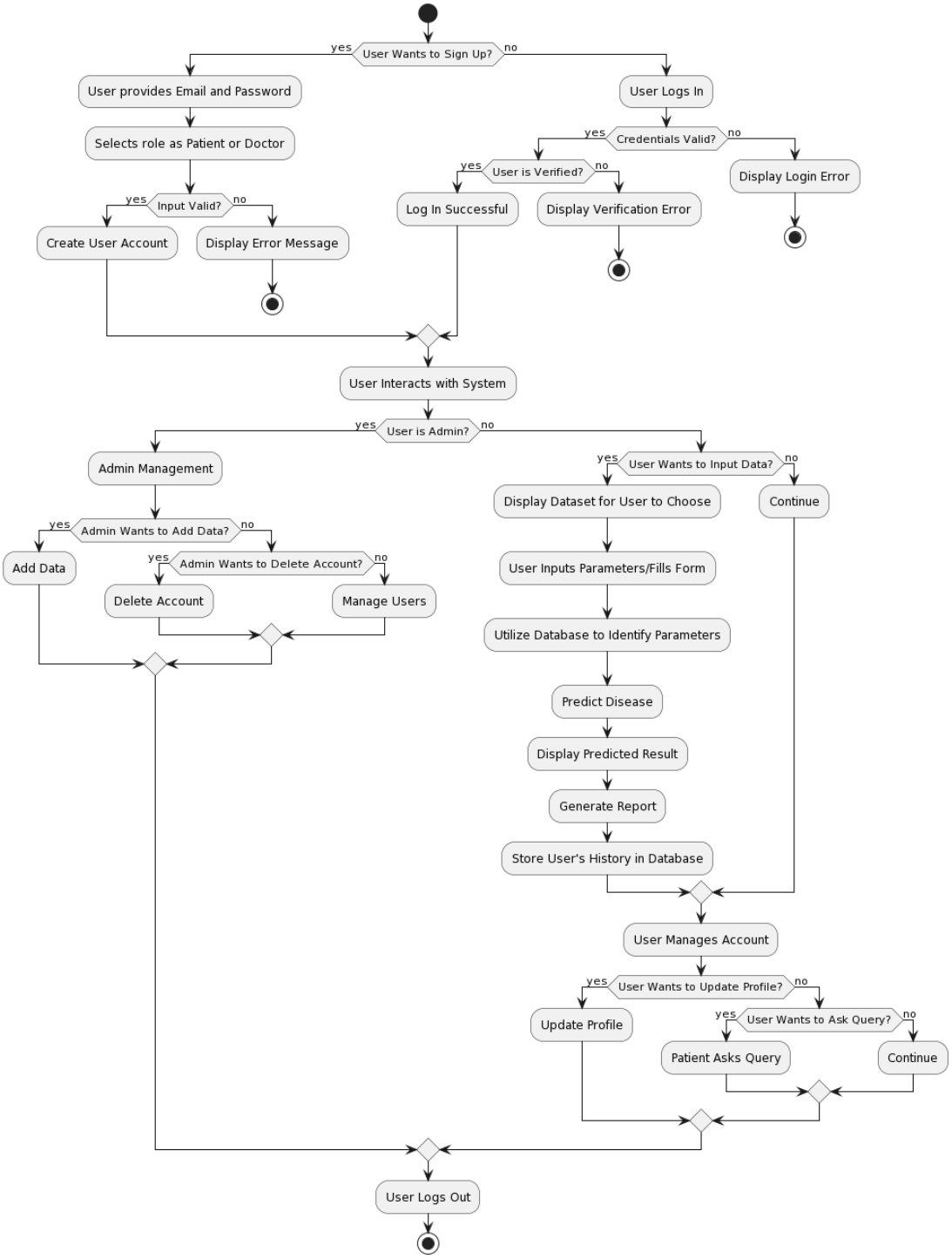

Figure 3.2: Algorithm Flow Chart

3.5 System Sequence Diagram

The System Sequence Diagram for CardioPulse (CP) encapsulates the sequential events initiated by the user within the application. This serves as a chronological representation of key interactions betwee user and CardioPulse sys during the CVD Prediction process.

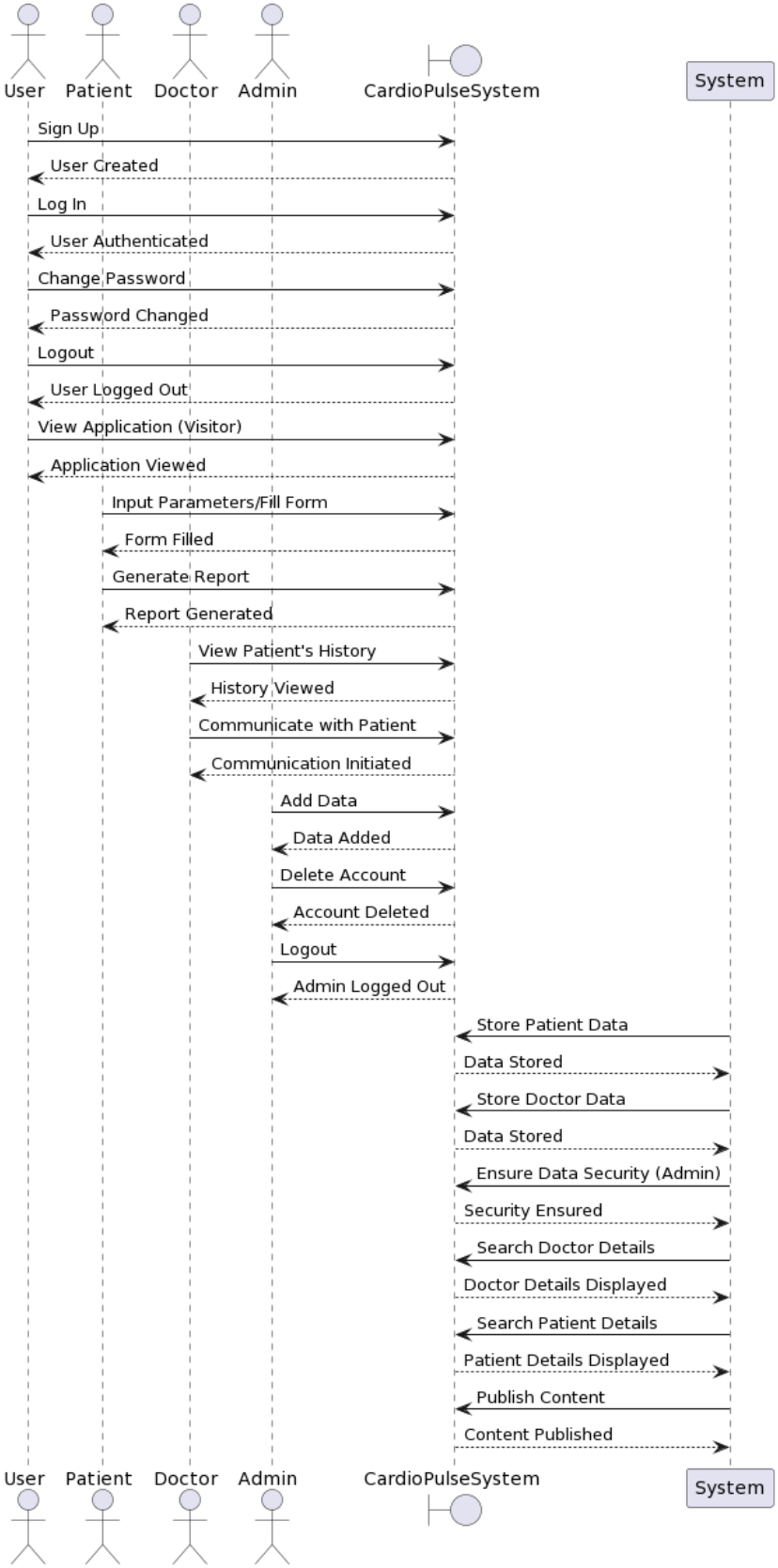

Figure 3.3: System Sequence Diagram

3.6. Sequence Diagram

The sequence diagram for the CardioPulse app illustrates the dynamic interactions between key actors and system components during the disease diagnosis process.

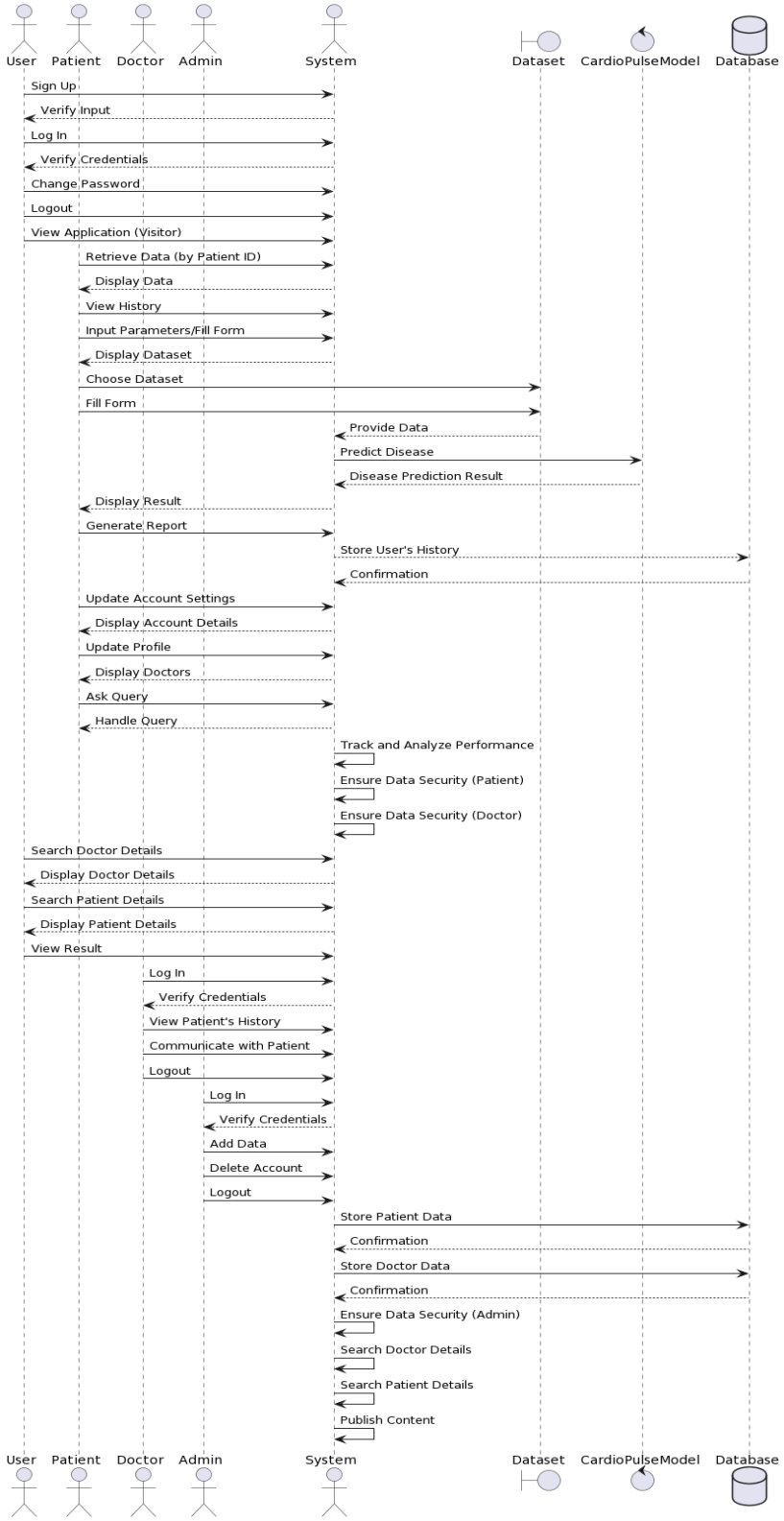

Figure 3.4: Sequence Diagram

3.7. Collaboration Diagram

Collaboration diagram or communication diagram of the proposed system shows the interaction and relation of the external entities with the system use cases or objects. The collaboration diagram shows the interaction of user with login, registration, account and mobile app form user interfaces. The communication of the user with the system has been represented step by step in the following figure.

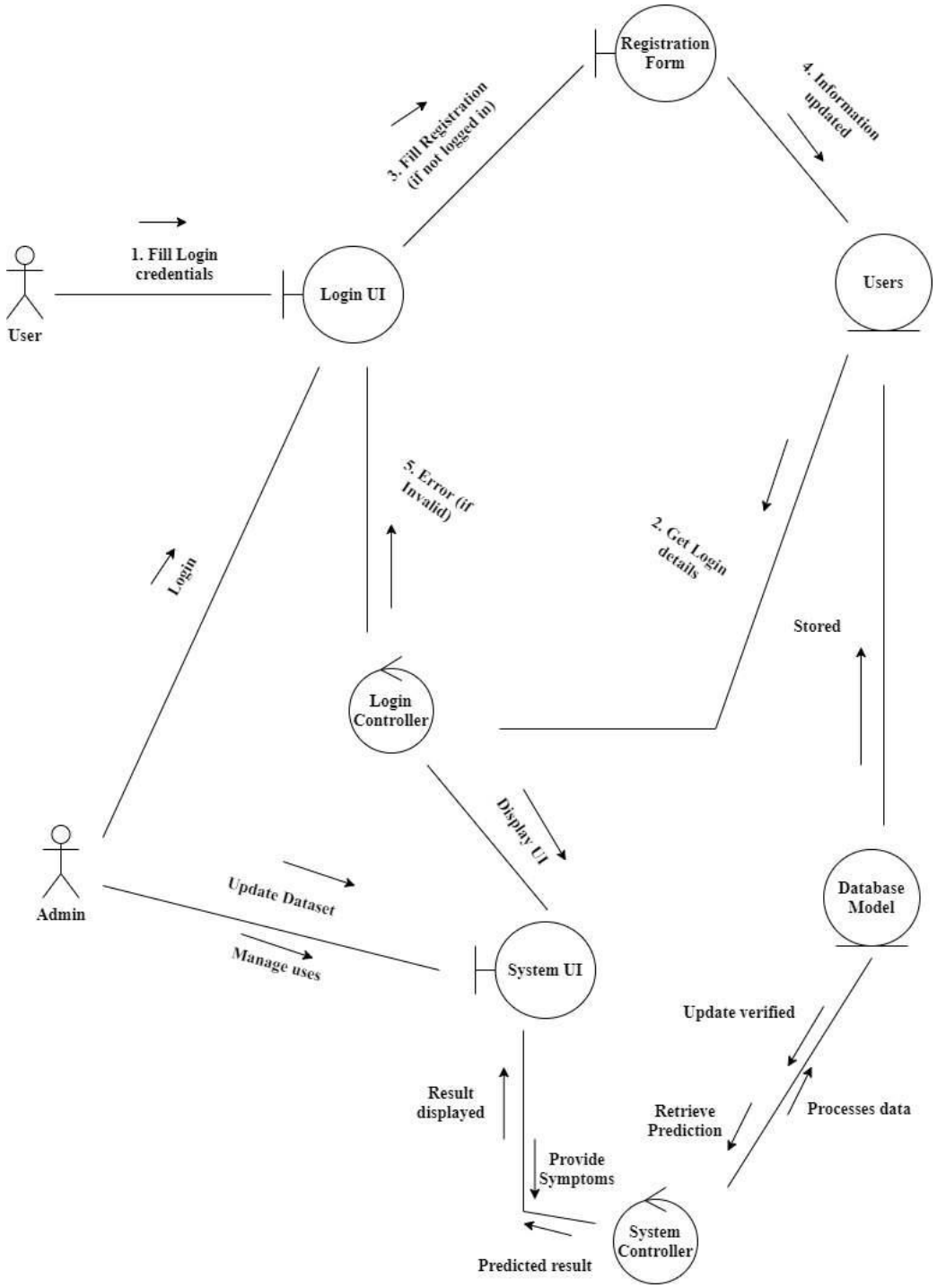

Figure 3.5: Collaboration Diagram

3.8. Operation Contracts

A UML Operation contract identifies system state changes when an operation happens. Effectively, it will define what each system operation does. An operation is taken from a system sequence diagram. It is a single event from that diagram. A domain model can be used to help generate an operation contract.

Operation Contract Syntax of RegisterUser:

Name: RegisterUser
Responsibilities: Register a new user in the CardioPulse system.
Cross References: User Registration Use Case
Exceptions:

- User already exists with the same username.
- Invalid input provided during registration.

Preconditions:

- The user is not currently registered in the system.
- The provided username is unique.

Postconditions:

- A new user with the specified username is registered.

  The user is now logged in.

Operation Contract Syntax of SignInUser:

Name: SignInUser
Responsibilities: Sign in an existing user to the CardioPulse system.
Cross References: User Sign-In Use Case
Exceptions:

- Invalid username or password.
- User is already signed in.

Preconditions:

- The user is not currently signed in.

Postconditions:

- The user is now signed in.

Operation Contract Syntax of ViewUserRecords:

Name: ViewUserRecords
Responsibilities: Allow the user to view their historical records.
Cross References: View Records Use Case
Exceptions:

- No records available.

Preconditions:

- The user is signed in.

Postconditions:

- The user is viewing their historical records.

Operation Contract Syntax of SelectDataset:

Name: SelectDataset
Responsibilities: Allow the user to select a dataset for prediction.
Cross References: Select Dataset Use Case
Exceptions:

- No datasets available.

Preconditions:

- The user is signed in.

Postconditions:

- The user has selected a dataset.

Operation Contract Syntax of SubmitFormData:

Name: SubmitFormData
Responsibilities: Submit user-filled data for prediction.
Cross References: Submit Form Use Case
Exceptions:

- Invalid form data.

Preconditions:

- The user has selected a dataset.

Postconditions:
The submitted data is processed for prediction

Operation Contract Syntax of CaptureCardioData:

Name: CaptureCardioData
Responsibilities: Capture and record cardiovascular data from the user.
Cross References: Cardiovascular Data Input Use Case
Exceptions:

- Invalid or incomplete cardiovascular data.
- Technical issues during data capture.

Preconditions:

- The user is logged into the system.

Postconditions:

- Cardiovascular data is successfully recorded in the system.

Operation Contract Syntax of PredictDiseaseRisk:

Name: PredictDiseaseRisk

Responsibilities: Utilize the CardioPulse model to predict disease risk based on user data.
Cross References: Disease Risk Prediction Use Case
Exceptions:

- Insufficient data for prediction.
- Technical issues during the prediction process.

Preconditions:

- The user has submitted valid cardiovascular data.

Postconditions:

- Disease risk is predicted and available for the user.

Operation Contract Syntax of GenerateHealthReport:

Name: GenerateHealthReport
Responsibilities: Generate a comprehensive health report based on user history and predictions.
Cross References: Health Report Generation Use Case
Exceptions:

- No relevant data available for report generation.

Preconditions:

- The user has an existing health history.

Postconditions:

- A health report is generated and accessible to the user.

Operation Contract Syntax of CommunicateWithDoctor:

Name: CommunicateWithDoctor
Responsibilities: Enable communication between the user and their assigned doctor.
Cross References: User-Doctor Communication Use Case
Exceptions:

- Technical issues preventing successful communication.
- The doctor or user is not available for communication.

Preconditions:

- The user is logged into the system.

Postconditions:

- Successful communication is established between the user and doctor.

Operation Contract Syntax of UpdateUserSettings:

Name: UpdateUserSettings
Responsibilities: Allow users to update their account settings and preferences.
Cross References: User Settings Update Use Case
Exceptions:

- Invalid or incomplete information provided for the update.

Preconditions:

- The user is logged into the system.

Postconditions:

- User account settings are successfully updated.

3.9. Design Class Diagram

The structure of the proposed CardioPulse system has been represented by class diagram which shows the various elements of the proposed system. Each class contains respective variables and functions. The relationship among classes has also been defined. The class diagram for the CardioPulse application includes several interconnected components that facilitate comprehensive cardiovascular health management. The User class serves as the base with attributes like userId, username, password, and methods for user actions such as signUp(), signIn(), and updateProfile(). The Patient class extends User, adding health-specific attributes (patientId, healthHistory, bloodType) and methods to manage patient data (retrieveData(), viewHistory(), generateReport()). The Doctor class, also extending User, includes attributes like doctorId, specialization, and methods for viewing patient history and communication.

The Admin class handles administrative tasks with methods for adding data and deleting accounts. The CardiovascularData class contains attributes critical for disease prediction, such as age, sex, restingBloodPressure, and the method predictDisease(). The PredictionResult class captures prediction outcomes, including prediction and confidenceLevel. The Report class documents health reports, while the Communication class manages messaging between doctors and patients.

The Dataset class manages the datasets used for training the predictive model, and the CardioPulseModel class encapsulates the machine learning model used for disease prediction. Finally, the Database class stores cardiovascular data and prediction results. Relationships among these classes include inheritance (Patient and Doctor from User), associations (Patient with CardiovascularData and Report), and utilization (CardioPulseModel with CardiovascularData and PredictionResult), ensuring an integrated approach to cardiovascular health management within the application.

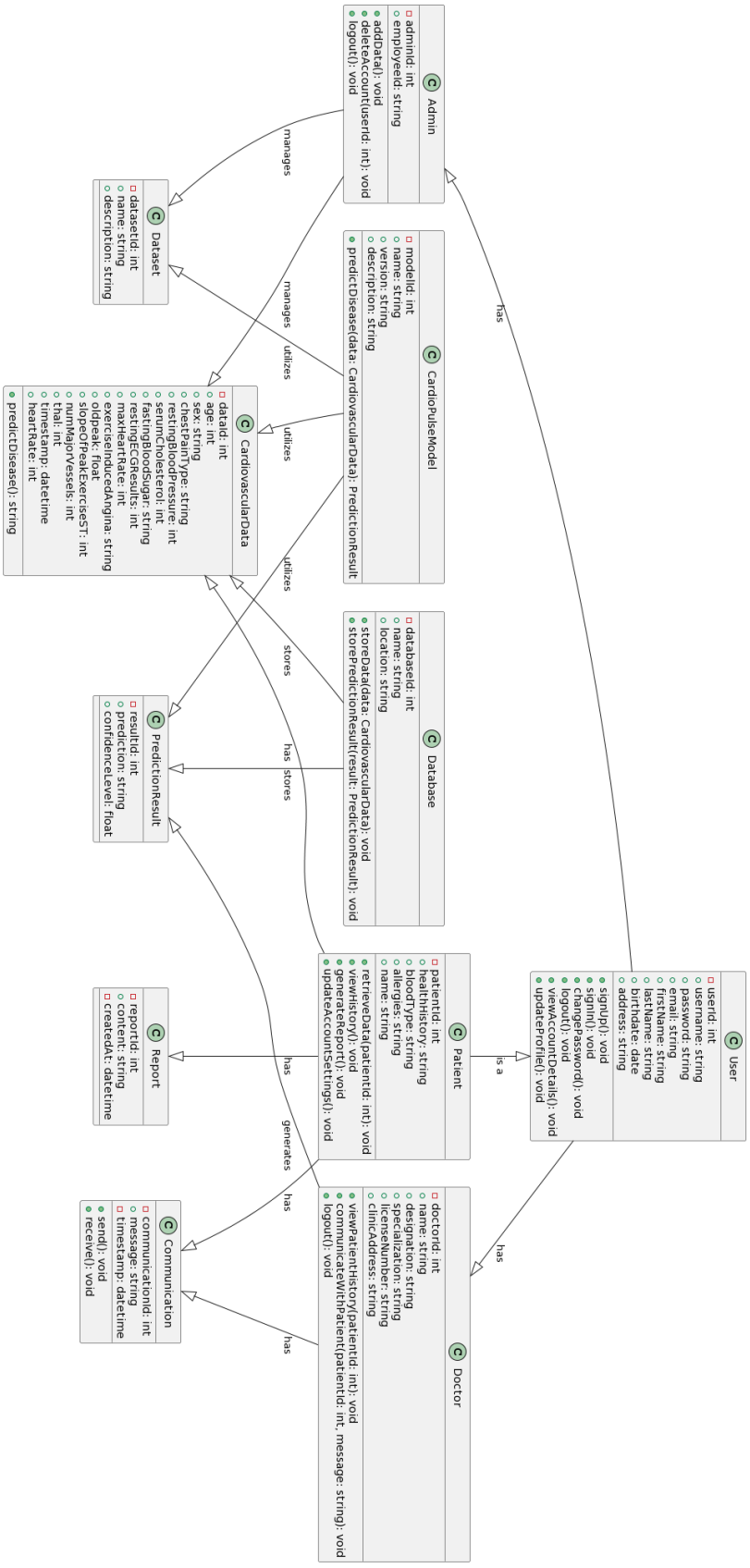

Figure 3.6: Design Class Diagram

3.10. State chart diagram

In our CardioPulse system, the State Chart Diagram captures the dynamic behavior of the User entity as it progresses through various states while interacting with the application. This diagram is instrumental in illustrating how different events trigger transitions between states, influencing the behavior of operations.

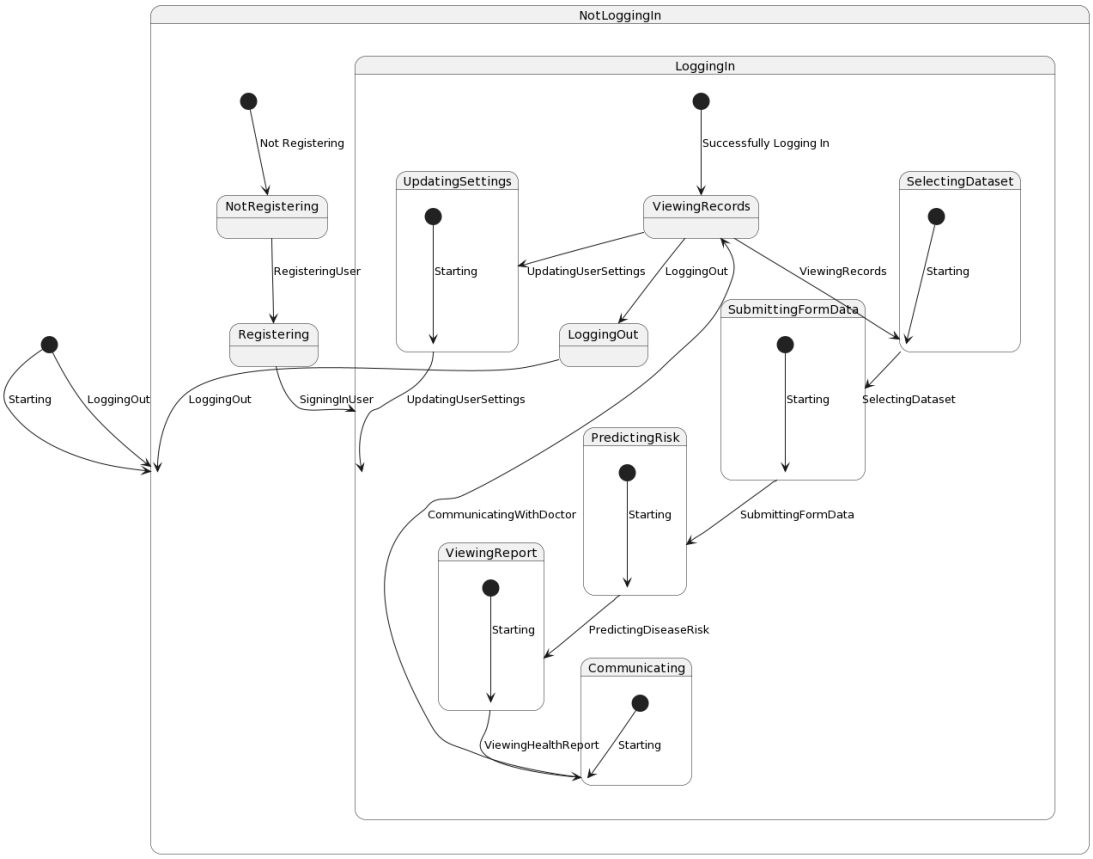

Figure 3.7: State chart diagram

3.11. Database Model

The database model of the system visually represents the system database and relation of its components. The database diagram has been designed using Crow's foot notation. Each entity has key and field of attributes. The relation cardinality among the entities has also been described in the database diagram. The database diagram of the cardiovascular disease prediction system is given in the following.

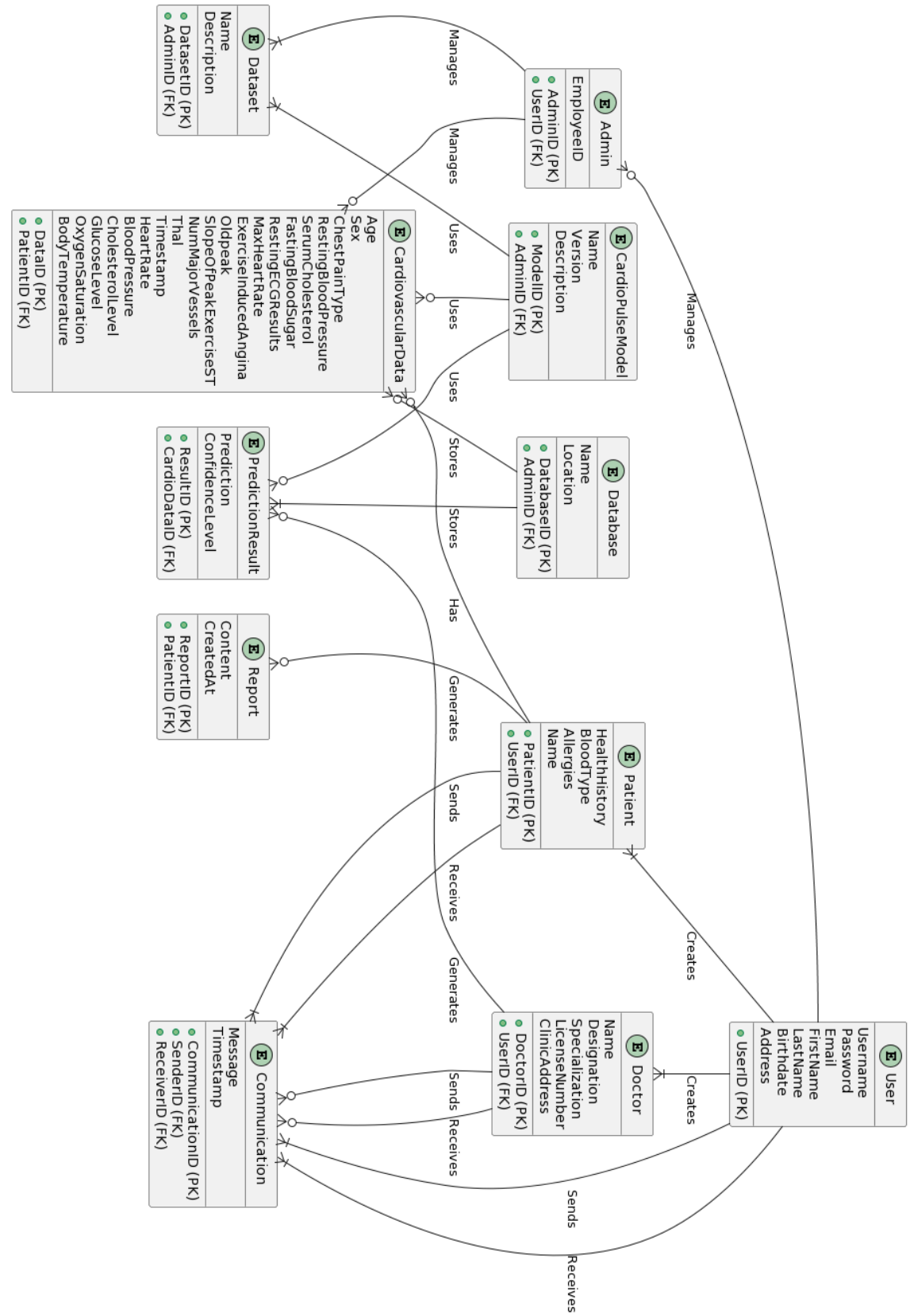

Figure 3.8: Database Model

3.12. Activity Diagram

The activity diagram that shows the activities going into the system of the CardioPulse, is given below.

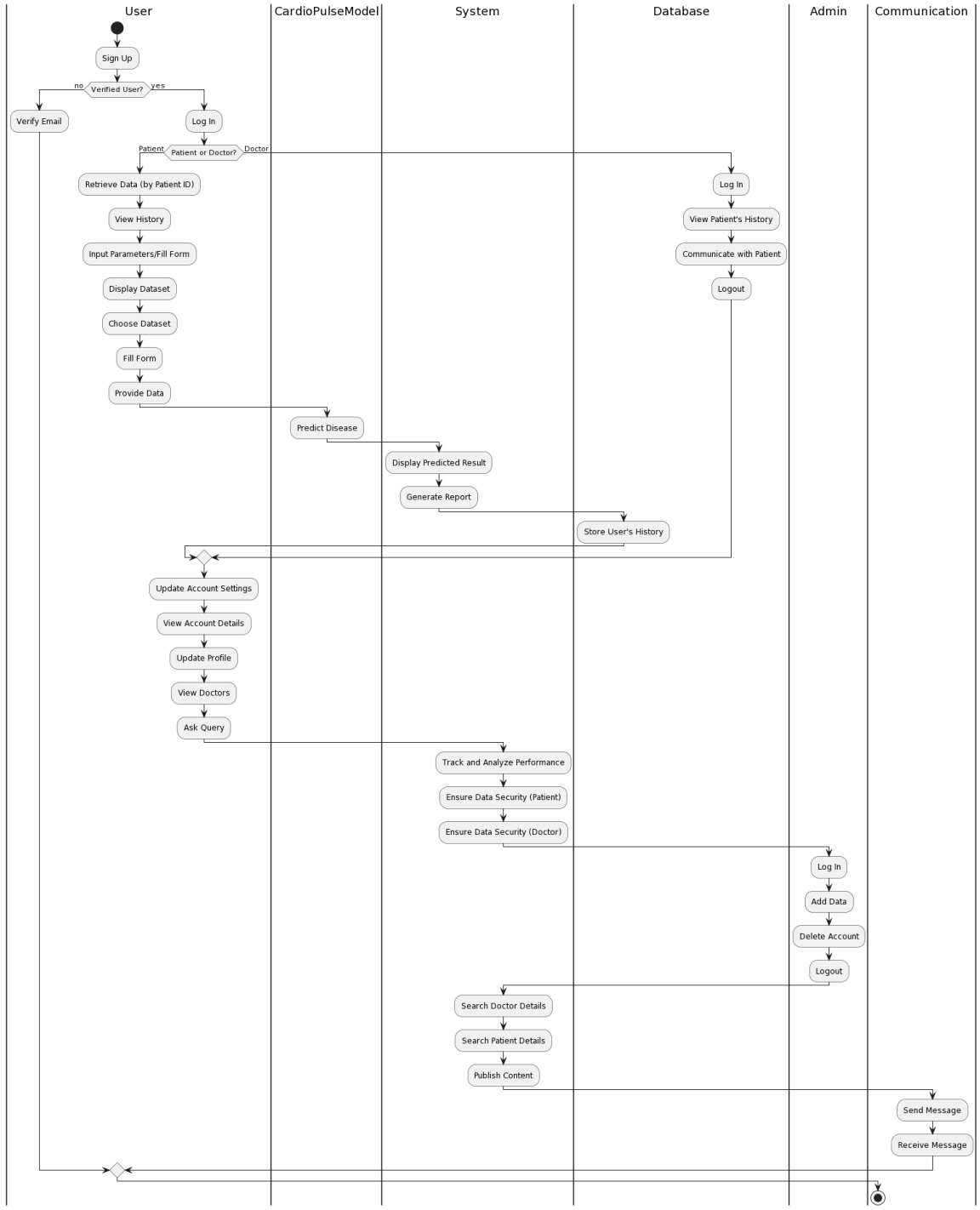

Figure 3.9: Activity Diagram

Chapter 4:

User Interface Design

4.1. Introduction

User interface for cardiovascular disease prediction and treatment recommendation system has been designed to facilitate the user to interact with the system with minimal effort. The user interface consists of the modules required for the proper functionality of the proposed system. Following is the site's appearance and scheme to navigate through the system with ease.

4.2. Site Maps

Site map of the system provides an overview of the information architecture in a hierarchy. The following site map illustrates the proposed system starting from main page and navigating the pages in structure. New page shows that a new page will open related to the previous link. Risk calculator will open into new form which will take input from the user and then predict the result in future page. Future page depends upon the input of the previous page or link.

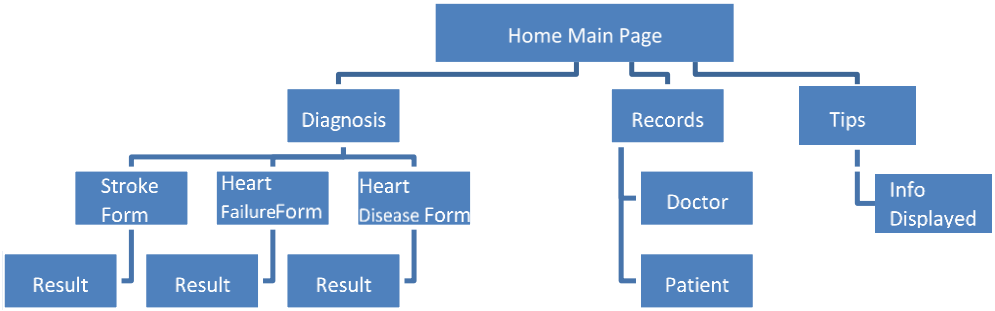

Figure 4.1: Site Map

4.3. Story boards

The story board for cardiovascular system depicts the verbal and visual interaction of the user with the system. The story boards are designed to give an idea of the structure of the system, the alignment of the widgets and text placement in the mobile app. The terms button, links, textbox, and label have been shown. Each page has its own set of buttons, text fields, and labels. The labels represent the text that will be displayed. Links are used to display how the button will work to interact with the other pages within the system. The home represent the main view fo the dashboard the mobile app is offering to the user and each text button icon will take the user to its particular page. The placement of the icons, app bar, bottom navigation bar is also defined in the story board. The mobile app must not necessarily exactly like story board. The story boards of all the mobile pages are given in this section. The story board for home dashboard of the system, login and form page is given below.

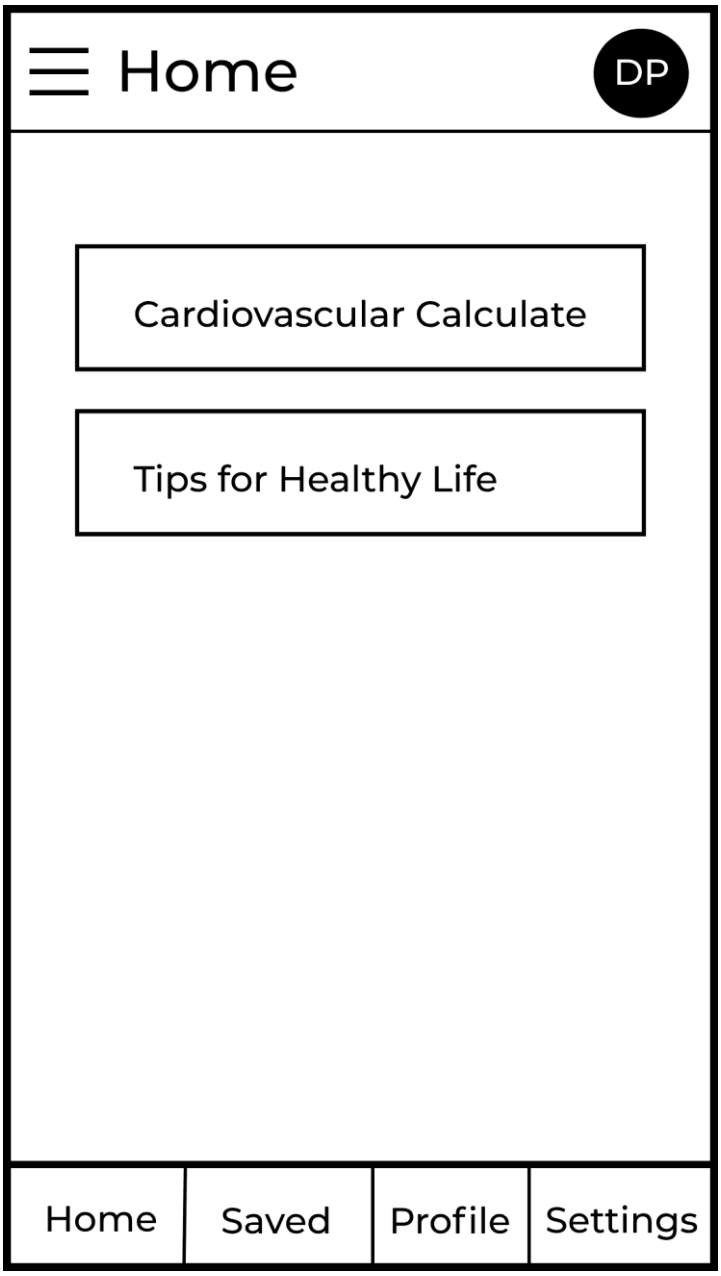

Figure 4.2: Story Board

The story board for Login user interface is given below. It defines all the text boxes, links, images, labels and buttons required to be defined at a particular position in the user interface of the log in page. User can login thorugh email and also by using the google or facebook sign in/up options. An options to signup is also given at the end of the screen page. The consistency of the design is to be achieved in the mobile app design; therefore, it is also included in the story board. The login story board is given in the below figure.

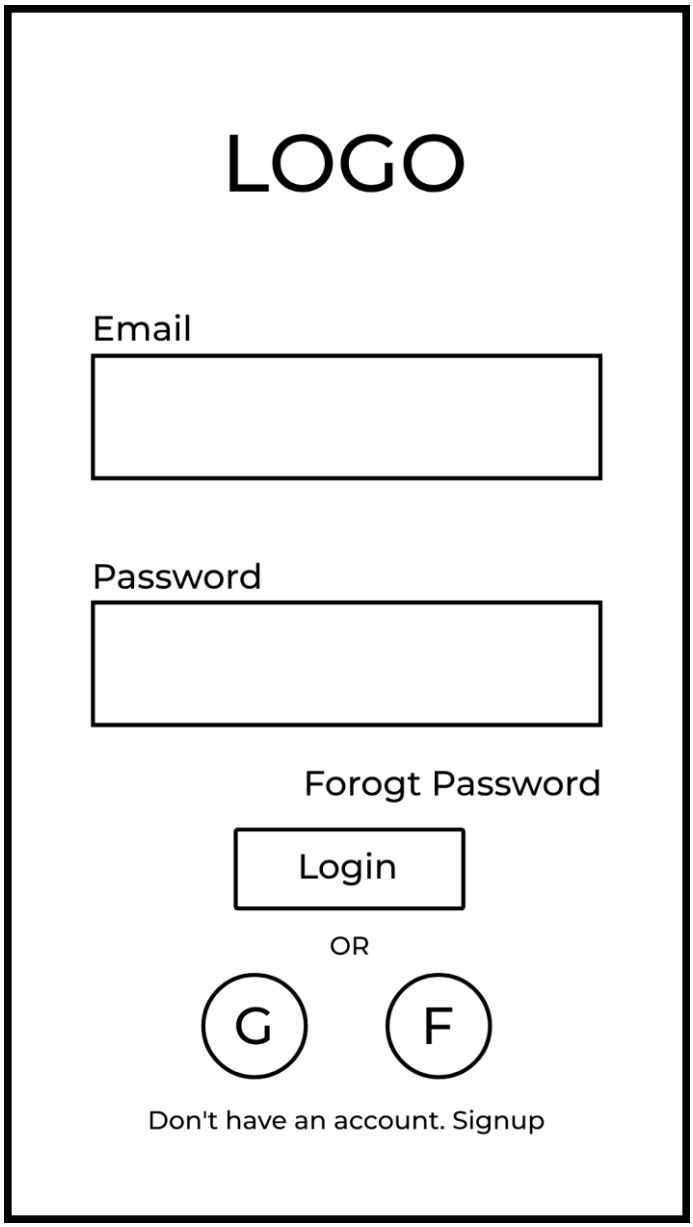

Figure 4.3: Login UI Storyboard

After successful login of the user, home page of the mobile app will be available with active account of the user. The home page of the mobile app has been designed in the following story board. The user will be able to input symptoms to get the prediction of the disease. The risk calculating page having the symptom form is designed below. The form will have various parameters according to the selected type of from ranging from Heart disease, Stroke, heart failure from which have different sets of parametric values. This page will have the same layout as well. The story board of the risk calculator is given in the below figure.

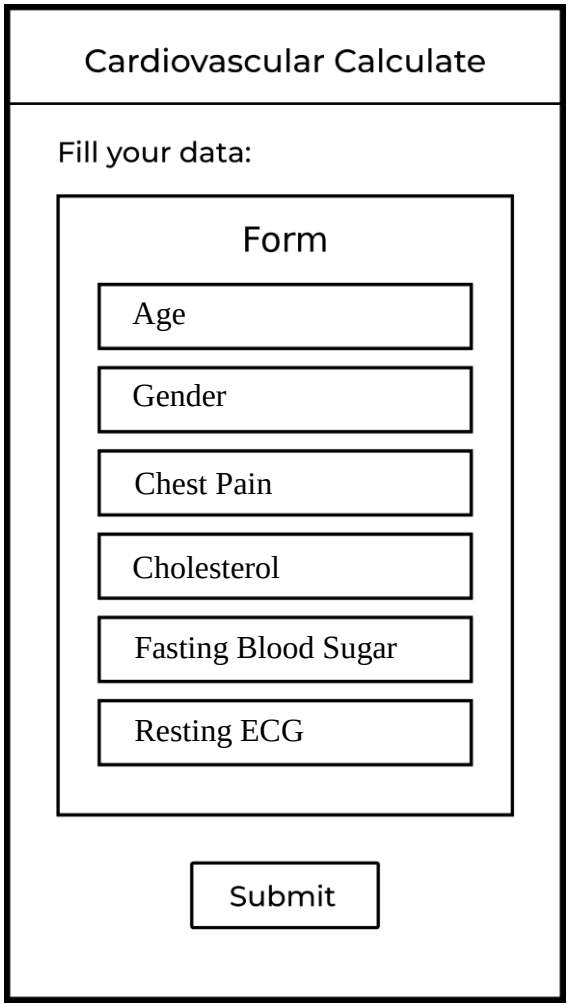

Figure 4.4: Risk Calculator storyboard

After the completion of input of patient's symptoms, the result will be generated. The result page has been designed on a new page so that it will show the complete report of the user's input values or parameters along with the predicted statement and recommended treatment, if any. The result user interface has the same layout with the predicted result at the center of the page. Then the results are displayed in the new page designed for that results activity.

4.4. Prototype

A prototype is an early sample, model, or release of a product built to test a concept or process. It is a term used in a variety of contexts, including semantics, design, electronics, and software programming. System analysts and users generally use a prototype to evaluate a new design to enhance precision. Prototyping serves to provide specifications for a real, working system rather than a theoretical one. In some design workflow models, creating a prototype (a process sometimes called materialization) is the step between the formalization and the evaluation of an idea.

A prototype can also mean a typical example of something such as in the use of the derivation 'prototypical'. This is a useful term in identifying objects, behaviors and

concepts, which are considered the accepted norm.

4.5. Product Interfaces

The system has been designed according to the user requirements. The user interface design of the cardiovascular disease prediction and treatment recommendation system is given below.

After starting the mobile app, a splash screen appears. The splash screen of the app is design below.

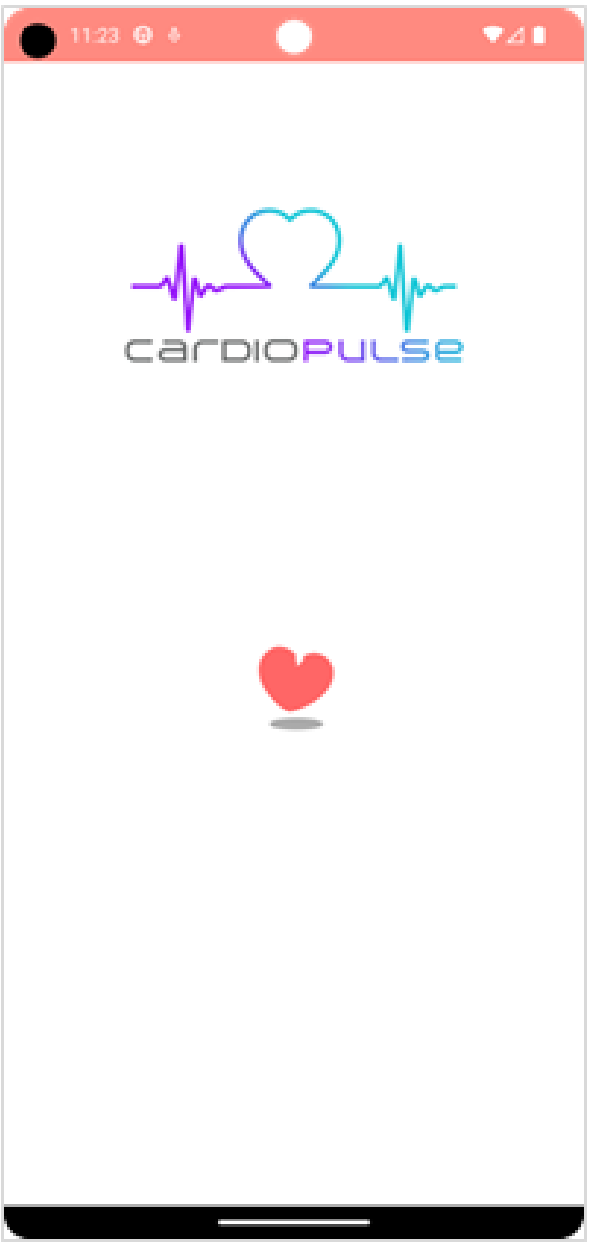

Figure 4.5: Splash Screen

After splash screen, the login screen appears. That contain the filed for the email and password, or you can also login through the google sign up. The login screen design of the app, displayed below.

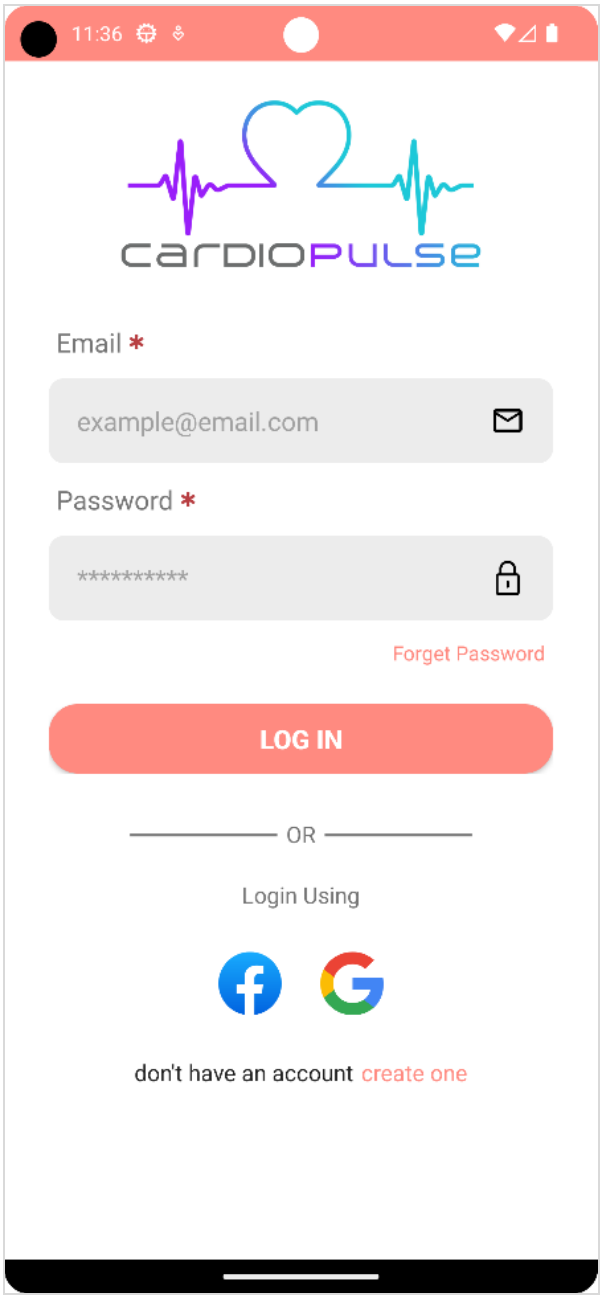

Figure 4.6: Login Page

The new user can register themselves here by filling the this registration form**.** The signup screen design of the app, displayed below.

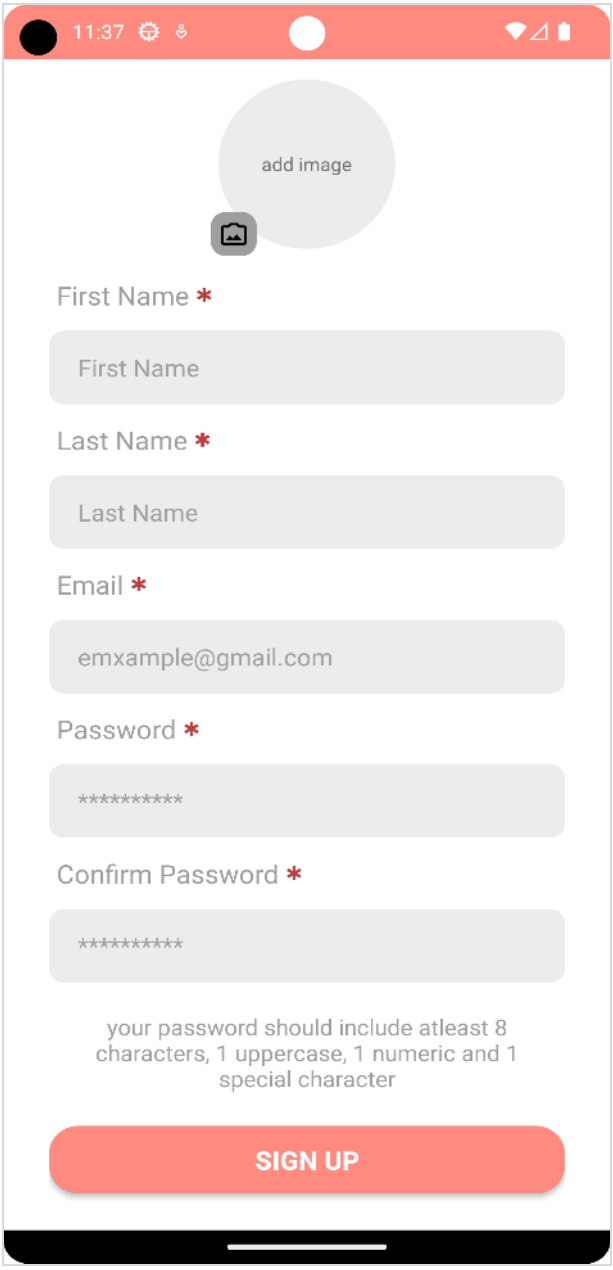

Figure 4.7: Register Page

The user can also go for forget password incase the user don’t remember the password. Below is the forgot password screen.

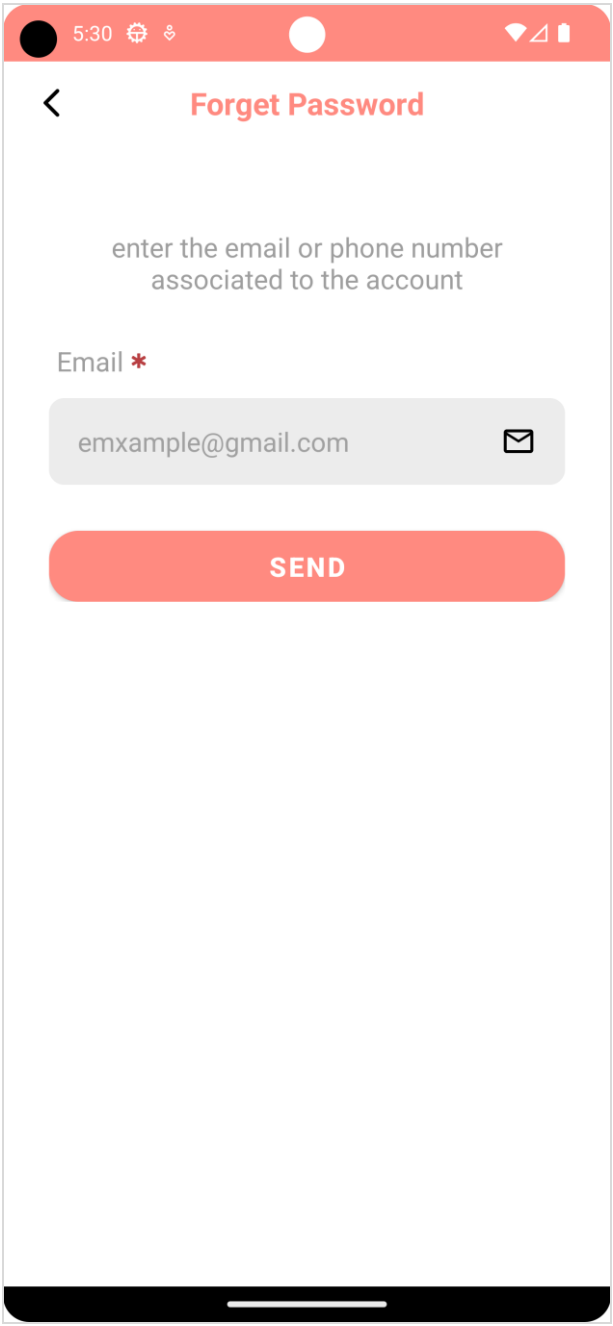

Figure 4.8: Forget Password

The main dashboard home screen of the app is displayed below. It contain the main components of the risk assessment and the tips.

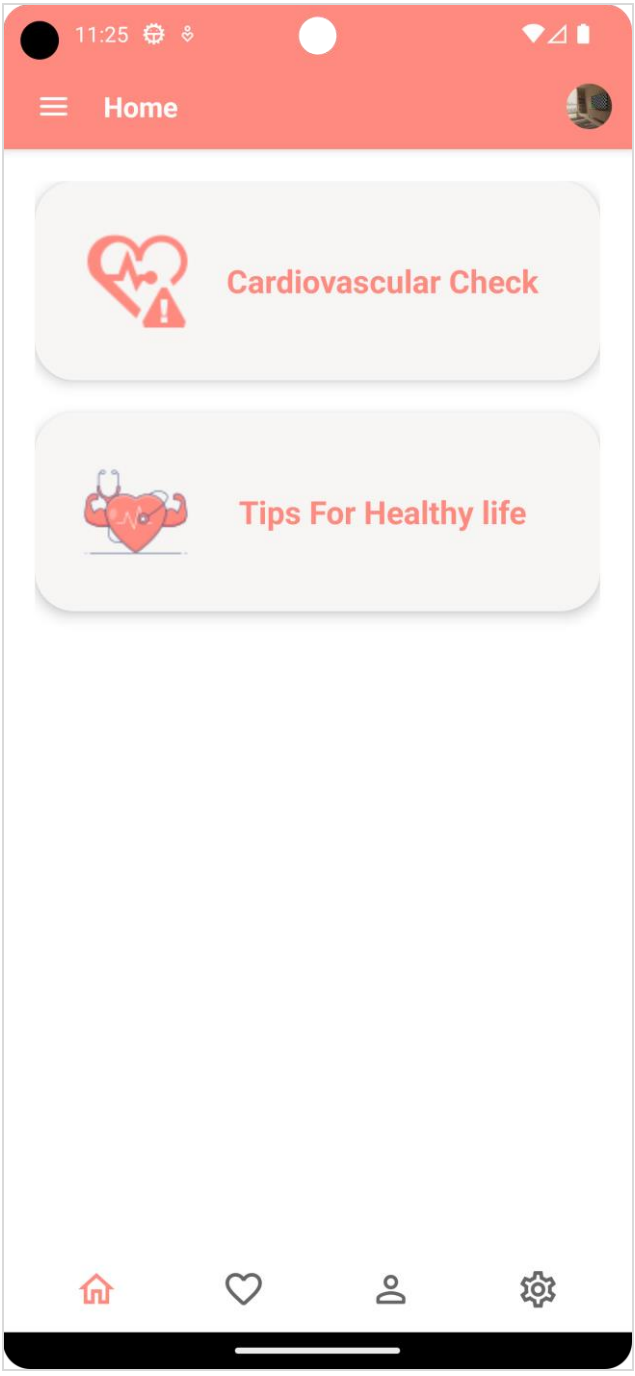

Figure 4.9: Home Screen

Following is the form that the user will fill to calculate the CHD identification. The cardiovascular data form is designed below.

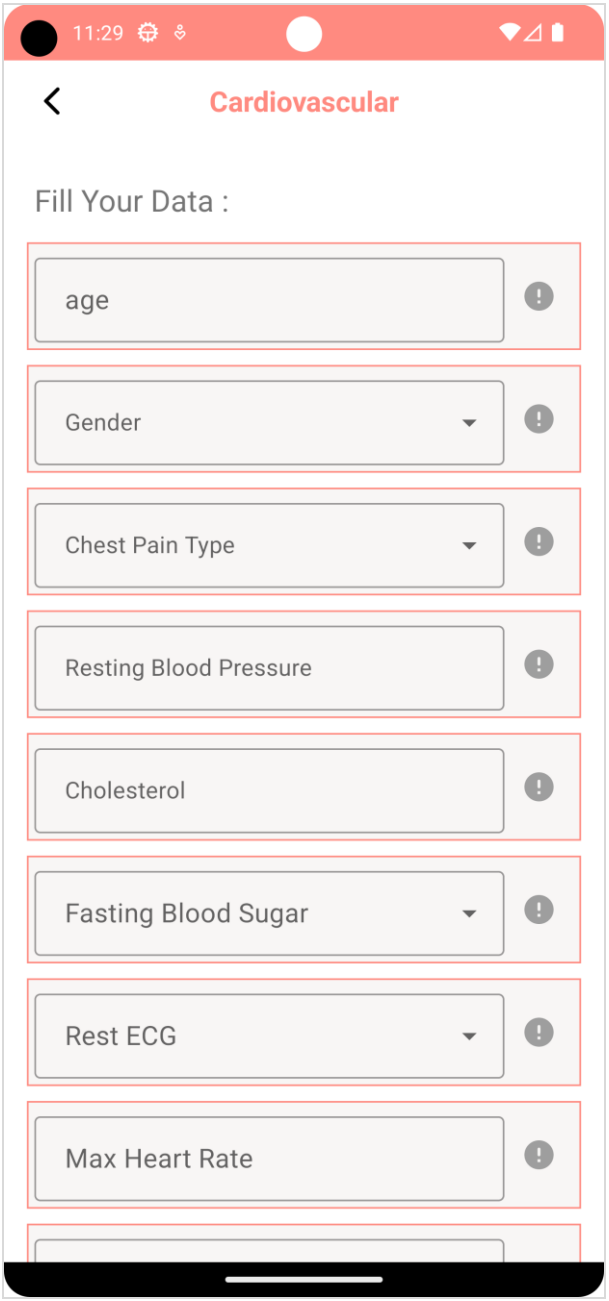

Figure 4.10: Data Form Screen

The tips for healthy life is shown in the following screen displayed below. It contains the tips to improve your heart condition and health.

Figure 4.11: Tips for Healthy Life

The profile screen contains the user details like name, email, picture etc that he/she can change later on as well. Except the email. The designed profile screen, displayed below.

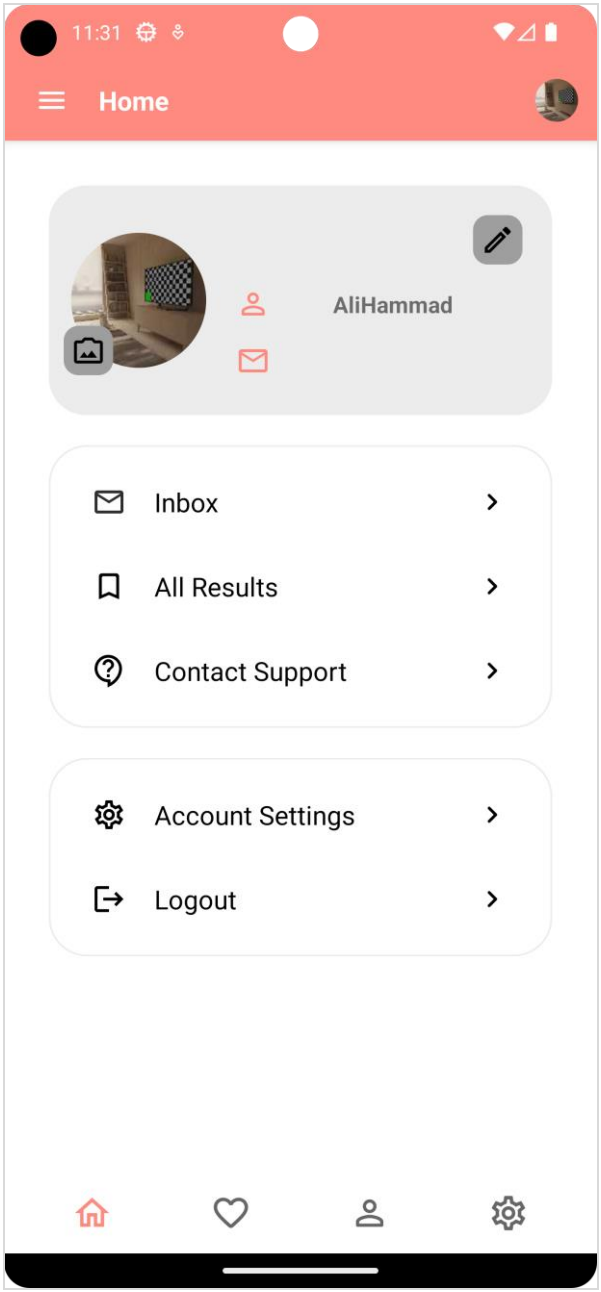

Figure 4.12: User Profile

4.6. Navigational maps:

Navigational map shows what an action will result in for user interacting with the system. It shows the relation of a button or link to other pages. Following is the navigational map of the proposed system.

The user first have to signup so when he click on signup then the user is redirected to the signup screen where he will fill form to signup.

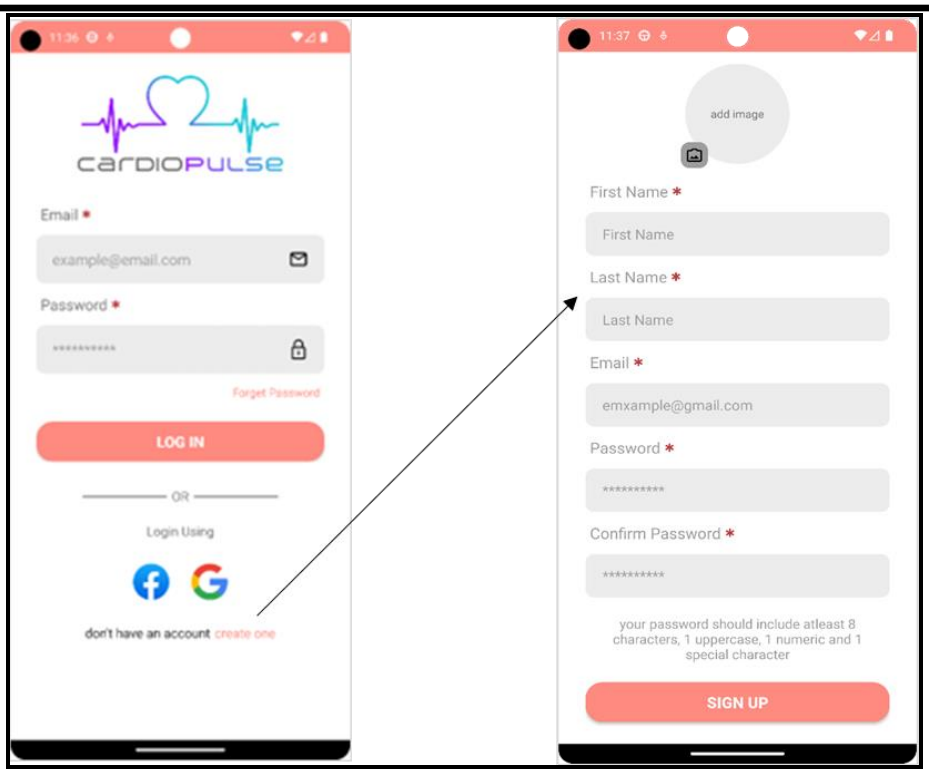

Figure 4.13: Register

Then the main dashboard home page screen will be shown to the user after successful signup. After he click the cardiovascular calculate button, the cardiovascular data form screen will be opened. Where he just have to enter his/her details.

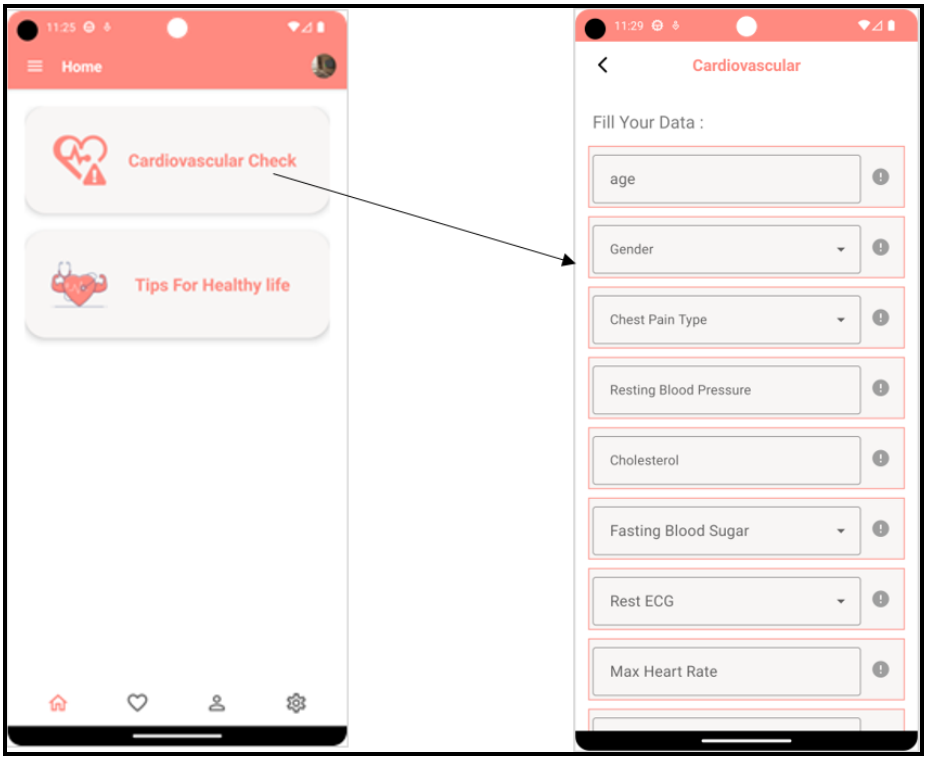

Figure 4.14: Disease Detection

After filling the form there will be a button to submit the info and then the results will be shown to the user in a new screen. User can save the results that will be shown to the user in the favourite/saved section.

Figure 4.15: Saving Results

The user can view the tips for the healthy life where there will be tips to improve the heart condition and healthy life.

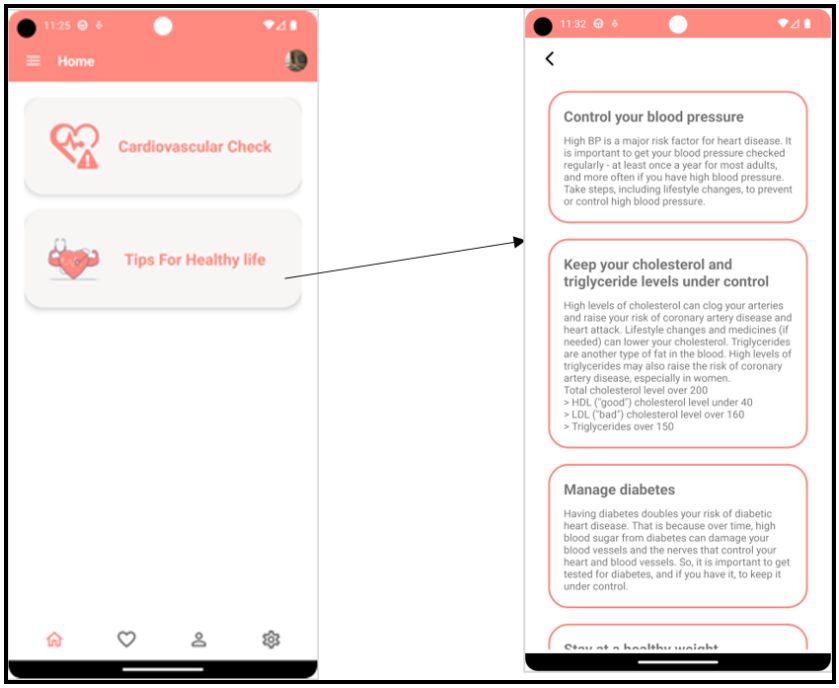

Figure 4.16: Tips/Recommendations

The user clicks on the profile tab then he is redirected the profile screen. Where the user can view and edit his details.

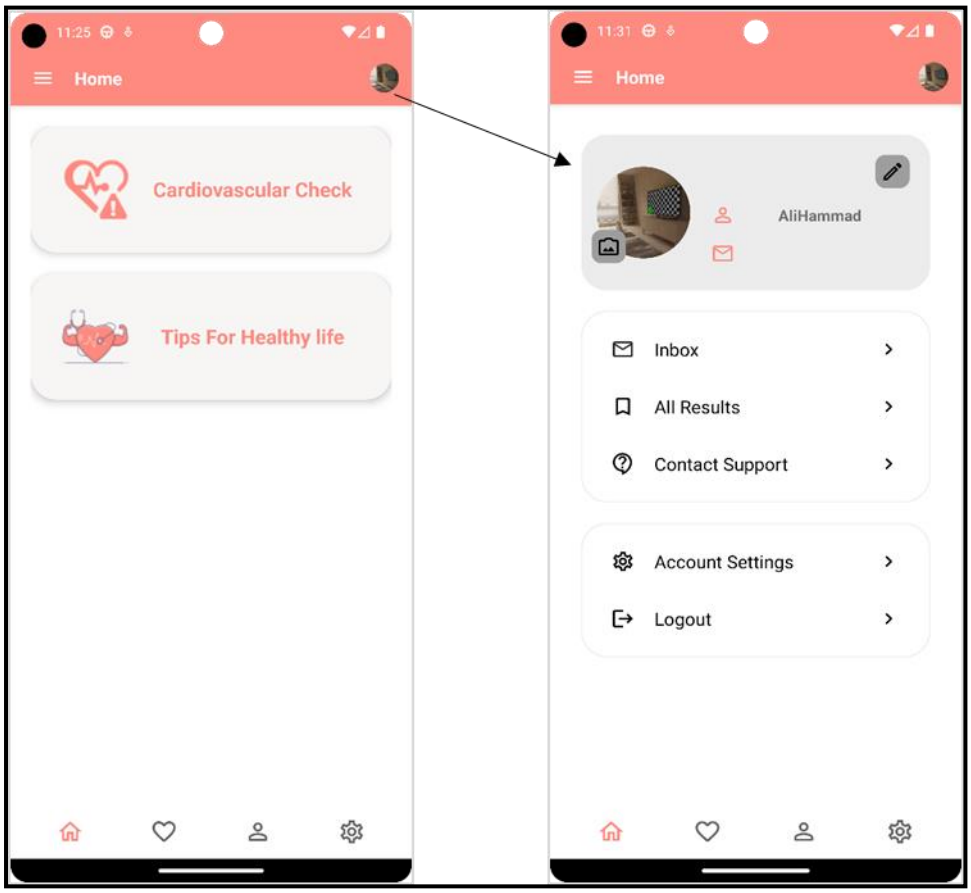

Figure 4.17: User Profile

4.7. Trace-ability Matrix

Traceability matrix of the user interface of the proposed system defines the features to be built in accordance with the requirements defined in the requirement specification document. Each use case has been given a rank of priority and referenced feature. Traceability matrix for UI design is given below.

Table 4.1: Traceability matrix for UI design

N#	Feature built	DB Table	UI ID	Use Case ID	Use Case Name	Rank
1.0	Sign Up	Registration	1	UC_1	UC_SignUp	Medium
2.0	Login	Login	2	UC_2	UC_Login	Highest
3.0	Logout	User Account	3	UC_3	UC_Logout	Lowest
4.0	View	Record	4	UC_4	UC_ViewHistory	Highest
5.0	Delete	Delete	5	UC_5	UC_DeleteAccount	Lowest
6.0	Upload	Upload.	6	UC_6	UC_UploadData	Lowest
7.0	Manage Account	Manage Account	7	UC_7	UC_ManageAccount	Lowest

8.0	Result	Result, Prediction Analysis	8	UC_10	UC_GenerateResult	Highest
9.0	Report	Result	9	UC_11	UC_GenerateReport	Highest
10.0	Input Parameters	Enter Parameters	10	UC_12	UC_EnterParameters	Highest

Chapter 5:

Software Testing

5.1 Introduction:

Testing is one of the essential deliverables in software life cycle. Software testing is based on a set of criteria on which the system is assessed. In this phase, we have defined dynamic aspects of testing phase with respect to out project. We have subjected our project to the standard software testing criteria. All the testing activities have taken place without being restricted by the complexity, or criticality of the functionality. According to our proposed system, basic testing criteria is not useful for every test phase. Testing methodology and its various aspects have been defined further in the document. Software testing artifacts that are included for cardiovascular disease prediction and treatment recommender system are defined below.

1. Test Plan
2. Test Design Specification
3. Test Case Specification
4. Test Procedure Specification
5. Test Item Transmittal Report
6. Test Log
7. Test Incident Report
8. Test Summary Report

5.2. Black box plan/White box plan/Grey box plan

5.2.1. Black Box Testing

Black box testing is a method of software testing which is used to examine the functionalities of the software system mainly focusing on input and output of the application. It checks the behavior of the system by utilizing software requirements. In view of our proposed application, we have used black box testing to analyze the system behavior with respect to the requirement specifications. We have tested the functionalities of the application without considering the code of the functionalities. After defining our testing method, we will further explore the plan below.

5.2.1.1. Types of Black Box Testing

Types of black box testing implemented for our proposed system are described below.
Functional testing
We have used functional testing to examine the functional requirements of our system.

5.2.1.2. Tools used for Black Box Testing:

We have used various techniques for black box testing of our proposed project which are listed
below:

1. Decision table testing
2. Equivalence partitioning method

We have manually performed these testing techniques using the following.

5.2.2. White Box Testing

White Box testing is a software testing technique which is used to test internal code and design of the application to verify predefined set of input with respect to output. In view of our proposed application, we have used White Box Testing to analyze the internal

design of the application by examining its code. We have applied various techniques to test the application with respect to its code implementation. After defining our testing method, we will further explore the plan below.

5.2.2.1. Types of White Box Testing

Types of White box testing implemented for our proposed system are described below.

5.2.2.2. White Box Testing Tools

We have used various techniques for white box testing of our proposed project which are listed below: 1. Basis path flow.

5.2.3. Grey Box Testing

Grey Box Testing or Gray box testing is a software testing technique to test a software product or application with partial knowledge of internal structure of the application. The purpose of grey box testing is to search and identify the defects due to improper code structure or improper use of applications.
In this process, context-specific errors that are related to mobile systems are commonly identified. It increases the testing coverage by concentrating on all of the layers of any complex system.
Gray Box Testing is a software testing method, which is a combination of both White Box Testing and Black Box Testing method.

- In White Box testing internal structure (code) is known
- In Black Box testing internal structure (code) is unknown
- In Grey Box Testing internal structure (code) is partially known

In Software Engineering, Gray Box Testing gives the ability to test both sides of an application, presentation layer as well as the code part. It is primarily useful in Integration Testing and Penetration Testing.

5.2.3.1. Techniques used for Grey box Testing

- **Matrix Testing:** This testing technique involves defining all the variables that exist in their programs.
- **Regression Testing**: To check whether the change in the previous version has regressed other aspects of the program in the new version. It will be done by testing strategies like retest all, retest risky use cases, retest within a firewall.
- **Orthogonal Array Testing or OAT**: It provides maximum code coverage with minimum test cases.
- **Pattern Testing:** This testing is performed on the historical data of the previous system defects. Unlike black box testing, gray box testing digs within the code and determines why the failure happened

5.3. Test plan

5.3.1. Purpose

The purpose of this test plan is to describe the scope, resources and schedule of the activities. This plan has been designed to identify tasks to be performed for the testing of

features of this project. The test plan has been described with respect to the outline below

5.3.2. Outline

A test plan shall have the following structure:

a. Test plan identifier
b. Introduction
c. Test items
d. Features to be tested
e. Features not to be tested
f. Approach
g. Item pass/fail criteria
h. Suspension criteria and resumption requirements
i. Test deliverables
j. Testing tasks
k. Environmental needs
l. Responsibilities
m. Staffing and training needs
n. Schedule
o. Risks and contingencies
p. Approvals

5.2.2.1. Test plan identifier

The test plan identifier for this project is SE_24.

5.2.2.2. Introduction

The features of the applications to be tested are summarized below.

• Login module
• Admin control panel
• Registration
• Cardiovascular disease symptom Form
• Result generation
• Saving result
• Logout

5.2.2.3. Test items

The items which will be tested are given below with the referenced use case id.

Table 5.1: Test items

Item	**Reference**
Login module	UC_02
Admin control panel	UC_07
Registration	UC_01
Cardiovascular disease symptom Form	UC_13
Result generation	UC_10
Saving result	UC_11

Logout	UC_03

5.2.2.4. Features to be tested

The items which will be tested are given below along with their respective features.

Table 5.2: Features to be tested

Item	Reference	Features
Login module	UC_02	Username, password
Admin control panel	UC_07	Account request approval
Registration	UC_01	Name, Email, password
Cardiovascular disease symptom Form	UC_13	Parameters
Result generation	UC_10	Result page
Saving result	UC_11	Generate report
Logout	UC_03	Logout

5.2.2.5. Features not to be tested

The features or items which will not be included for testing are given below.

Table 5.3: Features not to be tested

Features	Reason
Admin panel	This is handled manually.
Result generation process	This is calculated at the backend.
Alert notifications	This is dealt with in specific conditions.
Error notifications	This is dealt with in specific conditions.
User data	Data privacy of the user.

5.2.2.6. Approach

We will use Black box testing and white box testing for the features of our project. We have opted to adopt manual testing approach for the most part of this testing. Testing will be performed according to the test plan. Major techniques used in the testing are given below. For black box testing, we will use the following techniques.

1. Decision Table Testing
2. Boundary value analysis

These testing techniques will be applied on major features of the application. The fields must be executed at least once to check the value limit in case of an undefined error. For white box testing, we will use the following. 1. Basis path analysis It will be applied on the code of main modules.

5.2.2.7. Item pass/fail criteria

The criteria of each item for test pass or fail depends upon the module being tested. For the majority of the feature of this mobile application if the item meets the expected

output, the feature will be considered to have passed.

5.2.2.8. Suspension criteria and resumption requirements

In case of identification of item that is not associated with the test plan or does not match the test criteria, the testing activity may partially or completely be suspended.

5.2.2.9. Test deliverables

The deliverable documents include the following:

a. Test plan;
b. Test design specifications;
c. Test case specifications;
d. Test procedure specifications;
e. Test item transmittal reports;
f. Test logs;
g. Test incident reports;
h. Test summary reports.

Input and output test data has been identified during test case formation.

5.2.2.10. Testing tasks

Testing tasks required to perform testing on this project are as follows. Identification of test case scenario, dependency of one task on the other, skills in testing techniques applied.

5.2.2.11. Environmental needs

The environmental needs for testing the proposed application are a computer system with internet connection.

5.2.2.12. Responsibilities

The project team consists of the following members with the assigned responsibilities.

Ali Aoun:	Model training,
Hammad:	Requirement engineering, Implementation
Muneeba :	Designing, Implementation
Fariha :	Testing

5.2.2.13. Staffing and training needs

Necessary skills required are knowledge on how to enter values in the form and generating result.

5.2.2.14. Schedule

Milestone for schedule of testing is one of the last phases of software development. It is required to be completed after completion of mobile application development within 2 to 3 weeks.

5.2.2.15. Risks and contingencies

Risks that may occur during the phases of plan are delay in server down may cause contingency. Unavailability of the required tool may delay the process. Therefore, access to backup data or alternative tool and additional internet connection may be required to meet the scheduled delivery.

5.2.2.16. Approvals

The approval form is given below.

Table 5.3: Approval

Project Title: Cardiovascular Disease Prediction		
Project Plan	SE_22	SECTION A
Identification of change	No potential change has occurred.	NONE
Approval		
Title	Name	Signature
Project Manager	Mr. Attique Ur Rehman	

5.4. Test design specification

5.4.1. Purpose

Test cases are design on the basis of test plan. The design of the test case is given below. The proposed mobile application has following main modules:

1. Login
2. Sign Up
3. Prediction form
4. Result/Prediction
5. Logout
6. Save

Each module will have different test cases with various test scenarios.

Table 5.4: Test Design specification

Test case Design	**Test case ID**	**Designed by**	**Executed by**
TC_01	Log in	Ali Aoun	Muneeba
TC_02	Check Result	Hammad	Ali Aoun
TC_03	CVD Form	Muneeba	Hammad
TC_04	Log out	Ali Aoun	Muneeba
TC_05	Registration	Hammad	Ali Aoun
TC_06	Saving data	Fariha	Hammad
TC_07	Wrong Credentials for registration	Ali Aoun	Fariha
TC_08	Wrong credentials for log in	Hammad	Ali Aoun

TC_09	Missing required fields	Fariha	Hammad
TC_10	Invalid form data	Ali Aoun	Fariha
TC_001	Register Doctor	Hammad	Ali Aoun
TC_002	Change password	Fariha	Hammad

.

5.5. Test Case Specification

5.5.1. Purpose

The purpose of test case specification is to define test cases on the basis of test designs. The proposed application has following main modules:

1. Login
2. Sign Up
3. Prediction form
4. Result/Prediction
5. Logout
6. Save

Testing techniques have been applied to the mobile application modules which are given below

5.5.2. Outline

The outline for testing specifications is described in detail below.

5.5.2.1. Test Cases

The test cases designed above will be specified in this section.
Test case for login feature has been described below.

Table 5.5: Test Case for Login Page

Test Case ID	TC_01
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	High
Test design date	10-03-2024
Module Name	Login
Test executed by	Muneeba Hussain
Test Title	Login Validation
Test execution date	15-03-2024

Description	To validate successful user login.
Pre-Conditions	User must be registered.
Dependencies	N.A.
Steps	1. Launch Application 2. Enter valid Username and valid Password 3. Click Login
Test data	Username =”ABC”, Password = 123
Expected Outcome	Login Successfully.
Actual Outcome	Login Successfully
Status	Pass.

Test case for generating result is designed below.

Table 5.6: Test Case for Result

Test Case ID	TC_02
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	Medium
Test design date	10-03-2024
Module Name	Check Result
Test executed by	Hammad
Test Title	Check Result
Test execution date	13-03-2024
Description	To validate result generation with valid parameters.
Pre-Conditions	Form must be filled and data in the fields must be valid
Dependencies	N.A.

Steps	1. Visit the mobile app. 2. Enter valid Parameters in the form 3. Click Check
Test data	Input valid parameters.
Expected Outcome	Result generated
Actual Outcome	Result generated
Status	Pass.

Test case for the cardiovascular disease form has been described below.

Table 5.7: Test Case for CVD form

Test Case ID	TC_03
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	Medium
Test design date	10-03-2024
Module Name	CVD Form
Test executed by	Hammad
Test Title	CVD Form
Test execution date	13-03-2024
Description	To check if valid input allows form filling without an error.
Pre-Conditions	None.
Dependencies	N.A.
Steps	1. Visit the mobile app.
	2. Enter valid parameters in the form.

Test data	Input valid parameters.
Expected Outcome	Form is filled Successfully
Actual Outcome	Form is filled Successfully
Status	Pass.

Test case for logout features has been designed below.

Table 5.8: Test Case for Logout

Test Case ID	TC_04
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	Medium
Test design date	10-03-2024
Module Name	Logout
Test executed by	Hammad
Test Title	Logout
Test execution date	13-03-2024
Description	To check successful logout.
Pre-Conditions	User must be logged in.
Dependencies	N.A.
Steps	1. Visit the mobile app. 2. Enter valid Parameters in the form.
Test data	Input valid parameters.
Expected Outcome	Logout Successfully
Actual Outcome	Logout Successfully
Status	Pass.

Test case for the registration feature is given as follows.

Table 5.9: Test Case for Registration

Test Case ID	TC_05
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	Medium
Test design date	10-03-2024
Module Name	Registration
Test executed by	Hammad
Test Title	Registration
Test execution date	13-03-2024
Description	To check successful user registration.
Pre-Conditions	None.
Dependencies	N.A.
Steps	1. Visit the mobile app. 2. Enter First name. 3. Enter last name. 4. Enter email 5. Enter password. 6. Confirm password.
Test data	Input valid parameters.
Expected Outcome	Successfully registered
Actual Outcome	Successfully registered.
Status	Pass.

Test case for generating report has been designed as follows.

Table 5.10: Test Case for Saving data

Test Case ID	TC_06
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	Medium
Test design date	10-03-2024
Module Name	Save data
Test executed by	Hammad
Test Title	Saving data
Test execution date	13-03-2024
Description	To check if data is saved successfully.
Pre-Conditions	User must fill the form with valid parameters.
Dependencies	N.A.
Steps	1. Visit the mobile app. 2. Enter valid parameters 3. Click save
Test data	Input valid parameters and click “Save”.
Expected Outcome	Saved successfully.
Actual Outcome	Saved successfully.
Status	Pass.

Test case for authenticating credentials is designed below. This test case describes the criteria for invalid credentials for registration feature.

Table 5.11: Test Case for Wrong Registration credentials

Test Case ID	TC_07
Test Design by	Ali Aoun

Test priority (Low/Medium/High)	Medium
Test design date	10-03-2024
Module Name	Registration
Test executed by	Hammad
Test Title	Wrong Registration credentials
Test execution date	13-03-2024
Description	To check the response of mobile app with wrong registration credentials.
Pre-Conditions	None.
Dependencies	N.A.
Steps	1. Visit the mobile app. 2. Enter First name. 3. Enter last name. 4. Enter email 5. Enter password. 6. Confirm password.
Test data	First name= “Ali Aoun” Last name= “” Email= “233@uskt.edu.pk” Password= “1234AVc!” Confirm password= “Abc123&”
Expected Outcome	Error message: Email or password is wrong.
Actual Outcome	Error message: Email or password is wrong.
Status	Pass.

Test case for authenticating login is designed below. This test case specifies the testing criteria for the invalid input of login credentials.

Table 5.12: Test Case for Wrong Login credentials

Test Case ID	TC_08
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	Medium
Test design date	10-03-2024
Module Name	Login
Test executed by	Hammad
Test Title	Wrong Login credentials
Test execution date	13-03-2024
Description	To check the response of mobile app with wrong login credentials.
Pre-Conditions	None.
Dependencies	N.A.
Steps	1. Visit the mobile app. 2. Enter Email 3. Enter password.
Test data	Email= “233@uskt.edu.pk” Password= “1234AVc!”
Expected Outcome	Error message: Email or password is wrong.
Actual Outcome	Error message: Email or password is wrong.
Status	Pass.

Test case for the registration form is designed below. This test case specifies the criteria of missing fields for the sign up page.

Table 5.13: Test Case for Missing Required fields for Sign Up

Test Case ID	TC_09
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	Medium
Test design date	10-03-2024
Module Name	Registration
Test executed by	Hammad
Test Title	Missing required fields for Sign up.
Test execution date	13-03-2024
Description	To check the response of mobile app with missing require fields
Pre-Conditions	None.
Dependencies	N.A.
Steps	1. Visit the mobile app. 2. Enter First name. 3. Enter last name. 4. Enter email 5. Enter password. Confirm password. 6. Click Register
Test data	First name= “Muneeba” Last name= “Javed” Email= 058@uskt.edu.pk Password = “ ” Confirm Password = “123ABc#”
Expected Outcome	Error message: Please enter the required field.

Actual Outcome	Error message: Please enter the Required field
Status	Pass.

Test case for the cardiovascular disease symptoms form is designed below. This test case specifies the criteria of entering invalid fields for the form.

Table 5.14: Test Case for invalid form input

Test Case ID	TC_10
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	Medium
Test design date	10-03-2024
Module Name	CVD form
Test executed by	Hammad
Test Title	Invalid form input
Test execution date	13-03-2024
Description	To check the response of form with invalid input parameters.
Pre-Conditions	None.
Dependencies	N.A.
Steps	1. Visit the mobile app. 2. Fill the form
Test data	Invalid parameters.
Expected Outcome	Error message: Please enter the correct value.
Actual Outcome	Error message: Please enter the correct value.

Status	Pass.

Basis path Testing Code: Register Patient(User)

```
public ActionResult DoctorPost( doctor d,HttpPostedFileBase Image)
      {
         bool check=false;

         ApplicationDbContext db = new ApplicationDbContext();
         var dbpatient = db.patients.ToList();
         foreach (var item in dbpatient)
         {
            if (d.Email == item.Email)
            {
               TempData["exist"] = "Email already
exist";                  check = true;
            }
         }

         if (check == true)
         {
            return View("Index");

}
el
se
{

            string filename = Path.GetFileName(Image.FileName);
string _filename = DateTime.Now.ToString("yymmssfff") +
filename;                                                  string path =
Path.Combine(Server.MapPath("~/Images"), _filename);
            Image.SaveAs(path);
            d.Image = "~/Images/" + _filename;
            db.doctors.Add(d);
db.SaveChanges();

            var mail = new MailMessage();
            mail.To.Clear();
            mail.To.Add("CardioUSKT@gmail.com");
mail.From                    =                    new
MailAddress("CardioUSKT@gmail.com");
            mail.Subject = "Registeration";
            mail.Body = d.FirstName + "\twants to register itself as a Patient." + "\n from (:)=>" + d.Email;
mail.IsBodyHtml = true;
            SmtpClient smtp = new
SmtpClient();                smtp.Host =
"smtp.gmail.com";                smtp.Port
= 587;
            smtp.UseDefaultCredentials = false;                smtp.Credentials = new
System.Net.NetworkCredential("CardioUSKT@gmail.com", "tufhajai");

            smtp.EnableSsl = true;
            smtp.Send(mail);
```

```
        return View("patientregister");
    }
```

The test case for the above code is designed below. It specifies the criteria of the doctor registration form.

Table 5.15: Test Case for Doctor registration

Test Case ID	TC_001
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	High
Test design date	10-03-2024
Module Name	Registration
Test executed by	Muneeba Javed
Test Title	Doctor Registration
Test execution date	13-03-2024
Description	To validate registration of doctor.
Approach	White Box testing
Pre-Conditions	None.
Dependencies	N.A.
Steps	1. Enter Credentials
	2. If Email already exists, system will return to main page 3. If email does not exist, the credentials will be added to the system. 4. Admin will receive the request to add user as a doctor. 5. If Admin will approve the request, then user will be registered as doctor.
Test data	Valid credentials that don't already exists.
Expected Outcome	Registered Successfully.
Actual Outcome	Registered Successfully
Status	Pass.

Code 2: Change Password

```
public async Task<ActionResult> ChangePassword(ChangePasswordViewModel model)
    {
        if (!ModelState.IsValid)
        {
            return View(model);
        }
        var result = await UserManager.ChangePasswordAsync(User.Identity.GetUserId(),
model.OldPassword, model.NewPassword);
        if (result.Succeeded)
        {
            var user = await
```

```
UserManager.FindByIdAsync(User.Identity.GetUserId());                if (user
!= null)
            {
               await SignInManager.SignInAsync(user, isPersistent: false, rememberBrowser: false);
            }
            return RedirectToAction("Index", new { Message = ManageMessageId.ChangePasswordSuccess });
          }
          AddErrors(result);
          return View(model);
        }
```

The following test case has been designed to validate the criteria of password change.

Table 5.16: Test Case for Change password

Test Case ID	TC_002
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	High
Test design date	10-03-2024
Module Name	Change password
Test executed by	Muneeba Javed
Test Title	Change password
Test execution date	13-03-2024
Description	To validate changing password of registered user
Pre-Conditions	User must be registered.
Dependencies	N.A.
Steps	1. If user is not in valid state, return to main page. 2. Else enter old password and new password 3. If old password is valid, password is changes successfully. 4. Otherwise, if invalid credentials are found then user is redirected to the main page.
Test data	Input valid old password.
Expected Outcome	Password changed successfully.
Actual Outcome	Password changed successfully.
Status	Pass.

Code 3: View Requests

```
public async Task<ActionResult> PatientRegister(int id)
      {
         var    db    =    new    ApplicationDbContext();
var  model  =  db.patient.Find(id);                        if
```

```
(ModelState.IsValid)
        {
            var user = new ApplicationUser { UserName = model.Email, Email = model.Email,
FirstName = model.FirstName, LastName = model.LastName };                    var result = await
UserManager.CreateAsync(user, model.Password);                if (result.Succeeded)
            {
                var rolestore = new RoleStore<IdentityRole>(new ApplicationDbContext());
                var rolemanger = new RoleManager<IdentityRole>(rolestore);                    await
rolemanger.CreateAsync(new   IdentityRole("Patient"));                                        await
UserManager.AddToRoleAsync(user.Id, "Patient");
```

The test case to specify the criteria of forgot password feature has been defined in the table as follows.

Table 5.17: Test Case for View requests

Test Case ID	TC_003
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	High
Test design date	10-03-2024
Module Name	View requests
Test executed by	Muneeba Javed
Test Title	View
Test execution date	13-03-2024
Description	To validate requests user to be registered.
Pre-Conditions	None.
Dependencies	N.A.
Steps	1. If there is a request, show it in the request page. 2. Else if user has been registered, remove request.
Test data	Login credentials and verification code.
Expected Outcome	Requests are visible.
Actual Outcome	Requests are visible.
Status	Pass.

Code 4: Admin approval

```
    [Authorize(Roles              ="Admin")]
public ActionResult doctorlist()
    {
        var doc1 = db.doctors.ToList();
return View(doc1);
    }

    [Authorize(Roles = "Admin")]
public ActionResult generaluserlist()
    {
```

```
        var user1 = db.Users.ToList();
return View(user1);
    }

}

}
```

Test case for the specification of the phone number confirmation is designed below.

Table 5.18: Test Case for admin approval

Test Case ID	TC_004
Test Design by	Ali Aoun
Test priority (Low/Medium/High)	High
Test design date	10-03-2024
Module Name	Registration
Test executed by	Muneeba Javed
Test Title	Admin approval
Test execution date	13-03-2024
Description	To validate doctor's certificate
Pre-Conditions	None
Dependencies	N.A.
Steps	1. If user is not in valid state, return to main page. 2. Else upload certificate 3. A request will be generated 4. Enter code to verify 5. Otherwise, if invalid credentials are found then user is redirected to the main page.
Test data	Input certificate.
Expected Outcome	Validated successfully.
Actual Outcome	Validated successfully.
Status	Pass.

5.6. Test procedure specification

5.6.1. Purpose

The test cases specified above have been tested using defined software testing procedures. The purpose of the test specification is to specify the procedure performed on the test cases with respect to the techniques applied.

5.6.2. Outline

Test procedure specification has been structured as follows:

5.6.2.1. Test procedure specification identifier

Each procedure has been given a unique specification identifier.
Procedure steps are as follows.

Table 5.19: Login form submission: decision table

Test specification ID	**Spec_Login**			
Inputs	**Case 1**	**Case 2**	**Case 3**	**Case 4**
Username	F	T	F	T
Password	F	F	T	T
Output (E/D)	E	E	E	D

Legend:
T- Correct input value
F- Wrong input value
E- Error message
D- Home page is displayed.

Decision Table Interpretation:
Case 1: Username and Password both are incorrect. Therefore, an error message is displayed.
Case 2: Username is correct and password is wrong. Therefore, an error message is displayed.
Case 3: Username is wrong and password is correct. Therefore, an error message is displayed. Case 4: Username and password both are correct. Therefore, user has successfully logged in and homepage is displayed.

Table 5.20: Password field: Equivalence partitioning

Test Case	Password field accepts minimum 6 characters.	
Equivalent partition	Input values in 0-5 and 6-10 partition are equivalent.	
Test Specification ID	Spec_password	
Scenario	1	2
Description	Enter 0 to 5 characters in password field	Enter 6 to 10 characters in password field
Expected Output	System should not accept	System should accept

5.6.3. Procedure details

The procedural test details are in the following table.

Table 5.21: Procedural detail

Sr.	Test case identifiers	Test Revisions	Test features	Test status
1	TC_01	1st	Successful Log in	Pass
2	TC_02	1st	Successful Check Result	Pass
3	TC_03	1st	Successful input CVD Form	Pass
4	TC_04	1st	Successful Log out	Pass
5	TC_05	1st	Successful Registration	Pass
6	TC_06	1st	Successful Saving data	Pass
7	TC_07	1st	Wrong Credentials for registration	Pass
8	TC_08	1st	Wrong credentials for log in	Pass
9	TC_09	1st	Missing required fields	Pass
10	TC_10	1st	Invalid form data	Pass
11	TC_001	1st	Successful Register Doctor	Pass
12	TC_002	1st	Successful Change password	Pass

5.7. Test item transmittal report

5.7.1. Purpose

In this report, we are presenting test items that are being transmitted for software testing. It enlists the responsible person, location, status and variations from requirements and design.

5.7.2. Outline

Following items are being transmitted to the software testing.

Table 5.22: Transmittal of CVD form

Transmittal report identifier	SE_transmitForm
Transmitted items	Reference: SE_22 Responsible members: Ali Aoun, Hammad, Muneeba,Fariha
Location	Test Plan: SE_trans
Status	Log in user checks the input parameters from the form.

Approvals	Approved By: Ali Aoun , Hammad, Muneeba,Fariha

The details for the transmittal of item account is given as follows.

Table 5.23: Transmittal of Account

Transmittal report identifier	SE_transmitAccount
Transmitted items	Reference: SE_22 Responsible members: Ali Aoun , Hammad, Muneeba,Fariha
Location	Test Plan: SE_transmitAccount
Status	Account requests as viewed by admin
Approvals	Approved By: Ali Aoun , Hammad, Muneeba,Fariha

5.8. Test log

5.8.1. Purpose

The purpose of documenting test log is to enlist the performed tests and its detail.

Table 5.24: Test log

Test log ID	Description	Event Entries	Activity
TC_01	Log in	Email, Password	Click Login
TC_02	Check Result	Form parameters	Click Check
TC_03	CVD Form	Form parameters	-
TC_04	Log out	-	Click Log out
TC_05	Registration	Name, Email, Password	Click Register
TC_06	Saving data	Form parameters	Click Save
TC_07	Wrong Credentials for registration	Name, Email, Password	Click Register
TC_08	Wrong credentials for log in	Email, Password	Click Login
TC_09	Missing required fields	Input fields	Input fields
TC_10	Invalid form data	Form parameters	Click Check
TC_001	Register Doctor	Name, Email, password, image file	Click Register
TC_002	Change password	Old password, new password, user ID	Click Change password

5.9. Test incident report

5.9.1. Purpose

Incidents occurring throughout the phase of testing are described below.

5.9.2. Outline

The incident report of the login item is given as follows.

Table 5.25: Test incident Login report

Test incident report identifier	SE_INC
Summary	Test Items: Username, Password Test case ID: TC_01
Incident description	a. Inputs: Username, Password b. Expected results: Successful login c. Actual results: Successful login d. Date and time: 7-13-2022 e. Procedure step: Click Log in f. Environment: Mobile app, mobile browser, internet connection g. Attempts to repeat: 2
Impact	Test plan: Risk identification

The test incident report of cardiovascular disease prediction form is given as follows.

Table 5.26: Test incident Form report

Test incident report identifier	SE_Incident
Summary	Test item: Invalid form input Test Case: TC_10
Incident description	a. Inputs: Form parameters b. Expected results: Display Error c. Actual results: Display Error d. Date and time: 6-7-2022 e. Procedure step: enter parameters on CVD form f. Environment: Mobile app, internet connectivity
Impact	Test plan: Risk analysis Test procedure: Validate input step by step

5.10. Test summary report

5.10.1. Purpose

The test summary report of cardiovascular disease prediction and treatment recommender system describes the test cases and testing approaches applied on each feature of the system. The system is validated to track the requirements and avoid any conflict rising due to negligence or misuse.

5.10.2. Outline

Test summary report is given below.

Table 5.27: Testing Summary report

Test summary report identifier .	SE_Summary
Summary	The modules that have been classified into test cases are 1. Login 2. Sign Up 3. Prediction form 4. Result/Prediction 5. Logout 6. Save White box testing techniques and Black box testing techniques have been applied on the test cases designed.
Comprehensive assessment	Almost all the features are working according to the requirements.
Summary of results	Majority of the test cases have achieved the Passed status.
Approvals	Approved By: Project Manager

Chapter 6:

Conclusion and Future Work

In conclusion, CardioPulse represents a significant advancement in the realm of cardiovascular health management. By harnessing the power of machine learning and data analytics, CardioPulse offers users personalized disease assessments and actionable insights, empowering them to take control of their cardiovascular health proactively. Throughout the development and implementation phases, CardioPulse has demonstrated its potential to revolutionize the way individuals monitor and manage heart health. The successful deployment of CardioPulse underscores its effectiveness in addressing the global health concern of heart diseases and highlights its role as a valuable tool in early risk detection and prevention. Looking ahead, there are several avenues for further enhancing CardioPulse and expanding its impact on public health. One potential area for future work is the integration of additional health metrics and data sources to enhance the accuracy and comprehensiveness of disease predictions. Incorporating real-time physiological data from wearable devices and leveraging advancements in remote monitoring technology could further refine CardioPulse's predictive capabilities. Additionally, ongoing research and development efforts could focus on refining the machine learning algorithms powering CardioPulse to adapt to evolving health trends and individual user profiles.

Furthermore, expanding the reach of CardioPulse through strategic partnerships with healthcare providers, insurance companies, and public health agencies could facilitate greater access to the platform and encourage widespread adoption among diverse populations. Collaboration with medical professionals and researchers could also facilitate the validation and refinement of CardioPulse's predictive models, ensuring its efficacy across different demographic groups and healthcare settings. Moreover, ongoing user feedback and usability testing will be essential for continuously improving the user experience and ensuring that CardioPulse remains intuitive, user-friendly, and accessible to individuals of all backgrounds. Incorporating features such as personalized health recommendations, interactive coaching modules, and community support forums could further enhance user engagement and adherence to healthy lifestyle behaviors.

The future of CardioPulse holds immense promise for advancing cardiovascular health management and promoting proactive wellness strategies on a global scale. Through continued innovation, collaboration, and user-centric design, CardioPulse has the potential to make a lasting impact on public health and contribute to a healthier future for individuals worldwide.

REFERENCES

[1] T. Sharma, S. Verma, and Kavita, ‘‘Prediction of heart disease using cleveland dataset: A machine learning approach,’’ Int. J. Recent Res. Aspects, vol. 4, no. 3, pp. 17–21, 2017.

[2] P. S. Kohli and S. Arora, ‘‘Application of machine learning in disease prediction,’’ in Proc. 4th Int. Conf. Comput. Commun. Autom. (ICCCA), Dec. 2018, pp. 1–4.

[3] A. H. Gonsalves, F. Thabtah, R. M. A. Mohammad, and G. Singh, ‘‘Prediction of coronary heart disease using machine learning: An experimental analysis,’’ in Proc. 3rd Int. Conf. Deep Learn. Technol., 2019, pp. 51–56.

[4] World Health Organization. (2021, June 11). “Cardiovascular Diseases (CVDs)”

[5] Zubair, Faiza & Nawaz, Syed & Nawaz, Ammara & Nangyal, Hasnain & Amjad, Naila & Khan, Muhammad. (2018). Prevalence of cardiovascular diseases in Punjab, Pakistan: a crosssectional study. Journal of Public Health. 26. 10.1007/s10389-018-0898-4.

[6] P. E. Rubini, C. A. Subasini, A. V. Katharine, V. Kumaresan, S. G. Kumar, and T. M. Nithya, ‘‘A cardiovascular disease prediction using machine learning algorithms,’’ Ann. Romanian Soc. Cell Biol., vol. 25, no. 2, pp. 904–912, 2021. [Online]. Available

Appendixes:

APPENDIX A: FYP CODE SNIPPETS OR PSEUDOCODE

FYP (Final Year Project) code snippets and pseudo code are concise representations of programming logic or algorithms used in a final year project. Code snippets are short sections of actual code written in a specific programming language, showcasing practical implementation. Pseudo code, on the other hand, is a high-level, human-readable description of the logic or algorithm without strict adherence to any programming language syntax, making it a valuable tool for planning and explaining complex processes. Both serve as educational aids to help understand, develop, and document their project's technical aspects.