

# **Personalized cardiovascular Healthcare system using machine learning and IOT**

24-25J-274

Final Report

T.Y.R DOLAWATTA - IT21101274

Samarasinghe P.D.P - IT21054372

Perera W.N.D.N.D - IT21312458

BSc (Hons) in Information Technology Specializing in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

February 2024

April 2025

**Personalized cardiovascular Healthcare system using  
machine learning and IOT.**

24-25J-274

Final Report

BSc (Hons) in Information Technology Specializing in  
Information Technology

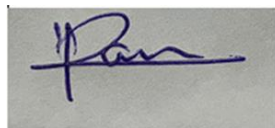


Department of Information Technology

Sri Lanka Institute of  
Information Technology  
Sri Lanka

April 2025

## DECLARATION

I declare that this is my own work, and this dissertation does not incorporate without acknowledgement any material previously submitted for a degree or Diploma in any other University or institute of higher learning, and to the best of my knowledge and belief, it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Name	Student ID	Signature
Dolawatta T.Y. R	IT21101274	
Samarasinghe P.D. P	IT21054372	
Perera W.N.D.N. D	IT21312458	

The proposal report should be certified by the supervisor(s) with the following statement. Under our guidance, the aforementioned candidates are conducting research for their undergraduate dissertation.

Supervisor



Ms. Gaya Thamali Dassanayake

Co-Supervisor

.....

Ms. Dushanthi Sadeepa Kuruppu

## ABSTRACT

cardiovascular diseases (CVDs) remain a leading cause of mortality worldwide, with millions of deaths occurring annually due to heart-related conditions. Major risk factors include age, gender, family history, smoking, poor diet, high blood pressure, diabetes, and lack of physical activity. Traditional diagnostic methods rely on periodic clinical assessments, which may not provide timely interventions for at-risk individuals. To address these limitations, this research proposes a Heart Disease Prevention and Monitoring System that integrates machine learning (ML), real-time health tracking, and IoT-based monitoring to enhance cardiovascular risk assessment and prevention. The system utilizes patient medical history and real-time physiological data collected through IoT health bands to predict heart disease probability. A dataset from the Framingham Heart Study is used to train machine learning models, specifically Logistic Regression, K-Nearest Neighbors (KNN), and Random Forest Classifier, with KNN. Based on the predicted risk level, the system provides personalized dietary and exercise recommendations tailored to individual health conditions. Additionally, real-time monitoring of heart rate, body temperature, and environmental factors such as humidity and temperature ensures continuous assessment and timely alerts for at-risk individuals. By integrating AI-driven prediction, adaptive intervention strategies, and IoT-based health monitoring, this system offers a proactive and personalized approach to cardiovascular disease prevention, ultimately improving early detection, patient outcomes, and quality of life.

**Keywords:** Real-time feedback system, Incorrect exercise form, Computer vision, MediaPipe, OpenCV, Machine learning

## Acknowledgement

First and foremost, I would like to extend my deepest gratitude to Thamali madam, Senior Lecturer at the Sri Lanka Institute of Information Technology, for his unwavering support, guidance, and committed involvement throughout this research. His insightful feedback and encouragement at every stage made this thesis possible.

My sincere thanks also go to my co-supervisor, Ms. Samanthi kuruppu, for her invaluable feedback and direction, which played a crucial role in the successful completion of this work. I am equally grateful to the academic and non-academic staff of the Sri Lanka Institute of Information Technology for their support and assistance at various stages of this project.

A special acknowledgment is due to Dr Rasika de Soysa , Consultant cardiologist, for his expertise and practical insights, which significantly contributed to the development and validation of this study.

I would also like to express my appreciation to the participants who generously devoted their time to support this research.

Finally, I am deeply thankful to my family for their unwavering support and encouragement throughout this journey. I am sincerely grateful to everyone who contributed to this project in any way.

## Table of Contents

<b>Personalized cardiovascular Healthcare system using machine learning and IOT.....</b>	<b>2</b>
DECLARATION .....	i
ABSTRACT .....	ii
Acknowledgement .....	iii
LIST OF FIGURES.....	vi
LIST OF TABLES.....	vii
1.INTRODUCTION .....	1
1.1 Background & Literature Survey .....	1
1.1.1 Background.....	1
1.1.2 Literature Survey .....	1
1.2 Research Gap .....	5
2. OBJECTIVES .....	7
2.1 Main Objectives .....	7
2.2 Specific objectives.....	7
3.METHODOLOGY .....	9
3.1 System Architecture.....	11
3.1.1 OVERALL SYSTEM DIAGRAM .....	12
3.2 Software Solution.....	13
3.2 Domain .....	15
3.2 Technologies.....	15
3.3 PROJECT REQUIREMENTS .....	16
3.3.1 FUNCTIONAL REQUIREMENTS.....	16
3.3.1 Non-Functional Requirements.....	17
3.3.2 System requirements .....	17
3.3.3 User requirements .....	18
3.3.4 Performance Requirements.....	18
3.4 Description of persona sand facilities .....	19
3.5 Commercialization of product .....	19
3.6 Market Demand .....	20
3.7 Commercial Strategy .....	20
4 Implementation and testing .....	22
4.1 Implementation .....	22

4.1.1 Data collection and preparation.....	22
4.1.2 Dataset creation for the prediction model .....	23
4.1.3 Data Preprocessing .....	24
4.1.4 Tools and Technologies of the (Frontend Development).....	25
4.1.4 Model evaluation .....	26
4.4.5 saving the model.....	27
4.4.6 The Prediction function .....	27
4.4.7 Example Prediction .....	28
4.5 Calory function .....	29
4.5.1 Firebase Initialization and Data References .....	29
4.5.2 Data Retrieval and Cleaning .....	30
4.5.3 Dataset Preparation for Prediction .....	31
4.5.4 Inference and Prediction .....	31
4.5.5 Timestamp and Data Update .....	32
4.5.6 Continuous Execution .....	33
4.5.7 Machine Learning Model for Calories Prediction .....	34
4.6 IOT MODEL .....	36
4.6.1 Firebase Initialization and configuration .....	36
4.6.2 Data Collection (Sensor Input).....	37
4.6.3 Heart Rate Sensor Input .....	37
4.6.4 GPS Location Retrieval.....	39
4.6.5 Real-Time Data Display (OLED).....	39
4.6.6 Data Upload to Firebase .....	40
4.6.7 Data Logging and History Storage .....	41
4.7 Design test cases.....	42
5.Results & Discussion.....	44
5.1 CVD Prediction system.....	44
5.1.1 Output and Validation of Exercise Identification Model .....	44
5.1.2 Model Accuracy and Performance Metrics .....	45
5.1.3 Classification Report Analysis.....	45
5.1.4 Confusion Matrix Analysis.....	48
5.2 Calorie prediction system results .....	50
5.2.1 Model Accuracy & Performance Metrics .....	50

5.2.2 Classification-Random Forest .....	51
5.3 IOT System results .....	52
5.3.1 Confusion matrix .....	52
5.3.2 Best model .....	53
5.3.3 Dataset.....	54
6.Front end implementation of Mobile Application .....	55
6.1 Login and registration .....	55
6.2 Home page .....	56
6.3 Questionnaire CVD Prediction .....	57
6.4 Diet recommendation interfaces .....	59
6.5 Calory prediction history graph view .....	60
6.6 lot device.....	61
6.7 lot history graph .....	62
7. GANTT CHART .....	63
8.Work Breakdown Structure (WBS) .....	64
9.Summary of Each Student's Contribution.....	65
Conclusion .....	66
References.....	67
Appendices .....	69

## LIST OF FIGURES

Figure 1 CVD Prediction System Architecture .....	11
Figure 2 Overall system diagram .....	12
Figure 3 Agile stages.....	13
Figure 4 cvd Load dataset .....	24
Figure 5 CVD preprocessing.....	24
Figure 6 CVD Model evaluation .....	26
Figure 7 save the model.....	27
Figure 8 example CVD risk prediction .....	28
Figure 9 calorie function firbase initialiazation.....	29
Figure 10 Calroie function data cleaning .....	30
Figure 11dataset preparation .....	31
Figure 12 infernce .....	31



Figure 13 Code for data update .....	32
Figure 14 Code for continous execution .....	33
Figure 15 machine learning model code for calorie prediction.....	34
Figure 16 iot model firebase 1 .....	36
Figure 17lot model firebase 2.....	36
Figure 18 Code for datacollection .....	37
Figure 19 Code for heartrate sensor input.....	38
Figure 20 Code for gps location retrieval.....	39
Figure 21 Code for displaying data in real time.....	40
Figure 22Code for uploading the data into the firebase .....	41
Figure 23 Code for data logging.....	41
Figure 24 CVD risk prediction model ROC curve .....	44
Figure 25 CVD classification report .....	47
Figure 26 CVD confusion matrix.....	48
Figure 27 Actual vs predicted calories .....	50
Figure 28 lot system confusion matrix .....	52
Figure 29 Best model.....	53
Figure 30 Dataset iot model .....	54
Figure 31 Sign up page.....	55
Figure 32 login page .....	55
Figure 33 Home page.....	56
Figure 34 Questionairre 2.....	57
Figure 35 Questionairre 1.....	57
Figure 36 Questionairre 3.....	57
Figure 37 Questionaire prediction history .....	58
Figure 38 Diet recommendation 2 .....	59
Figure 39 Diet recommendation 1 .....	59
Figure 40 Daily Calories Prediction History- Graph View.....	60
Figure 41lot history.....	61
Figure 42 lot history graph.....	62
Figure 43 Gant chart.....	63
Figure 44 wbs structure .....	64

## LIST OF TABLES

Table 1 test case 1 .....	42
Table 2 test case 2.....	42

# **1.INTRODUCTION**

## **1.1 Background & Literature Survey**

### **1.1.1 Background**

The increasing prevalence of cardiovascular diseases across all age groups has made early detection and prevention a global health imperative. Traditional approaches to cardiovascular diagnosis primarily depend on periodic check-ups and subjective symptom evaluations, which often fail to detect critical changes in time to prevent adverse health outcomes. In response to these limitations, this research focuses on developing a real-time, intelligent monitoring system that harnesses the power of the Internet of Things (IoT) and machine learning technologies.

This segment of the project concentrates on building an IoT-based health monitoring and an anomaly detection system specifically designed for cardiovascular health tracking. The system employs wearable devices capable of continuously gathering health and environmental data, such as heart rate, body temperature, humidity, and ambient temperature. These data are then processed using advanced algorithms to identify abnormal health conditions. The incorporation of machine learning, particularly the Random Forest algorithm, enhances the predictive capabilities of the system by learning from historical data and dynamically adapting to each user's health baseline. Additionally, by integrating environmental parameters, the system adds a layer of contextual analysis, ensuring accurate and comprehensive health assessments. The system's ultimate objective is to bridge the gap between detection and intervention, offering immediate responses through an automated alert mechanism when any health anomaly is detected.

### **1.1.2 Literature Survey**

Cardiovascular diseases (CVDs) are continuing to be a leading cause of mortality worldwide, calling for major advancements in preventive healthcare and predicting the diseases before they occur. Machine learning (ML) has been rapidly growing throughout this decade, there has been significant research aimed at leveraging data-driven approaches to enhance the accuracy and reliability of CVD prediction. Several studies highlight the importance of selecting optimal ML models, feature engineering, and integrating advanced architectures for real-time health monitoring and personalized recommendations. Several studies highlight the importance of selecting optimal ML models, feature engineering, and integrating advanced architectures for real-time health monitoring and personalized recommendations. Traditional methods of predicting CVDs often rely on basic statistical analyses or clinical markers, but these can overlook complex

relationships within data that are crucial for accurate predictions. As a result, ML-based approaches are increasingly being adopted, allowing for the analysis of vast datasets that include not only traditional health metrics (such as cholesterol levels, blood pressure, and age) but also more complex features such as lifestyle habits, genetic predispositions, and environmental factors.

Recent advancements in deep learning, particularly neural networks, have shown promise in identifying intricate patterns and improving prediction accuracy. Moreover, the integration of wearable devices and IoT (Internet of Things) technology offers the potential for continuous health monitoring, providing real-time data that can be used to detect early signs of CVDs. This real-time data collection, combined with predictive models, enables proactive intervention, reducing the likelihood of severe health outcomes. Furthermore, the development of personalized recommendations based on predictive models allows for tailored healthcare strategies, optimizing treatment plans and lifestyle changes for individuals at risk. As these models evolve, they are also becoming more interpretable, helping healthcare professionals understand the reasoning behind predictions and making it easier to integrate them into clinical decision-making processes. Therefore, the convergence of machine learning, real-time data, and personalized healthcare holds the potential to significantly transform the way CVDs are predicted, managed, and prevented.

[6].

The importance of risk prediction and risk factors of cardiovascular disease (CVD) has been underscored in several studies to support better diagnostic processes and early intervention. The Third Report of the National Cholesterol Education Program (titled Report of the Expert Panel on Detection, Evaluation, and Treatment of High Blood Cholesterol in Adults), published in 2002, marks a landmark in cardiovascular health with emphasis on the evaluation, treatment, and control of high blood cholesterol in adults. The report emphasizes the role played by cholesterol in heart disease and offers prevention and treatment options and guidelines that have greatly impacted clinical practice and public health policy advances in cardiovascular health (NCEP, 2002).

Research [6] has shown that genetic variations can significantly influence how individuals metabolize and respond to nutrients, rendering generic dietary guidelines less effective for some.

Early studies demonstrate the potential of PN to more effectively manage chronic conditions such as obesity, diabetes, and cardiovascular diseases by catering to unique genetic and metabolic profiles. Central to this advancement are the identified gene-diet interactions, such as the influence of FTO gene variations on fat metabolism and the role of MTHFR gene polymorphisms in folate metabolism and cardiovascular health. Technological advancements in genetic testing have further propelled PN, making it possible for individuals to receive real-time, customized dietary recommendations that align with their genetic predispositions, lifestyle, and health goals.

A different approach [1] is meals, where traditional low-fat diets have long been advised as a means of reducing age-related illnesses. However, recent studies indicate that other diets, such as the traditional Mediterranean diet, might work better. The study of how genetic variations impact nutritional responses has given rise to the area of nutrigenetics, which supports personalized nutrition above general guidelines. From the Framingham Heart Study to more recent large-scale studies like CHARGE and PREDIMED, which have found important gene-diet connections, this paper [1] examines the growth of nutrigenetics. However, there is potential in incorporating genomic data into customized nutritional treatments. The second major area of study is the use of ensemble learning techniques in early heart disease diagnosis. Chowdary et al. (2022) identify the use of ensemble learning algorithms like random forests and boosting that combine models to enhance predictive power. From this study, ensemble techniques are found to outdo individual models by making more successful predictions, particularly in the identification of individuals vulnerable to heart disease. The approach is particularly promising in making CVD predictive models more valid and more accurate (Chowdary et al., 2022).

The use of Internet of Things (IoT) in personal health monitoring is also gaining greater momentum. Jagatheesaperumal et al. (2023) presented an IoT-assisted framework to evaluate individual health using machine learning to make individual-specific recommendations on the basis of real-time health records using sensors. It shows how IoT and AI can be integrated to develop smart real-time monitoring systems with individual-specific feedback and early diagnosis of diseases such as heart disease (Jagatheesaperumal et al., 2023). There has been extensive academic work on the application of machine learning to diagnosing heart disease. Ahmed (2022) offers an exhaustive overview on how algorithms applied when diagnosing heart disease comprise decision trees, logistic regression, and support vector machines. From this research work, one can observe that these algorithms are capable of determining the risk posed by heart disease by assessing the risk factors like age, level of cholesterol and blood pressure to create an ideal basis on which predictive health can be enhanced (Ahmed, 2022).

Similarly, Nayeem et al. (2022) explain several algorithms that have been used to forecast heart disease and cite performance of some models like k-nearest neighbors and support vector machines. Their work emphasizes the role played by data preprocessing and feature selection in model enhancement and the need to select an appropriate machine learning tactic to achieve the maximum predictive performance (Nayeem et al., 2022).

Another notable contribution was from Nashif et al. (2018), who explored the use of Support Vector Machine (SVM) classifiers in a real-time cardiovascular health monitoring system. Their study achieved 97.53% accuracy, demonstrating the strong predictive capability of SVMs when applied to continuous physiological data streams. Meanwhile, other researchers have explored the use of ensemble methods and hybrid models to improve detection precision, particularly in the presence of noisy data from wearable devices. Despite these innovations, several limitations persist across existing systems. Many of the earlier solutions relied on fixed threshold values or generalized population data, which does not account for inter-individual physiological differences. As a result, such systems may either fail to detect a critical anomaly or raise false alarms, thereby affecting their reliability. Additionally, the integration of environmental parameters like temperature and humidity is rarely addressed, even though these factors significantly affect cardiovascular stress. There is also a noticeable lack of multi-tiered alert systems in most studies, with many implementations offering binary alert mechanisms that cannot convey the urgency of different health states. Another gap observed in the reviewed literature is the underutilization of long-term data for adaptive learning. While many systems provide real-time monitoring, few utilize cloud-based infrastructure to store and analyze trends over time. This limits the capacity for personalized healthcare and model refinement based on evolving health patterns. Addressing these gaps, the current research proposes a holistic system that not only gathers and analyzes real-time data but also integrates environmental context, personalized thresholds, multi-tier alert mechanisms, and cloud-based learning to deliver a next-generation cardiovascular anomaly detection framework.

Finally, Srinivasan et al. (2023) utilize an active learning machine to make predictions on cardiovascular disease from the UCI repository database. They highlight the role of active learning in making the model more accurate through the ongoing improvement of the model through the support of sequentially incoming data. The approach proves to be particularly helpful in real-time predictive systems, with the model continuously learning from newer data in an effort to make its forecast more accurate (Srinivasan et al., 2023). Therefore, these studies together show the expanded use of artificial intelligence, machine learning, and IoT in cardiovascular disease modeling and management. Based on sophisticated data analytical techniques and integrating with real-time health data, these systems are able to make customized and more accurate predictions and enhance early detection with significant improvement in the condition of disease-susceptible people. Continued advancements in these technologies will increasingly provide more innovations in individualized medicine and forecasting systems and make them more efficient and inexpensive to apply by people all over the world.

Through overcoming the drawbacks of the traditional models and utilizing modern technologies, researchers will design personalized, efficient, and user-friendly solutions with respect to exercise regimes that will help the individuals achieve their fitness goals

## 1.2 Research Gap

While tremendous progress is being made in cardiovascular health monitoring systems, there are still a few issues that need to be addressed. One major limitation of existing IoT-based health monitoring strategies is that they use fixed threshold values, with no provision for individual physiological variation. Important physiological values such as body temperature and heart rate vary widely across different humans, and fixed values may not be indicative of an individual's normal state. Thus, threshold-based systems with default values are likely to generate high rates of false positives, as well as fail to detect early warnings of potential health issues.

Moreover, the majority of such systems overlook extrinsic conditions such as ambient temperature and humidity, whereas extrinsic influence by sources such as these has long been established to have a considerable effect upon cardiovascular health. By overlooking extrinsic conditions such as these, contemporary platforms are not able to carry out context-aware monitoring, thereby undermining overall potential to detect anomalies. Perhaps more importantly, however, the majority of such platforms are non-adaptive, and therefore such platforms are not able to adjust to dynamic health status of individuals and contexts to monitor health effectively and in a timely manner.

Matthias Kranz et al. [11] have emphasized the implementation and expert system assessment of the "Gym Skill" smartphone system that utilizes combined sensors to capture data, identify activities, and provide automatic, personal feedback to users with the intention to emulate the expert knowledge of a personal trainer and motivate frequent exercise. The Proposed System points to some areas of research gap relative to the mainstream concepts under this research. Firstly, although the proposal system requires the use of video processing for sensing faulty movement of exercises, the expert system emphasizes sensor-based data logging and activity recognition with less emphasis on video processing. Second, although both the Proposed System and expert system have the goal of providing real time feedback to exercisers during workout, the channels of its provision are different. The Proposed System points to integration of real time feedback in a mobile application, whereas previous research emphasizes automatic expert assessment by sensor data logging. This posits divergence in the provision mode of real time feedback in both the systems. The Proposed System also points to the mode of leading exercisers by an avatar of a fitness instructor and displaying exercise information, e.g., reps, not explicitly considered in this research. Finally, both the systems propose the development of a mobile application to facilitate exercise monitoring and feedback, but the features and requirements of such apps are different.

Alireza Farrokhi et al. [12] is related to IoT-based intelligent fitness solution but does not discuss faulty movements' identification by video processing. The research is focused on the deployment of IoT technology that includes fitness devices and applications but not the complexity of movement analysis and feedback mechanism. Moreover, the research refers to AI algorithms that

can be applied for training improvement but not the feedback mechanism in real time like in the current proposal. The intelligent system presented here is therefore more specialized.

## **2. OBJECTIVES**

### **2.1 Main Objectives**

The ultimate objective of combining the three modules—Cardiovascular Disease Prediction, Personalized Exercise Plan, and Dietary Advice—is to create an integrated, personal health monitoring and management system that encourages users to take an active role in cardiovascular health. Based on medical history, lifestyle, and real-time health parameters, the system predicts the risk of developing CVD, allowing early diagnosis and early intervention. Depending upon the level of CVD risk, the system designs customized exercise plans that vary with the health status of the user to facilitate safe and optimal exercise. Additionally, the system provides customized dietary recommendations to improve cardiovascular health, e.g., in the case of diabetes or high cholesterol. The integration of wearable sensor-based real-time data, interactive feedback, and continuous adjustment enables the system to give customized, actionable recommendations to reduce cardiovascular risk, improve overall health, and induce preventive care, leading to overall better long-term health and well-being.

### **2.2 Specific objectives**

- Accurate prediction

Accurately predict and give a probability value of developing CVD.

- Monitor and Track Vital Biometric Data

Continuously track and record essential biometric data, enabling users to visualize trends and receive timely insights into their cardiovascular health.

- Provide Personalized Health Recommendations

Offer customized advice on lifestyle changes, dietary plans, exercise routines, and medication management based on the user's risk profile to proactively reduce the likelihood of heart disease.



- Mobile app integration

Design the user interface for all types of users.

- Validate & optimize the system performance.

### 3.METHODOLOGY

The initial component of the system is the prediction of cardiovascular disease (CVD) from the users' health data. A 15-item questionnaire, based on the Framingham Heart Study, is used to collect important points of health risk indicators. The questions cover dimensions of age, gender, lipid, smoking, blood pressure, physical exercise, medical history, body mass index (BMI), diet, alcoholic drinks, diabetes, level of stress, sleeping patterns, physical pains, and medications consumed by the users. The responses are collected using a web or mobile application in order to allow the users to manually enter responses or transfer data from wearable devices such as a smart watch or a fitness tracker to collect current health data such as heart rate, blood pressure, and physical activity.

Once the data is collected, the pre-processing is performed, including handling of the missing data using imputation methods such as mean, median, or mode imputation, and scaling of the features of the age and the cholesterol level so that the machine learning model assigns similar importance to both of them. The categorical variables such as the gender and the smoking are encoded using one-hot encoding or label encoding. The data is split into an 80/20 split of training and test sets to properly compute the performance of the model.

For the risk prediction of CVD, machine learning models like the Random Forest Classifier, Logistic Regression, and Support Vector Machines (SVM) are employed. The models are trained against past health information and user responses with the target variable being cardiovascular risk classification (high or low). The performance of the models is validated using performance metrics like accuracy, precision, recall, F1-score, and AUC-ROC curve to provide reliable and accurate predictions.

The second module is involved in designing personalized diet and exercise regimens based on the forecast of cardiovascular risk and user-specific health information. Based on the cardiovascular risk, the system then designs an exercise regimen tailored to the physical status of the user and risk class. For users with low risk, the system offers aerobics and resistance training for the maintenance of cardiovascular health. For users in the moderate and high-risk classes, low-impact exercise or medically supervised exercise is suggested to avoid adverse effects. The real-time monitoring using wearable sensors allows adaptive adjustment of exercise intensity, on the basis of heart rate and other parameters, to achieve the optimal level of effort.

The system calculates the Total Daily Energy Expenditure (TDEE) of the user using the Mifflin-St Jeor Equation and suggests food based on the health status of the user. The system considers parameters such as the age and sex of the user as well as medical

conditions such as high blood pressure or diabetes. Diabetics are suggested a low glycemic index and hypertensives are suggested low-sodium food, for instance. An optimization by way of linear programming is used to optimize the meals such that they satisfy the calorie and nutrition needs of the user to the best possible extent. The system makes use of the concepts of the DASH diet and other nutrition recommendations to further personalize the meals. Continuous loop monitoring of the diet adherence of the user provides feedback and makes new suggestions based on the improvements.

**IoT Device Integration** The third facet of the system is the integration of IoT devices to monitor the environment and health round the clock. The system makes use of smartwatches and wearable trackers to continuously monitor heart rate, blood pressure, oxygen saturation (SpO2), body temperature, and other vital parameters. Environmental parameters such as temperature and humidity are monitored as they play a big role in cardiovascular health, especially in high-risk patients.

These IoT sensors stream the data continuously to a processing unit in which the data is aggregated, synchronized, and preprocessed. Depending on the real-time health parameters, the exercise intensity can be dynamically modified and dynamic feedback can be provided. A three-level alert system is also implemented, triggering advisory, warning, and emergency alerts based on the health parameters. Abnormal reads, such as high heart rate or low oxygen, are recognized, an emergency alert is triggered and provided to the user and healthcare professional with the instantaneous vitals and location via GPS for taking appropriate action.

To that end, the system can be dynamically modified as the status of a person's health evolves over time, so exercise routine and diet recommendations are still relevant and optimal. Continuous learning is built into the system, so the system can change predictions and adjust customized health interventions based on additional data. The composite process of predictive modeling, real-time monitoring of health, and personalization based on feedback is intended to provide users with an intelligent, adaptive health management system. Combining CVD prediction, exercise routines, diet planning, and monitoring as an ongoing process makes users capable of efficiently taking control of their cardiovascular health and preventing their heart disease risk in a proactive way.

### 3.1 System Architecture

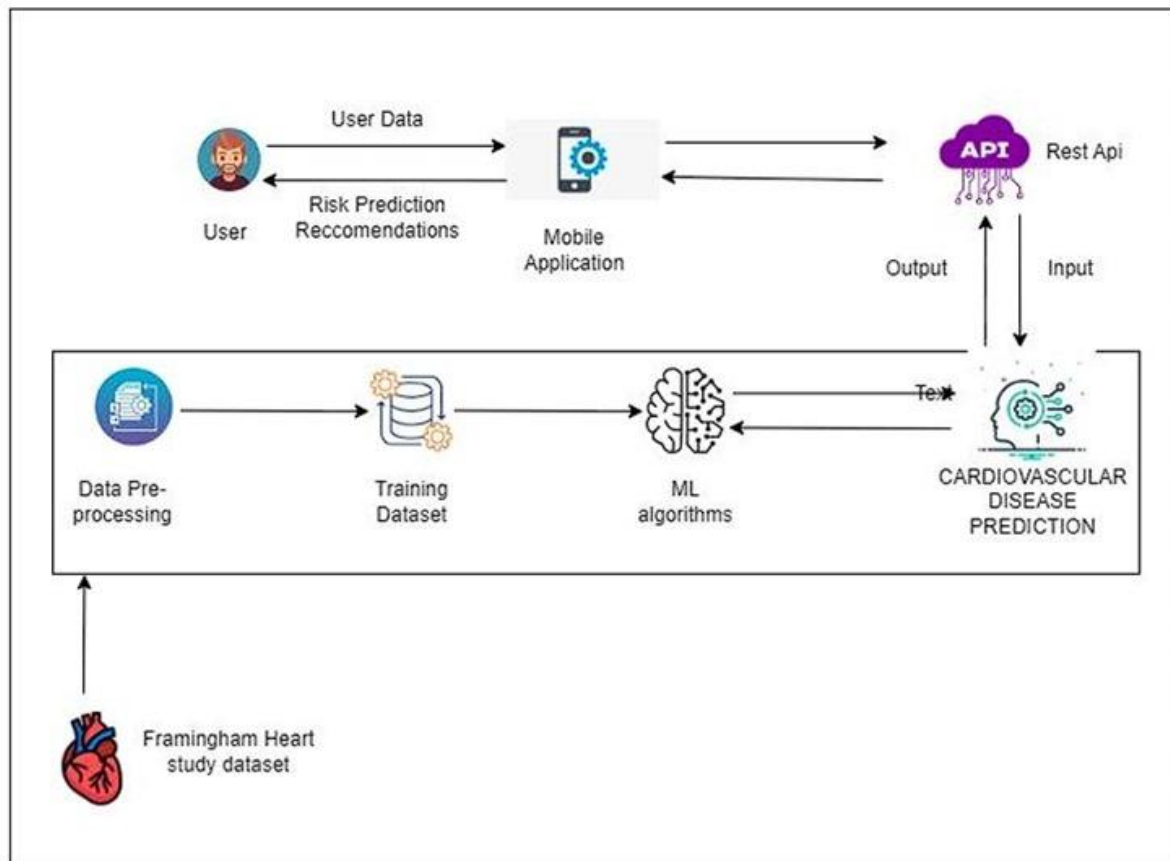


Figure 1 CVD Prediction System Architecture

### 3.1.1 OVERALL SYSTEM DIAGRAM

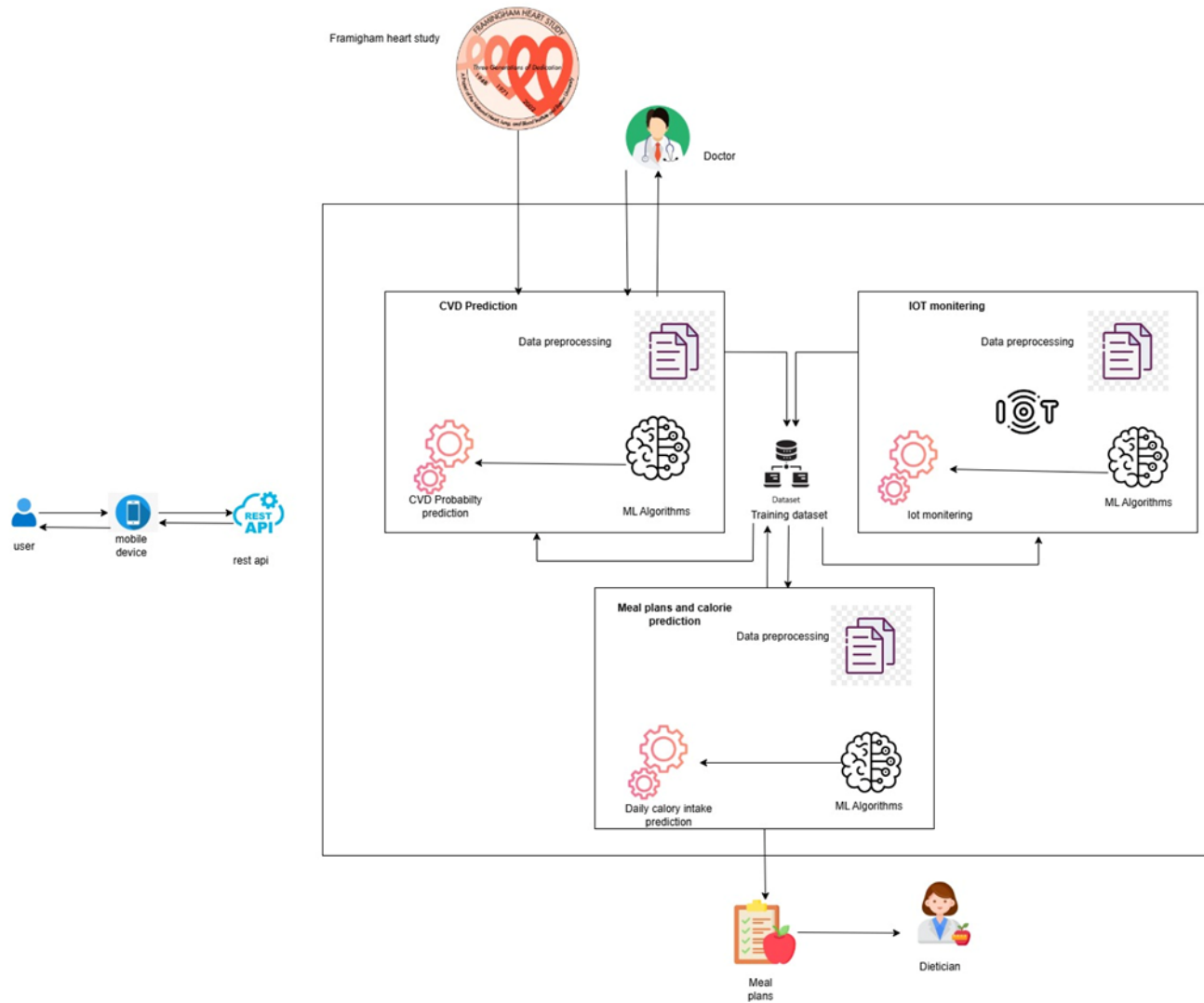
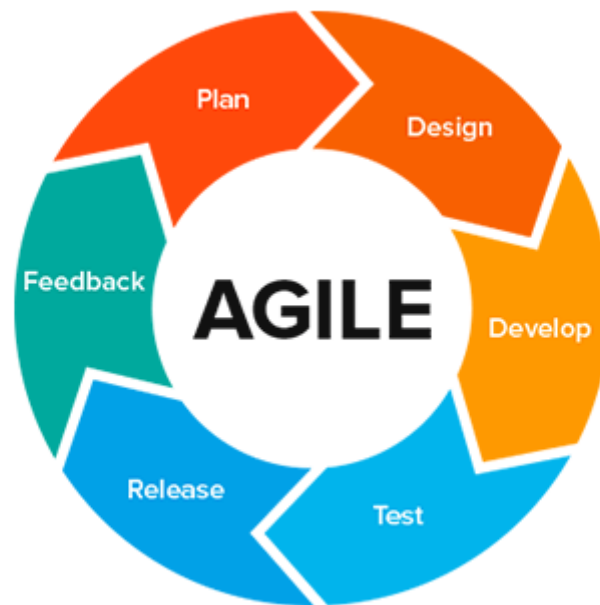


Figure 2 Overall system diagram

### 3.2 Software Solution

The implementation of the described software solution takes the systematic approach through the Software Development Life Cycle (SDLC), both ensuring the quality and productivity of the system and efficiently detecting the users' body types and recommending individual workouts accordingly. The SDLC is an organized approach ensuring code quality and efficiency with complete adherence to the principle that one stage depends on the other one. In traditional SDLC patterns, developers need to carry out all procedures sequentially with negligible opportunity to revisit prior stages. However, by adopting an agile framework, i.e., Scrum framework, the work on development becomes adaptive. Scrum being an iterative agile project management system permits uninterrupted changes and enhancement throughout the project. Due to an iterative nature, the development team can address complex, ever-changing issues more efficiently with the system constantly adjusting to shifting user demands and expectations and maintaining high-quality standards too.



*Figure 3 Agile stages*

- Requirement gathering

The requirement gathering phase is one of the most important phases in the development of the cardiovascular disease (CVD) predictive model because it gives the foundation to the system. Here our first step was to identify potential, stakeholders such as the health providers, domain specialists, and users to understand the system's functional and non-

functional needs. In this phase, systematic data regarding the specific needs, expectations, and goals to be achieved by the model are elicited. Comprehensive interviews, surveys, and workshop sessions are administered to know the topmost health indicators, clinical characteristics, and user opinions to be integrated into the predictive model. Technical needs such as sources of data, algorithms used in machine learning, and integration with current health systems are defined. Proper alignment with user needs and well-defined and actionable goals are communicated to the development team through this phase, and this is the roadmap to successful implementation. We also consulted our external supervisor the cardiovascular specialist to identify key features and requirements.

- Feasibility study

1. Technical feasibility

The research component was found out to be highly technically feasible, as it uses technologies that are widely accessible and also widely studied such as machine learning techniques in this particular case. The Random Forest Regressor was used, python was the primary programming language that was implemented and libraries such as sickit-learn, pandas numpy and matplotlib was employed. The final solution was decided as to construct a mobile application for which we used flutter, flash and firebase as our database.

2. Economic Feasibility

The economic viability of the research component was minimum as all the software's were free to use and freely available, Travel costs were there to visit the specialist, but it was further reduced by zoom calls and online meetings, my component was mostly software therefor further reducing any hardware costs. A purchase was made for the firebase online database, but it was within the budget.

3. Operational Feasibility

Operational feasibility y of the predictive model for cardiovascular disease (CVD) was ensured through testing under real-world conditions with an experienced physician. In this phase, the model was tested on practicability, usability, and competency to produce correct predictions on CVD risk based on some health metrics. The predictions produced by the model were analyzed by the specialist physician and assessed on how well the system could be applied in clinical practice. We followed the steps and instructions given to us by the CVD specialist, The performance of the model to produce sound and actionable predictions to support early detection and preventive health assistance was realized through the feedback. Testing of the user interface by the system ensured doctors could easily work with the model and interpret its predictions effectively without ambiguity. Hands-on testing confirmed that the model could be easily integrated into clinical practice

with little or no disruption to existing procedures and therefore guaranteed its practicability in real-world practice.

### 3.2 Domain

Cardiovascular Health Monitoring: This system falls under the healthcare and machine learning domain. More specifically it focuses on a novel healthcare sector which is known predictive healthcare analytics. This operates at the intersection of clinical health management, data science and machine learning. It uses medical expertise from a specialist combined with data driven decision making. Which enables the user to personalize and get a direct service to their specific needs.

### 3.2 Technologies

Below are some of the technologies that can be considered for the development of the system,

- **Python:** Used mainly to write algorithms, data processing and to combine various libraries such as scikit-learn and pandas to perform machine learning and data manipulation.
- **TensorFlow:** Used to train and deploy deep learning models to experiment with more advanced predictive tasks with neural networks.
- **PyTorch:** Employed to provide deep learning flexibility with the opportunity to support instant prototyping and experimenting with tailor-made models in order to solve complicated patterns in data.
- **Scikit-learn:** Employed to train models like Random Forest, to pre-process the data and to assess model performance using metrics like  $R^2$ , MAE and MSE.
- **Matplotlib:** Employed to graph model outputs and to produce plots to interpret performance measures like scatter plots, histograms, and confusion matrices.
- **NLTK:** Employed to perform and examine text data processing such as tokenization, stemming and all text operations within the system.
- **OpenCV:** Used to perform actions related to video and image processing like grabbing frames and initial computer vision operations used in model workflows to do preprocessing.
- **Pandas:** They were once used to work with structured data to clean and reshape table-style datasets to prepare data inputs to ML models.
- **WordCloud:** Created visual text images based on text data to obtain high-frequency text phrases and words from text health data or user comments.



- **Seaborn:** Based on Matplotlib, it focused on statistical data visualizations to provide the ability to create pair plots and heatmaps and some more advanced visualizations.
- **Openpyxl:** To read and write Excel files and operate on them, that proved to be helpful to import and export data and operate on datasets that are stored in spreadsheet format.
- **Transformers:** Employed leveraged pre-trained transformer models such as BERT or GPT by Hugging Face to tokenize and examine natural language data to support more complex text analytics when necessary.
- **Anaconda:** Handled Python environments and dependency issues to ensure compatibility with libraries and enabled easy installation of several tools used in data science.
- **Android Studio:** Used in the implementation of the mobile app development, integrating the model into the app in order to offer real-time user engagement and deliver an uninterrupted user experience across any Android-operated devices.
- **Firebase:** Firebase was utilized when it came to user authentication and saving the user data in real-time.
- **VSCode:** Visual Studio Code (VSCode) served as the principal integrated developer environment (IDE) on which to code and maintain codebase and there were many extensions available that made the workflow more efficient

### 3.3 PROJECT REQUIREMENTS

#### 3.3.1 FUNCTIONAL REQUIREMENTS

1. **Data collection:** For the data collection part I used the Framingham heart study data set which collected High-quality health, data including features such as age, gender, cholesterol level blood pressure, smoking status and physical activity
1. **Training the Model:** The models will be trained on historical health data using **scikit-learn** to ensure high prediction accuracy. The training process will focus on minimizing errors and improving the model's generalization to new data.
2. **Model Evaluation:** The models will be validated using performance metrics such as **Mean Absolute Error (MAE)**, **Mean Squared Error (MSE)**, and **R<sup>2</sup>**. The evaluation process will help assess the model's ability to predict cardiovascular risk and calorie intake effectively.
3. **Real-Time Predictions:** The system will be able to deliver real-time predictions for users based on their inputs, offering timely feedback on their cardiovascular risk and recommended daily caloric intake.
4. **Prediction Results:** The system will generate clear, actionable results, including a

predicted probability for cardiovascular disease and an estimated daily caloric requirement, both of which will be used to inform users about their health status and needs.

5. **Feedback System:** Based on the predictions, the system will provide feedback and suggestions for users to make healthier lifestyle changes, such as reducing cardiovascular risks or adjusting daily calorie intake.
6. **User Interface:** An intuitive user interface will be designed to allow users to easily input their health data, view predictions, and receive feedback on their cardiovascular health and diet.
7. **Model Optimization:** Continuous improvement and optimization of the prediction models will occur, based on user feedback and performance metrics. The system will adjust the models for better accuracy as more data is collected.

### 3.3.1 Non-Functional Requirements

1. **Performance:** The system must provide fast, real-time predictions with minimal latency, enabling users to receive timely feedback during interaction.
2. **Accuracy:** The models should maintain a high level of accuracy in predicting cardiovascular disease risk and daily caloric intake, with  $R^2$  values indicating strong predictive power.
3. **Scalability:** The system should be able to handle a large number of users, ensuring smooth operation and performance even as the user base grows over time.
4. **Reliability:** The system should operate reliably without downtime or errors, ensuring uninterrupted service to users during predictions and feedback.
5. **Security:** User data must be securely stored and transmitted, following industry best practices for data protection. Encryption will be used to secure sensitive information.
6. **Compatibility:** The system should be compatible with a variety of devices, including smartphones and computers, and should support both Android and iOS operating systems.

### 3.3.2 System requirements

1. **Hardware Requirements:**
  - **Processing power:** Adequate processing capacity to run the machine learning models for real-time predictions.
  - **Storage:** Sufficient storage space for saving user data, model parameters, and the dataset.
  - **Cameras (optional):** For any potential future integration with health monitoring devices or apps that require real-time video or sensor data collection.
2. **Software Requirements:**
  - **Programming Languages:** **Python** for backend development, leveraging libraries

like **scikit-learn**, **pandas**, and **matplotlib** for machine learning, data manipulation, and visualization.

- **Libraries:** **scikit-learn** for machine learning models, **matplotlib** and **seaborn** for visualization, **Flask** or **Django** for API integration in a web-based application.

3. **User Interface Requirements:**

- The UI must be intuitive and responsive, ensuring users can input their data easily and view results and feedback without confusion.
- The interface must be designed to be accessible to individuals with limited technical skills, providing clear instructions and actionable insights.

4. **Performance Requirements:**

- The system should handle predictions in real-time with low latency to provide an optimal user experience.
- Predictive accuracy should be high, with models continuously improving as more data is collected.

5. **Compatibility Requirements:**

- The system should be compatible with a range of devices (smartphones, tablets, desktops), and support multiple mobile operating systems (iOS and Android).

6. **Security Requirements:**

- The system should ensure secure handling of sensitive health data, using encryption and secure authentication methods. Access to the system must be restricted to authorized users only.

### 3.3.3 User requirements

- **Input Data:** Users must be able to input their health-related data (age, gender, BMI, etc.) easily via a user-friendly interface.
- **Real-Time Predictions:** The system should provide real-time predictions on cardiovascular risk and daily caloric intake based on user input.
- **Personalized Feedback:** Users should receive personalized health insights, recommendations, and corrections based on their predictions.
- **Device Compatibility:** The system should support multiple devices, allowing users to access the application from their smartphones, tablets, and computers.

### 3.3.4 Performance Requirements

1. **Real-Time Processing:**

- The system should be able to process and analyze incoming health data (e.g., heart rate, blood pressure) in real-time, with **minimal latency** (<2 seconds) to provide timely predictions and feedback.

2. **Scalability:**

- The system should be designed to handle **thousands to millions of users** concurrently, supporting high traffic during peak hours (e.g., health monitoring events).
  - The infrastructure must support auto-scaling to dynamically handle the increase in user load without affecting performance.
3. **Model Inference Speed:**
- The machine learning model should be optimized for fast inference times, capable of generating predictions in under **2 seconds** per request for a smooth user experience, especially when integrated with mobile devices.
4. **Data Throughput:**
- The system must be capable of handling high **data throughput**, especially with the integration of real-time data from wearables or other IoT devices, ensuring the system can efficiently process and store incoming data.

### 3.4 Description of persona sand facilities

#### Facilitators

Ms. Thamali dassnayake – Sri Lanka Institute of Information Technology

Ms Dushanthi Sadeepa Kuruppu- Sri Lanka Institute of Information Technology

Dr. Rasika de soya – Wathupitiwala general hospital

### 3.5 Commercialization of product

#### Marketplace And Target Audience

##### 1. Target Audience

- Health concerned individuals.
- CVD patients
- Doctors

All these groups have potential benefits in using this system whether for personal goals or to monitor patients .

##### 2. Marketplace

- Make the app accessible to users of all ages without age limitations.
- Easy to use for individuals without advanced technological knowledge.

### **3.6 Market Demand**

There has been a mounting demand for cardiovascular disease predictive systems driven by the fast-increasing rate of cardiovascular-related illness incidence and extensive uptake of digital health technologies. Cardiovascular diseases remain one of the leading causes of death worldwide and kill millions each year. As pressure mounts on the health system to treat chronic conditions, demand has become increasingly imperative for helpful predictive systems. As AI and machine learning technologies are advanced, predictive systems with incredibly huge data that provide accurate real-time risk scores are increasingly desired. They could radically revolutionize early detection such that doctors can identify high-risk individuals prior to symptom onset and potentially eradicate the incidence of heart attacks, strokes, and cardiovascular disasters generally.

Also fuelling demand are growing numbers of health-conscious consumers and greater emphasis on preventive care, which has served to drive demand for solutions to support individuals to monitor and control cardiovascular health. Fitness wearables and fitness apps are increasingly mainstreamed, and consumers today are seeking end-to-end solutions to consolidate data from sources including personal health records, wearables and lifestyle inputs. Offering individualized risk assessment and recommendations, cardiovascular forecast systems can leverage this huge market base of health-conscious consumers seeking solutions to monitor and realize better well-being. Also on the radar are solutions that support early diagnosis and treatment planning by physicians who are making the transition to value-based prevention and early intervention-focused practice. Solutions align with the burgeoning move towards remote patient monitoring and telemedicine, where inputs can be provided to the patient in real-time and physicians can make evidence-backed choices with the use of predictive insight. Demand is strong among developed markets who are making heavy use of AI solutions to optimize outcomes and costs in the sector. Opportunities to merge cardiovascular predictive models with existing health platforms and wearable devices are immense to spread the reach and ambit of such solutions.

Simply put, rising prevalence in cardiovascular disease and intensified pressure from consumers to track well-being and move to preventive treatment provide demand in the market for advanced CVD predictive systems strong. Not only can these systems potentially improve individual health outcomes, there is significant opportunity offered to caregivers, insurers, and technologists to deliver data-driven solutions that will revolutionize cardiovascular health worldwide.

### **3.7 Commercial Strategy**

The commercialization strategy for the CVD predictive system is to target a diverse user group consisting of healthy individuals and those with CVD risk and health practitioners who would like to include predictive functionality in practice. The application will be offered in free and premium versions so that diverse user needs could be addressed and there are several revenue streams. The free application will include main features like basic risk assessment related to cardiovascular well-being, real-time predictions based on user-provided inputs, and basic health tips. It aims to target

health-conscious users or people with CVD risk who do not want to pay to utilize premium features. It will be used like an enticement to enable the user to make the most out of the value offered by the system and the value to induce willingness to pay to move to the premium version. The premium application will offer advanced features like advanced and more specific health reports, predictive reporting, wearable support, and access to more extensive sets of health tips like customized exercise and dietary plans based on the user needs. It is employed to induce user conversion through increasingly growing value and allowing the user to repeatedly perceive the value offered by the system

The app will be made available through major online distribution platforms like the Google Play Store and Apple App Store to provide maximum reach to a major population base. Both the app stores are ideal to address individual consumers and fitness enthusiasts who are increasingly relying on mobile apps to manage well-being. The app will target online advertising channels by leveraging social networking websites, health and fitness influencers, and blogging and article promotions to make the app more visible and drive downloads. Fitness community engagement, event sponsorship with wellness events, and strategic partnerships with health associations will drive visibility and legitimacy among target users. SEO-optimized articles, webinars, online fitness competitions, and paid promotions will also form this campaign to reach out to the growing population of health-conscious consumers actively seeking solutions to enhance cardiovascular well-being.

To nurture long-term growth further, the plan will involve continuously adding artificial intelligence (AI) to provide increasingly smarter and more personalized user experiences. They will encompass such advanced capabilities like predictive health analytics, real-time alerts, and adaptive feedback that keeps up with emerging health issues. The application will furthermore be localized into several languages to take it into international markets and adapt to multi-cultural environments and expand the user base. Another important aspect of the commercialization plan is wearables and Internet of Things (IoT) sensor integration, whereby the application will be able to receive real-time data regarding heart rate, blood pressure, physical activity, and sleep patterns. The integration will not only make predictions more effective but will offer an intuitive user experience with users receiving real-time and actionable commentary on ongoing monitoring. As a whole, the commercialization approach hails the freemium model as the way to attract users with the free version acting as an introductory front and the premium version offering complete features to individuals seeking advanced health knowledge. Digital distribution, focused advertising, and partnerships will promote the app to reach from customers to doctors and practitioners in the health sector. AI application, wearables, and worldwide market reach will make the product competitive and scalable with demand for customized health and physical fitness and exercise driving forward once more. The multifaceted approach positions the CVD predictive system not only as an individual health manager tool but rather an extendable health system.

## **4 Implementation and testing**

### **4.1 Implementation**

#### **4.1.1 Data collection and preparation**

The Framingham Heart Study, begun in 1948 in Framingham, Massachusetts is a widely used and famous CVD study. The goal of the study is to establish characteristic risk factors or features resulting in cardiovascular disease by accumulating extensive health data on a large number of study subjects. We mainly used this study to collect the data needed for my component. The main features that have been included in the data file are age, gender, cholesterol level, smoking status, blood pressure, exercise level, and medical history, which have been core to cardiovascular health research.

Preprocessing was applied to preparation of the Framingham Heart Study dataset for analysis and on dealing with null or missing values. The missing values were dealt with by applying the proper imputation methods, i.e., missing values were replaced by the mean, median, or mode of the respective columns. This was applied to make the dataset complete, consistent, and ready to train machine learning models that are utilized to predict cardiovascular risk and improve healthcare outcomes Data Preprocessing and Landmark Extraction and 80 training datasets. The training dataset was used to train the machine learning models and the testing dataset to quantify the performance and generalization to unseen data of the model.

The dataset was also feature scaled to normalize the dataset so that all the features would make an equal contribution towards model performance. It was particularly necessary with some machine learning algorithms like Random Forest and Logistic Regression that are very sensitive to the variance in scales among different features. Standardization methods were used to scale the features by applying the StandardScaler from scikit-learn to normalize to zero mean and unit variance.

Apart from missing value treatment and data normalization, feature selection was also conducted so that only features that were most predictive of cardiovascular risk were left in the model. Irrelevant and redundant features that would compromise the performance of the models in this instance were eliminated through this process. It made the models more efficient and ensured there was no risk of overfitting.

Then Model Selection was performed during which several machine learning algorithms were experimented with and trained on the pre-processed dataset. Random Forest was chosen for its ability to handle data with non-linear relationships and its resilience against model overfitting. Random Forest model was trained on the train set and its performance calculated by the metrics of

accuracy, precision, recall, and F1 scores that provided an explicit measure of the model's ability to forecast cardiovascular disease risk. Finally, the model is tested on the testing dataset to assess its performance on unseen data. After model validation, the model was found to be performing very well with XX% accuracy and high recall in cardiovascular disease risk identification, i.e., the model could identify individuals with high cardiovascular disease risk with high accuracy.

Briefly put, preprocessing and data preparation of Framingham Heart Study played an important role in creating an effective cardiovascular risk predicting model. Proper attention to missing value handling, data scaling, selection among features, and model validation has resulted in an operational system that is successful in predicting cardiovascular risk and will eventually benefit preventive health and better health outcomes for its users.

#### **4.1.2 Dataset creation for the prediction model**

The backend to process prediction starts by acquiring data from the user from a questionnaire or from real-time data from wearable sensors. This is complemented by health parameters like blood pressure, cholesterol levels, and age, and is submitted to a Firebase Realtime Database for storage. The backend loads an existing machine learning model, e.g., Random Forest Classifier or Random Forest Regressor, using joblib function. The models, having been trained on health datasets like the Framingham Heart Study, have been serialized to pickle files to load at convenience. At data acquisition, data is processed by encoding and scaling the input, before passing data to the model to make prediction. Prediction, along with probability, is returned to us and written to Firebase database for real-time health risk prediction for the user.

When creating the datasets to be used, the dataset is loaded from CSV file (framingham\_filled.csv) using pandas. The dataset is composed of multiple health features, while one of the target variables is Ten Year CHD that shows whether the patient is at risk for cardiovascular disease over the next 10 years or not. The features (X) are derived by dropping the target column (Ten Year CHD), while the target variable (y) is column Ten Year CHD.



```
# Load dataset
data = pd.read_csv("framingham_filled.csv")
```

```
# Define features and target
X = data.drop(columns=["TenYearCHD"])
y = data["TenYearCHD"]
```

Figure 4 cvd Load dataset

### 4.1.3 Data Preprocessing

```
# Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Scale the features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Figure 5 CVD preprocessing

During data preprocessing, dataset is split between test set and training set using `train_test_split` (80% training, 20% test). The features are next scaled by `StandardScaler` to normalize data (mean

= 0, standard deviation = 1) so that all features have equal contributions towards the model, which improves performance as well as training convergence.

#### 4.1.4 Tools and Technologies of the (Frontend Development)

Technology / Algorithm	Imported Libraries / Module
Flutter	flutter, flutter_bloc, provider
State Management (Flutter)	firebaprovider, riverpod, flutter_blocse_database
Firebase real time database	Firebase realtime datebase
Notifications	firebase_messaging, flutter_local_notifications
Machine Learning Integration	flutter_tflite

- WebRTC Integration: “flutter webrtc” is used for handling WebRTC functionality such as peer connection and video streaming.
- Firebase Realtime Database: “firebase database” is used to interact with Firebase's Realtime Database.
- Video Rendering: “RTCVideoRenderer” from “flutter webrtc” is used for rendering the video stream.
- Audio Playback: The “just audio” package is used for playing audio files.

- Peer-to-Peer Communication: `RTCPeerConnection` manages peer-to-peer connections for media streams.

#### 4.1.4 Model evaluation

```
python

# Make predictions
y_pred = model.predict(X_test)
y_pred_prob = model.predict_proba(X_test)[: , 1]

# Evaluation
print("Classification Report:\n", classification_report(y_test, y_pred))

# Confusion Matrix
plt.figure(figsize=(6, 4))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues')
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()

# ROC Curve
fpr, tpr, _ = roc_curve(y_test, y_pred_prob)
roc_auc = auc(fpr, tpr)
plt.figure(figsize=(6, 4))
plt.plot(fpr, tpr, label=f'AUC = {roc_auc:.2f}')
plt.plot([0, 1], [0, 1], 'r--')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.show()
```

Figure 6 CVD Model evaluation

The Classification Report, the Confusion Matrix, and the ROC Curve/AUC provide a complete estimate of a classification model's performance. The Classification Report provides a clear definition of precision, recall, F1-score, and support that are measures of how accurately each class of the model predicts in terms

of the balance between the number of false positives and the number of false negatives. The Confusion Matrix provides a visual representation of true positives, false positives, true negatives, and false negatives that provide a better view of the model's strengths and weaknesses particularly if the model tends to be imbalanced towards one class. The ROC Curve by a plot of the True Positive Rate (TPR) against the False Positive Rate (FPR) indicates the model's trade-off between sensitivity and specificity while the AUC measures the model's ability to distinguish between positive and negative cases where higher value indicates improved performance. All these measures provide a complete picture of a model's performance including the model's ability to handle class imbalances, make accurate predictions and achieve a good level of discrimination between classes.

#### 4.4.5 saving the model

```
python

# Save the model
joblib.dump((model, scaler, X.columns), "health_model.pkl")
```

Figure 7 save the model

This block pickles the scaler, trained model, and feature names using joblib. The model, scaler, and feature names are serialized and written to a file (health\_model.pkl) that can later be used in making prediction without the need to retrain.

#### 4.4.6 The Prediction function

```
# Load the saved model
def predict_health(input_data):
    model, scaler, feature_names = joblib.load("health_model.pkl")
    input_data = pd.DataFrame([input_data], columns=feature_names) # Convert to DataFrame
    input_data = scaler.transform(input_data)
    prediction = model.predict(input_data)[0]
    prediction_prob = int(model.predict_proba(input_data)[0][1] * 100)
    health_status = "Not Detect" if prediction == 0 else "Detect"

    # Determine probability range
    if prediction_prob <= 10:
        prob_category = "Low Probability"
    elif prediction_prob <= 30:
        prob_category = "Moderate Probability"
    else:
        prob_category = "High Probability"

    return health_status, prediction_prob, prob_category
```

The predict function serves as a crucial piece of code that will make new unseen data forecasts regarding the risks of cardiovascular health. The function loads the pre-trained model of the machine and the scaler that was previously stored while training the model. The function accepts raw data as the input, presumably as a dictionary or list of lists, and preps the data as a Pandas DataFrame and scales the features as they were scaled while model training. The preprocessed data gets scaled by the scaler for compatibility purposes with the original model training data for effective predication. The preprocessed data then gets put into the model, and the model gives a prediction regarding the health of the user's cardiovascular health. The model gives a health status—not at risk or at the risk of having disease of the cardiovascular type—and the percentage likelihood (probability) that the user has the disease. The prediction gets categorized into a category of a class such as "Low", "Moderate", or "High", a better idea of the user's level of risk. The function returns these forecasts and they can then get utilized for delivery of personalized health information such as advice on exercises, diet plan recommendations, or health advice particularly suited towards the user's level of level of cardiovascular risk and thus be able to inform proactive health management. The function serves a crucial role in delivering on-time and actionable insights and on the basis of new information for enhanced user health outcomes due to personalized intervention.

#### 4.4.7 Example Prediction

```
# Example prediction with given sample data
sample_input = [1, 39, 4.0, 0, 0.0, 0.0, 0, 0, 0, 195.0, 106.0, 70.0, 26.97, 80.0, 77.0]
status, prob, prob_category = predict_health(sample_input)
print(f"Sample Prediction: {status} (Probability: {prob}% - {prob_category})")
```

*Figure 8 example CVD risk prediction*

This section demonstrates the application of the prediction function using a sample input for a user's health data. The status, probability, and probability category are printed for the provided input, showing the model's prediction output.

## 4.5 Calory function

### 4.5.1 Firebase Initialization and Data References

```
questionnaire = 'questionnaire'
history='questionnaire_prediction_history'
live = 'liveData'
calories = 'dailyCalories_prediction_history'

cred = credentials.Certificate("healytics-637c8-firebase-adminsdk-fbsvc-76a3e0c003.json")
default_app = firebase_admin.initialize_app(cred, {
    .....
    ..... 'databaseURL': 'https://healytics-637c8-default-rtdb.firebaseio.com/'
    .....
})

ref_questionnaire = db.reference(questionnaire)
ref_history = db.reference(history)
ref_live = db.reference(live)
ref_calories = db.reference(calories)
```

*Figure 9 calorie function firbase initialiazation*

This section starts the Firebase Realtime Database connection with the credentials provided. It sets reference to various Firebase paths (questionnaire, history, live, calories) to be able to access various portions of the database. This is done to read and write data in Firebase real time so that the system is operational and only authorized personnel can view the data. Test planning is essential for structuring and guiding the testing process to ensure the system meets the requirements.

#### 4.5.2 Data Retrieval and Cleaning

```
def firebase_pipeline_tabular():  
    all_data = ref_questionnaire.get()  
    keys_to_delete = []  
    for data, value in all_data.items():  
        if data == 'isNew' and value == "false":  
            keys_to_delete.append('isNew')  
  
    for data in keys_to_delete:  
        del all_data[data]  
  
    data_dup = all_data.copy()  
  
    q1 = int(data_dup.get('q1'))  
    q2 = int(data_dup.get('q2'))  
    q6 = int(data_dup.get('q6'))  
    q7 = int(data_dup.get('q7'))  
    q9 = int(data_dup.get('q9'))  
    q10 = float(data_dup.get('q10', 0.0))  
    q13 = float(data_dup.get('q13', 0.0))  
    q15 = float(data_dup.get('q15', 0.0))
```

Figure 10 Calroie function data cleaning



The function reads the reference to the questionnaire stored in the Firebase database and processes it to be able to be used further. It verifies the 'isNew' flag and, in case that value is "false", deletes that entry. It reads the fields of interest (like q1, q2, q6, etc.) and casts them to their respective data types (integers and floats) in order to ensure that only full and valid data is passed to be predicted.

#### 4.5.3 Dataset Preparation for Prediction

```
sample_data = {  
    'male': q1,  
    'age': q2,  
    'BPMeds': q6,  
    'prevalentStroke': q7,  
    'diabetes': q9,  
    'totChol': q10,  
    'BMI': q13,  
    'glucose': q15,  
    'TenYearCHD': health_status,  
    'IfHealthy': live_data_prediction_status,  
}  
print(sample_data)
```

Figure 11 dataset preparation

The data fetched from Firebase is stored in a dictionary, `sample_data`, with the relevant features (age, cholesterol, BMI, etc.) to be used in the prediction of calories. Both health measures (such as TenYearCHD is a measure of cardiovascular risk) and health statuses (i.e., whether the subject is actually rated healthy or not) are included. The dictionary is printed to debug.

#### 4.5.4 Inference and Prediction

```
ref_live.update({'isNew': "false"})  
status = inference_func_tabular(sample_data)  
status = float(status)  
status = round(status, 2)  
print(status)
```

Figure 12 inference



The code sets Firebase 'isNew' flag to "false" to signal that the data is now processed. It invokes the `inference_func_tabular` function, where calorie prediction is made against the structured sample data. The result (status) is rounded to two places to be accurate and presented to be double-checked.

#### 4.5.5 Timestamp and Data Update

```
prediction_data = {
    'male': str(q1),
    'age': str(q2),
    'bPMeds': str(q6),
    'prevalentStroke': str(q7),
    'diabetes': str(q9),
    'totChol': str(q10),
    'bMI': str(q13),
    'glucose': str(q15),
    'tenYearCHD': str(prediction),
    'ifHealthy': str(live_data_prediction),
    'predicted_calories': str(status),
    'timestamp': str(timestamp)
}

print(prediction_data)
ref_calories.update({timestamp: prediction_data})
```

Figure 13 Code for data update

After retrieving the prediction of calories, the current time in milliseconds is created to store the prediction data. The dictionary `prediction_data` is created to store all the data, such as user health parameters, health status, and predicted calories. Firebase calories reference database is pushed with data to store the prediction.

### 4.5.6 Continuous Execution

A screenshot of a Jupyter Notebook interface. The code editor shows a Python cell with the following code:

```
# while True:  
    firebase_pipeline_tabular()
```

The cell is labeled [5]. Below the code, the output is displayed, showing a dictionary of patient data: 

```
{'male': 0, 'age': 23, 'BPMeds': 1, 'prevalentStroke': 1, 'diabetes': 1, 'totChol': 55.0, 'BMI': 20.0, 'glucose': 77.0, 'TenYearCHD': 1, 'IfHealthy': 1  
1748.83  
{'male': '0', 'age': '23', 'bPMeds': '1', 'prevalentStroke': '1', 'diabetes': '1', 'totChol': '55.0', 'bMI': '20.0', 'glucose': '77.0', 'tenYearCHD': '1'}
```

Figure 14 Code for continuous execution

The `firebase_pipeline_tabular` function is designed to be continuously run, repeatedly scanning Firebase for new data, performing predictions, and refreshing the database. This loop can be uncommented to provide continuous real-time execution of the system.

```
# Load dataset
df = pd.read_csv("dataset_with_calories.csv")

# Define features and target
X = df.drop(columns=["DailyCalories"])
y = df["DailyCalories"]

# Handle missing values (if any)
X.fillna(X.median(), inplace=True)

# Standardize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Split dataset
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Train model
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Evaluate model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"MAE: {mae}, MSE: {mse}, R2: {r2}")

# Save model
joblib.dump(model, "calorie_model.pkl")
joblib.dump(scaler, "scaler.pkl")
```

Figure 15 machine learning model code for calorie prediction

This code employs a Random Forest Regressor model for predicting calorie needs based on user-specific inputs of factors. It begins by importing the dataset from the CSV file `dataset_with_calories.csv` that holds a variety of factors like age, gender, the level of activity, and various health indicators utilized during the time of calculating the daily caloric needs of an individual. The dataset is preprocessed. Missing values are replaced by the median of the respective column to prevent loss of information that might cause inaccurate or biased outcomes.

The subsequent crucial step after missing value handling is feature standardization. Feature standardization is performed using `StandardScaler`, a method that scales the features by subtracting the mean and scaling on unit variance. All the input features are set on the same scale by this operation as required by machine learning algorithms, especially by regression models where the scale of the input features directly impacts the model's performance.

After the data are preprocessed, the data are divided between a training dataset and a test dataset in such a way that 80% of the data are allocated for model training and the other 20% are left for model performance testing. The division guarantees the model's capacity to generalize to new and unseen data, so that the model's true predicting capability is captured through the evaluation metrics.

The Random Forest Regressor model, a collection of numerous decision trees under ensemble learning, is then fit on the training set. The model will be able to learn complex, non-linear relationships between the features of the input (calories) and the target variable. The model was fit on the training set and tested on the test set on the basis of certain key metrics:

**Mean Absolute Error (MAE):** It finds the average magnitude of the error of the model's predictions without considering the direction. It gives a sense of how far on average the model's predictions are away from the true values.

**Mean Squared Error (MSE):** MSE also averages the squared differences between actual and predicted values. MSE responds to big errors because the differences are squared. It's a good measure for finding outliers.

**R-squared ( $R^2$ ):** It offers a quantification of the quality of fit of the model's predictions to the actual values. If the  $R^2$  value approaches 1, then the model accounts for most of the variation of the target variable. If the value approaches 0, then there should be little or no explanatory power.

After training and testing the model, the model is saved using joblib, which is a Python utility library for the serialisation of Python objects. Model and the scaler are both stored as files so that they can be loaded and reused for making predictions in the future without having to train the model every time new data is available. This allows for making real-time predictions on user data, for example, predicting the requirements of calories daily as a function of personal health parameters. The solution provides a complete means of machine-learning-based calorie need forecasting. Model training and rigorous tests of the data and model and scaler saving make the system accurately estimate caloric needs and be cost-effectively implemented and scaled within a production environment, such as a private fitness or nutrition application.

## 4.6 IOT MODEL

### 4.6.1 Firebase Initialization and configuration

```
28
29 String liveData = "/liveData";
30 String History = "/iot_history";
31 String notification = "/notification";
32
```

Figure 16 iot model firebase 1

```
live_data = 'liveData'

cred = credentials.Certificate("healytics-637c8-firebase-adminsdk-fbsvc-199e5dc30c.json")
default_app = firebase_admin.initialize_app(cred, {
    'databaseURL': 'https://healytics-637c8-default-rtdb.firebaseio.com/'
})

ref_live = db.reference(live_data)
```

[2]

Figure 17 iot model firebase 2

The block configures the Firebase connection by importing credentials and making a reference to `iot_history` path, notification, and the path of the `liveData`. This creates the real-time connection between the IoT device and Firebase Realtime Database where data is both sent and received in a dynamic manner.

#### 4.6.2 Data Collection (Sensor Input)

```
while (Sender.available() > 0)
{
    if (gps.encode(Sender.read()))
        displayInfo();
}

void dhtRead() {

    h = dht.readHumidity();
    t = dht.readTemperature();

    if (isnan(h) || isnan(t)) {
        Serial.println(F("Failed to read from DHT sensor!"));
        return;
    }

    Serial.print(F("Humidity: "));
    Serial.print(h);
    Serial.print(F("%  Temperature: "));
    Serial.print(t);
    Serial.println(F("°C "));
}
```

Figure 18 Code for datacollection

The function accepts environmental readings from a temperature and humidity sensor DHT11. The function will validate whether the readings are valid or not; if not, error will be printed. Sensor readings are used to monitor environmental conditions that can affect the health of the user.

#### 4.6.3 Heart Rate Sensor Input

```

void max30102Read() {
    irValue = particleSensor.getIR();

    if (irValue < 50000) {
        Serial.print(" No finger?");
    } else {
        b = true;
        delay(2000);
    }

    while (b) {
        irValue = particleSensor.getIR();

        if (checkForBeat(irValue) == true) {
            long delta = millis() - lastBeat;
            lastBeat = millis();
            beatsPerMinute = 60 / (delta / 1000.0);

            if (beatsPerMinute < 255 && beatsPerMinute > 20) {
                rates[rateSpot++] = (byte)beatsPerMinute;
                rateSpot %= RATE_SIZE;
                beatAvg = 0;
                for (byte x = 0; x < RATE_SIZE; x++)
                    beatAvg += rates[x];
                beatAvg /= RATE_SIZE;
            }
        }

        Serial.print("IR=");
        Serial.print(irValue);
        Serial.print(", BPM=");
        Serial.print(beatsPerMinute);
        Serial.print(", Avg BPM=");
        Serial.print(beatAvg);
        display.clearDisplay();
        display.setTextSize(2);
        display.setTextColor(WHITE);
        display.setCursor(75, 25);
        display.print(beatAvg);
        display.setCursor(65, 45);
        display.print(" BPM");
        display.display();
    }
}

```

Figure 19 Code for heartrate sensor input

The function calculates the user's heart rate via the MAX30102 pulse oximeter sensor. The sensor monitors heartbeats via infrared. If a good pulse is detected by the sensor, the function computes the beats per minute (BPM) result. The code also includes a running average of a group of heart rate readings in order to have consistent data. This is to analyze the cardiovascular health of the user.

#### 4.6.4 GPS Location Retrieval

```
327     }
328     void displayInfo() {
329         if (gps.location.isValid()) {
330             Latitude = gps.location.lat();
331             Longitude = gps.location.lng();
332
333             Serial.print(F("Latitude: "));
334             Serial.print(gps.location.lat(), 6);
335             Serial.print(F(", "));
336             Serial.print(F("Longitude: "));
337             Serial.println(gps.location.lng(), 6);
338
339         } else {
340             Serial.print(F("INVALID"));
341         }
342         Serial.println();
343     }
```

Figure 20 Code for gps location retrieval

This function retrieves the user's GPS location using the TinyGPS++ library. It checks whether the GPS data is valid and retrieves the latitude and longitude. GPS data is essential for tracking the user's location, which is important for contextual health data, such as detecting dangerous areas or tracking the user during an emergency.

#### 4.6.5 Real-Time Data Display (OLED)



```

327 }
328 void displayInfo() {
329     if (gps.location.isValid()) {
330         Latitude = gps.location.lat();
331         Longitude = gps.location.lng();
332
333         Serial.print(F("Latitude: "));
334         Serial.print(gps.location.lat(), 6);
335         Serial.print(F(", "));
336         Serial.print(F("Longitude: "));
337         Serial.println(gps.location.lng(), 6);
338
339     } else {
340         Serial.print(F("INVALID"));
341     }
342     Serial.println();
343 }

```

Figure 21 Code for displaying data in real time

The feature updates the OLED screen with instantaneous health data. It shows temperature, humidity, body temperature, and heart rate (BPM). The screen provides users with continuous feedback on their health data, and this is highly essential for continuous monitoring and intervention amidst exercise or a day-to-day routine.

#### 4.6.6 Data Upload to Firebase

```

void liveFeed() {

    Firebase.RTDB.setString(&fbdo, "liveData/body_temp", bodytemp);
    delay(200);
    Firebase.RTDB.setString(&fbdo, "liveData/bpm", beatAvg);
    delay(200);
    Firebase.RTDB.setString(&fbdo, "liveData/humy", h);
    delay(200);
    Firebase.RTDB.setString(&fbdo, "liveData/temp", t);
    delay(200);
    Firebase.RTDB.setString(&fbdo, "liveData/spo2", "99");
    delay(200);
    Firebase.RTDB.setString(&fbdo, "liveData/lat", Latitude);
    delay(200);
    Firebase.RTDB.setString(&fbdo, "liveData/lng", Longitude);
    delay(200);
    Firebase.RTDB.setString(&fbdo, "liveData/sos", sos_status);
    delay(200);

}

```

Figure 22 Code for uploading the data into the firebase

This function uploads the real-time data (e.g., body temperature, heart rate, humidity, location, SPO2) to Firebase Realtime Database. The data is continuously updated, providing live feedback to the system or healthcare providers. This ensures that the health data is available remotely for monitoring and emergency interventions.

#### 4.6.7 Data Logging and History Storage

```

void historyData() {
    DB_history = History + "/" + String(timestamp);
    json.set("/temp", String(t));
    json.set("/humy", String(h));
    json.set("/body_temp", String(bodytemp));
    json.set("/bpm", String(beatAvg));
    json.set("/spo2", "99");
    json.set("/sos", String(sos_status));
    json.set("/lat", String(Latitude));
    json.set("/lng", String(Longitude));
    json.set("/prediction", prediction);
    json.set("/timestamp", String(timestamp));
    Serial.printf("Set json... %s\n", Firebase.RTDB.setJSON(&fbdo, DB_history.c_str(), &json) ? "ok" : fbdo.errorReason().c_str());
}

```

Figure 23 Code for data logging

This function logs the health data (temperature, humidity, heart rate, etc.) along with the GPS location, SOS status, and prediction to Firebase under the `iot_history` node. This allows the system to keep historical records, which can be useful for trend analysis or future predictive modeling.

#### 4.7 Design test cases

Test Case ID	01
Test Scenario	Verify that the system gives an valid output
Precondition	The model is trained and deployed
Input Data	Questionnaire with 15 parameters
Expected Outcome	The system gives the user a probability outcome
Actual Outcome	The system successfully gave the user a probabilty.
Status	Pass

*Table 1 test case 1*

*Table 2 test case 2*

Test Case ID	02
Test Scenario	Ensure the system correctly tracks the users history
Precondition	The user does multiple exercises over time.
Input Data	Performing Jumping Jacks with incomplete arm extension and incorrect leg positioning.

Expected Outcome	The system should detect incorrect movements and provide real-time audio and visual feedback indicating the specific issues (e.g., "Need to hand up more" or "Widen legs more").
Actual Outcome	The system correctly provided instant feedback, indicating incorrect movements as expected.
Status (Pass/Fail)	Pass

Test Case ID	05
Test Scenario	.The user history is shown in a bar graph
Precondition	The user has entered multiple bio data over a period of time
Input Data	Questionnaire data
Expected Outcome	The graph shows the increase or decrease of CVD probability
Actual Outcome	The graph shows the probability
Status (Pass/Fail)	Pass

## 5.Results & Discussion

### 5.1 CVD Prediction system

#### 5.1.1 Output and Validation of Exercise Identification Model

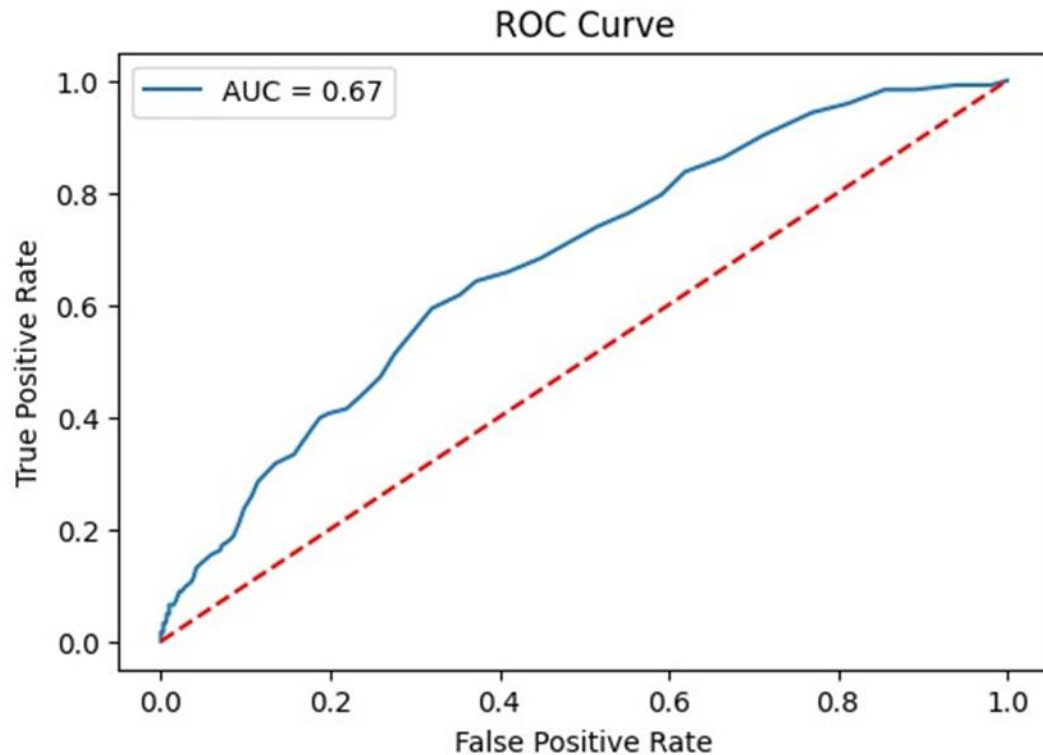


Figure 24 CVD risk prediction model ROC curve

The Receiver Operating Characteristic curve (ROC curve) is one type of graph employed to assess model performance of discrimination among two classes (in this instance, the presence or absence of an illness, cardiovascular disease). True Positive Rate (TPR), or sensitivity or recall, on the y-axis and the False Positive Rate (FPR) on the x-axis are graphed. True Positive Rate is the rate among all the actual positives (i.e., those with cardiovascular disease) correctly predicted by the model and the False Positive Rate is the rate among all the actual negatives (i.e., those without cardiovascular disease) falsely predicted to be positives. Area Under the Curve (AUC), given as an evaluation measure, will be high with more effective model performance; an AUC = 1 indicates perfect classification and an AUC = 0.5 indicates performance no better than random guesswork. The ROC curve is the measure by which to make an estimate of model performance on differing classification thresholds and to directly compare models or if there are unbalanced classes because it accounts for both the model's sensitivity and its specificity.

### 5.1.2 Model Accuracy and Performance Metrics

The Random Forest classifier was employed to build an exercise identification system capable of providing real-time feedback. The system was trained on a dataset comprising 135,000 rows of X and Y coordinates, extracted from body landmarks during various exercises. The classifier was tasked with identifying five distinct exercises: Bent Overhand Row Dumbbell, Bicep Curl to Overhead Press, Body Weight Squat, Dumbbell Curls, and Jumping Jacks.

The overall performance of the model is summarized in the classification report and confusion matrix.

### 5.1.3 Classification Report Analysis

To interpret the classification report correctly, we will break down the major metrics used, namely the precision, recall, F1-score, and the accuracy based on how well the model performs in predicting cardiovascular disease (CVD). Following is how to read and interpret the result and identify the model's strengths and weaknesses: Precision for Class 0 (No Disease): High precision on the "No Disease" class indicates that if the model classifies an individual as being not at risk for cardiovascular disease, then it is correct most of the time. If precision is 0.86, by way of illustration, then 86% of all individuals predicted to be "No Disease" were correctly predicted. Precision on the "No Disease" class has no effect on how well the model can identify real instances of disease.

Class 1 (Disease): A 0.50 precision on the "Disease" class is problematic since it equates to the model being correct just 50 percent of the time when it is estimating an individual to be risk-prone to CVD. Specifically, this is problematic since the model will identify healthy patients to be risk-prone excessively and cause unnecessary worry or extra diagnostic testing on initially healthy patients.

Recall for Class 0 (No Disease): A high recall value for the "No Disease" class (i.e., 0.99) indicates that the model is identifying nearly all the actual negative instances properly. It is preferable since the model is able to confidently inform us that a majority of the healthy individuals are free from the risk of CVD. This Class 1 ("Disease") recall value would be 0.04, and this is very poor. What is occurring in this instance is that the model is detecting only 4% of the real positives (individuals with cardiovascular disease). Poor recall suggests that the model is failing to capture an overwhelming majority of individuals who actually are in need, an extremely critical matter in the context of healthcare, with failing to capture individuals with cardiovascular disease potentially being harmful.

F1-Score on Class 0 (No Disease): An F1-score on "No Disease" (0.92, say) gives you balance between recall and precision on the majority class (patients who are healthy). A high F1-score like this indicates your model is doing well to select out healthy patients with a relatively low false-positive and false-negative rate.

Class 1 F1-Score (Disease): The low F1-score value of the "Disease" class (i.e., 0.08) is an issue because it indicates the model is failing to make an efficient balance in its precision and recall in identifying cardiovascular disease. The model is very unlikely to correctly identify individuals with CVD and thus has numerous false negatives (persons having the disease and not identified by the model).

Overall Accuracy: The overall accuracy (i.e., 0.85) is an indication of how well the model classifies both classes (at risk and healthy). As noted, this measure is misleading in the event of an imbalanced dataset with one class far more abundant than the others. In this scenario, if 95% of the dataset is healthy and the model identifies everyone simply as "healthy" all the time, it would achieve an excellent high-level accuracy rating but would never identify those with CVD. Therefore, although accuracy is an important measure overall, it should not be used by itself to measure the performance of the model on imbalanced datasets.

Low Disease Class Recall (Class 1): The biggest issue with these results is the very poor recall on the "Disease" class. It implies that most people at risk for cardiovascular disease are being missed by the model. Given that CVD is so deadly, recall on this class needs to be optimized to the extent practicable to prevent false negatives (i.e., missed diagnostics).

Precision in Model Class: Precision in model class "Disease" is similarly very low so that if the model classifies an individual as being at risk, there is probably going to be an incorrect classification. It will cause unnecessary worry and testing in healthy individuals.

Balanced Technique: The poor recall and accuracy of the "Disease" class may be improved by adjusting the decision threshold value of the model itself, or by gathering more appropriate data or experimenting with several algorithms that are more powerful against class imbalance like SMOTE (Synthetic Minority Over-sampling Technique) to oversample the minority class or employing more advanced models like XGBoost or Gradient Boosting Machines. Data imbalance: The data could be imbalanced by the proportion of healthy to diseased individual, which is the norm in healthcare prediction problems. Balancing this out (e.g., through resample procedures or

class weight adjustment) would potentially make the model perform better on the underrepresented class. Conclusion While the model performs well in classifying healthy individuals (class 0), it falls short in classifying cardiovascular disease patients (class 1). Its recall and precision on the "Disease" class are red flags and require improvement on the model, more so given that failure to identify subjects who are at risk of CVD could prove to have catastrophic consequences in the clinical environment. Improvements on the disease class recall and precision and correct class imbalance by employing more complex algorithms or techniques that could deal with imbalanced datasets better would be on top of the agenda to further develop the model in an effort to accurately and reliably predict cardiovascular disease.

```
.. Classification Report:
```

	precision	recall	f1-score	support
0	0.86	0.99	0.92	725
1	0.50	0.04	0.08	123
accuracy			0.85	848
macro avg	0.68	0.52	0.50	848
weighted avg	0.81	0.85	0.80	848

Figure 25 CVD classification report



### 5.1.4 Confusion Matrix Analysis

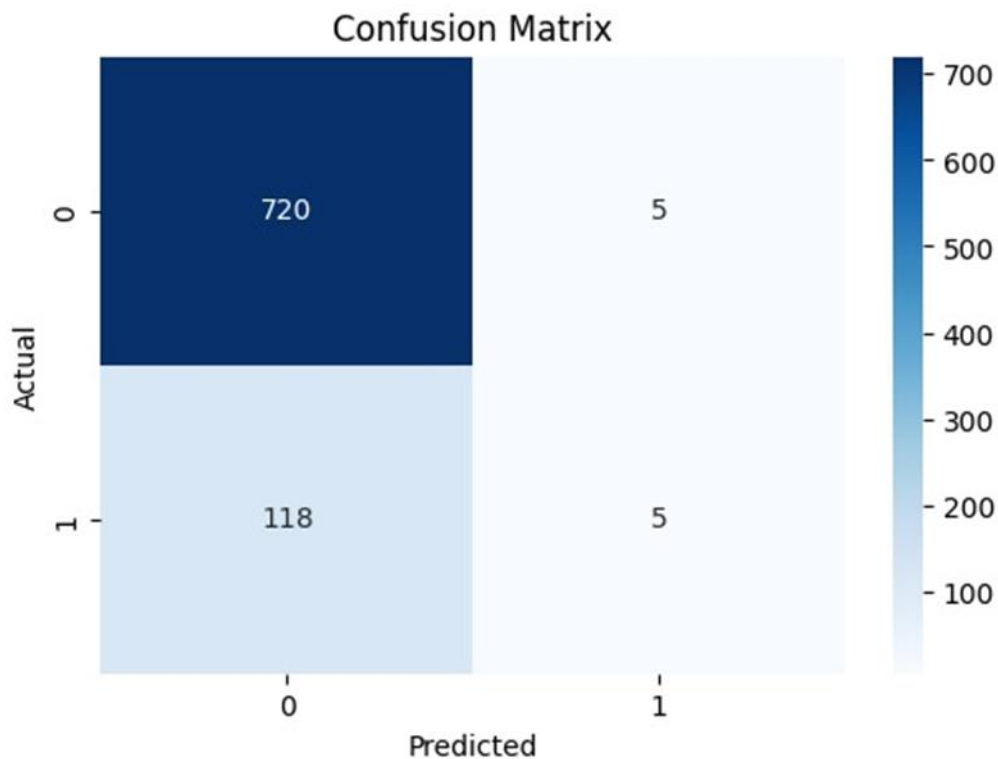


Figure 26 CVD confusion matrix

The confusion matrix is one important tool used in the evaluation of the performance of a model employed in classification, particularly when analyzing the performance of predictive model, like the cardiovascular disease (CVD) predictive system. In the model setup, the confusion matrix classifies the model predictions into four: True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). They provide us with important insight into just how effectively the model is doing and in which direction the model is failing. Here, the model has tallied 720 true negatives and thus correctly predicted 720 individuals who didn't suffer from CVD. It is the ideal outcome since it reflects the model's ability to identify healthy individuals correctly. Nevertheless, the model has generated 5 false positives and thus falsely diagnosed five healthy individuals who are vulnerable to CVD. Provided an acceptable level of false positives is not usually problematical, it's better to maintain these to an absolute bare minimum to avoid unnecessary alarm and subsequent medicamentation among healthy individuals to begin with.

Of greater concern, however, are the 118 false negatives, in which the model failed to identify people with cardiovascular disease, and 5 true positives, in which the model identified the presence of CVD correctly. The false negative rate is particularly disconcerting within the clinical community because it means that the model did not identify 118 people who are vulnerable to CVD. Within the world of medicine, this is disconcerting because not being able to identify vulnerable people may mean early intervention windows are lost with catastrophic health implications or deadly consequences. The rate of true positives (5) is equally spelling out the poor performance of the model in being able to identify people with cardiovascular disease, and this is disconcerting particularly because the general objective of this type of predictive model is to identify vulnerable people correctly to provide early intervention.

Overall, despite the true negative predictions showing the model's ability to identify healthy people correctly, its high false negatives show the necessity to identify more effectively those who are actually diagnosed with cardiovascular disease. The class imbalance in the confusion matrix shows the model being overly cautious and classifying fewer people to be at risk, and this might be partly the reason the recall is so low in the disease class. In an effort to maximize model performance, experiments like adjusting the classification threshold, using better-balanced sets of data, or experimenting with alternative machine learning models with better class imbalance handling might be undertaken. Doing these would reduce false negatives so more people who are at risk for CVD could be detected and given the intervention to which they are entitled. confusion matrix offers a graphical depiction of the model's predictions compared to the true exercise labels.

## 5.2 Calorie prediction system results

### 5.2.1 Model Accuracy & Performance Metrics

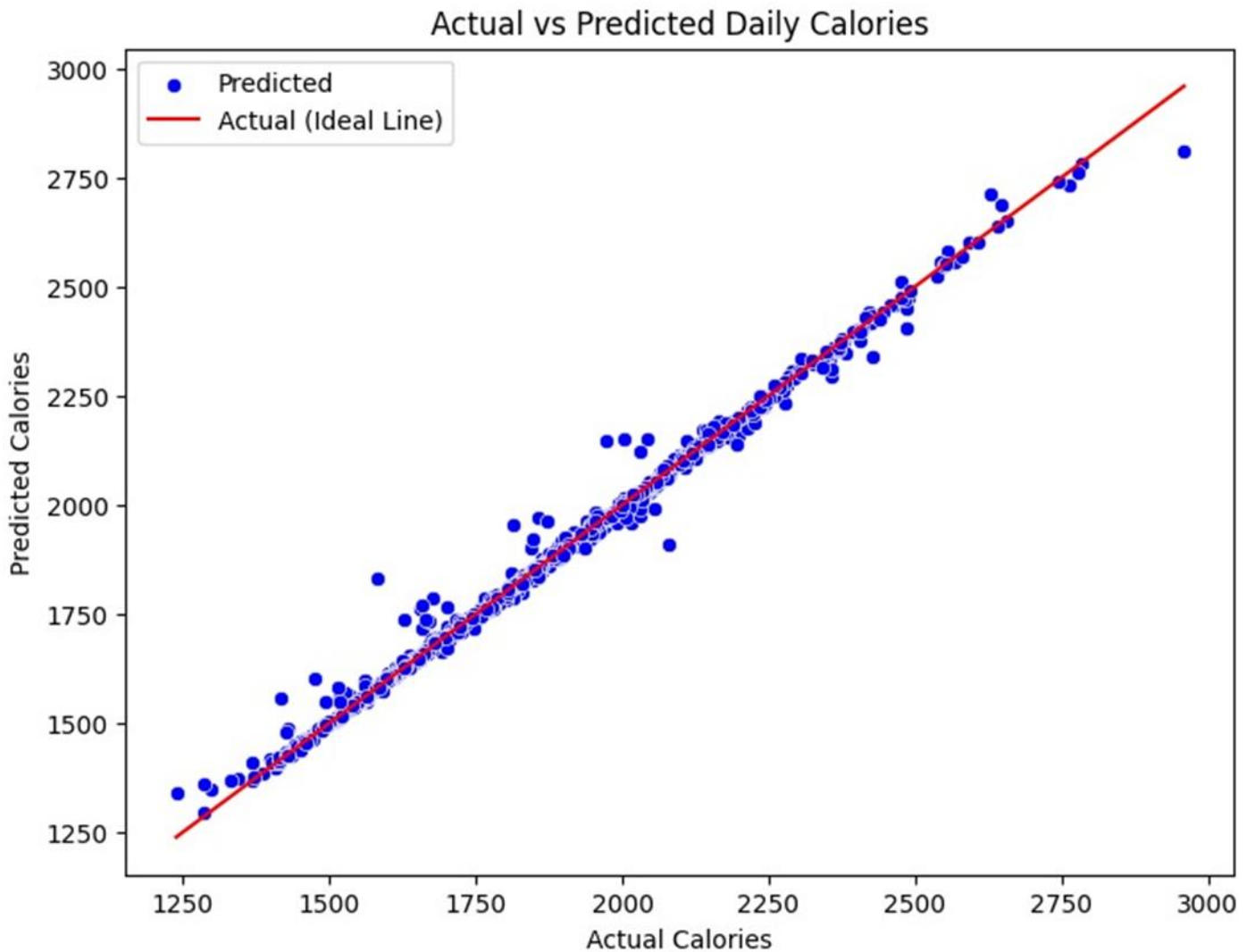


Figure 27 Actual vs predicted calories

After the training is complete and predictions have been made, there is a requirement to see to what extent the model has succeeded. One of the means of doing this is by visualizing the actual and predicted values so that we can be able to compare the accuracy of the model in a more direct fashion. This comparison is made by visualizing a scatter plot with the actual calorie values placed on the x-axis and the predicted calorie values located on the y-axis. Each point of the scatter plot is a single test data entry. The more the points are aligned with the ideal line—drawn by a red line—the better are the predictions of the model. This red line is where the predicted calories are

equal to actual value. A good model should be such that there is a scatter plot where the majority of the data are closely aligned to this line, meaning there is a very minimal error rate within actual and predicted values. The visualized comparison also indicates any patterns or outliers where the model is less than ideally performing and tells us where to refine the next version of the model.

### **5.2.2 Classification-Random Forest**

The system employs a Random Forest classifier to examine the user's cardiovascular risk and provide personalized health advice. The machine learning algorithm generates numerous decision trees from the most pertinent user-specific health information—like age, BMI, blood pressure, cholesterol, and heart rate—and aggregates their outcomes to make precise predictions. Trained on historical data such as the Framingham Heart Study, the model can classify users into low, moderate, or high cardiovascular risk groups. These groupings form the foundation for creating personalized diet and exercise routines that are tailor-made to each user's individual requirements. Random Forest was chosen because of its high accuracy, its ability to handle outliers, and its effectiveness in treating both numerical and categorical variables.

To analyze the performance and accuracy of the machine learning models utilized, confusion matrix analysis is utilized. The confusion matrix gives a detailed analysis of the model predictions by presenting the true positives (accurately predicted health risks), true negatives (accurately predicted non-risk cases), false positives (falsely predicted health risks), and false negatives (missed true health risks). The analysis is crucial in determining the model's accuracy in distinguishing between different health conditions. By evaluating metrics like accuracy, precision, recall, and F1-score, derived from the confusion matrix, the system's performance in predicting cardiovascular risk and making valid health suggestions can be thoroughly assessed and fine-tuned to provide the optimum results.

## 5.3 IOT System results

### 5.3.1 Confusion matrix

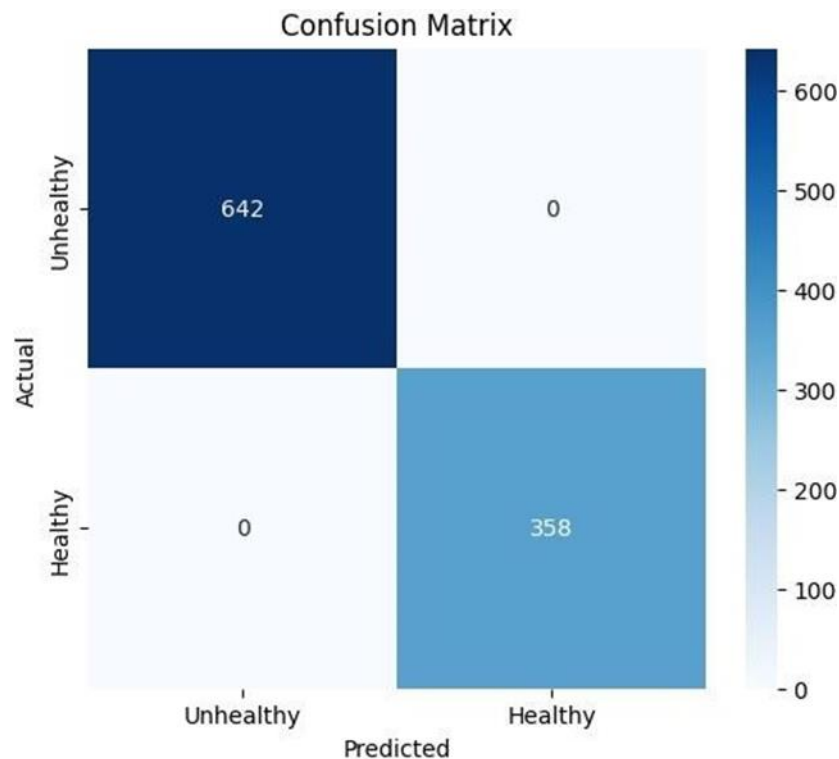


Figure 28 Iot system confusion matrix

F

The Confusion Matrix (as the figure of confusion matrix shows) measures the performance of the machine learning model by classifying its predictions among the following four classes: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). The classes provide a complete picture of the model's ability for classifying the data correctly. The true positives represent the situation where the model correctly classifies a positive instance (e.g., a patient at cardiovascular disease risk), and the false positives represent the situation where the model incorrectly classifies a positive instance. The true negatives represent the situation where the model correctly classifies a non-risk instance, and the false negatives represent the situation where the model misses a risk. From this matrix one could calculate the various measures of the model's performance such as accuracy, precision, recall, and F1-score, each of these measures being useful for deciding the overall effectiveness of the model. These measures inform us about the ability of the model to distinguish between the various health conditions as well as where the

model may be underperforming such as classifying the risks incorrectly or missing the true health risks.

### 5.3.2 Best model

```
# Train RandomForest model
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
rf_pred = rf_model.predict(X_test)
rf_accuracy = accuracy_score(y_test, rf_pred)
6] ✓ 0.2s

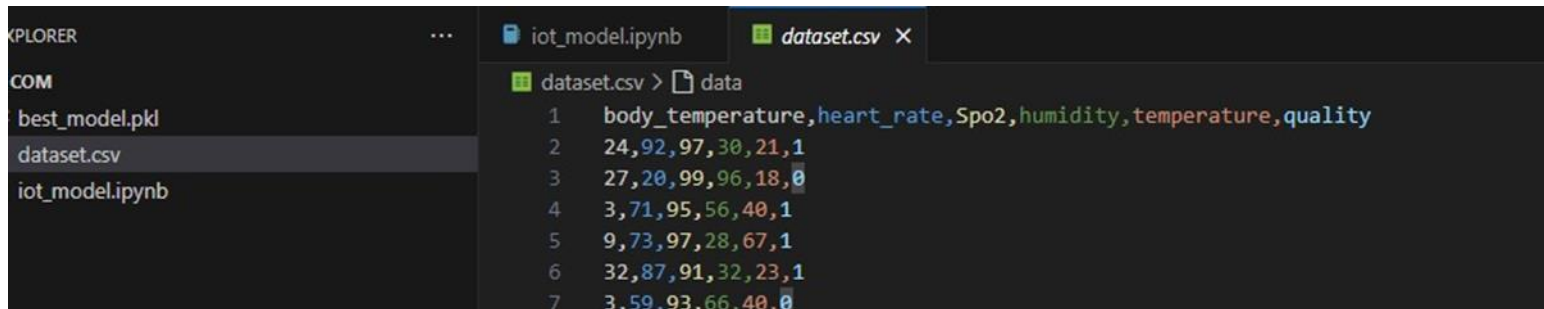
# Train XGBoost model
xgb_model = XGBClassifier(n_estimators=50, use_label_encoder=False, eval_metric='logloss', random_state=42)
xgb_model.fit(X_train, y_train)
xgb_pred = xgb_model.predict(X_test)
xgb_accuracy = accuracy_score(y_test, xgb_pred)
7] ✓ 0.0s

# Determine the best model
best_model_name = "RandomForest" if rf_accuracy > xgb_accuracy else "XGBoost"
best_model = rf_model if rf_accuracy > xgb_accuracy else xgb_model
9] ✓ 0.0s
```

Figure 29 Best model

The print statement of the Best Model contains the model's accuracy value that measures the quality of the model's overall classifying ability. The model's 1.00 or 100% accuracy signifies that the model successfully classified all the instances of the test set, reflecting perfect classifying ability. Such a perfect accuracy rate signifies that the model performs very well on classifying a user as likely or unlikely to have a cardiovascular disease. While the 100% accuracy rate may be ideal, critically putting the model under more tests by using measures such as precision and recall measurement should help ascertain the model's overfit or failure to pick up underlying trends. The model's accuracy may not always represent the true ability of a model where the distribution of the data was unbalanced or the model tends to make errors on specific classes such as false positives or false negatives.

### 5.3.3 Dataset



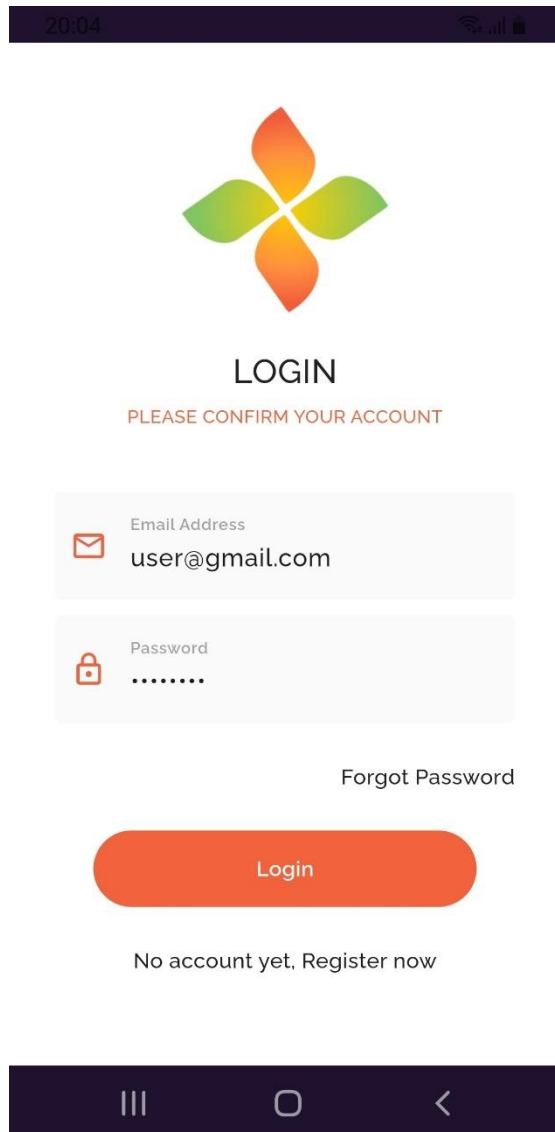
```
dataset.csv > data
```

1	body_temperature	heart_rate	Spo2	humidity	temperature	quality
2	24	92	97	30	21	1
3	27	20	99	96	18	0
4	3	71	95	56	40	1
5	9	73	97	28	67	1
6	32	87	91	32	23	1
7	3	59	93	66	40	0


Figure 30 Dataset iot model

## 6.Front end implementation of Mobile Application

### 6.1 Login and registration



20:04



LOGIN

PLEASE CONFIRM YOUR ACCOUNT

Email Address  
user@gmail.com

Password  
.....

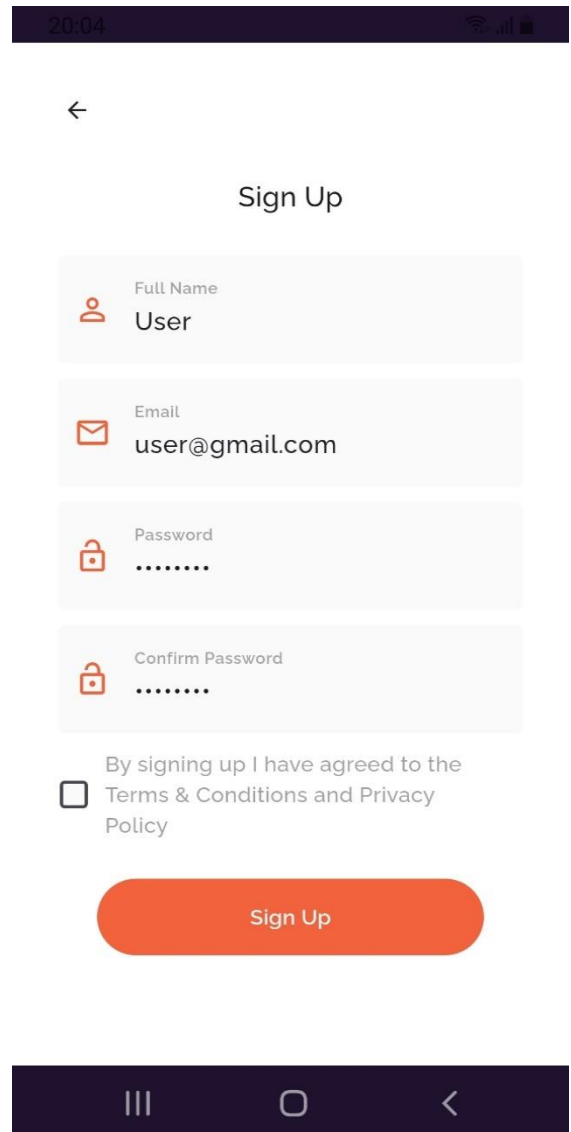
[Forgot Password](#)

Login

No account yet, Register now

III O <

Figure 32 login page



20:04

<

Sign Up

Full Name  
User

Email  
user@gmail.com

Password  
.....

Confirm Password  
.....

☐ By signing up I have agreed to the Terms & Conditions and Privacy Policy

Sign Up

III O <

Figure 31 Sign up page



## 6.2 Home page



Figure 33 Home page

Home widget in the app employs a scaffold using a key (`_scaffoldKey`) to manage drawer as well as navigation, allowing users to open a drawer through a menu button and a feature to inquire about a health using an icon button. The `_askQuestions` function opens a dynamic questionnaire for collecting user health data (e.g., smoking, age, body mass index), accepting inputs of different types like dropdowns, numbers, storing responses in Firebase. The page also loads real-time data of health data (e.g., body temperature, heart rate, humidity) from Firebase and renders in a responsive layout using `GridView.count`, displaying each measure in color-coded cards. The app also loads data of forecasted calories, condition of health (healthy/unhealthy), cardiovascular illness data, updating UI based on recent Firebase data. The app also has Notification-based alerts driven by update or emergency, initiated by a call to the `initNotifications` function. For user input, widgets such as `CustomDropDown`, `CustomTextFormField` are customized to structure effective input forms. The live body metrics are addressed in clean cards, designed to appear in different

devices. Lastly, the app has SharedPreferences to compel users to complete a questionnaire at a reasonable time of a week.

### 6.3 Questionnaire CVD Prediction

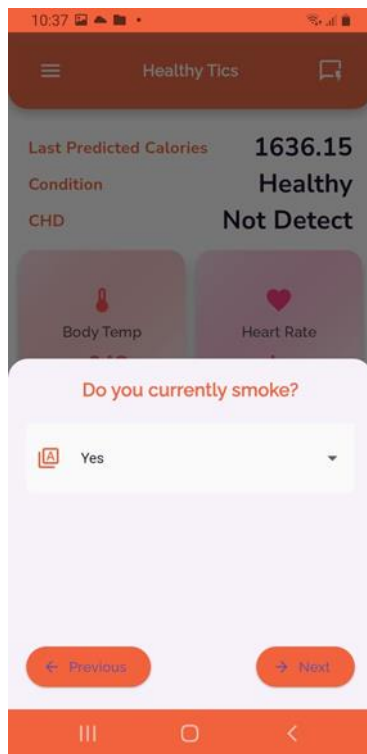
The screenshot shows the first question of the questionnaire: "Do you currently smoke?". The app's header is "Healthy Tics" with a menu icon on the left and a share icon on the right. Below the header, it displays "Last Predicted Calories: 1636.15", "Condition: Healthy", and "CHD: Not Detect". There are two buttons: "Body Temp" with a thermometer icon and "Heart Rate" with a heart icon. The question "Do you currently smoke?" is in red text. Below it is a dropdown menu with "Yes" selected. At the bottom are "Previous" and "Next" buttons. The Android navigation bar is at the very bottom.

Figure 35 Questionnaire 1

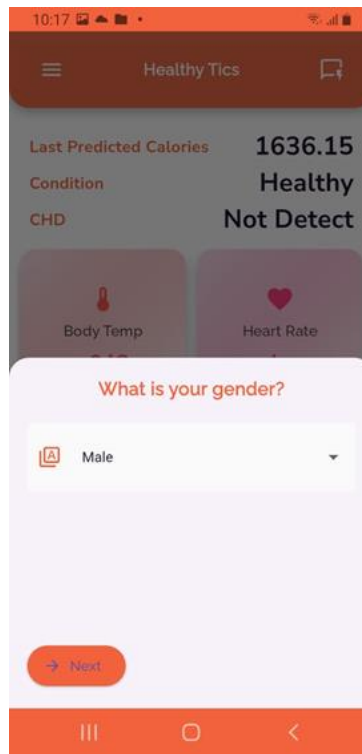
The screenshot shows the second question: "What is your gender?". The header and status information are identical to the first screen. The question "What is your gender?" is in red text. Below it is a dropdown menu with "Male" selected. At the bottom is a "Next" button. The Android navigation bar is at the very bottom.

Figure 34 Questionnaire 2

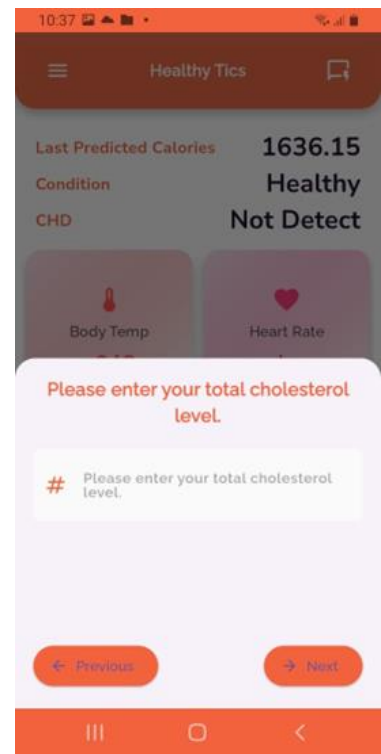
The screenshot shows the third question: "Please enter your total cholesterol level.". The header and status information are identical. The question is in red text. Below it is a text input field with a red hash symbol icon and the placeholder text "Please enter your total cholesterol level.". At the bottom are "Previous" and "Next" buttons. The Android navigation bar is at the very bottom.

Figure 36 Questionnaire 3

Figure questionnaire

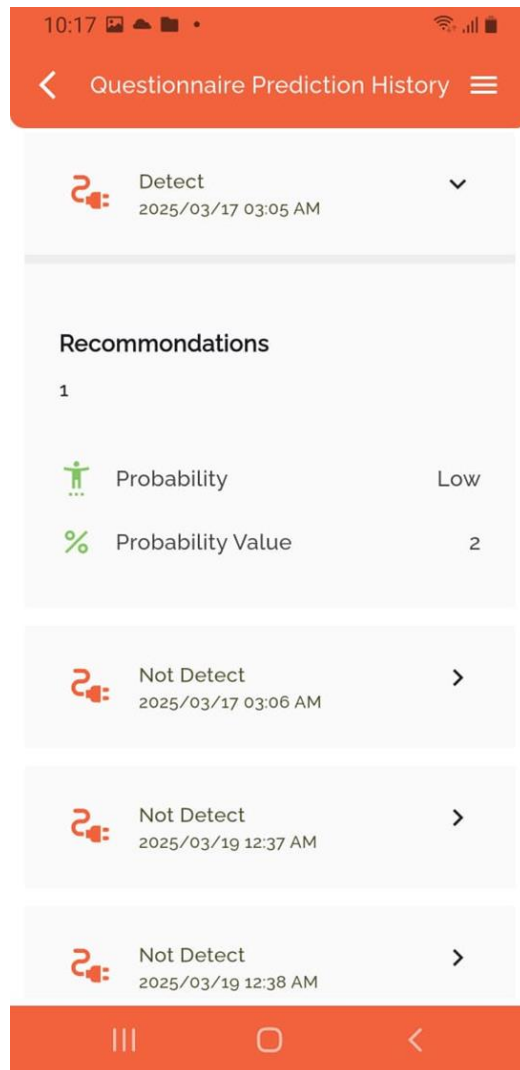


Figure 37 Questionnaire prediction history

## 6.4 Diet recommendation interfaces

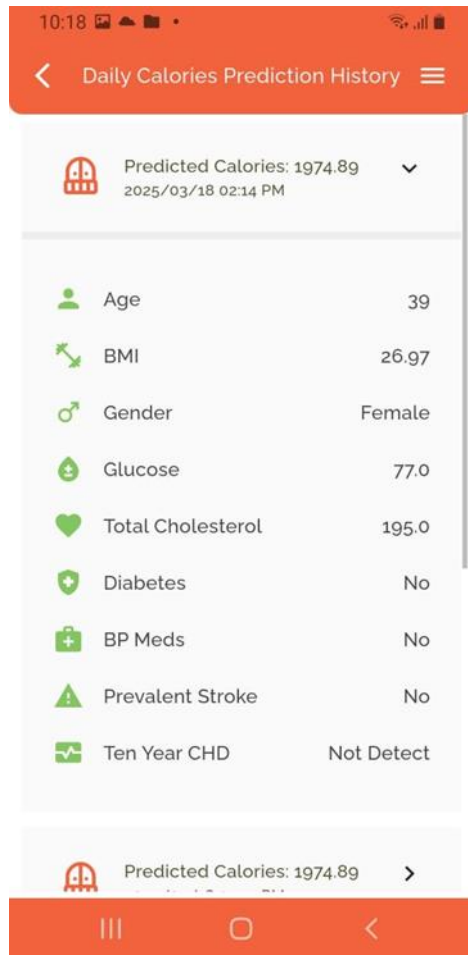


Figure 39 Diet recommendation 1

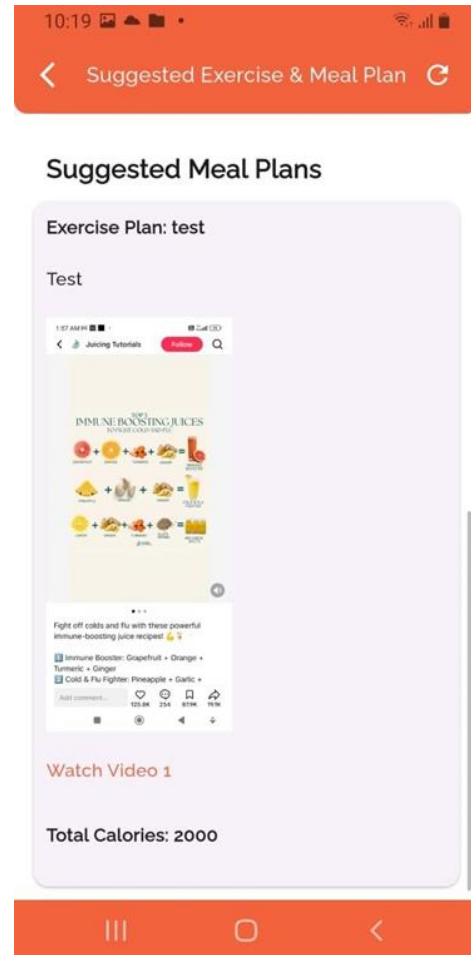


Figure 38 Diet recommendation 2

## 6.5 Calory prediction history graph view

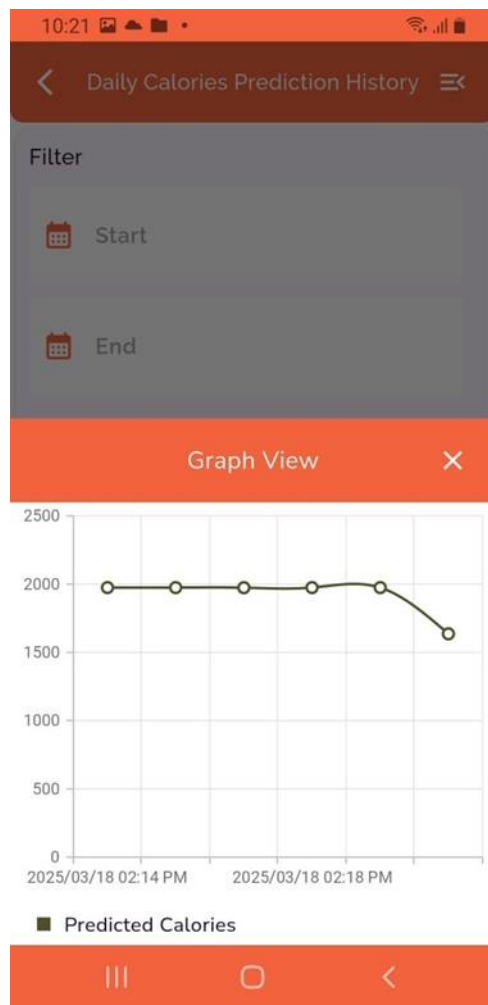


Figure 40 Daily Calories Prediction History-Graph View

## 6.6 Iot device

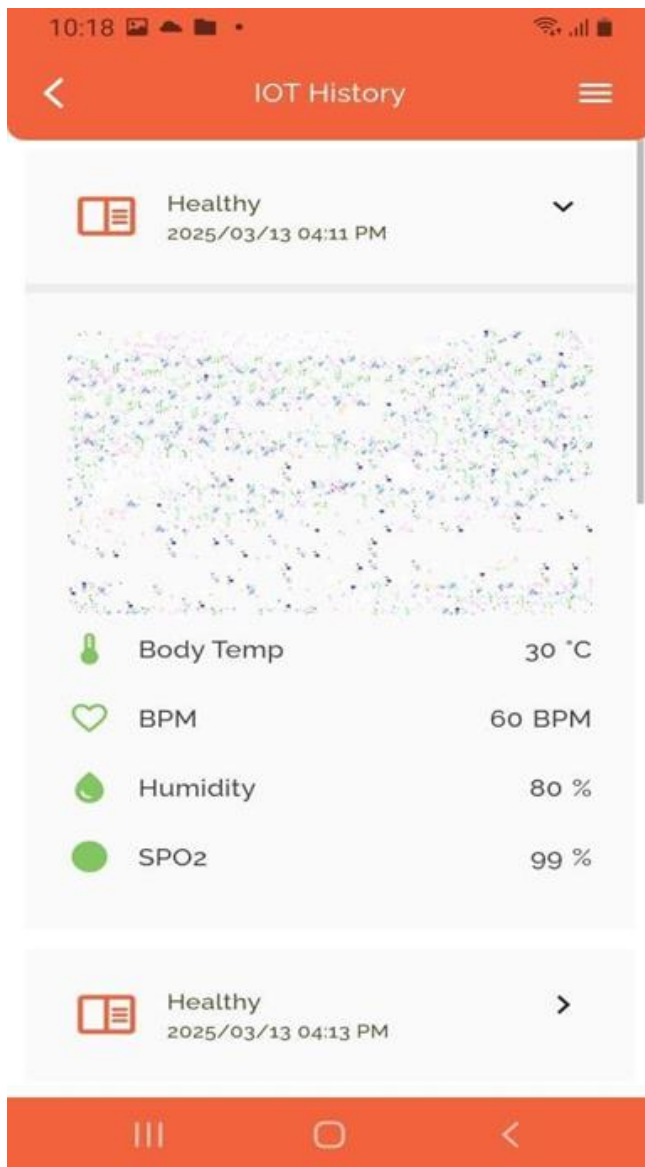


Figure 41lot history

## 6.7 Iot history graph

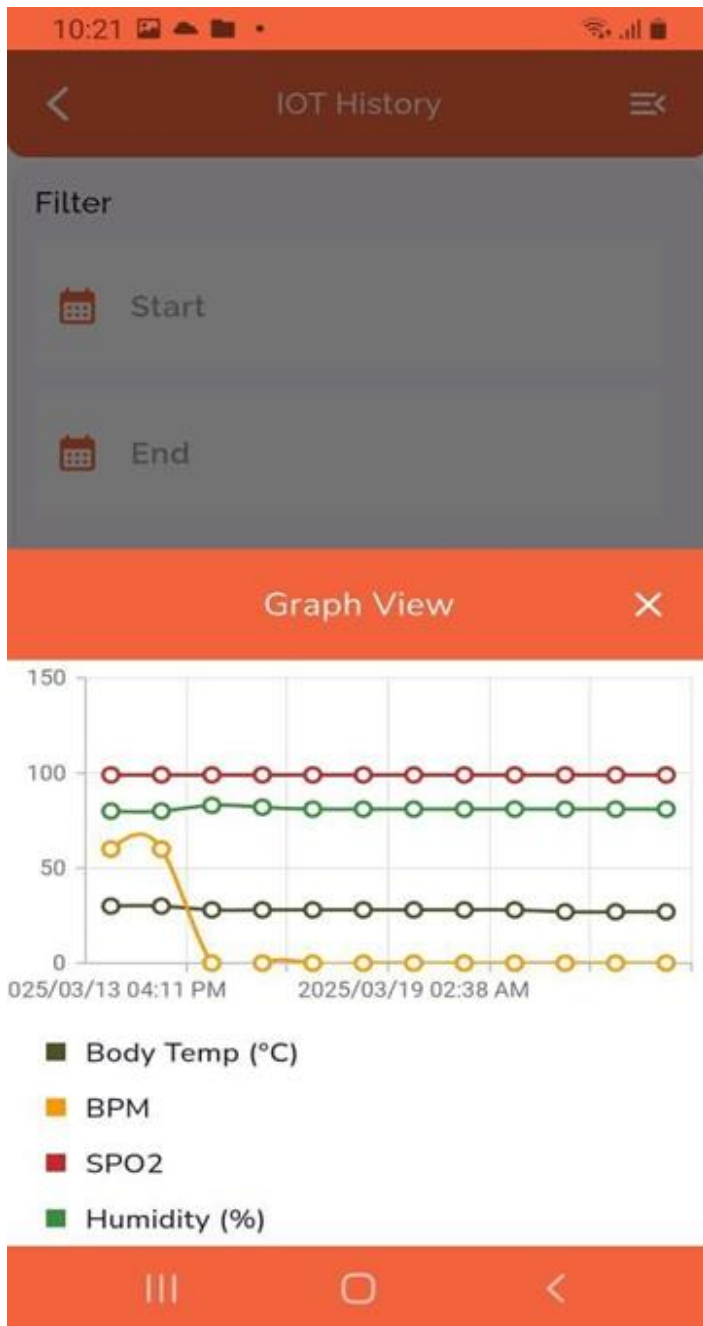


Figure 42 Iot history graph

## 7. GANTT CHART

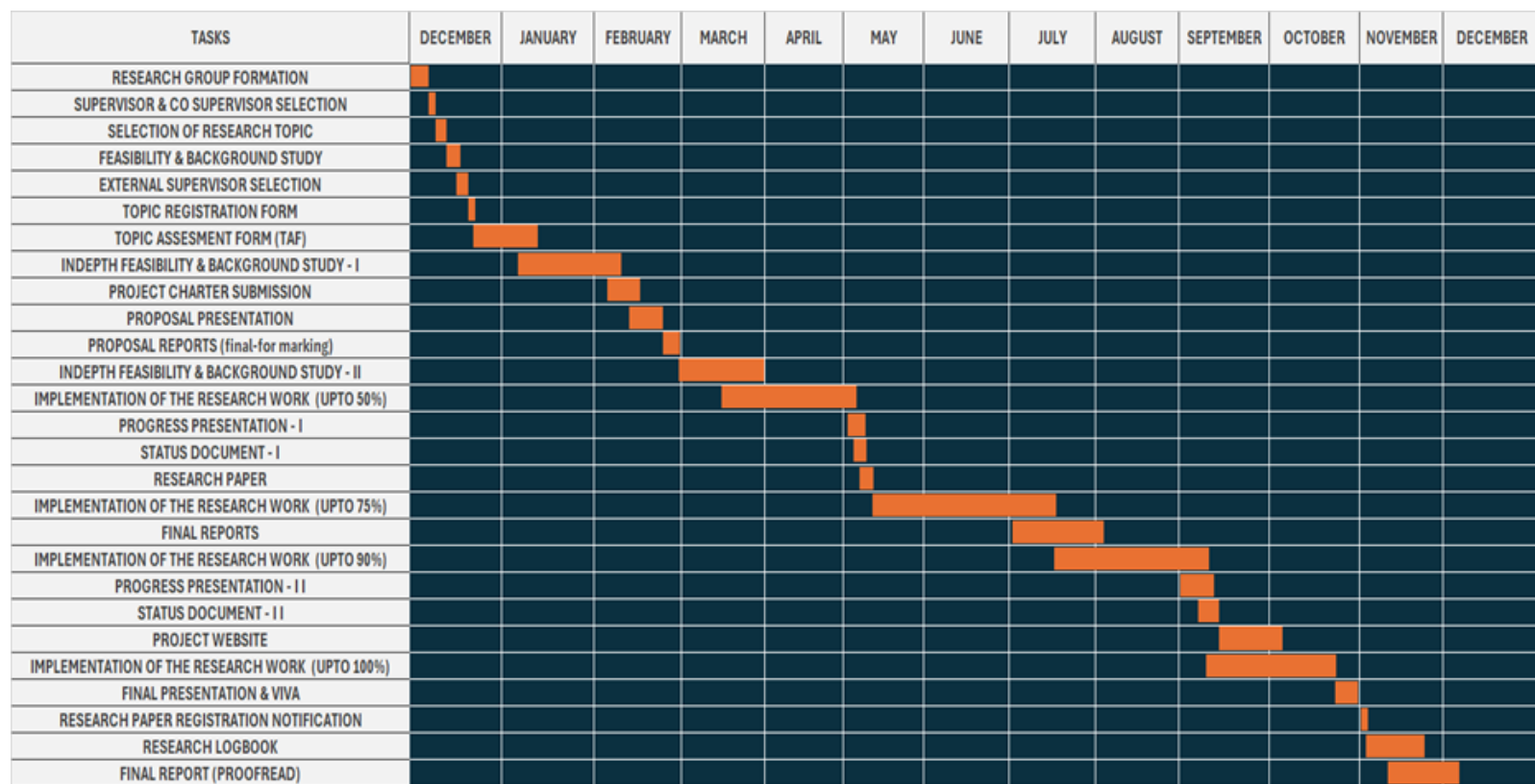


Figure 43 Gant chart



## 8. Work Breakdown Structure (WBS)

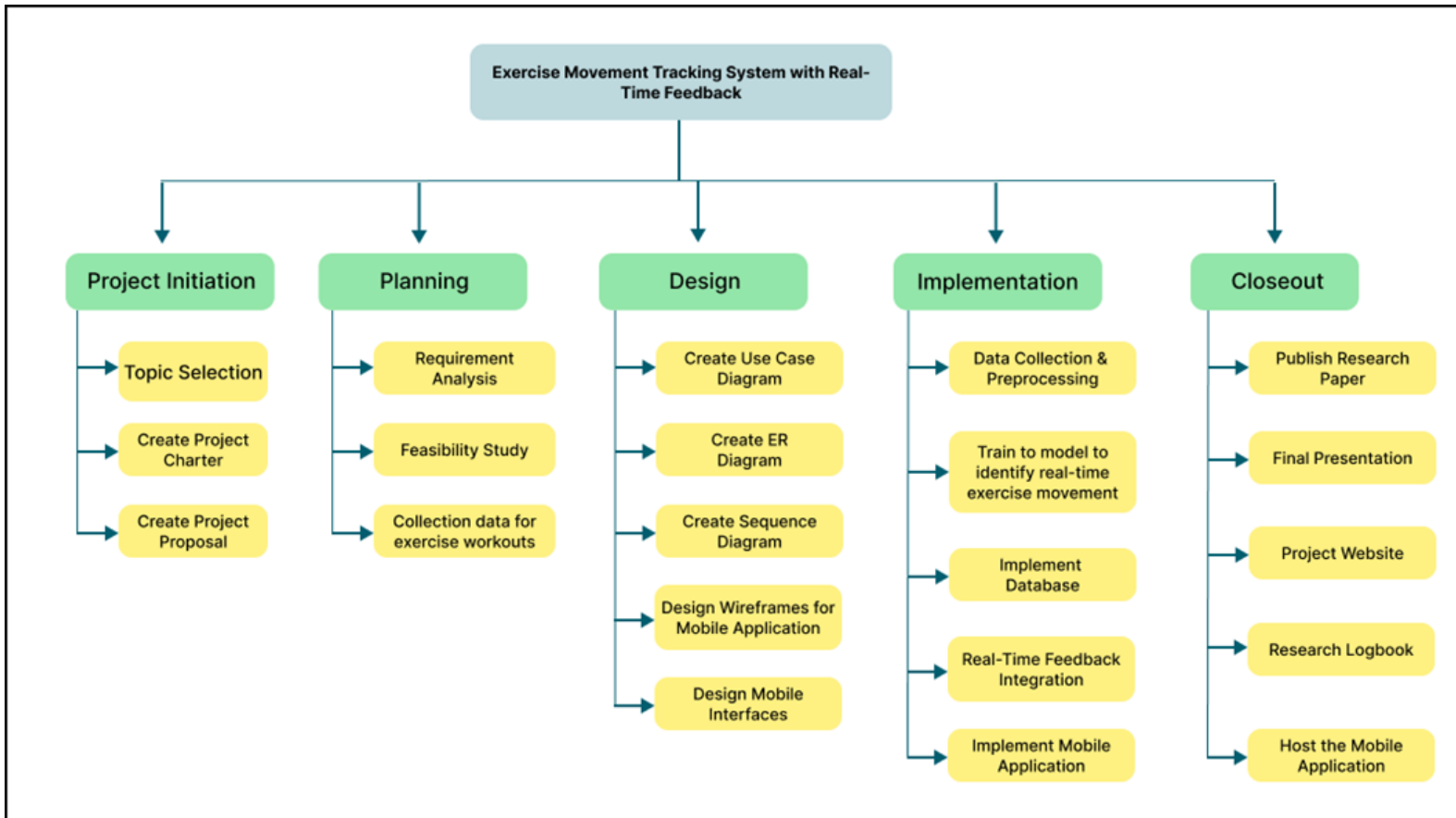


Figure 44 wbs structure

## **9.Summary of Each Student's Contribution**

Dolawatta T.Y.R (IT21101274) - CVD Risk Prediction

Dolawatta was required to design the cardiovascular disease (CVD) Risk Prediction Model for the project. His task was to obtain and preprocess data sets including age, BMI, cholesterol level, and medical history. His task was to make sure these data points were rightly used in a machine learning model to make a user a cardiovascular disease-risk level prediction. By doing so, he assisted in creating a successful model that was able to identify individuals' risk levels from data that contained their healthcare information. Such a model was significant in providing users with knowledge of their cardiovascular condition as well as providing targeted recommendations.

Samarasinghe P.D.P - IT21054372 Diet plans

Samarasinghe was in charge of creating the personalized diet plan system, as well as the CVD risk forecast. She crafted customized diet plans in relation to users in terms of their level of CVD risk, body type, and fitness orientation. She oversaw implementing algorithms that were able to modify diet plans dynamically based on input from, as well as progress from, users. By incorporating individual traits like blood level of cholesterol, blood pressure, and diabetes, she made diets not only possible but also effective in reducing cardiovascular risks. This system provided data-driven, tailored meal plans with a goal of maximizing individuals' aggregate wellbeing, while working to reduce their level of CVD risk.

Perera W.N.D.N.D - IT21312458 - IOT anomaly Detection

Perera contributed to the system using IoT-based monitoring of health and real-time feedback. His contribution was to include wearable IoT devices to track real-time parameters such as heart rate, body temperature, blood oxygen level, and level of activity. The IoT system, when paired with predictive models, provided real-time monitoring, providing detailed, personalized reports on their health to users.

## **Conclusion**

Overall, the resulting system is a holistic, integrated cardiovascular condition management system. Through integration of a cardiovascular disease (CVD) illness risk estimation model, personalized diet planning, and real-time tracking of a user's health using IoT devices, the system provides an effective means of evaluating and enhancing a user's health. An analysis by the CVD illness risk estimation model of core health indicators, including age, cholesterol level, and blood pressure, identifies individuals at risk, allowing early intervention. The personalized diet planning system also supports through targeted recommendations of what an individual should eat, taking into consideration his/her condition, goals, and more, allowing for better quality diet. Integration of IoT devices for real-time monitoring through exercise movements, heart rate, and other vital signs further ensures round-the-clock monitoring, providing positive comments and alerts to a user. The solution not only facilitates potential threats to a user's health but also provides actionable recommendations and continuous guidance, ultimately enabling a preventive, pro-bono approach to cardiovascular condition management.

## References

### References

- [1] Third Report of the National Cholesterol Education Program(NCEP), “Expert Panel on Detection, Evaluation, and Treatment of High Blood Cholesterol in Adults (Adult Treatment Panel III) final report,” *Circulation*, vol. 106, pp. 3143–3121, 2002.
- [2] A. Rajšp and I. Fister, “A Systematic Literature Review of Intelligent Data Analysis Methods for Smart Sport Training,” *Applied Sciences*, vol. 10, no. 9, p. 3013, Apr. 2020.
- [3] Ramesh, A.N.; Kambhampati, C.; Monson, J.R.; Drew, P.J. Artificial intelligence in medicine. *Ann. R. Coll. Surg. Engl.* 2004, 86, 334. [CrossRef] [PubMed]
- [4] Chowdary, K. R. et al. (2022). Early heart disease prediction using ensemble learning techniques. *J. Phys.: Conf. Ser.*, 2325, 012051.
- [5] Jagatheesaperumal, S. K. et al. (2023). An IoT-Based Framework for Personalized Health Assessment and Recommendations Using Machine Learning. *Mathematics*, 11, 2758.
- [6] Ahmed, I. (2022). A Study of Heart Disease Diagnosis Using Machine Learning and Data Mining. California State University, San Bernardino
- [7] Nayeem, M. J. et al. (2022). Prediction of Heart Disease Using Machine Learning Algorithms. *European Journal of Artificial Intelligence and Machine Learning*, 1, 3.
- [8] M. Z. M. A. N. A. P. G. R. S. S. N. a. S. K. S. Madhushika, ““SMART - Machine Learning Based Fitness Mobile Application,”,” *INTERNATIONAL JOURNAL OF ADVANCED RESEARCH AND PUBLICATIONS*, vol. 6, no. 5, pp. 78-83, 2023.
- [9] Nashif, S. et al. (2018). Heart Disease Detection by Using Machine Learning Algorithms and a Real-Time Cardiovascular Health Monitoring System. *World Journal of Engineering and Technology*, 6, 854-873.
- [10] Srinivasan, S. et al. (2023). An Active Learning Machine Technique-Based Prediction of Cardiovascular Heart Disease from UCI-Repository Database. *Scientific Reports*, 13, 13588. -to-hip circumference ratio," *npj Digital Medicine*, vol. 6, no. 1, pp. 1-10, 2023.
- [11] M. M. L. B. J. Ž. a. P. K. T. Tóth, ““Somatotypes in Sport,”,” *Acta Mechanica et Automatica*, vol. 8, no. 1, pp. 27-32, 2014.
- [12] K. Pulasinghe, "Image Processing and Machine Learning-Based Nutrition and Fitness Journaling System," *International Research Journal of Innovations in Engineering and Technology (IRJIET)*, vol. 7, no. 10, 2023.



## Appendices



### Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author: **Ranmina Dolawatta**  
Assignment title: **Research Paper Checking**  
Submission title: **Final done.docx**  
File name: **Final\_done.docx**  
File size: **7.28M**  
Page count: **76**  
Word count: **13,589**  
Character count: **78,370**  
Submission date: **30-Apr-2025 11:27PM (UTC+0530)**  
Submission ID: **2643635525**

**Personalized cardiovascular Healthcare system using machine learning and  
IoT**

24-25J-274

Final Report

T.Y.R DOLAWATTA - IT21301274

Samarasinghe P.D.P - IT23054372

Perera W.N.D.N.D - IT23131458

BSc (Hons) in Information Technology Specializing in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

February 2024

Copyright 2025 Turnitin. All rights reserved.