

**REVOLUTIONIZING REMOTE HEALTH
MONITORING: AUTONOMOUS DETECTION OF
CARDIAC ABNORMALITIES WITH CUSTOMIZED
DIETARY PLANNING**

Subasinghe Arachchige Sanuthi Vihansa	– IT21134180
Pannila Vithanage Poorna Prabhathiya Senadheera	– IT21126888
Dissanayake Mudiyanseelage Sathira Dinal Wijeratne	– IT21138386
Hilarina Melani Christy	– IT21127946

Final Report

BSc (Hons) in Information Technology Specializing in Software
Engineering

Department of Software Engineering
Sri Lanka Institute of Information Technology

August 2024

**REVOLUTIONIZING REMOTE HEALTH
MONITORING: AUTONOMOUS DETECTION OF
CARDIAC ABNORMALITIES WITH CUSTOMIZED
DIETARY PLANNING**

Subasinghe Arachchige Sanuthi Vihansa	– IT21134180
Pannila Vithanage Poorna Prabhathiya Senadheera	– IT21126888
Dissanayake Mudiyanseelage Sathira Dinal Wijeratne	– IT21138386
Hilarina Melani Christy	– IT21127946


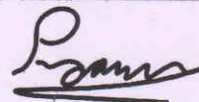


The dissertation was submitted in partial fulfilment of the requirements
for the B.Sc. (Honors) degree in Information Technology Specializing in
Software Engineering

Department of Software Engineering


August 2024

DECLARATION

We declare that this is our 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 our 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. Also, hereby grant to Sri Lanka Institute of Information Technology the non-exclusive right to reproduce and distribute our dissertation in whole or part in print, electronic or other medium. We retain the right to use this content in whole or part in future works (such as article or books).

Name	Student ID	Signature
Vihansa S.A.S	IT21134180	
Senadheera P.V.P.P	IT21126888	
Wijeratne D.M.S.D	IT21138386	
Christy H.M	IT21127946	

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.


.....
Signature of Supervisor
(Dr. Dilshan De Silva)

23/8/2024
.....
Date

ABSTRACT

Cardiovascular diseases remain the leading cause of mortality worldwide, yet access to comprehensive cardiac diagnostics and personalized care is often limited, particularly in remote or underserved areas. This study introduces an innovative solution that leverages advanced deep learning—specifically a modified Attention U-Net framework—to reconstruct a complete 12-lead electrocardiogram from a single Lead I input with an overall Pearson Correlation Coefficient of 0.883, significantly enhancing the accessibility of accurate cardiac assessments. The proposed method achieves an overall accuracy of 98.6% in diagnosing a wide range of cardiac abnormalities, including ischemic heart disease, cardiac conduction disorders, and myocardial infarctions, with a sensitivity of 97.9% and a specificity of 99.2%. Additionally, this research integrates personalized dietary planning tailored to individual patient profiles, factoring in age, gender, Body Mass Index, diabetes, cholesterol levels, and heart disease status. This dual approach not only facilitates timely and accurate diagnosis but also promotes improved cardiovascular health through customized dietary interventions. The findings underscore the potential for this comprehensive system to revolutionize remote cardiac care, making high-quality healthcare more accessible and personalized, with far-reaching implications for global health outcomes.

ACKNOWLEDGEMENT

First and foremost, we would like to express our deepest gratitude to our supervisor, Dr. Dilshan De Silva, for his unwavering support, insightful guidance, and invaluable feedback throughout the entire duration of this research project. His expertise and encouragement were pivotal to the successful completion of our undergraduate research. We would also like to extend my sincere thanks to Dr. P.V.W. Senadheera, whose generous assistance in connecting us with cardiology specialists and continuous support was crucial for the progression of our research.

Additionally, we are grateful to the cardiology doctors and medical professionals who provided their valuable time and expertise, which significantly enriched the depth of our study. Finally, we would like to thank our parents and friends for their constant encouragement. This research would not have been possible without the collective effort and support of all these individuals.

Table of Contents

DECLARATION	i
ABSTRACT.....	ii
ACKNOWLEDGEMENT	iii
LIST OF FIGURES	vi
LIST OF TABLES	ix
LIST OF APPENDICES	x
1. INTRODUCTION	1
1.1 Background Literature	1
1.2 Literature Review.....	4
1.2.1 ECG Acquisition from the Palm	4
1.2.2 Reconstructing Multiple ECG Leads	4
1.2.3 Heart Disease Diagnosis	6
1.2.4 Dietary Plan Prediction	7
1.3 Research Gap	9
1.3.1 ECG Acquisition from the Palm	9
1.3.2 Reconstructing Multiple ECG Leads	14
1.3.3 Heart Disease Diagnosis	16
1.3.4 Dietary Plan Prediction	19
1.4 Research Problem	22
1.5 Research Objectives	24
2. METHODOLOGY	27
2.1 Methodology	27
2.1.1 ECG Acquisition from the Palm	27
2.1.2 Reconstructing Multiple ECG Leads	29

2.1.3	Heart Disease Diagnosis	31
2.1.4	Dietary Plan Prediction	37
2.2	Commercialization Aspects of The Product	48
2.3	Testing & Implementation	50
2.3.1	ECG Acquisition from the Palm	50
2.3.2	Reconstructing Multiple ECG Leads	55
2.3.3	Heart Disease Diagnosis	63
2.3.4	Dietary Plan Prediction	72
3.	Results & Discussion	81
3.1	Results.....	81
3.1.1	ECG Acquisition from the Palm	81
3.1.2	Reconstructing Multiple ECG Leads	84
3.1.3	Heart Disease Diagnosis	93
3.1.4	Dietary Plan Prediction	100
3.2	Research Findings	107
3.3	Discussion	109
4.	SUMMARY OF EACH STUDENT’S CONTRIBUTION	113
4.1	IT21134180 – Vihansa S.A.S	113
4.2	IT21126888 – Senadheera P.V.P.P	113
4.3	IT21138386 – Wijeratne D.M.S.D	114
4.4	IT21127946 – Christy H.M.....	114
5.	CONCLUSION.....	116
	REFERENCES	118
	APPENDICES	124

LIST OF FIGURES

Figure 2.1.1.1: Palm Analysis and ECG generation System Architecture	27
Figure 2.1.2.1: Reconstruction of 12-lead ECG from a single lead component diagram	30
Figure 2.1.3.1: PTBXL dataset	31
Figure 2.1.3.2: Training and Evaluation	34
Figure 2.1.3.3: Early stopping callback method	35
Figure 2.1.4.1: BMI Calculation	41
Figure 2.1.4.2: Medical Report Analysis	42
Figure 2.1.4.3: Component Diagram	45
Figure 2.3.1.1: Palm ECG acquisition system	51
Figure 2.3.1.2: Unfiltered lead I ECG signal acquired from palms	52
Figure 2.3.1.3: Filters implementation for palm acquired ECG signals	53
Figure 2.3.1.4: Lead off detection.....	54
Figure 2.3.2.1: Application of the Butterworth filters in code level.....	56
Figure 2.3.2.2: Application of Savitzky-Golay filter in code level	56
Figure 2.3.2.3: Application of Min-Max normalization in code level.....	57
Figure 2.3.2.4: Code segment of splitting the dataset.....	57
Figure 2.3.2.5: CardioFit's Modified Attention U-Net model architecture.....	58
Figure 2.3.2.6: Application of the pure 1D convolution layer in code level	58
Figure 2.3.2.7: Application of the 1D convolution layer with batch normalization in code level	58
Figure 2.3.2.8: Application of 1D transposed convolution layer in code level	59
Figure 2.3.2.9: Implementation of the Attention Gate mechanism in code level	60
Figure 2.3.2.10: Definition of batch size and epochs	61
Figure 2.3.3.1: Import libraries	64
Figure 2.3.3.2: Handling of missing values	65
Figure 2.3.3.3: Signal Normalization.....	66
Figure 2.3.3.4: Noise reduction mechanism	66
Figure 2.3.3.5: Code of segmentation and labeling	67

Figure 2.3.3.6: Code of Data Preparation	68
Figure 2.3.3.7: Routing Configuration.....	69
Figure 2.3.3.8: Model Loading and Prediction	69
Figure 2.3.3.9: Model Stored in Firebase.....	71
Figure 2.3.4.1: Snapshot of the dataset created	75
Figure 2.3.4.2: Final Evaluation Random and Fasting blood glucose report adapter	78
Figure 2.3.4.3: Final Evaluation Lipid Profile Report adapter	78
Figure 3.1.1.1: Filtered lead I ECG signal acquired from palms	81
Figure 3.1.1.2: Frequency spectrum comparison.....	82
Figure 3.1.2.1: Error metrics of the trained model: MSE and MAE.....	84
Figure 3.1.2.2: Evaluating Pearson correlation coefficient.....	85
Figure 3.1.2.3: Evaluating R^2 value	85
Figure 3.1.2.4: Architecture of model 1	86
Figure 3.1.2.5: Pearson correlation coefficient of model 1.....	87
Figure 3.1.2.6: R^2 value of model 1	87
Figure 3.1.2.7: Architecture of model 2.....	88
Figure 3.1.2.8: Pearson correlation coefficient of model 2.....	88
Figure 3.1.2.9: R^2 value of model 2	88
Figure 3.1.2.10: Architecture of model 4.....	89
Figure 3.1.2.11: Pearson correlation coefficient of model 4.....	90
Figure 3.1.2.12: R^2 value of model 4	90
Figure 3.1.2.13: Architecture of model 5.....	91
Figure 3.1.2.14: Pearson correlation coefficient of model 5.....	91
Figure 3.1.2.15: R^2 value of Model 5.....	91
Figure 3.1.3.1: Training Accuracy of Baseline Model	96
Figure 3.1.3.2: Training Loss of Baseline Model	97
Figure 3.1.3.3: Confusion Matrix of Baseline Model.....	98
Figure 3.1.4.1: Flask Server Acts as a Middleware Between AWS and Flutter App	101
Figure 3.1.4.2: Diet Home Page.....	104
Figure 3.1.4.3: Medicine Report Upload Screen	104
Figure 3.1.4.4: Medicine Report Analysis Screen	105

Figure 3.1.4.5: Dietary Advice Proposing Screen	105
Figure 3.1.4.6: Medicine Reminder Home Page.....	106

LIST OF TABLES

Table 1: Comparison of Key Features Between the Proposed System and Existing Studies (ECG Acquisition from the Palm Component).....	14
Table 2: Comparison of Former Studies (Reconstructing Multiple ECG Leads Component).....	15
Table 3: Comparison of Previous Work and Proposed System (Heart Disease Diagnosis Component).....	17
Table 4: Summary of Research Gaps (Dietary Plan Prediction Component).....	20
Table 5: Technologies, architectures and algorithms used	30
Table 6: BMI Categorization	38
Table 7: Method of Obtaining Essential Factors	39
Table 8: Medical Report Analysis Data	73
Table 9: Factors Considered	74
Table 10: Report Analysis Tools used	76
Table 11: Comparison of metrics between Modified Attention U-Net and proposed model.....	85
Table 12: Comparison of the overall performance metrics.....	85
Table 13: Summary of Performance Metrics Across Different Model Architectures	92
Table 14: Tools and Technologies Used.....	102

LIST OF APPENDICES

Appendix A: Plagiarism report	124
-------------------------------------	-----

1. INTRODUCTION

1.1 Background Literature

Cardiovascular diseases (CVDs), often referred to as cardiac diseases, have emerged as the leading cause of mortality worldwide. The global burden of cardiac diseases highlights the urgent need for innovative and effective diagnostic tools and treatment strategies. Extensive research has demonstrated the significant role of diet in cardiovascular health, establishing a clear link between dietary habits and the incidence of CVDs [1]. Despite these findings, existing healthcare systems often struggle to provide accessible and personalized care, particularly in remote and underserved areas. This research seeks to address these challenges by integrating advanced AI-based diagnostics with personalized dietary advice, thereby enhancing the theoretical grounding in the field of remote health monitoring and AI-based healthcare solutions.

In the realm of cardiac diagnostics, the 12-Lead electrocardiogram (ECG) is widely recognized as the gold standard. This non-invasive test records the electrical activity of the heart and provides valuable insights into its functioning and health. The 12-Lead ECG captures a comprehensive electrical snapshot of the heart from twelve different angles, thereby enabling the detection of a wide array of cardiac abnormalities.

However, the acquisition of a 12-Lead ECG typically necessitates a visit to a hospital or a healthcare facility. This requirement poses significant challenges, particularly for patients residing in remote areas or those with mobility issues. Furthermore, the process of obtaining a 12-Lead ECG can be time-consuming and may require the expertise of a trained healthcare professional.

While there exist alternative ECG devices that offer fewer leads, these often fall short in terms of accuracy and reliability. A reduced number of leads may not capture the complete electrical activity of the heart, potentially leading to missed or inaccurate

diagnoses. As such, the 12-Lead ECG remains the most trusted and reliable tool for cardiac assessment [2].

To address these challenges, this study proposes a novel approach to reconstruct a complete 12-Lead ECG from a single Lead I ECG, which can be captured through the palms using portable devices. This approach leverages convolutional neural networks (CNNs) to extrapolate the information from the Lead I ECG and generate the remaining 11 leads. By bridging the gap between traditional, facility-based ECG acquisition and the need for accessible, reliable cardiac diagnostics, this method has the potential to make comprehensive cardiac assessment more convenient without compromising accuracy. The theoretical grounding of this approach is supported by advancements in AI-based diagnostics and remote health monitoring, which provide a strong foundation for innovation and its potential impact on global healthcare.

In addition to improving cardiac diagnostics, this research also addresses the significant gap in personalized dietary interventions for cardiovascular health. The study proposes an integrated system that combines ECG reconstruction with personalized dietary planning based on key factors such as age, gender, BMI, and health conditions like heart disease, diabetes, and cholesterol levels. By tailoring dietary recommendations to individual needs, this system aims to prevent and manage CVDs more effectively. The conceptual model of combining ECG reconstruction with dietary planning underscores the importance of a holistic approach to healthcare, where both diagnostics and lifestyle interventions are aligned to improve patient outcomes.

This proposed solution addresses real-world problems, particularly the inaccessibility of comprehensive cardiac diagnostics in remote areas and the lack of tailored dietary interventions. Patients in these regions often face challenges related to the availability, cost, and limitations of current ECG systems and dietary advice tools. By proposing a system that overcomes these barriers, this research contributes to improving global healthcare accessibility, especially in rural areas. The potential global impact of this technology is significant, as it could lead to better healthcare outcomes and reduced mortality rates from CVDs if adopted widely.

Furthermore, this paper fills gaps in the current literature by addressing the limitations of existing cardiac monitoring systems and personalized care solutions. The integration of AI-based ECG reconstruction with personalized dietary planning represents a novel approach that not only enhances the accuracy and accessibility of cardiac diagnostics but also provides a comprehensive, individualized healthcare solution. This research, therefore, makes a critical contribution to the ongoing efforts to improve global healthcare and reduce the burden of cardiovascular diseases.

This research builds on key theories in neural networks, signal processing, and personalized medicine to propose a novel approach for ECG acquisition and reconstruction. Foundational concepts like the Universal Approximation Theorem (UAT) and CNNs demonstrate the ability of neural networks to model complex relationships. In signal processing, principles such as Fourier Analysis support accurate signal reconstruction. In personalized medicine, the concept of Stratified Medicine emphasizes tailored healthcare interventions, which parallels our approach in customizing ECG signal processing. This study uses CNNs within a modified Attention U-Net framework to reconstruct a full 12-Lead ECG from a single Lead I signal obtained from the palms. This method extends the UAT by effectively mapping the relationship between a single lead and a complete set of ECG leads. By integrating AI with healthcare, particularly through CNNs, we redefine how comprehensive ECG data can be generated from minimal inputs, transforming cardiac diagnostics, especially in remote areas. This interdisciplinary approach strengthens both the theoretical foundation and practical application of our framework.

The paper is organized into several key sections. Section II provides an overview of existing research related to ECG acquisition methods, multi-lead ECG reconstruction, heart disease diagnosis, and dietary advice, highlighting the gaps this study aims to address. Section III outlines the design and implementation of the proposed system, including the hardware setup for ECG acquisition, the deep learning model used for multi lead ECG reconstruction, the heart disease diagnosis approach, and the development of the personalized dietary advice system. Section IV summarizes the key components of the research, its limitations and suggestions for future work.

1.2 Literature Review

1.2.1 ECG Acquisition from the Palm

Lourenço, Fred, and Silva [3] present an experimental apparatus designed for palm-based ECG biometrics, which aligns with traditional biometric systems in terms of usability and non-intrusiveness. Their device integrates electrodes and signal conditioning circuitry on a surface where the user can rest their palm, eliminating the need for conductive gels or access to more intimate body parts. The authors utilized dry Ag/AgCl electrodes and a custom signal conditioning circuit requiring only two contact points. This approach, leveraging a virtual ground, simplifies the traditional three-lead ECG setup to a more user-friendly two-lead configuration.

Adil, Baruah and Roy [4] proposed a real-time ECG acquisition system utilizing only two electrodes, addressing common issues such as impedance matching, power-line noise, and muscle activity interference. Their design incorporates an AD8232 analog front end for amplification and an Atmega328p microcontroller for digital signal conversion and transmission. The system effectively mitigates noise and maintains signal integrity through a combination of hardware and software filtering techniques. The authors highlight the use of a second-order low-pass digital filter and a high-pass filter to eliminate baseline drift and power-line interference. Their experimental setup successfully acquired ECG signals from the palm with minimal noise, demonstrating the system's robustness and potential for integration into portable and IoT-based health monitoring systems.

1.2.2 Reconstructing Multiple ECG Leads

Research indicates the potential for reconstructing accurate ECGs with fewer leads [5], [6], [7], [8]. Kapfo et al. [7], for instance, combined a discrete wavelet transform (DWT) with Long Short-Term Memory (LSTM) networks in a patient-tailored system

to reconstruct standard 12-lead ECGs from just three input leads, demonstrating strong correlation and low error in their results. Similar success with LSTMs was reported by Jangjay et al. [8]. Nonetheless, practical hurdles exist with these methods since they still necessitate electrodes across the chest and limbs. The wearables industry demands solutions that further streamline data collection.

To address this challenge, Gundlapalle and Acharyya [9] introduced a novel technique utilizing CNNs, LSTMs, and MLPs to reconstruct a full 12-lead ECG from the singular input of lead II. They achieved noteworthy correlation and regression coefficients. Yoon et al. [10] explored the use of a generative adversarial network (GAN) for single-lead reconstruction, employing a U-net and patch discriminator architecture to attain favorable signal quality. According to the researchers' knowledge, these represent the predominant studies dedicated to single-lead ECG reconstruction.

However, significant limitations exist in both works. The use of shorter segmented signals (1-2.5 seconds) potentially simplifies reconstruction but undermines practicality, as traditional ECGs necessitate 10-second recordings [11]. Seamlessly recombining these segments for real-world use poses an unresolved complexity. Additionally, the deep learning algorithms employed are both computationally intensive (particularly Yoon et al.'s GAN [10], [12]) and can introduce instability. Furthermore, Gundlapalle and Acharyya's potential data leakage in their train/test split raises concerns about the generalizability of their findings.

Garg, Venkataramani and Priyakumar [13] discusses an approach to reconstruct multi-lead ECGs from a single lead using a modified Attention U-Net framework. It addresses the gap between reduced lead set data and standard 12-lead ECG interpretation. The model's ability to reproduce 11 leads from lead II with high Pearson correlation, low mean square error, and satisfactory R squared values is emphasized, showcasing its potential for real-life application in disease classification tasks.

1.2.3 Heart Disease Diagnosis

CVDs continue to be the largest cause of worldwide deaths, highlighting the importance of precise timely diagnosis. The standard 12-lead ECG is used by clinicians to diagnose a wide variety of cardiac problems, hence, providing electrical information about the heart's electrical activity.

The source of [14], indicates that the 12 lead ECG signals were directly input to a one-dimensional neural network in combination with a Long Short-Term Memory artificial recurrent neural network, with the intention of diagnosing cardiac abnormalities with the aid of deep learning techniques. The temporal features are studied by this model, with the intention of proving the diagnosis of results. Furthermore, the CNN-BiLSTM classification method for the diagnosis of diseases, while considering a few evaluation metrics with the intention of assessing the performance. However, this source of work uses a dataset of 6,877 recordings of ECG signals obtained from various time duration, for which an additional processing was done, to bring all the ECG raw data signal to a fixed rate of 15 seconds overall.

In addition to that, a previous research work [15], where it contributed to account of six cardiac abnormalities being predicted in total, comprising of the 1st degree AV block, Right bundle branch block, Left bundle branch block, Sinus bradycardia, Atrial fibrillation, Sinus Tachycardia. A CNN was used, that adapted itself to uni-dimensional signals. A total of four residual blocks with two convolutional layers per block were used. The respective outcome of results of each of these convolutional layers was then used as a source of input into a rectified linear unit (ReLU).

Furthermore, another study [16] reveals the proposal of a deep neural network that considers an input of raw 12 lead ECG signals of 30 seconds each, where extraction of deep features are done with the intention of predicting or rather diagnosing diseases of Sinus Rhythm, Atrial fibrillation, Right Bundle Branch block, Premature atrial contraction (PAC), Premature Ventricular contraction (PVC), ST-segment depression (STD) and ST-segment elevation (STE). Extraction of the required features needed for the diagnosing segment was done via two sections in this study, where statistical

features were extracted from the ECG signals fed. This ensured that the variance, standard deviation and the related values were extracted. While the second form of extraction was done using the Shannon entropy of signal processing.

Moreover, the research study published by Bo Heden [17] aimed at diagnosing acute myocardial infarction with the aid of the 12 lead ECG used as inputs. The neural network used here comprises of one input layer, a hidden layer and output layer. Where the hidden layer comprising of 15 neurons provided an encoded unit of output as to whether the ECG was classified as Myocardial Infarction or not. Since it only detects if a specific patient is diagnosed with Myocardial Infarction or not, its potential scope of figuring out if patients have been diagnosed with potential other cardiac diseases has been restricted, thereby being of limited use. Hence, it is important that there is one solution that can diagnose plenty of cardiac diseases accurately, where all niche aspects of cardiac diseases are also considered.

1.2.4 Dietary Plan Prediction

Consuming a healthy diet throughout the life-course helps to prevent malnutrition in all its forms as well as a range of noncommunicable diseases (NCDs) and conditions. However, increased production of processed foods, rapid urbanization and changing lifestyles have led to a shift in dietary patterns. People are now consuming more foods high in energy, fats, free sugars and salt/sodium, and many people do not eat enough fruit, vegetables, and other dietary fiber such as whole grains [18].

As a result, an epidemic in the cardiovascular diseases could be observed in a significant manner. According to statistics diet related risks affected for nearly ten million deaths worldwide [19]. It is estimated over 80% of deaths from CVD take place in low-income and middle-income countries [20]. Therefore, it is important to develop effective affordable strategies for the prevention and treatment of CVDs.

The increasing focus on health in the modern era has led to a growing trend of individuals seeking to improve their dietary habits. This often involves adopting

healthier practices like reducing sugar and carbohydrate intake or increasing physical activity. However, adhering to rigid, unguided dietary changes can be detrimental to an individual's well-being. For instance, while reducing sugar consumption is generally beneficial, eliminating it can disrupt metabolic processes. Therefore, individualized, and well-informed approaches to healthy eating are crucial [21].

To cater this demand of increased interest in healthy living numerous dietary plans have raised in the present. Anyhow, the effectiveness and continuity of such emerged diet solutions are still at a risk. Most of the individuals who follow a specific diet plan immersed as a trend is in lack of the knowledge on the actual calorie intake that is required by one's own self. As a result, many other health problems such as vitamin deficiencies, lack of energy to perform the daily metabolic activities arises.

Most dietary plans which is available in the society fails as they do not capture overall requirement. A proper diet plan should consider the individuals, daily calorie intake, daily calorie burn out amount, BMI, age, gender, special medical conditions that the user has example diabetes or cholesterol.

The prominent reason why this research component is aimed to propose a dietary advice but not a dietary plan is because of the above-mentioned reason. In this research component the researchers aim consider the patients age, BMI, gender, medical report analysis (To get inputs as to whether the user is having high cholesterol or sugar) and cardiac health and proposed a customized dietary advice. Which could aid the user to know the right amount of carbohydrates, proteins, fats and oils that is necessary for the individual to perform his metabolic functions.

1.3 Research Gap

1.3.1 ECG Acquisition from the Palm

The area of face-based ECG acquisition is an emerging field with limited research available. One notable study in this area is detailed by Bin Li et al. [22]. This research investigates the feasibility of generating ECG signals from facial videos using remote rPPG. The study employed a combination of explicit and implicit methods to reconstruct ECG signals from the rPPG data extracted from facial recordings. The researchers used deep learning models to analyze the facial video data, applying algorithms to correlate the facial pulse signals with ECG waveforms.

In this study, researchers developed a method to reconstruct ECG signals using a combination of advanced image processing and machine learning techniques. The facial recordings were processed to separate relevant physiological features from background noise, after which these features were analyzed and refined to optimize the quality of the physiological data. The final step involved using a Generative Adversarial Network (GAN) to generate realistic ECG signals from the processed facial data. The study demonstrated a moderate to strong correlation between the predicted and actual ECG signals but did not provide details on the number and types of ECG leads generated, leaving some questions about its practical applicability.

The results from this study demonstrated that while it is possible to reconstruct ECG signals from facial videos, the reliability and accuracy of these signals are significantly influenced by various factors. The study highlighted those environmental conditions, such as lighting and camera quality, as well as subject movement, greatly affect the quality of the rPPG signals. These limitations suggest that while the approach holds potential, it is still in the early stages of development and may not yet be suitable for clinical applications.

There was another notable study done by Zhu et al [23] which worked on ECG reconstruction via photoplethysmography (PPG), where researchers achieved a high

accuracy of 0.98 averaged correlation in reconstructing ECG signals from PPG data using a discrete cosine transform (DCT) mapping approach. This inspired me to obtain a high-quality rPPG signal from facial videos, similar to PPG signals, so that it would be possible to generate accurate ECG signals.

Given these findings, there was initially a motivation to explore face-based ECG acquisition in this research. The potential for a non-invasive, accessible monitoring technology that could utilize everyday devices like smartphones or tablets was particularly appealing. However, the research on face-based ECG acquisition remains limited, with significant gaps in practical and clinical applicability. The scarcity of comprehensive studies in this area underscores the challenges of developing a reliable method for generating ECG signals from facial analysis. Despite the initial promise, the field is still in its infancy, and more research is needed to address the technical limitations and ensure that the method can be used reliably in real-world scenarios.

Given the challenges encountered with facial analysis for ECG acquisition, it became evident that alternative non-invasive methods were necessary. Facial analysis, while innovative, faced significant obstacles such as unreliable signal quality due to environmental factors like lighting and motion, as well as difficulties in obtaining consistent rPPG signals. These challenges limited the practicality of face-based methods, making it difficult to achieve the accuracy required for reliable ECG monitoring.

This led to the exploration of palm-based ECG acquisition as a viable alternative. To my knowledge, there are only two studies done that explore obtaining ECG signals from the palms, including the studies done by Lourenço et al. [3] and Adil et al. [4]. These studies highlight the potential of using palm-based systems to acquire ECG signals with fewer electrodes, making the process more comfortable and less intrusive for users. The device described by Lourenço et al. utilized dry Ag/AgCl (silver/silver chloride) electrodes to acquire ECG signals from the palms. Notably, this approach was primarily intended for biometric identification purposes rather than for medical diagnostics. The study demonstrated that ECG signals captured from the palms could be used as a unique biometric identifier, providing an additional layer of security in

biometric systems. However, the focus on biometrics means that the system was not specifically designed or validated for clinical use, which introduces limitations when considering its application for medical ECG monitoring.

The study done by Adil et al. [4] further explores the use of palm-based systems by proposing a real-time ECG acquisition system that uses just two electrodes. This system addresses common issues such as impedance matching, power-line noise, and muscle activity interference, which are often encountered in traditional ECG setups. The use of an analog front end (AFE) such as the AD8232 and a microcontroller (ATmega328p) allows for the efficient processing and digitization of the ECG signals, making the system portable and suitable for integration into various health monitoring applications. One of the key advantages of palm-based ECG acquisition is its accessibility and simplicity. By placing electrodes on the palms, this method avoids the discomfort and inconvenience associated with traditional electrode placements on the chest and limbs. The use of portable devices like the AD8232 and Arduino UNO Rev3 further enhances the system's usability, allowing for easy integration into various applications, including remote health monitoring and IoT-based systems.

This approach aligns with the broader goal of making cardiac monitoring more accessible, particularly in remote or resource-limited settings where traditional ECG equipment may not be available. By simplifying the process and reducing the number of electrodes needed, palm-based ECG systems offer a practical solution for continuous and unobtrusive health monitoring.

Despite its potential, several research gaps remain in the development of palm-based ECG systems. First, there is a need for more extensive validation studies that compare the accuracy of palm-based ECG acquisition with traditional 12-lead systems. These studies should include diverse populations and a variety of clinical scenarios to ensure the reliability and generalizability of the results. Second, challenges in miniaturizing the technology for truly portable and user-friendly applications need to be addressed. While current systems demonstrate feasibility, further advancements in hardware design and signal processing are required to create devices that are both compact and robust enough for everyday use.

Finally, issues related to signal interference and consistency must be thoroughly investigated. Factors such as skin impedance, electrode placement variability, and environmental conditions can significantly impact the quality of the ECG signals obtained from the palms. Research focused on optimizing electrode materials, improving noise reduction techniques will be crucial in overcoming these challenges.

While palm-based ECG acquisition offers a promising alternative to traditional methods by simplifying the process of obtaining lead I from the palms, its true potential lies in its ability to be integrated with advanced techniques for reconstructing a full 12-lead ECG from this single lead. This integration could significantly enhance the practicality and effectiveness of palm-based ECG systems, making comprehensive cardiac monitoring more accessible and user-friendly. However, reconstructing a 12-lead ECG from fewer leads, particularly from just a single lead like lead I, presents its own set of challenges.

The current state of research in 12-lead ECG reconstruction has made significant strides. Techniques using Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs), and Generative Adversarial Networks (GANs) have shown promising results. For example, Kapfo et al. [7] proposed networks to predict the standard 12-lead ECG from just three input leads. Their study reported an impressive average correlation coefficient of 0.98, suggesting that their model effectively captured the general trends of the ECG signals. However, their work also had significant drawbacks, particularly the high root mean squared error (RMSE) of 78, indicating substantial deviations between the reconstructed and actual signals, which raises concerns about the accuracy of their reconstructions despite the high correlation.

Another study conducted by Jangjay et al. [8] contributed to the field by using LSTM networks to achieve an average correlation coefficient of 0.95 for their reconstructed signals. This was a solid performance, showing that their model could effectively reconstruct ECG signals with high fidelity. However, like Kapfo et al., Jangjay et al.'s work was dependent on multiple input leads, which limits its practical applicability.

Yoon et al. [10] employed a GAN that utilized a U-Net as the generator and a patch discriminator as the discriminator to reconstruct the 12-lead ECG from a single lead. The study achieved a mean Frechet Distance (FD) score of 11.321 and a mean squared error (MSE) of 0.038, indicating moderate success in producing accurate reconstructions. However, their approach was complicated by the inherent instability and high computational demands of GAN training. Additionally, they relied on 11 separate models, each responsible for reconstructing one of the 11 leads, and further limited their approach by segmenting the ECG signals to only 2.5 seconds. This segmentation is a significant drawback, as comprehensive and accurate interpretation, especially in a clinical setting, generally requires a 10-second ECG recording [11].

Another study carried out by Gundlapalle et al. [9] proposed a method that combined CNNs, LSTMs, and multi-layer perceptrons (MLPs) to reconstruct the 12-lead ECG from just one lead, reporting a correlation coefficient of 0.973 and a regression coefficient of 0.959. Their approach was innovative, but the complexity of the model made it difficult to implement and like Yoon et al., they also used 11 different models, each to reconstruct a single lead. Additionally, the model segments the ECG signals into 1-second intervals which is a drawback as previously mentioned.

Lastly a study carried out by Garg et al. [13] presented a novel approach using a modified Attention U-Net framework for single-lead to multi-lead ECG reconstruction. Their model used only lead II as input to reconstruct the entire 12-lead ECG, achieving a Pearson correlation coefficient of 0.805, an MSE of 0.0122, and an R-squared value of 0.639. While their correlation coefficient was slightly lower than some previous studies, their model excelled in maintaining low reconstruction errors and practical applicability. Their approach had significant advantages, such as the ability to reconstruct a full 10-second ECG signal in one pass and using a single combined model for all 11 leads. This reduces the complexity, and computational resources compared to models that reconstruct each lead individually. Furthermore, they demonstrated that their reconstructed signals were effective in real-world scenarios by maintaining comparable performance to original signals in a cardiovascular disease classification task. The original signals achieved an accuracy of 45.6% and an AUROC (Area Under

the Receiver Operating Characteristic Curve) of 0.810, while the reconstructed signals achieved an accuracy of 42.6% and an AUROC of 0.752.

The comparison in Table 1 highlights the distinct capabilities of the proposed system in relation to existing studies. While Adil et al. [4] focused on acquiring a single ECG lead from the palm and Garg et al. [13] developed a method for reconstructing a 12-lead ECG from a single lead, our system integrates these functionalities into a unified platform. It not only acquires a palm-based ECG signal but also reconstructs the full 12-lead ECG at a higher accuracy, displays it on a smart tablet. This comparison underscores the comprehensive and practical nature of our proposed solution, aimed at enhancing accessibility and usability in cardiac health monitoring.

Table 1: Comparison of Key Features Between the Proposed System and Existing Studies (ECG Acquisition from the Palm Component)

Reference	Ability to acquire ECG	Ability to Obtain 12-Lead ECG	System for ECG Visualization
Adil et al. [4] (Two-Electrode Palm ECG)	✓	X	X
Garg et al. [13] (12 Lead ECG reconstruction)	X	X	X
Proposed System	✓	✓	✓

1.3.2 Reconstructing Multiple ECG Leads

In this research component, we identified 3 unique key contributions to find the research gaps. By comparing with the recent studies, a summary of key contributions from recent studies and this study are provided in Table 2.

Table 2: Comparison of Former Studies (Reconstructing Multiple ECG Leads Component)

Application Reference	Use Lead I as the Initial Signal (Input)	Reconstruct ECG signals with a duration of 10 seconds or longer	Obtaining Standard 12-Lead ECG
Reconstruction of 12-lead ECG using a DWT with LSTM network by Kapfo <i>et al.</i> [7]	X	✓	✓
Reconstruction of 12-lead ECG using LSTM network by Jangjay <i>et al.</i> [8]	X	✓	✓
Reconstruction of 12-lead ECG using CNNs, LSTMs, and MLPs by Gundlapalle and Acharyya [9]	X	X	✓
Reconstruction of 12-lead ECG using GAN, a U-net and patch discriminator architecture by Yoon <i>et al.</i> [10]	✓	X	✓
Reconstruction of 12-lead ECG using a modified Attention U-Net framework by Garg, Venkataramani and Priyakumar [13]	X	✓	✓
Proposed System (CardioFit AI)	✓	✓	✓

This study addresses these gaps by introducing a modification of the model proposed by Garg, Venkataramani and Priyakumar [13], specifically designed to capture all three critical factors. They have been able to produce reconstructed ECG signals with a much higher Pearson correlation coefficient in most of the leads.

1.3.3 Heart Disease Diagnosis

The need for accurate and rapid diagnosis of cardiac conditions is critical in the medical field, especially when addressing life-threatening conditions such as arrhythmias and myocardial infarctions. Traditional methods of diagnosing these conditions through ECG data rely on conventional electrode placements across the chest and limbs, which can be cumbersome, time-consuming, and challenging in emergency scenarios. The conventional approach also often requires expert analysis, which may not be immediately available, leading to delays in treatment.

Previous research has explored various innovations in ECG data acquisition and analysis. For instance, studies like those by Barandas et al. [24] and Ghaffari et al. [25] have focused on improving the accuracy of cardiac disease diagnosis using machine learning models trained on conventional 12-lead ECG data. However, these studies do not address the convenience and immediacy required in real-world applications, particularly in out-of-hospital settings where time is of the essence. Additionally, the application of AI in ECG analysis has been explored extensively, yet the integration of these techniques into portable, user-friendly platforms like mobile applications remains underdeveloped.

Another significant gap in the literature is the lack of research on alternative methods of lead placement that can simplify the process without sacrificing diagnostic accuracy. While Smith et al. [26] have examined the feasibility of single-lead ECG devices, their studies do not fully address the possibility of reconstructing a full 12-lead ECG from minimal lead placements, which could revolutionize the field by making ECGs more accessible in non-clinical settings.

In addressing the communication and action gaps during emergencies, recent works have explored integrating ECG analysis with telemedicine. However, the majority of these studies focus on data transmission and remote monitoring rather than real-time decision-making and emergency notifications [27], [28]. For example, Kumar et al. demonstrated the potential of using cloud-based platforms to store and analyze ECG

data, but their work lacks real-time alert mechanisms that could notify caregivers immediately upon detecting a critical condition.

The proposed system aims to bridge these gaps by introducing a novel method of capturing and analyzing ECG data. It leverages palm-based electrode placement to reconstruct a 12-lead ECG, integrating this with a machine learning model trained to diagnose cardiac conditions rapidly. This system also includes a real-time emergency notification feature that sends the patient's geolocation to pre-defined contacts if a critical condition is detected. This innovative approach addresses the challenges of traditional ECG methods, improving accessibility, speed, and accuracy while enabling immediate intervention during emergencies.

By developing a comprehensive mobile application that facilitates real-time diagnosis and alerts, this research offers a significant advancement over previous studies. The proposed system not only enhances the accuracy of diagnosis through novel lead placement and advanced data processing but also ensures that emergency situations are handled swiftly and effectively without the need for immediate expert intervention.

Table 3: Comparison of Previous Work and Proposed System (Heart Disease Diagnosis Component)

Aspect	Research A	Research B	Research C	Proposed System
Diagnosis Methodology	Traditional 12-lead ECG analysis	Conventional 12-lead ECG with AI models	Single-lead ECG with deep learning	Palm-based electrode placement with 12-lead reconstruction using machine learning
Lead Placement	Conventional chest and limb placement	Conventional chest and limb placement	Single-lead ECG	Novel palm-based lead placement
Data Processing Techniques	Basic signal processing with AI for arrhythmia detection	AI for diagnosing myocardial infarction and arrhythmias	Deep learning for arrhythmia detection	Machine learning models for cardiac condition diagnosis

				using minimal lead ECG data
Emergency Handling	No real-time alerts; relies on clinician intervention	No real-time alerts	No real-time alerts	Immediate geolocation-based alerts to predefined contacts upon critical diagnosis
Mobile-Based Implementation	Not implemented	Not implemented	Limited to cloud-based monitoring	Full mobile application integration with real-time diagnosis and emergency alerts
Accessibility	Hospital setting with expert interpretation required	Hospital setting with expert interpretation required	Out-of-hospital, but lacks user-friendliness	User-friendly, accessible for both clinicians and non-experts

1.3.4 Dietary Plan Prediction

The landscape of fitness applications has seen significant growth improving their products adapting to the modern technology, with many platforms offering exercise tutorials and diet plans aimed at helping users achieve their fitness goals and maintain a healthy lifestyle. However, a notable gap exists in these applications' approach to dietary guidance. Most fitness apps currently available in the market focus on providing predefined diet plans that do not account for the individual's unique physiological characteristics or ongoing health metrics. These static diet plans fail to adapt to the dynamic nature of human metabolism and health, which vary significantly based on factors such as age, gender, BMI, heart condition, blood glucose levels, and cholesterol levels.

Predefined diet plans can often lead to suboptimal health outcomes because they do not consider the user's real-time health data. These plans typically ignore critical metrics such as blood glucose and cholesterol levels, which are vital for creating personalized dietary recommendations [29]. Furthermore, studies have shown that diet plans that do not adapt to individual health conditions, such as heart disease or diabetes, can sometimes exacerbate these conditions rather than help manage them [30].

The proposed solution addresses this gap by offering customized dietary advice rather than a one-size-fits-all diet plan. This dietary advice is tailored based on the user's age, gender, BMI, heart condition, blood glucose levels, and cholesterol levels. Unlike existing applications, which generally require extensive user input and still fail to provide truly personalized advice, the proposed solution minimizes user input to essential factors such as age, gender, height, and weight. More critical health indicators, such as blood glucose and cholesterol levels, are obtained through scanning the patient's medical reports, and heart conditions are assessed through ECG readings.

Studies by Thomas et al. and Johnston et al. suggest that personalized dietary recommendations that consider a range of health metrics are more effective in promoting long-term health and preventing chronic diseases [31], [32]. For instance, the inclusion of blood glucose levels in dietary advice is crucial for individuals at risk

of or managing diabetes, as highlighted by the research of Evert et al [33]. Similarly, tailoring dietary recommendations based on cholesterol levels can help in managing cardiovascular risk, as supported by the findings of Lichtenstein et al [34].

The proposed solution thus fills a significant gap in the current market by moving beyond static diet plans. It provides dynamic, evidence-based dietary advice that is closely aligned with the user's current health status. This approach not only enhances the relevance and effectiveness of the dietary advice provided but also aligns with the latest research advocating for personalized nutrition. This solution's focus on integrating multiple health indicators into a cohesive dietary advice model is in line with the growing recognition in the scientific community that effective dietary interventions must be tailored to the individual's specific health needs [35].

Table 4: Summary of Research Gaps (Dietary Plan Prediction Component)

Current Approaches	Identified Research Gaps	Proposed Solution
Static Diet Plans	Do not account for real-time health data and individual variability	Provide dynamic dietary advice based on personalized health metrics
Limited Health Metrics Consideration	Ignore critical factors like blood glucose, cholesterol levels, and heart condition	Include age, gender, BMI, heart condition, blood glucose, and cholesterol levels
Extensive User Input Required	Requires user to input a large amount of data manually	Minimize user input by automating data collection through medical reports and ECG
Lack of Real-Time Adaptability	Static plans fail to adapt to changes in the user's health status	Continuous updating of dietary advice as health metrics change

While existing fitness applications offer general diet plans that may not cater to individual health needs, the proposed solution fills this gap by providing tailored dietary advice based on a thorough consideration of various health metrics. This innovation in dietary recommendation stands on the cutting edge of personalized

nutrition, making it a potentially more effective tool for users seeking to manage their health through diet. By addressing the gaps identified in current approaches, the proposed solution enhances the relevance, personalization, and effectiveness of dietary advice in fitness applications.

1.4 Research Problem

Cardiovascular diseases (CVDs) are the leading cause of death globally, accounting for an estimated 17.9 million deaths annually [44]. Millions of people suffer from cardiac arrhythmias, and critical misdiagnoses of heart diseases often result in preventable deaths [45]. A prominent reason for the spread of non-communicable diseases (NCDs) is unhealthy dietary patterns [46]. Given this context, the core problem that this research aims to address is the need for a non-invasive, cost-effective, and accessible method for obtaining a reliable 12-lead ECG, which is a crucial tool for the early detection and continuous monitoring of cardiac conditions.

Traditional methods of obtaining a 12-lead ECG involve using multiple electrodes placed on specific parts of the body, which require direct skin contact, specialized equipment, and trained personnel to operate [47]. This approach, while effective, is impractical in many settings, particularly in remote or resource-limited environments where access to healthcare facilities and advanced medical devices is limited. Early detection and monitoring of cardiac abnormalities are essential for improving patient outcomes, yet the limitations of traditional ECG acquisition methods create barriers to widespread and regular use.

This research initially explored using facial analysis to obtain a 12-lead ECG by extracting remote photoplethysmography (rPPG) signals from facial video data. The aim was to develop a contactless method for ECG acquisition using advanced image processing and machine learning techniques to analyze subtle changes in skin color caused by blood flow [48]. However, the practical challenges such as environmental factors, lighting, and facial motion highlighted the need for an alternative approach. Consequently, the research shifted to palm-based ECG acquisition, using a three-electrode setup on the palms to obtain lead I, which was then used as input for a deep learning model to reconstruct the remaining 12 leads [49].

This approach offers greater reliability and practicality, especially in environments with limited healthcare access. Moreover, by integrating palm-based ECG acquisition with machine learning for multi-lead ECG reconstruction, this research provides a novel, accessible solution to the global burden of cardiovascular diseases. This method could be particularly useful in remote areas or low-resource settings, where standard ECG equipment is often unavailable.

The rising prevalence of non-communicable diseases (NCDs), particularly cardiovascular diseases, underscores the need for more comprehensive and personalized healthcare approaches. Unhealthy dietary patterns and the global rise in obesity further exacerbate the burden of these conditions. Traditional methods of dietary advice often overlook crucial physiological factors such as cholesterol, glucose levels, and heart health, leading to suboptimal outcomes. This research highlights the importance of personalized diet advice that integrate not only demographic and anthropometric data but also key physiological indicators, including ECG readings.

By offering a non-invasive, remote method to obtain ECG signals, the system addresses a critical gap in cardiac monitoring. This approach enables individuals to access important diagnostic tools in the comfort of their homes or other convenient locations, significantly improving access to care. Coupled with personalized dietary advice tailored to an individual's physiological profile, this system has the potential to not only reduce the risk of cardiovascular diseases but also empower patients to take control of their health. Ultimately, the combination of advanced ECG acquisition and personalized nutrition guidance offers a holistic solution to improving heart health and preventing NCDs.

1.5 Research Objectives

This report outlines the objectives for developing an advanced smart tablet platform aimed at revolutionizing self-assessment ECG tests and personalized health management. The platform's primary goal is to generate a complete 12-lead ECG signal from a single ECG lead (Lead I) with a high degree of accuracy, while also providing personalized dietary advice based on patient health records. The project aims to facilitate early diagnosis of critical cardiac diseases, improve the accessibility of ECG monitoring, and enhance patient engagement in their health management.

Generation of a Complete 12-Lead ECG from Lead I Signal - Develop and validate a method to reconstruct the entire 12-lead ECG pattern from a single ECG lead (Lead I), ensuring the generated signals are accurate and comparable to traditional ECG readings.

Identification of Critical Cardiac Diseases - Utilize the reconstructed 12-lead ECG signals to accurately diagnose critical cardiac conditions, enabling informed medical discussions and early intervention.

Personalized Dietary Advice Based on Health Records - Analyze patient health data, including demographic, anthropometric, and physiological parameters, to provide tailored dietary recommendations that promote a healthy lifestyle.

Specific Objectives:

Data Collection and Analysis: Collect Comprehensive Health Data- Gather datasets containing demographic information (age, gender), anthropometric measurements (weight, height), and physiological data (glucose and cholesterol levels).

Analyze Medical Reports: Review and interpret patient medical reports to provide detailed health assessments and identify potential risk factors.

Mapping Diagnosis to Dietary Advice:

Develop Analytical Learning Algorithms - Explore and implement suitable analytical learning algorithms to map diagnosis results with precise dietary recommendations.

Pilot Testing for Accuracy - Conduct pilot studies to test the accuracy of the software in providing reliable diet advice and diagnosis, ensuring that the system meets clinical standards.

Model Training and ECG Reconstruction:

Determine Optimal ECG Lead - Identify the optimal ECG lead for obtaining accurate readings that contain the most comprehensive cardiac data. This will involve using a customized ambulatory ECG device.

Deep Learning Model Implementation- Train a deep learning model, using the Attention U-Net Framework, to reconstruct the remaining 11 ECG leads from the acquired Lead I signal. The model will analyze the cardiac data in the input signal to generate accurate and clinically useful ECG patterns.

User Engagement and Health Management:

Develop a User-Friendly Interface - Create an intuitive interface optimized for tablets, allowing users to visualize their ECG signals in real-time, thus promoting early detection and continuous monitoring of cardiac conditions.

Medicine Management - Build a mechanism that enables patients to upload medical prescriptions and set medicine reminders based on the provided information, enhancing adherence to treatment plans.

Remote Health Monitoring Integration:

Integrate with Remote Health Systems - Investigate the potential to integrate the ECG capturing technique with remote health monitoring systems. This would include real-time data transmission, location tracking, and automatic recommendations based on detected health conditions.

The primary objective of this project is to create a smart tablet platform that simplifies the process of conducting self-assessment ECG tests while ensuring the accuracy and clinical relevance of the generated data. By leveraging advanced deep learning techniques and integrating personalized health management features, this platform aims to provide a comprehensive and user-friendly solution for cardiac monitoring and overall health improvement.

2. METHODOLOGY

2.1 Methodology

2.1.1 ECG Acquisition from the Palm

This component focused on developing a palm-based ECG acquisition system capable of reliably capturing a Lead I ECG signal from the palms, which would then be used as the foundation for reconstructing a full 12-lead ECG.

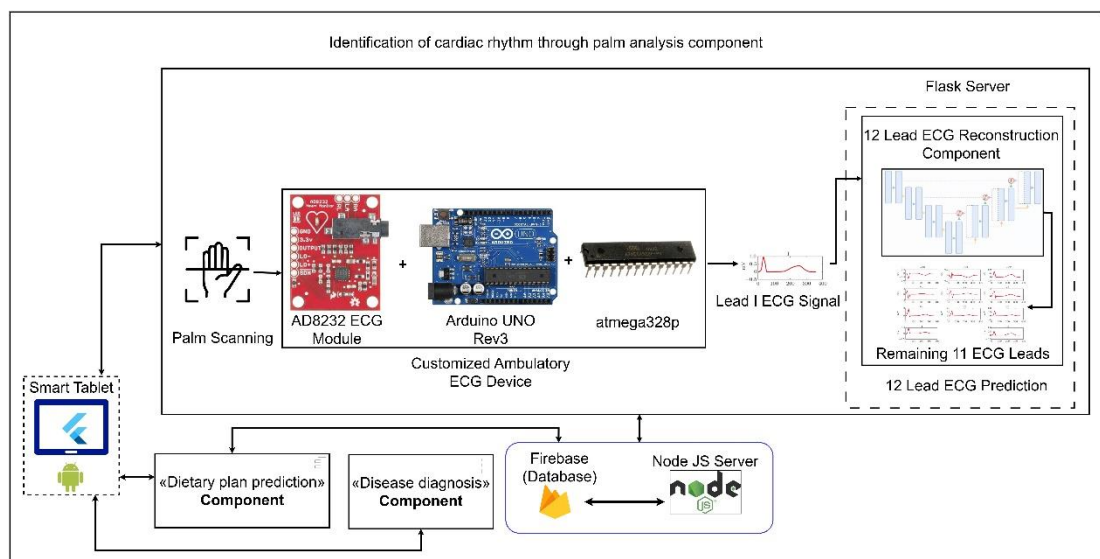


Figure 2.1.1.1: Palm Analysis and ECG generation System Architecture

As seen in Figure 2.1.1.1 the core components of the system included an AD8232 ECG module and an Arduino UNO Rev3 with an ATmega328p microcontroller. The AD8232 was selected for its high common-mode rejection ratio (CMRR) and low power consumption, making it well-suited for portable, battery-powered applications. The Arduino UNO facilitated Analog-to-Digital Conversion (ADC) and initial signal processing, with data transmitted to a PC via USB for further analysis. It consists of disposable gel electrodes, and they were chosen for their ability to maintain stable and low impedance contact with the skin, which is essential for capturing clear biopotential signals. The electrode configuration, with the positive electrode (LA) on the left palm,

the negative on the right palm (RA), and the reference electrode (RL) on the right leg, was specifically selected to optimize signal acquisition while minimizing noise.

Lead I was chosen for its simplicity and its effectiveness in providing a clear view of the heart's electrical axis, making it an ideal lead for reconstructing the full 12-lead ECG. The system captured the Lead I signal, which was then amplified, conditioned, and digitized at a sampling rate of 200 Hz to ensure sufficient resolution for detailed analysis. Several noise reduction techniques were employed to enhance signal quality, including a 50 Hz notch filter to eliminate power-line interference, a high-pass filter to address baseline drift, and a low-pass filter to reduce muscle noise. These techniques ensured that the ECG signals retained their diagnostic integrity, free from significant artifacts.

Once the ECG signals were filtered and digitized, they were transmitted to a PC for further processing. A custom Python script was developed to apply additional digital filters as necessary, refining the signal for the next phase. The processed Lead I signal was then used as input for a modified Attention U-Net deep learning model, hosted on a Flask server, which reconstructed the remaining 11 ECG leads. This comprehensive approach allowed for the transformation of a single Lead I signal into a full 12-lead ECG, making the system a valuable tool for accurate and portable ECG monitoring, particularly in remote health assessments and telemedicine applications.

2.1.2 Reconstructing Multiple ECG Leads

The proposed system demonstrates the capability to reconstruct standard ECG leads by thoroughly analyzing a single ECG lead. The complexities of this analysis procedure are explained in Figure 2.1.2.1, which offers a graphic depiction of the methodical and complex approach used by this specific system component. The standard ECG leads reconstruction process is demonstrated by this graphical representation, which also helps to clarify the analytical details.

As part of the suggested remedy, a customized ambulatory ECG device is identified as an equipment for this research component. The device is essential to the acquisition of ECG signals since it integrates with the smart tablet in a fluid manner.

The ECG lead I signal obtained from the specified device is effortlessly transferred to the smart tablet application. That ECG signal is then subjected to a second step in the application framework, where it is processed using a deep learning model that is intended to reconstruct the remaining ECG leads. Based on the Attention U-Net Framework, which evolved from CNN principles, the deep learning model's architecture is built [13]. The computational power of an NVIDIA Tesla P100 graphics processing unit (GPU) is utilized to increase the effectiveness of the training process for this deep learning model. This significantly accelerates the computational effort and speeds up the training stage.

During the final phase of the procedure, the 7 leads (II, V1, V2, V3, V4, V5, V6) that were reconstructed using the deep learning model and other calculated 4 leads (III, aVR, aVL, aVF) using lead I & II are combined with the ECG lead I that was obtained from the ambulatory ECG device. This combined ECG data set is sent to the Firebase database. This final phase makes sure that the enriched ECG data is properly compiled and stored, which makes it easier to retrieve and analyze the data later on inside the larger telemedicine framework.

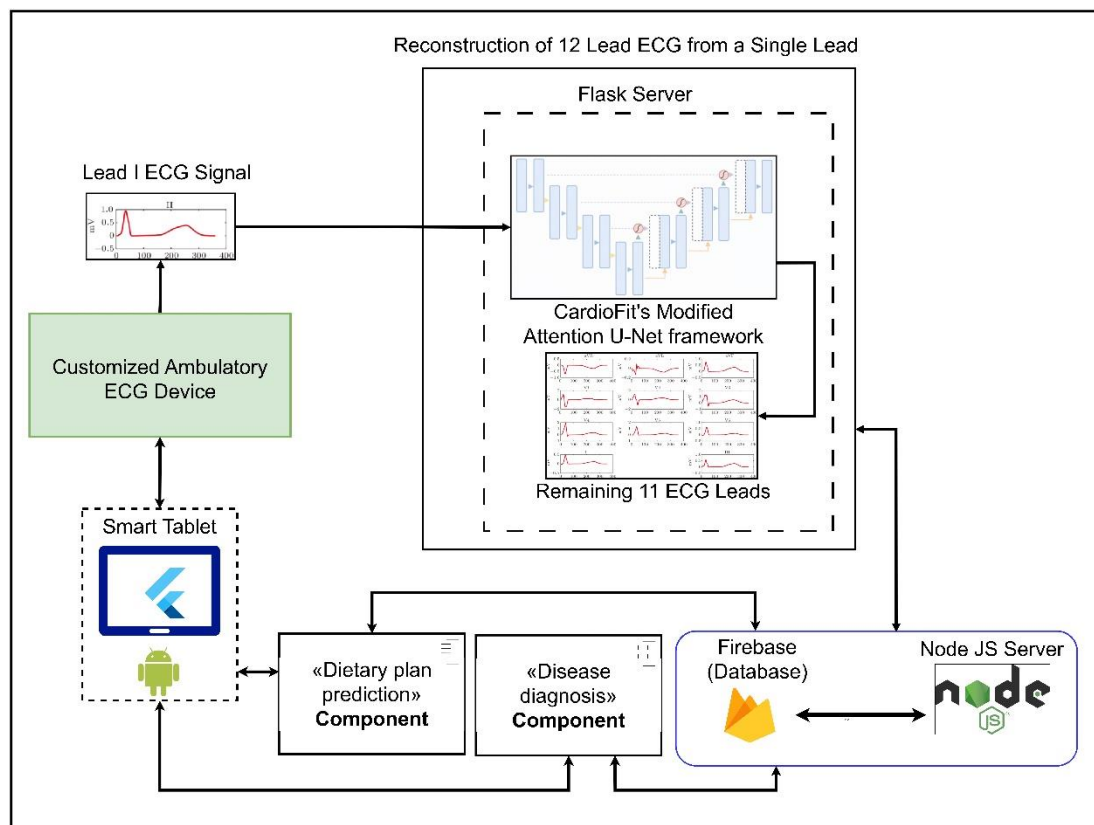


Figure 2.1.2.1: Reconstruction of 12-lead ECG from a single lead component diagram

Table 5 provides a detailed overview of the technologies, architectures, and algorithms used in the palm analysis component of cardiac rhythm detection.

Table 5: Technologies, architectures and algorithms used

Technologies	Flutter, Python, TensorFlow, Keras, Arduino
Architectures	Attention U-Net
Algorithms	CNN

2.1.3 Heart Disease Diagnosis

This research was done with the aim of making a new method and proving its efficiency in diagnosing diseases of the heart by using an ECG 12-lead signal taken from the face and palms unconventional electrode placement. The methodology was the various aspects of the project: data acquisition, preprocessing, model development, and post-diagnosis actions, one of which was live location sharing. We will elucidate each of these steps in detail in the section below.

Data Acquisition



Figure 2.1.3.1: PTBXL dataset

This investigation used the PTB-XL dataset, a wide range of ECG records included several cardiac conditions that were found in people of kindly diverse [36]. The dataset was acquired, and it might be procured from it. In the process of extracting relevant features such as age, sex, and ECG recordings from the dataset, this information was also extracted. The ECG signals were taken with the help of a particular electrode placement method, where electrodes were placed on the palms rather than the chest and limb conventionally. This unexampled method has contributed to a new way of heart disease diagnosis, which ungrudgingly crossing even the geographical divide may be, the suggested patient friendly method is much more understandable and approachable.

Data Preprocessing:

Several crucial procedures were taken during the preprocessing stage to guarantee the data's accuracy and applicability. The age and sex data in the dataset was first cleansed, and missing values were properly handled. To remove noise and artifacts, which is crucial for accurate diagnosis, the ECG signals received are treated. The raw ECG signals were then filtered in a low-pass mode to suppress the interference high-frequency noise [37]. The filter was designed using a movable average window to ensure that the signal was rather than abrupt, it was smooth. The signals were additionally standardized to a fixed range in turn translated to the improvement of the accuracy of the machine learning models.

Thereafter, the 12-lead ECG signals were subsequently divided into training, validation, and test sets. Stratified sampling was the method which made sure that the distribution of diagnostic labels was the same across all sets. The training set was utilized to train the models, the validation set to adjust hyperparameters, and the test set to see how the final model performed.

Model Development:

The main goal of this study was to create a deep learning model that could identify heart disorders based on 12-lead ECG signals that were taken from the palms. The design of the model architecture, training, assessment, and deployment were among the crucial stages of the model development process.

Model Development:

The main goal of this study was to create a deep learning model that could identify heart disorders based on 12-lead ECG signals that were taken from the palms. The design of the model architecture, training, assessment, and deployment were among the crucial stages of the model development process.

Model Architecture Design:

In order to capture the temporal patterns in the ECG signals, a basic 1D convolutional neural network (CNN) was used as the initial model architecture. Layers like Conv1D, MaxPooling1D, Flatten, and Dense layers were incorporated in this model, which culminated in a softmax output layer for multi-class classification. The model was assessed using binary cross-entropy loss and other performance metrics, including accuracy, recall, precision, and AUC (Area Under the ROC Curve). The model was trained using the Adam optimizer with a learning rate of 0.001 [38].

- Input Layer

The input layer receives the 12-lead ECG data, where 5000 represents the number of time steps and 12 corresponds to the number of leads.

$$X \in \mathbb{R}^{5000 \times 12}$$

- Convolutional Layers

The convolutional layers extract features from the ECG signals using 1D convolution

$$Y = \text{ReLU}(W * X + b)$$

where W and b are the weights and biases, and $*$ denotes the convolution operation.

- Global Average Pooling (GAP) Layer

The GAP layer reduces each feature map to a single value by averaging, reducing dimensionality while retaining important information.

- Dropout Layer

Dropout was applied to prevent overfitting, randomly setting a fraction of input units to zero during training.

- Output Layer

The final dense layer with sigmoid activation outputs probabilities for multi-label classification of 23 possible diagnoses.

$$\hat{Y} = \sigma(WZ + b)$$

where σ *denotes* the sigmoid function.

The remaining blocks were added to a more complex architecture that was created to increase model performance. By including shortcut connections that helped address the vanishing gradient issue that deep networks frequently face, this architecture enabled the model to learn more complicated characteristics. Multiple convolutional layers, residual blocks, global average pooling, dropout layers to minimize overfitting, and a sigmoid output layer for multi-label classification made up the final design.

Training and Evaluation:

```
# Attempt to stratify the data by the combination of diagnoses.
folds = list(StratifiedKFold(n_splits=5, shuffle=True, random_state=42).split(data, unique_combos))
dev_idx, test_idx = folds[0]
train_idx, val_idx = train_test_split(dev_idx, test_size=0.1, random_state=209)
print('Size of Train Split:', len(train_idx))
print('Size of Test Split:', len(test_idx))
print('Size of Validation Split:', len(val_idx))

# Split data df
data_train = data.iloc[train_idx].reset_index(drop=True)
data_val = data.iloc[val_idx].reset_index(drop=True)
data_test = data.iloc[test_idx].reset_index(drop=True)

# Split labels df
labels_train = data_labels.iloc[train_idx].reset_index(drop=True)
labels_val = data_labels.iloc[val_idx].reset_index(drop=True)
labels_test = data_labels.iloc[test_idx].reset_index(drop=True)

# Create data generators
train_gen = ECG_DataGen(data_train, labels_train, data_col, batch_size, sample_len)
val_gen = ECG_DataGen(data_val, labels_val, data_col, batch_size, sample_len)
test_gen = ECG_DataGen(data_test, labels_test, data_col, len(data_test), sample_len)

Size of Train Split: 15695
Size of Test Split: 4360
Size of Validation Split: 1744
```

Figure 2.1.3.2: Training and Evaluation

The training procedure for the ECG-based cardiac disease detection model was carefully planned to avoid overfitting and guarantee strong performance in a variety of heart situations. The dataset was divided into training, validation, and testing sets

using a stratified K-fold approach in order to preserve class balance—a crucial component in medical datasets with potentially unbalanced classes. 15,695 samples were used for training, 1,744 samples were used for validation, and 4,360 samples were used for testing in the final splits. This balanced distribution made it possible to assess the model's generalization skills in-depth.

Early halting was used as a crucial preventative technique against overfitting during training. A patience parameter of three epochs and a monitor parameter configured to measure validation loss (val_loss) were used to define the EarlyStopping callback. This configuration made sure that training would stop and the model's best weights—those with the lowest validation loss—would be restored if the validation loss did not decrease for three consecutive epochs. This tactic was essential for preventing overfitting to the training set and preserving the model's good generalization on unknown data.

```
# Define the callback
early_stop = EarlyStopping(monitor='val_loss', patience=3, restore_best_weights=True)
```

Figure 2.1.3.3: Early stopping callback method

Real-time performance tracking was done using validation data to oversee and direct the training process. Through this monitoring, performance might be optimized by fine-tuning hyperparameters including learning rate, batch size, and model architecture. The finished model showed notable gains in accuracy, recall, and precision, highlighting its efficacy in accurately diagnosing and differentiating between different heart diseases.

Several methods were used to evaluate performance on the test set. Confusion matrices offered in-depth understanding of the model's classification abilities, emphasizing its capacity to accurately classify various cardiac states. The trade-offs between recall and precision were further clarified by precision-recall curves, which are crucial in medical settings where false negative results can have detrimental effects. Furthermore, Area Under the Curve (AUC) values were calculated to evaluate the

overall discriminative capacity of the model, providing a thorough assessment of its effectiveness.

A model that obtained high accuracy and consistency across multiple classes was the result of the combination of early halting, stratified data splits, continuous monitoring, and comprehensive assessment metrics. This made the model a dependable tool for identifying heart illnesses based on ECG data.

Deployment:

After attaining adequate performance, the model was put into use on a Flask server and hosted on PythonAnywhere, enabling cardiac diagnostics in real time. For this deployment, a RESTful API that takes in ECG data, runs it through the trained model, and outputs diagnostic predictions had to be created. Furthermore, a geolocator module was included in the Flutter mobile application as well. This module locates the patient and notifies their loved ones about the necessity of hospital admission based on the severity of the identified conditions.

2.1.4 Dietary Plan Prediction

The success of developing a reliable and effective dietary advice system hinge on a thorough and systematic approach to requirement gathering. This process is crucial as it forms the foundation for creating a robust dataset that can deliver personalized dietary recommendations.

Requirement Gathering:

The initial phrase of the project was focused on gathering the comprehensive requirements necessary for developing an accurate and effective dietary advice system. The requirement gathering process involved the following steps,

1. Expert consultations: To ensure that the dietary advice system is grounded in expert knowledge, consultations were held with several experts in the domain. I engaged with experienced nutritionists and dietitians who provided valuable insights into the factors that must be considered when creating personalized recommendations. It was with their guidance and advice that a dietary advice section was selected instead of following the tradition of proposing a diet plan.
2. Collaboration with Colombo Research Institute: The initial milestone of the requirement gathering originated at the nutrition department of Colombo Research Institute. This collaboration was instrumental in understanding the scientific and clinical aspects of the nutrition domain that are crucial for developing a credible dietary advice system. The nutritionists at the department provided the latest research trends, nutrition requirements that a healthy person must consume, the right quantity of food that needs to be consumed and steps on how a diet should be prepared.

Diet dataset preparation: To further refine the requirements, I collaborated with dietitians at Hemas hospital Wattala. This conjunction helped me in gathering practical insights from a clinical setting, ensuring that dataset is relevant and applicable to real

world scenarios. The domain expertise of the dietitian was valuable in understanding and preparation of the dataset matching the scenarios chosen.

Dataset Development:

Based on the requirement gathering and insights collected a comprehensive dataset was developed. The dataset was designed to incorporate critical and important factors that influence dietary needs and recommendations. The factors considered are as follows,

1. Age: Different age groups have different nutritional requirements. Therefore, it is important to consider age as a factor when providing dietary advice. For the ease of dataset creation age is classified for three categories as,
 - a. Age 01 - Age 19
 - b. Age 20 – Age 64
 - c. Age 64+
2. Gender: Gender differentiates in nutritional and metabolism, thereby it is considered as an important factor.
3. Body Mass Index (BMI): BMI is a key indicator of an individual's health status and was included to provide recommendations that align with an individual's weight category. People are grouped into four categories considering the BMI obtained,

Table 6: BMI Categorization

BMI	Category
BMI less than 18.5 kg/m ²	Underweight
BMI \geq 18.5 kg/m ² & BMI $<$ 24.9 kg/m ²	Normal
BMI \geq 25.0 kg/m ² & BMI $<$ 29.9 kg/m ²	Overweight
BMI \geq 30.0 kg/m ²	Obese

4. **Heart Condition:** Individuals with existing heart conditions require specialized dietary advice. The dataset includes parameters for tailoring recommendations to support cardiovascular health.
5. **Blood Glucose Levels:** Managing blood glucose levels is crucial for individuals with diabetes or prediabetes. The dataset includes guidelines for diets that help maintain stable glucose levels.
6. **Blood Cholesterol Levels:** The dataset also considers cholesterol management, providing dietary advice that supports healthy cholesterol levels and reduces the risk of cardiovascular disease.

Method of obtaining the essential factors:

To provide effective dietary advice the following methods were employed to obtain the necessary factors,

Table 7: Method of Obtaining Essential Factors

Factor	Method	Description
Age	User input	Obtained by asking the users to enter their age in their user profile, which is then categorized into relevant categories when proposing the dietary advice
Gender	User input	Gender information is obtained through user inputs, allowing the system to provide dietary recommendations in a customized manner.
Height	User input	Obtained by asking the users to enter their height in their user profile, which is then used for BMI calculation. Height is obtained in centimeters (cm).
Weight	User input	Obtained by asking the users to enter their height in their user profile, which is then used for BMI calculation. Weight is obtained in kilogram (kg).

BMI	System calculation	<p>When the user enters the height and weight the BMI of the person is calculated based on the following equation,</p> $\text{BMI} = \frac{\text{Weight (kg)}}{\text{Height (m)}^2}$
Heart condition	ECG reading analysis	<p>The heart condition is derived from ECG (Electrocardiogram) analysis. If any cardiac arrhythmia (irregular heartbeat) is detected in the ECG readings, the system records this condition as a Boolean value, “Yes” if arrhythmia is present, and “No” if no arrhythmia is observed. This information is crucial for tailoring dietary advice to support cardiovascular health, particularly in individuals with identified cardiac conditions.</p>
Blood glucose condition	Medical report analysis	<p>Blood glucose levels are obtained from the analysis of medical reports. This data is critical for creating dietary recommendations for individuals with diabetes or those at risk of developing diabetes.</p>
Blood cholesterol condition	Medical report analysis	<p>Obtained from the analysis of medical reports. This data is critical for creating dietary recommendations for individuals with diabetes or those at risk of developing cholesterol.</p>

BMI Calculation:

To calculate the Body Mass Index, it is necessary to convert the user's height from centimeters to meters before applying the BMI formula. The user inputs their height in centimeters (cm) and weight in kilograms (kg), and the system performs the necessary conversion and calculation to provide the BMI value in kilograms per square meter (kg/m²).

```
//Function to calculate BMI
void calculateBMI(String weight, String height) {
    if (weight != "" && height != "") {
        double bmi = double.parse(weight) /
            (double.parse(height) / 100 * double.parse(height) / 100);
        double roundedBMI = double.parse((bmi).toFixed(2));

        _caluclateBMIController.text = roundedBMI.toString();
    } else {
        _caluclateBMIController.text = "";
    }
}
```

Figure 2.1.4.1: BMI Calculation

As mentioned in the above snapshot since the height provided by the user is initially input in centimeters. This height is then converted to meters by dividing the value by 100, as BMI requires height in meters for accurate calculation. Once the height is converted, the BMI is calculated using the standard formula: $BMI = \text{Weight (kg)} / \text{Height (m)}^2$. This calculation is performed by parsing the input values, which are initially provided as strings, into double data types to ensure precise mathematical operations.

After the BMI is computed, the result is rounded to two decimal places to enhance clarity and readability. Finally, the calculated BMI value is displayed to the user via the `_calculateBMIController`, with the output being shown only when valid inputs for both weight and height are provided. This process ensures that the BMI calculation is accurate and user-friendly, delivering essential data for subsequent health assessments or dietary recommendations.

Medicine Report Analysis:

The medicine report analysis involves a systematic process to extract and analyze data from uploaded medical reports. The process is designed to be efficient, accurate, and user-friendly, ensuring that users can easily upload reports and receive meaningful analyses.

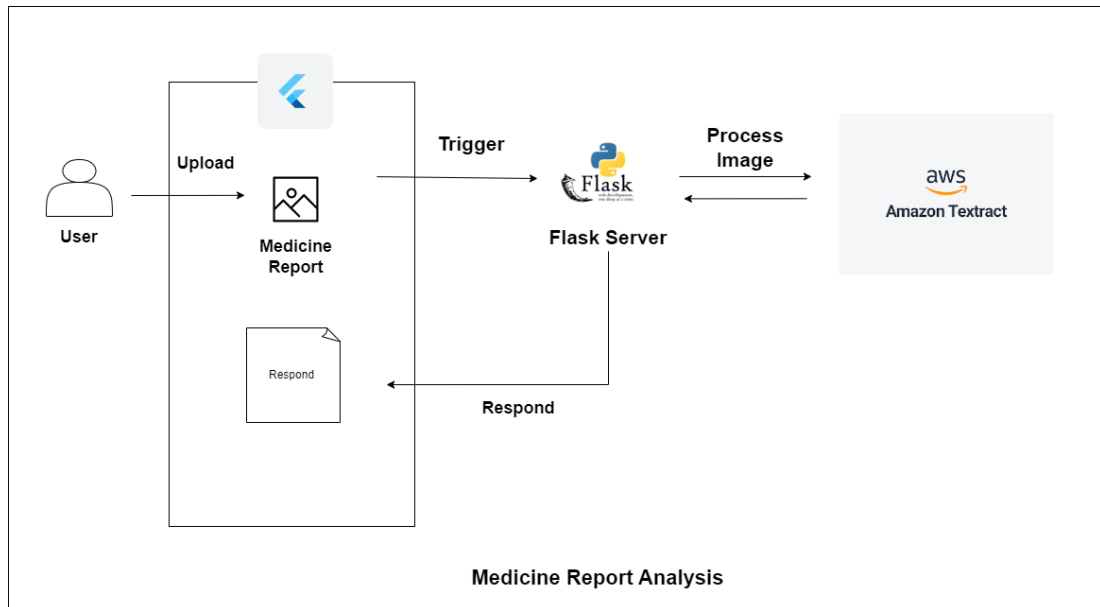


Figure 2.1.4.2: Medical Report Analysis

The following are the steps on how report analysis feature is executed,

1. User Interaction and Report Upload

The process begins with the user interacting with the application through a Flutter-based interface. The user is prompted to upload their medical reports using the app's file picker. The application supports multiple report types, including Fasting Blood Sugar, Urine Full Report, Lipid Profile, and Full Blood Count Report. Once the user selects a report type and uploads the corresponding image file(s), the application prepares the data for analysis.

2. Image Handling and Preprocessing

Upon image upload, the selected images are stored temporarily within the application. The app uses the `ImagePicker` package to facilitate the selection and uploading of images from the user's gallery. The uploaded images are then displayed in a table format within the app, allowing the user to review their selections. If the user decides to remove an image, they can do so easily through the app's interface.

3. Triggering the Analysis Process

After the user has uploaded the necessary reports, they can trigger the analysis process by clicking the "Analyse" button. This action initiates the image processing

workflow. The app sends the images to a backend server implemented using Flask, which interfaces with Amazon Textract, a service provided by AWS for document text extraction.

4. Backend Processing and Image Analysis

The Flask server receives the images and processes them using Amazon Textract. Textract analyzes the images, extracting textual data, including medical values such as glucose levels, lipid profiles, and other relevant metrics. The server then structures this extracted data into a JSON format, which is returned to the Flutter app for further processing.

5. Data Parsing and Result Compilation

Once the data is returned to the app, it is parsed and organized into a tabular format. Each report type has specific components, such as "Fasting Blood Sugar," "Cholesterol-Total," "HDL-C," etc., which are matched against the extracted data. The app dynamically generates rows in a data table for each component, displaying the measured values and corresponding units.

6. Diagnosis and User Feedback

Based on the extracted and parsed data, the app performs a comparative analysis to determine potential diagnoses. For instance, if certain glucose or cholesterol levels are detected, the app will classify the results into categories such as "Diabetes Mellitus," "Pre-Diabetes," or "High Cholesterol." The application then updates the user profile with these new data points and provides an overall diagnosis. This diagnosis is presented to the user along with the detailed report data.

7. User Interface and Navigation

The user can navigate back to the report selection screen or proceed to view the analysis results. The interface is designed to be intuitive, with clear prompts and actions, ensuring that users can efficiently manage and review their health data.

Throughout the process, the app ensures that user data is handled securely. All communication with the backend server is encrypted, and sensitive data is stored and

processed in compliance with relevant privacy regulations. The app uses secure methods to handle image uploads, API requests, and data storage, ensuring the confidentiality and integrity of user information.

The medicine report analysis methodology is a multi-step process that leverages modern technologies like Flutter, Flask, and Amazon Textract to provide users with insightful health data analysis. By combining a user-friendly interface with powerful backend processing, the application aims to make health data management accessible and meaningful for users.

Additional System Features:

Medicine Prescription Reading and Pill Reminder

This feature is designed to enhance the medication management process by leveraging modern technology. The system simplifies the process of reading prescriptions, setting up medication reminders, and managing alarms, ensuring users can maintain their medication schedules accurately and efficiently.

Prescription Reading

The system incorporates an AI-powered prescription reading feature that allows users to upload an image of their prescription. The uploaded image is processed using a backend service, which extracts relevant information such as medicine names, dosages, frequencies, and durations. This data is then presented to the user, allowing them to review and confirm the details before setting up reminders.

Users can upload a prescription image via device camera or gallery. The uploaded image is analyzed by an GPT 4o service, which extracts key details and presents them in user friendly format. The extracted data is then populated into appropriate fields for setting up reminders. If by any chance the extracted data has any errors, the users have the flexibility to edit the relevant fields.

Additionally, the system provides flexibility to the users to set up their medicine reminders manually as well. Here the users could manually enter the medicine name,

dosage, frequency, duration and any additional instructions if there is and set up pill reminders.

Setting up Alarm Reminders

Once the prescription details are confirmed or manually entered, the system sets up reminders using the `Alarm` package. These reminders are crucial in helping users take their medications on time. The system provides an intuitive interface for managing these reminders. Here the users could view, edit or delete their reminders from the list of scheduled alarms. The alarms are displayed with key details like dosage, time and additional notes (if any). Also, the users could enable or disable individual alarms without deleting them allowing for temporary changes in the schedule without looking the reminder settings.

By integrating prescription reading with an advanced pill reminder system, this solution significantly enhances medication adherence, providing a reliable tool for users to manage their health effectively. The combination of AI-driven automation and manual customization ensures that users have full control over their medication schedules, tailored to their specific needs.

Component Diagram:

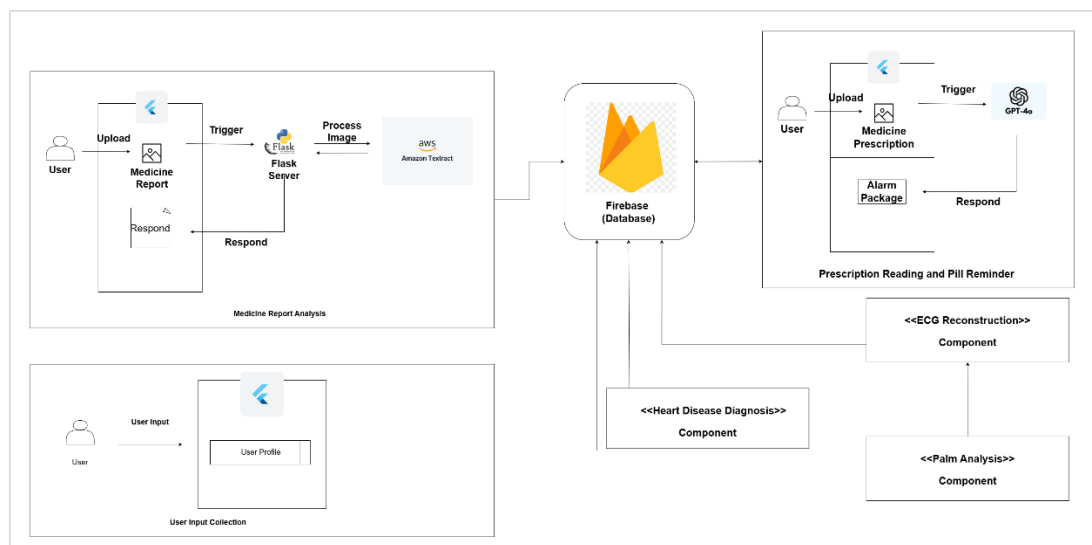


Figure 2.1.4.3: Component Diagram

The above component diagram outlines the architecture of the system designed for analyzing medical reports, reading prescriptions, managing medication reminders, and providing dietary advice in a customized manner. This system integrates several technologies and methodologies to ensure a comprehensive and user-friendly experience.

Medical Report analysis component

This component handles the analysis of medical reports uploaded by users. The process begins with the user uploading their medical report through a Flutter-based interface. The uploaded report is processed by a Flask server, which triggers the analysis using Amazon Textract—a service from AWS that extracts text and data from scanned documents. The extracted data is then structured into a format that can be used to provide insights and factors to produce dietary advice to the user. The processed results are sent back to the Flutter application, where the user can view the analysis.

Collection of User inputs

This component is responsible for gathering user inputs necessary for composing dietary advice. Users enter details such as age, gender, height, weight into their user profile. This information is used for calculating BMI and other health metrics that is then used for personalized dietary advice.

Prescription Reading and Pill Reminder

This component is designed to assist users in managing their medication schedules. Users can upload their prescription images, which are processed using an AI service that extracts critical details like medicine names, dosages, frequencies, and durations. These details are then used to set up alarms and reminders using the Alarm package. The system allows users to view, edit, or delete these reminders, ensuring they maintain their medication schedule accurately.

Firebase

The system uses Firebase as its backend infrastructure. Firebase is used to store and manage all user data, including profiles, medical reports, prescription details, and reminders.

This system provides a comprehensive solution for medication schedules, and dietary advice. By integrating various technologies like Amazon Textract, Flask, and AI services, the system offers an intuitive and effective platform for users to monitor and manage their health. The combination of prescription reading, pill reminders, and dietary advice ensures that users receive timely and personalized advice, enhancing their overall health management and decline the growing trend in non-communicable diseases and limit people from following unhealthy diet plans.

2.2 Commercialization Aspects of The Product

Our novel solution, comprising an ECG diagnostic tool and a remote health monitoring system with customized nutrition recommendations, is being commercialized to address various segments of the healthcare industry. Our approach combines the dietary management system with state-of-the-art cardiac diagnostic and remote health monitoring technologies to offer a holistic solution that will satisfy consumers and healthcare professionals alike.

Due to the rising incidence of cardiovascular illnesses and the increased need for remote health monitoring, the worldwide healthcare industry is moving toward easily accessible, non-invasive, and portable solutions. The ECG gadget has a great deal of potential for use in telemedicine, at-home diagnostics, and emergency care because it can recreate 12-lead data from a single lead. Better treatment of chronic diseases including diabetes, obesity, and cardiovascular disorders is also addressed by the individualized meal plan function built into the remote health monitoring system.

These technologies have a wide range of potential customers, from individual consumers looking for individualized health insights to healthcare professionals working in hospitals and clinics. The diet plan tool helps individuals take proactive control of their health by utilizing important health metrics including blood glucose, cholesterol, and BMI. This product is well-positioned to take a sizable chunk of the market because to the growing need for personalized healthcare solutions, particularly in areas where wearables and digital health technologies are being used more widely.

The main advantage over competitors is found in the distinct features of the health monitoring system and the ECG diagnostic tool. Compared to existing systems that are cumbersome and need clinical settings, the gadget delivers unrivaled portability and ease by reconstructing a full 12-lead ECG using just one lead. This makes it perfect for usage at home, in remote healthcare settings, and in places with limited resources.

In addition, the diet plan system's integration offers a more comprehensive approach to health management than many of the current exercise and diet apps. In a

competitive market, this feature stands out for combining AI-driven insights and providing individualized suggestions. It also has the ability to work with insurance companies and healthcare providers for wider applications.

We use a multi-pronged approach to commercialization. Direct marketing will be done to customers, telemedicine providers, and medical professionals regarding the ECG diagnostic tool. With its customized eating plan, the health monitoring system will use a subscription-based business model with various access tiers, ranging from basic health monitoring to advanced analytics and remote medical services. Hospitals and clinics may integrate these technologies seamlessly into their current healthcare systems through licensing opportunities, like our relationship with Hemas Hospitals.

Furthermore, collaborations with insurance providers and corporate wellness initiatives will be investigated as a means of encouraging consumers to embrace these technologies for prophylactic healthcare. The system's capacity to produce data-driven insights for research and the management of chronic diseases will also open up possibilities for cooperation with academic institutions and pharmaceutical businesses.

We intend to take use of distribution networks and strategic alliances to guarantee a successful market launch. The product's widespread adoption will be facilitated by partnerships with fitness technology businesses, telemedicine platforms, and healthcare providers. Through our relationship with Hemas Hospitals, we will be able to reach Sri Lanka and other areas with strong smartphone penetration and need for digital health solutions through a phased go-to-market strategy.

Digital marketing and influencer collaborations will be used to promote the diet plan function, which will appeal to both individual customers and businesses wishing to start wellness initiatives. A major emphasis will be on regulatory compliance, making sure the product complies with guidelines established by organizations such as the FDA, EMA, and NMRA.

2.3 Testing & Implementation

2.3.1 ECG Acquisition from the Palm

System Design and Hardware Integration:

The implementation of the palm-based ECG acquisition system was centered around creating a portable, cost-effective solution capable of capturing and reconstructing 12-lead ECG signals from a Lead I input. The system was designed using key components such as the AD8232 Analog Front End (AFE) and the Arduino UNO Rev3 equipped with an ATmega328p microcontroller. The system design prioritized portability and simplicity. The AD8232 AFE was chosen for its ability to effectively amplify and condition the weak biopotential signals captured from the body, specifically from the palms. The Arduino UNO Rev3, with its ATmega328p microcontroller, was responsible for digitizing these amplified signals at a sampling rate of 200 Hz, ensuring that the ECG data was captured with sufficient resolution for accurate analysis. The three-electrode configuration, with the positive electrode on the left palm, the negative electrode on the right palm, and the reference electrode on the right leg, was specifically selected to optimize signal clarity and minimize noise. The integration of these hardware components into a compact and user-friendly setup, as illustrated in Figure 2.3.1.1, allowed for easy deployment of the system in various environments, including home-based monitoring and remote healthcare settings.

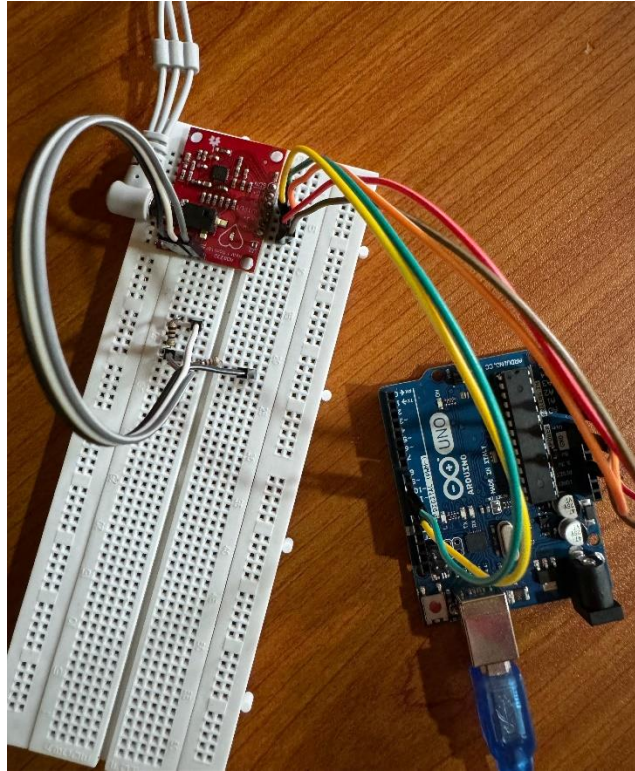


Figure 2.3.1.1: Palm ECG acquisition system

Signal Processing and Filtering Techniques:

Signal processing was a critical aspect of the system's development, aimed at enhancing the quality of the ECG data by removing noise and preserving the essential components of the signal. While the AD8232 provided initial amplification, additional noise reduction techniques were necessary to further improve signal clarity. A 50 Hz notch filter was implemented to mitigate power-line interference, a common source of noise in ECG acquisition, particularly in electrically noisy environments. Baseline drift, often caused by slow, low-frequency movements such as breathing, was addressed using a high-pass filter, while muscle noise and other high-frequency artifacts were reduced using a low-pass filter. Both second-order and fourth-order filters were employed to achieve a balance between effective noise reduction and signal preservation, ensuring that the ECG signals retained their diagnostic value. The necessity of these noise reduction techniques is highlighted in Figure 2.3.1.2, which shows the raw, unfiltered ECG signal acquired from the palms.

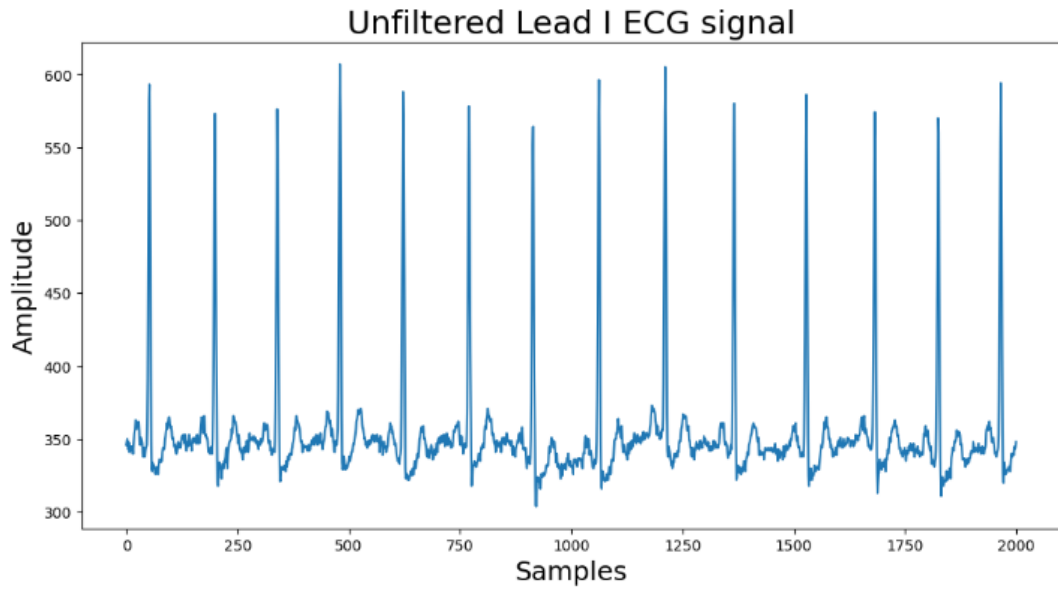


Figure 2.3.1.2: Unfiltered lead I ECG signal acquired from palms

Software Development for Data Transmission and Analysis:

Once the signals were filtered and digitized, they were transmitted to a PC via USB for further processing. A custom Python script was developed to manage this data, applying additional digital filters as necessary to refine the signal. The script was designed to be both efficient and user-friendly, facilitating seamless integration with the subsequent deep learning models used for 12-lead ECG reconstruction. Figure 2.3.1.3 illustrates the implementation of these filters.


```

6  def apply_savitzky_golay_filter_x(data):
7      window_length = 11
8      polyorder = 3
9      x = data
10     x = x.squeeze()
11     x = savgol_filter(x, window_length, polyorder)
12     return x[:, np.newaxis]
13
14     def butter_highpass_filter(data, cutoff_frequency, fs, order=4):
15         nyquist = 0.5 * fs
16         normal_cutoff = cutoff_frequency / nyquist
17         b, a = butter(order, normal_cutoff, btype='highpass', analog=False)
18         filtered_data = lfilter(b, a, data)
19         return filtered_data
20
21     def butter_lowpass_filter(data, cutoff_frequency, fs, order=4):
22         nyquist = 0.5 * fs
23         normal_cutoff = cutoff_frequency / nyquist
24         b, a = butter(order, normal_cutoff, btype='lowpass', analog=False)
25         filtered_data = lfilter(b, a, data)
26         return filtered_data
27
28     def apply_butterworth_filters_x(data):
29         x = data[np.newaxis, :, np.newaxis]
30         x = butter_highpass_filter(x, cutoff_frequency_high, fs, 4)
31         x = butter_lowpass_filter(x, cutoff_frequency_low, fs, 4)
32         return x
33
34     def apply_notch_filter(data):
35         sig = data.squeeze()
36         b, a = iirnotch(f0, Q, fs)
37         sig = filtfilt(b, a, sig)
38         sig = sig[:, np.newaxis]
39         return sig

```

Figure 2.3.1.3: Filters implementation for palm acquired ECG signals

System Validation and Connectivity Testing:

The testing phase was a crucial step in validating the system's functionality and accuracy. This phase involved the use of an Arduino sketch (Figure 2.3.1.4) to ensure proper connectivity of the palm electrodes and to verify that the system could reliably capture and process ECG signals. The code was specifically designed to monitor the connection status of the electrodes, providing immediate feedback on their status to ensure consistent and accurate signal acquisition.

```

1 void setup() {
2     // initialize the serial communication:
3     Serial.begin(9600);
4     pinMode(10, INPUT); // Setup for leads off detection L0 +
5     pinMode(11, INPUT); // Setup for leads off detection L0 -
6
7 }
8
9 void loop() {
10     if((digitalRead(10) == 1)|| (digitalRead(11) == 1)){
11         Serial.println('!');
12     }
13     else{
14         // send the value of analog input 0:
15         Serial.println(analogRead(A0));
16     }
17     //Wait for a bit to keep serial data from saturating
18     delay(1);
19 }

```

Figure 2.3.1.4: Lead off detection

2.3.2 Reconstructing Multiple ECG Leads

Data Acquisition and Data Pre-processing:

For this research project to train the deep learning model, data acquisition is necessary. This is why PTB-XL dataset [39] which contains 21,799 clinical 12-lead ECGs from 18,869 unique patients were utilized. It should be noted that these data are gender balanced with males accounting for 52% and females 48%. Every ECG spanned over ten seconds, and it contained essential electrical cardiac activities. Raw signal data had a complete set of twelve leads as a basis for our analysis. The pre-processing steps such as standardization of data format, artifact correction and signal enhancement were all done to allow accurate feature extraction leading to further analysis.

Although the dataset contains ECG samples at both sampling rates of 100Hz and 500Hz, the ECG samples recorded at a frequency of 500 Hz were decided to use instead of those recorded at 100 Hz. I arrived at this decision because a higher sample rate of 500 Hz offers more detailed information on ECG signals that can lead to better analysis and interpretation.

Empty records were removed in order to improve the quality of data. Thereafter, several digital signal processing methods were used to help reduce noise and improve the clarity of signals. In particular, I employed a Butterworth [40] high-pass filter having a cut-off frequency of 0.5Hz for lowering the level of low-frequency noise components, whereas another complementary Butterworth low pass filter was also used with a cut-off frequency of 200Hz for getting rid of any high-frequency noise artifacts. Figure 2.3.2.1 demonstrates how Butterworth low-pass and high-pass filters are used.

```
def butter_highpass_filter(data, cutoff_frequency, sampling_rate, order=4):
    nyquist = 0.5 * sampling_rate
    normal_cutoff = cutoff_frequency / nyquist
    b, a = scipy.signal.butter(order, normal_cutoff, btype='highpass', analog=False)
    filtered_data = scipy.signal.lfilter(b, a, data)
    return filtered_data
```

```
def butter_lowpass_filter(data, cutoff_frequency, sampling_rate, order=4):
    nyquist = 0.5 * sampling_rate
    normal_cutoff = cutoff_frequency / nyquist
    b, a = scipy.signal.butter(order, normal_cutoff, btype='lowpass', analog=False)
    filtered_data = scipy.signal.lfilter(b, a, data)
    return filtered_data
```

Figure 2.3.2.1: Application of the Butterworth filters in code level

Furthermore, to further refine the ECG signals, a Savitzky-Golay [41] filter was applied, smoothing out any remaining fluctuations or irregularities in the waveforms. Figure 2.3.2.2 represents the application of Savitzky-Golay filter in code level. To standardize the amplitude range across leads, Min-Max Normalization was implemented as Figure 2.3.2.3, scaling all ECG lead signals between 0 and 1. Subsequently, the dataset was partitioned into input (X) and output (Y) sets, with lead I designated as the input feature (X), and leads II, V1, V2, V3, V4, V5, and V6 forming the output features (Y). Finally, to facilitate model training and evaluation, the dataset was split into training and testing subsets at an 80%-20% ratio as the Figure 2.3.2.4, ensuring a robust assessment of model performance while preserving data integrity.

```
def apply_savitzky_golay_filter_y(data):
    result_array = np.empty_like(data)

    window_length = 31
    polyorder = 3

    for i in range(data.shape[0]):
        for lead in range(data.shape[2]):
            x = data[i, :, lead].reshape((1, data.shape[1], 1))
            x = savgol_filter(x.squeeze(), window_length, polyorder)
            result_array[i, :, lead] = x

    return result_array
```

Figure 2.3.2.2: Application of Savitzky-Golay filter in code level

```
def min_max_normalize(signal):
    min_val = np.min(signal)
    max_val = np.max(signal)
    normalized_signal = (signal - min_val) / (max_val - min_val)
    return normalized_signal
```

Figure 2.3.2.3: Application of Min-Max normalization in code level

```
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=41)
print("Train and Test sets splitted")
```

Figure 2.3.2.4: Code segment of splitting the dataset

Deep Learning Model Implementation:

Figure 2.3.2.5 illustrates the architectural blueprint of the deep learning model tailored for ECG multi-lead reconstruction. Designed to operate on individual ECG leads, each containing 5000 data points sampled at 500Hz over a duration of 10 seconds, the model aims to predict seven ECG leads, each also comprising of 5000 data points. The architecture primarily consists of convolutional layers, strategically arranged to extract hierarchical features from the input signals.

The model has the following convolutional layer's structure: firstly, 1D convolution layers which include Rectified Linear Unit (ReLU) activation functions are employed with kernel size of 3 and stride of 1 to capture local patterns in the input signals. Then, additional 1D convolution layers with ReLU activation functions and batch normalization are used, employing a kernel size of 3 and stride of 2 to down sample the feature maps so as to improve model efficiency. Lastly, I have used 1D transposed convolution with ReLU activations and batch normalizations characterized by a kernel size of 3 and a stride of 2 for signal reconstruction. In code level, Figure 2.3.2.6, Figure 2.3.2.7 and Figure 2.3.2.8 show how these convolution layers have been applied.

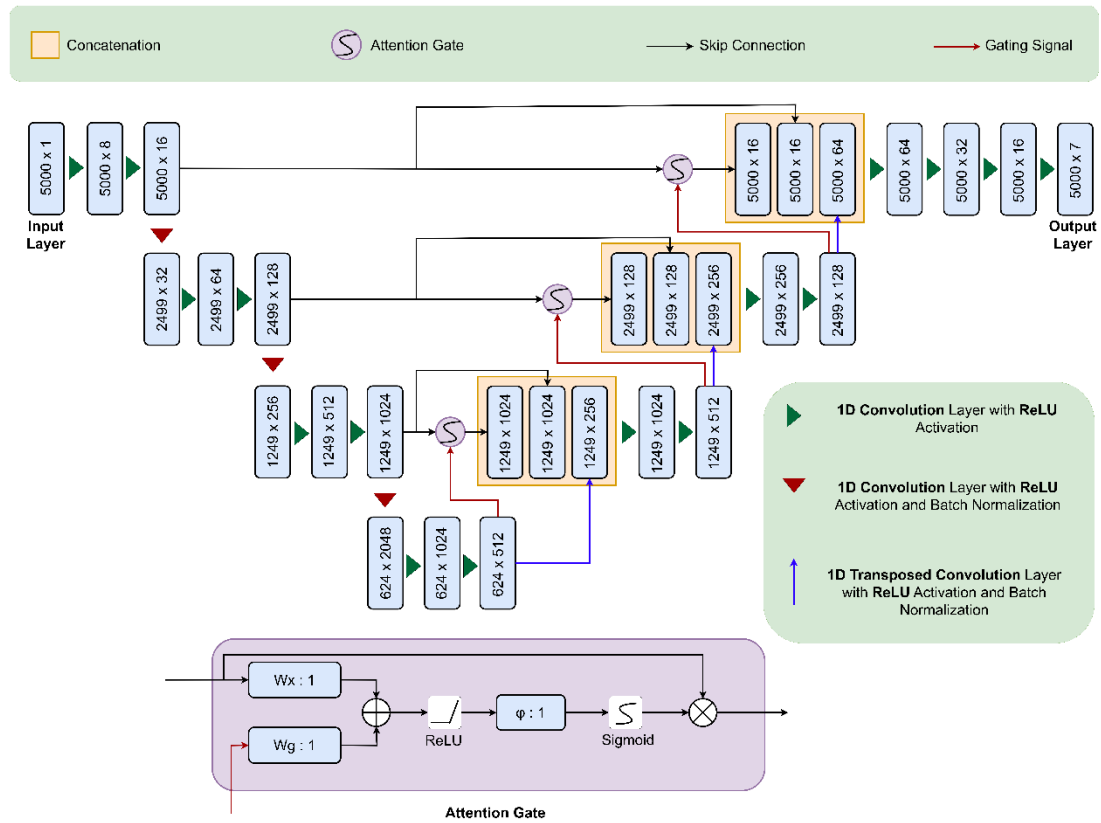


Figure 2.3.2.5: CardioFit's Modified Attention U-Net model architecture

```
def conv1D(x, num_filters):
    x = L.Conv1D(num_filters, 3, strides=1, padding='same')(x)
    x = L.Activation("relu")(x)
    return x
```

Figure 2.3.2.6: Application of the pure 1D convolution layer in code level

```
def conv1DBatchNorm(x, num_filters):
    x = L.Conv1D(num_filters, 3, strides=2)(x)
    x = L.Activation("relu")(x)
    x = L.BatchNormalization()(x)
    return x
```

Figure 2.3.2.7: Application of the 1D convolution layer with batch normalization in code level

```
def conv1DTransBatchNorm(x, num_filters):
    x = L.Conv1DTranspose(num_filters, 3, strides=2, output_padding=0)(x)
    x = L.Activation("relu")(x)
    x = L.BatchNormalization()(x)

    return x
```

Figure 2.3.2.8: Application of 1D transposed convolution layer in code level

Furthermore, within the attention gate mechanism embedded within the architecture, intricate signal processing occurs in a sequential fashion. Initially, the skip connection and the gating signal are concatenated to incorporate both local and global context into the processing pipeline. Subsequently, the concatenated signal undergoes segmentation via the ReLU activation function, facilitating feature extraction and refinement. Following this, the processed signal is passed through 1D transposed convolution layers to promote signal reconstruction. The resultant signal is then subjected to a sigmoid activation function, enabling non-linear transformations to enhance signal fidelity. Finally, the processed signal is modulated by multiplying it with the skip connection, thereby integrating contextual information and fine-grained details to enrich the reconstructed ECG leads. The code level process of attention gate mechanism is shown in Figure 2.3.2.9.

```

def fin_attention_Gate_new(x, gating, inter_shape):
    inter_shape = inter_shape.shape[2]
    print('intershape', inter_shape)

    shape_x = B.int_shape(x)
    print('\nshape_x', shape_x)
    shape_g = B.int_shape(gating)
    print('shape_g', shape_g)

    Wx = L.Conv1D(inter_shape, kernel_size = 2, strides = 2)(x)
    shape_Wx = B.int_shape(Wx)
    print('shape_Wx', shape_Wx)

    Wg = L.Conv1D(inter_shape, kernel_size = 1, strides = 1, padding = "same")(gating)
    Wg = L.Conv1DTranspose(inter_shape, 2, strides=1, output_padding=0)(Wg)
    print('Wg.shape', Wg.shape)

    concat_xg = L.add([Wg, Wx])
    print('concat', concat_xg)

    activation_xg = L.Activation("relu")(concat_xg)
    print('\nactivation_xg', activation_xg.shape)

    psi = L.Conv1D(inter_shape, kernel_size = 1, strides = 1, padding = "same")(activation_xg)
    print('psi', psi.shape)

    sigmoid_xg = L.Activation("sigmoid")(psi)
    shape_sigmoid = B.int_shape(sigmoid_xg)
    print('shape_sigmoid', shape_sigmoid)

    upsample_psi = L.Conv1DTranspose(inter_shape, 2, strides=2, output_padding=0)(sigmoid_xg)
    print('upsample_psi.shape', upsample_psi.shape)

    product_xg = L.multiply([upsample_psi, x])
    print(product_xg.shape)

    result = L.Conv1D(shape_x[2], 1, padding='same')(product_xg)
    result_bn = L.BatchNormalization()(result)

    return result_bn

```

Figure 2.3.2.9: Implementation of the Attention Gate mechanism in code level

Deep Learning Model Training:

The model underwent a training process that spanned 50 epochs. An epoch refers to one complete pass through the entire training dataset. By iterating over the dataset 50 times, the model had ample opportunity to learn the intricate patterns in the ECG signals. The training process was guided by the Adaptive Moment Estimation (ADAM) optimizer. ADAM is a popular optimization algorithm known for its efficiency and low memory requirements.

The batch size during training was set to 32. This means that the model was updated after every 32 samples. The number of epochs and the batch size are defined in the code level as shown in Figure 2.3.2.10. To have a more robust learning, the model was made to update more frequently by choosing smaller batch size. The learning rate was set at 0.0001. This is the degree of change in the model for every error predicted when updating weights of the model. Choosing the learning rate is challenging as a value too small may result in a long training process that could get stuck.

```
history = model.fit(x=x_train, y=y_train, batch_size=32, epochs=50, verbose=1)
```

Figure 2.3.2.10: Definition of batch size and epochs

A Kaggle Notebooks environment was employed to train the model, as it had 29 Gigabytes (GB) of Random Access Memory (RAM). The computational power for training was provided by an Nvidia Tesla P100 GPU that had a memory capacity of 16 GB. This high-performance GPU cut down on the time taken to train the model quite significantly. Regrettably, the TensorFlow version used in implementing this model is 2.15.0

Extracting and Calculating ECG Leads:

To obtain the remaining four ECG leads, the researchers utilize Einthoven's law [42] and Goldberger's equations [43]. Specifically, lead III is derived using Einthoven's law, as indicated in (1).

$$\text{Lead III} = \text{Lead II} - \text{Lead I} \quad (1)$$

Additionally, the augmented leads aVR, aVL, and aVF are calculated using Goldberger's equations, as outlined in (2), (3), and (4) respectively. This

comprehensive approach ensures that all necessary ECG leads are accurately reconstructed for subsequent analysis.

$$\text{Lead aVR} = (\text{Lead I} + \text{Lead II}) / (-2) \quad (2)$$

$$\text{Lead aVL} = (\text{Lead I} - \text{Lead III}) / 2 \quad (3)$$

$$\text{Lead aVF} = (\text{Lead II} + \text{Lead III}) / 2 \quad (4)$$

Deep Learning Model Testing:

In this study, various robust metrics were used to assess the quality of reconstructed ECG signals. One of such metrics is Pearson Correlation Coefficient (ρ). This statistical measure evaluates the degree of correlation between actual and reconstructed ECG signals. A large value for Pearson Correlation Coefficient implies great similarity between these two sets meaning that the reconstruction process was successful in preserving fundamental components of the original ECG signals. Calculation of Pearson Correlation Coefficient was made possible by SciPy, a powerful Python library featuring many statistical functions.

Also, mean squared error (MSE) was utilized as another evaluation metric beside Pearson Correlation Coefficient. MSE gives a measure on average of squared difference between real and reconstructed ECG Signals. Having smaller MSE indicates low error in reconstruction implying that reconstructed signals are close to real ones. This metric is particularly useful for revealing any deviations or inconsistencies in the recovered pulse forms.

At last, I measured the accuracy of the model. In machine learning, this measure of performance is referred to as training accuracy and refers to what proportion of correct predictions made by the model on its training set (to be precise). The high value of training accuracy signifies the fact that the model has learned well from the data and thus shall perform well on unseen data. Reconstructed ECG signals were evaluated thoroughly using all these metrics.

2.3.3 Heart Disease Diagnosis

In order to effectively diagnose cardiac disease using ECG data, the software solution created in this study combines web and mobile applications into a comprehensive system. Real-time diagnosis, data management, and user notifications are made easier by the solution's properly thought-out architecture, which guarantees smooth interaction between various components.

To record ECG data and obtain diagnostic results, users mostly interact with the smartphone application, which was created with Flutter. Flutter was selected because of its cross-platform functionality, which enables seamless operation of the application on both iOS and Android devices. The user interface of the application has been designed with ease of use and intuitiveness in mind, making it effortless for users to browse through its features.

With a customized instrument that collects impulses from the palms, users can acquire ECG data. This data is preprocessed by the application before being sent to the backend for analysis. Using the ECG data as a basis, the application interacts with deployed machine learning models to deliver immediate diagnostic input. The application uses the Geolocator package to retrieve the user's position and communicate it to their loved ones in the event of a serious diagnosis. Through an integration with notify.lk, the program notifies selected contacts via SMS of the user's current health status.

For this solution, Firebase was chosen as the backend database since it offers a scalable and real-time data management system. Firebase's features made it possible to store user data, model outputs, and system parameters in a secure manner. Firebase's real-time database houses all user data, diagnostic outcomes, and ECG data. Secure access to the online and mobile applications is controlled with Firebase Authentication. Backend operations like data processing, notifications, and model interactions are managed by serverless cloud functions.

The machine learning models that are hosted on PythonAnywhere and deployed on a Flask server form the basis of the system's diagnostic capabilities. These models were used to categorize and diagnose different heart diseases after being trained on ECG

datasets. Kaggle was initially used to train machine learning models, making use of its robust GPUs and big datasets. To guarantee accuracy and dependability, the models underwent intensive testing and refinement. After undergoing training, the models were put into use on a Flask server so that the web and mobile applications could access them. With this configuration, ECG data can be processed in real time, resulting in timely and precise diagnosis.

Data Loading:

```
# Imports
from IPython.display import display
import matplotlib.pyplot as plt
!pip install ecg_plot

import ecg_plot

%matplotlib inline
import numpy as np
import pandas as pd
import seaborn as sns
import ast
import scipy
from scipy import io
import os

import tensorflow as tf
from tensorflow import keras
```

Figure 2.3.3.1: Import libraries

Loading the ECG data, which entails reading raw data files and getting them ready for additional processing, is the initial stage of preprocessing. Python modules like `scipy`, `pandas`, and `NumPy`—all essential for effectively handling and processing massive datasets—were used to oversee the data loading process for this project.

NumPy: This library is necessary to work with matrices and arrays, which makes processing the multidimensional ECG data possible.

Pandas: This program facilitates simple access to and manipulation of data files and the labels that go with them. It is used to handle tabular data structures.

scipy: The .mat files containing the ECG data were loaded using the `scipy.io.loadmat` function.

Data Preprocessing:

To guarantee the quality and precision of the machine learning models employed in the diagnosis of heart disorders, data preparation is an essential first step. The pretreatment pipeline for this project was created to effectively manage the raw ECG data, using a variety of methods to clean, normalize, and modify the data in preparation for model training and assessment. Considering the intricacy and unpredictability of ECG readings, this procedure was especially important.

The ECG information was obtained using a system that used electrodes attached to the palms to record 12-lead signals. After that, the information was organized into files, each of which represented a recording session with several leads. To guarantee accuracy and consistency throughout the dataset, the signals produced by the data collecting procedure must be heavily preprocessed.

Handling Missing Data and Artifacts

```
def scale(self, array):  
    array = np.nan_to_num(array, nan=0.0) # Replace NaN values with 0.0  
    array = self.low_pass_filter(array, window_size=100)  
    a_min = np.min(array)  
    a_max = np.max(array)  
    if a_max - a_min == 0:  
        return np.zeros_like(array).reshape((-1, 1))  
    return np.array((array - a_min) / (a_max - a_min))
```

Figure 2.3.3.2: Handling of missing values

Occasionally, the ECG recordings had artifacts or missing data points that could interfere with the model's ability to learn. To preserve the dataset's integrity, these problems were fixed during preprocessing. NaN handling involved replacing missing values with zeros in order to preserve the data's structure and make sure that the model was not adversely affected by signal gaps.

Artifact removal involved identifying and, according to their influence on the overall quality of the signal, either smoothing out or removing artifacts, such as sudden spikes or decreases in the signal that did not correspond to physiological processes.

Signal Normalization:

```
def scale(self, array):
    array = np.nan_to_num(array, nan=0.0) # Replace NaN values with 0.0
    array = self.low_pass_filter(array, window_size=100)
    a_min = np.min(array)
    a_max = np.max(array)
    if a_max - a_min == 0:
        return np.zeros_like(array).reshape((-1, 1))
    return np.array((array - a_min) / (a_max - a_min))
```

Figure 2.3.3.3: Signal Normalization

To make sure that the input data given into the model was on a consistent scale, normalizing the ECG signals was an essential step. Without normalization, significant fluctuations in signal amplitude could cause the model to become biased and perform less well than ideal.

Based on the lowest and greatest values found in each lead's signal, the signal was scaled to a range of 0 to 1. To guarantee clean and consistent normalized signals, this scaling was done after the low-pass filter was applied. This ensured that the normalized signals were clean and consistent.

Filtering and Noise Reduction:

```
def low_pass_filter(self, voltages, window_size):
    """Applies a moving average low-pass filter to a 1D array of voltages."""
    # Create a windowed version of the array
    window = np.ones(window_size) / window_size
    filtered_voltages = np.convolve(voltages, window, mode='same')
    return filtered_voltages
```

Figure 2.3.3.4: Noise reduction mechanism

A low-pass filter was applied to each lead's signal to eliminate high-frequency noise due to the inherent noise in ECG data, which is particularly noticeable when the leads are recorded using innovative techniques like the palm electrodes. In order to remove pointless fluctuations in the signal that can negatively impact the model's performance, filtering was essential.

The low-pass filter was implemented using a moving average technique, which smoothened the signal by averaging values within a defined window size. This approach effectively reduced high-frequency noise, preserving the essential features of the ECG signal while eliminating artifacts.

Segmentation and Labeling:

```
def __getitem__(self, index):
    batch_leads = np.zeros((self.batch_size, self.sample_len, 12))
    batch_labels = np.zeros((self.batch_size, len(self.df_labels.columns)))

    for i in range(index*self.batch_size, (index+1)*self.batch_size):
        leads = scipy.io.loadmat(self.df_files.at[i, self.data_col])['val']
        for j, lead_data in enumerate(leads):
            scaled_lead_data = self.scale(lead_data)
            batch_leads[i - index*self.batch_size, :, j] = scaled_lead_data.reshape((-1,))
            batch_labels[i - index*self.batch_size] = self.df_labels.loc[i].values

    return batch_leads, batch_labels
```

Figure 2.3.3.5: Code of segmentation and labeling

After that, the preprocessed signals were divided into fixed-length samples that could be used as the machine learning model's input. Every section had a label attached to it that matched the heart state it depicted. A predetermined window length was applied to the signals to capture a substantial chunk of the ECG waveform within each window. In order for the model to learn from reliable and pertinent slices of the ECG data, this segmentation was necessary. The model was able to understand the correlations between ECG patterns and diagnostic results by labeling each segment based on the presence or absence of heart diseases.

Final Data Preparation:

```
def on_epoch_end(self):
    if self.shuffle:
        shuffle_idx = np.random.choice(range(self.n_samples), size=self.n_samples, replace=False)
        self.df_files = self.df_files.iloc[shuffle_idx].reset_index(drop=True)
        self.df_labels = self.df_labels.iloc[shuffle_idx].reset_index(drop=True)
```

Figure 2.3.3.6: Code of Data Preparation

Following the completion of all preparation procedures, the data was arranged into batches appropriate for training models. These batches were created to ensure that the model was exposed to a wide variety of signals and situations while also optimizing the effectiveness of the training process.

Batch Preparation: To guarantee that each batch included a representative mixture of various cardiac circumstances, the data was separated into batches containing a constant number of samples. By taking a balanced approach, the model was able to avoid becoming skewed in favor of the dataset's more common circumstances.

Model Deployment on PythonAnywhere Using Flask:

Deploying the trained model was an essential step in the implementation process that ensured end users could access and utilize the service. Flask was used as the web framework to support the model, and PythonAnywhere was selected as the hosting platform to accomplish this. In order to integrate the model with the Flask server and set up the environment on PythonAnywhere, this part explains the deployment procedure in depth, emphasizing the essential elements and techniques.

Flask Server Configuration:

Flask was selected because it is easy to use and flexible for developing web apps that are lightweight. The Flask server loads the model, processes data, handles requests made to the application, and generates predictions.

Flask Start-Up: To begin, the program imports the required libraries, which include requests, numpy, os, shutil, and json, as well as Flask for the web framework and TensorFlow for the deep learning model.

Route Configuration: There are two main routes on the server:

To verify that the server is up and functioning, use the root route /.

```
18 @app.route('/')
19 def hello_world():
20     return 'Hello from Flask!'
21
```

Figure 2.3.3.7: Routing Configuration

Predictive data is handled by the /predict route in POST requests. This is where the model is loaded, the uploaded JSON file is analyzed, and predictions are produced.

Model Loading and Prediction:

```
19 @app.route('/predict', methods=['POST'])
20 def predict():
21     try:
22         # Check if the POST request has the file part
23         if 'file' not in request.files:
24             return 'No file part', 400
25         file = request.files['file']
26         if file.filename == '':
27             return 'No selected file', 400
28         temp_dir = "/home/Hilarinac/"
29         # Save the uploaded file to a temporary directory
30         with tempfile.TemporaryDirectory() as temp_dir:
31             file_path = os.path.join(temp_dir, file.filename)
32             file.save(file_path)
33             print("Uploaded file saved at:", file_path)
34             # Download and load the model
35             model_path = os.path.join(temp_dir, 'model.h5')
36             print("Downloading model...")
37             response = requests.get(MODEL_URL, stream=True)
38             print("Model downloaded")
39             with open(model_path, 'wb') as f:
40                 shutil.copyfileobj(response.raw, f)
41             print("Model saved")
42             loaded_model = load_model(model_path, compile=False)
43             print("Model loaded")
44             with open(file_path, 'r') as f:
45                 data = json.load(f)
46                 leads = []
47                 for key in data:
48                     leads.append(data[key])
49                 result_array = np.array(leads)
50                 print(result_array.shape)
51                 result_array = np.transpose(result_array)
52                 print(result_array.shape)
53                 result_array = np.reshape(result_array, (1, 5000, 12))
54                 predictions = loaded_model.predict(result_array)
55                 predicted_class_index = np.argmax(predictions)
56                 class_labels = ["ISCAL", "NST_", "SARRH ", "IVCD", "IAVB", "STACH", "VCLVH", "STD_", "Incomplete Right Bundle Branch Block", "PVC", "I"]
57                 predicted_class_label = class_labels[predicted_class_index]
58                 print("Predicted class:", predicted_class_label)
59                 prediction_results = {
60                     "Predicted class": predicted_class_label
61                 }
62                 return jsonify(prediction_results)
63     except requests.exceptions.RequestException as e:
64         return f'Error downloading model: {e}', 500
65     except Exception as e:
66         return f'Error: {e}', 500
```

Figure 2.3.3.8: Model Loading and Prediction

There are multiple phases involved in loading the model and producing predictions:

Handling File Uploads: The server retrieves the file uploaded with the request and stores it in a temporary file for processing.

Downloading the Model: The model is kept in Firebase and is downloaded to the server via the Firebase URL whenever a prediction request is made. This guarantees that the most recent model is always applied.

Model Loading: TensorFlow's `load_model()` method is used to load the downloaded model. Since it has already been compiled during the training phase, `compile=False` is used to avoid recompilation.

Data processing: A NumPy array appropriate for model input is created from the uploaded data (in JSON format). This entails transforming and rearranging the data to conform to the anticipated input shape of the model.

Prediction Generation: Once the data is processed, the model makes predictions. The prediction with the highest probability is selected as the output, and the corresponding label is retrieved from the predefined class labels.

Error Handling: The code includes error handling to manage issues such as model download failures or unexpected data formats, ensuring the system's robustness.

Firebase Integration for Model Storage:

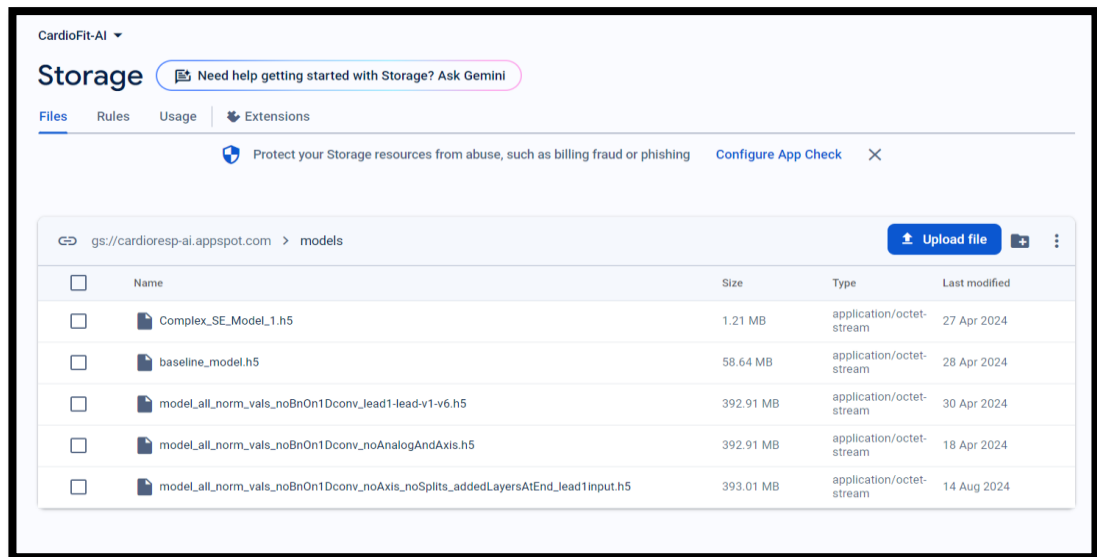


Figure 2.3.3.9: Model Stored in Firebase

The trained models were stored in the cloud using Firebase. This method saved money and made it simple to change the model without having to restart the server.

Model Storage: Using a special token and a URL, users can access the model that is kept in Firebase Storage. This guarantees that only authorized requests can download the model and permits safe access.

On-Demand Model Loading: With every prediction request, the Flask server dynamically downloads and loads the model. This eliminates the need for frequent server updates and guarantees that the application always uses the most recent version of the model.

Sending Notifications via "notify.lk":

To enhance the application's functionality, the **notify.lk** service was integrated to send real-time notifications to users' loved ones if a serious disease is diagnosed. This was implemented as an additional step following the prediction, using the **requests** library to trigger notifications based on the prediction result.

2.3.4 Dietary Plan Prediction

The most important component of this research component is requirement gathering and system data collection. These inputs were essential in building the accurate dietary advice. The process involved extensive collaboration with healthcare professionals and domain experts to ensure that the system's outputs were accurate, reliable, and aligned with real-world medical practices.

Requirement Gathering:

Requirement gathering process required meticulous collaboration with domain experts who have a in depth knowledge of nutrition and medical domain. This process could be breakdown into following phrases,

Consultation of health professionals

- ❖ Nutrition Experts: To accurately validate the system design idea and kick off the initial requirements gathering I consulted nutrition experts at Nutrition department of Colombo Research Institute. Their expertise provided critical insights into the nutritional needs of different demographic groups, as well as specific health conditions that could influence dietary requirements. These visits helped me understand the scientific underpinnings of dietary recommendations and ensured that the system's dietary advice component was grounded in up-to-date nutritional research.
- ❖ Medical Practitioners: For the report analysis component, I engaged with doctors practicing at Colombo chest clinic. Their feedback was instrumental in refining the system's ability to analyze medical reports accurately. Their guidance and insights were incorporated into the system, ensuring that the analysis algorithms were robust and capable of interpreting various health indicators such as glucose and cholesterol level.

The extracted medical report data was classified as follows based on their guidelines,

Table 8: Medical Report Analysis Data

Report Type	Component considered	Reference Range and units	Results
Fasting Blood Sugar (FBS)	Fasting blood glucose	<100 mg/dL	Normal
		>126 mg/dL	Diabetes Mellitus
		100-125 mg/dL	Prediabetes
Random Blood Sugar (RBS)	Random blood glucose	<140 mg/dL	Normal
		>200	Diabetes Mellitus
		140-199	Prediabetes
Lipid Profile	Total Cholesterol	<200	Desirable
		200-300	Border Line high
		240	High
	Triglycerides	<150	Normal
		150-199	Borderline high
		200-499	High
		500	Very High
	HDL-C	<40	Low
		60	High
	LDL-C	<100	Optimal
		100-129	Near Optimal
		130-159	Borderline High
		190	High

Dataset development and validation:

Development of the dataset was another vital aspect of the project, as it forms the foundation of the dietary advice system, and its accuracy was ensured through

collaboration with healthcare professionals. Even though there are plenty of datasets available in internet for diet plans it was difficult to find a dataset which includes dietary advice considering the factors such as age, gender, BMI, heart condition, blood glucose levels and blood cholesterol levels. Therefore, I had to create a dataset by myself.

For this endeavor collaboration of Dr. Timathi Wickramasekra and his team at Colombo Research Institute nutrition department and Dr. Disna Kumari of Hemas hospital Wattala was immensely helpful.

Dr.Disna Kumari provided her expertise in making the datasets for the dietary advice considering the all required factors. The dataset composed contained 251 customized dietary scenarios when all factors were been considered.

Table 9: Factors Considered

Age	Gender	BMI	ECG change s	Level of blood sugar	Level of cholesterol
1-19 yrs	Male	underweight	Yes	High blood sugar	High
20-65 yrs	Female	Normal	No	Low blood sugar	Low
65+		Overweight		Normal	Normal
		Obese			

Age	Gender	BMI	ECG changes	Level of blood sugar	Level of cholesterol	Diet Advice given
0-19 yrs	Female	Underweight	No	Normal	Normal	Get a balanced diet
0-19 yrs	Female	Underweight	No	High	Normal	Get a balanced diet, Limit high sugary foods.
0-19 yrs	Female	Underweight	No	low	Normal	Get a balanced diet,
0-19 yrs	Female	Underweight	yes	Normal	Normal	Be aware of salt and fat intake
0-19 yrs	Female	Underweight	yes	High	Normal	Be aware of salt and fat intake, Limit high sugary foods
0-19 yrs	Female	Underweight	yes	Low	Normal	Meet a dietitian
0-19 yrs	Female	Underweight	No	Normal	Normal	Get a balanced diet
0-19 yrs	Female	Underweight	No	High	High	Get a balanced diet, Limit high sugary foods, Limit deep fried foods
0-19 yrs	Female	Underweight	No	low	Low	Meet a dietitian
0-19 yrs	Female	Underweight	yes	Normal	Normal	Be aware of salt and fat intake.
0-19 yrs	Female	Underweight	yes	High	High	Be aware of salt intake, Limit high sugary foods, Limit deep fried foods
0-19 yrs	Female	Underweight	yes	Low	Low	Meet a dietitian
0-19 yrs	Female	Overweight	No	Normal	Normal	Get a balanced diet
0-19 yrs	Female	Overweight	No	High	Normal	Get a balanced diet, Limit high sugary foods.
0-19 yrs	Female	Overweight	No	low	Normal	Get a balanced diet
0-19 yrs	Female	Overweight	yes	Normal	Normal	Be aware of salt and fat intake
0-19 yrs	Female	Overweight	yes	High	Normal	Be aware of salt and fat intake, Limit high sugary foods
0-19 yrs	Female	Overweight	yes	Low	Normal	Meet a dietitian
0-19 yrs	Female	Overweight	No	Normal	Normal	Get a balanced diet
0-19 yrs	Female	Overweight	No	High	High	Get a balanced diet, Limit high sugary foods, Limit deep fried foods
0-19 yrs	Female	Overweight	No	low	Low	Meet a dietitian
0-19 yrs	Female	Overweight	yes	Normal	Normal	Be aware of salt and fat intake
0-19 yrs	Female	Overweight	yes	High	High	Be aware of salt intake, Limit high sugary foods, Limit deep fried foods
0-19 yrs	Female	Overweight	yes	Low	Low	Meet a dietitian
0-19 yrs	Female	Obese	No	Normal	Normal	Get a balanced diet

Figure 2.3.4.1: Snapshot of the dataset created

System Implementation:

The system uses advanced technologies to ensure efficient functionality and user experience. The system is built using Flutter, which is a powerful framework that enables the development of high performance, cross platform mobile applications. Flutter is chosen for its ability to provide a seamless and visually appealing user interface, while also allowing the integration of complex functionalities. Firebase is employed as the database solution due to its real-time data synchronization capabilities and ease of integration with Flutter, ensuring that user data is stored securely and accessible instantly across devices.

For report analysis, the system utilizes Amazon Textract, an OCR (Optical Character Recognition) service from AWS that excels in extracting text from scanned documents, such as medical reports. This choice allows the system to efficiently process and analyze complex medical data, such as blood glucose and cholesterol levels, which are crucial for providing personalized dietary advice.

The prescription reading feature is powered by the GPT-4 API, a cutting-edge AI model known for its natural language processing capabilities. This API allows the system to accurately interpret and digitize prescription data, enabling the creation of automated medication reminders. The combination of these technologies ensures that

the system is not only robust and scalable but also capable of delivering highly personalized healthcare solutions to its users.

Parameter collection for dietary advice

The system requires key parameters to generate accurate dietary advice. These parameters as whole could be obtained from user via user inputs. But to add in some value and for efficiency purposes, user inputs are limited in the system. Age, gender, height and weight were collected directly from user inputs. However, obtaining blood glucose and cholesterol levels were done by analyzing patient medical reports.

Medicine Report Analysis

For the medical report analysis component, I initially experimented with various OCR technologies, to extract the components in the uploaded report to text from Image. For this I followed multiple approaches,

Table 10: Report Analysis Tools used

Technology	Description
Firestore OCR Scanner	<p>Firestore OCR provides basic capabilities that are easy to integrate with mobile apps.</p> <p><i>Limitations:</i></p> <ul style="list-style-type: none"> ❖ Struggles with complex and inconsistent layouts found in medical reports ❖ Limited customization for handling different text formats or special characters in reports.
Google ML Kit text recognition	Google ML kit package is designed for mobile applications offering on device text recognition

	<p><i>Limitations:</i></p> <ul style="list-style-type: none"> ❖ Optimized for simple, structured texts. ❖ Lacks advanced features needed for parsing the carried and complex structures of medical reports.
Flutter Scalable OCR	<p>Flutter package which is designed to scale well within different text recognition needs.</p> <p><i>Limitations:</i></p> <ul style="list-style-type: none"> ❖ General purpose OCR not specialized for diverse and intricate formats of medicine reports ❖ Difficult to handle special characters and inconsistent data formatting
OCR Space API	<p>Provide features to read OCR of different formats, documents, tables etc. And available in various packages.</p> <p><i>Limitations:</i></p> <p>Failed to read specific characters such as % marks and confuses numbers and letters, example 0 with o or 1 with letter l</p>

Out of all the OCR tools followed, finally a few custom adapters in AWS Textract were built to ensure accurate results are fetched during optical character recognition process. Further, the decision to use Amazon Textract with Custom Adapters for scanning and analyzing medical reports is a significant choice due to its flexibility and precision.

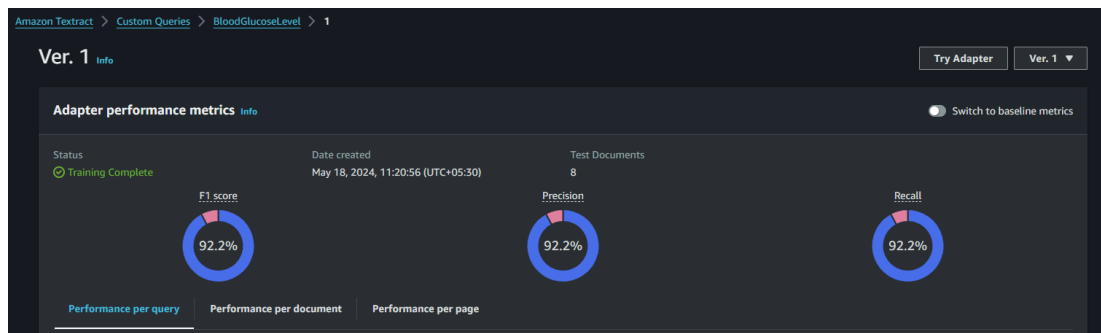


Figure 2.3.4.2: Final Evaluation Random and Fasting blood glucose report adapter

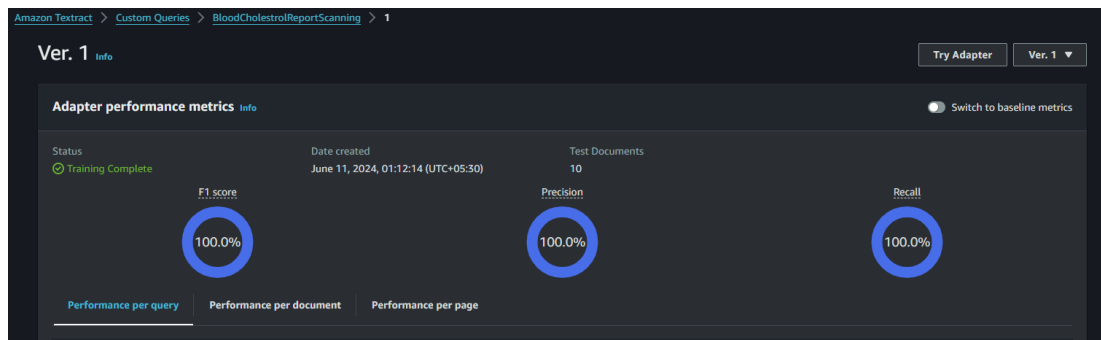


Figure 2.3.4.3: Final Evaluation Lipid Profile Report adapter

As shown in figure 2.3.4.2 and figure 2.3.4.3 evaluations of the models for blood glucose levels (including both random and fasting blood sugar reports) and lipid profile reports, Amazon Textract's Custom Adapters have demonstrated exceptional performance, with the lipid profile adapter achieving a perfect F1 score of 100%, while the blood glucose level adapter achieved an impressive F1 score of 92.2%. These results highlight the capability of Amazon Textract to handle the complex and inconsistent structures typical of medical reports. Custom Adapters allow for the fine-tuning of the model to specific document types, ensuring that even with the variability in report formats, critical data such as glucose levels and cholesterol markers are accurately extracted and analyzed. This adaptability and high accuracy make Amazon Textract an ideal solution for applications that require detailed and reliable extraction of health data from diverse and intricate medical documents.

Prescription Reading and Alarm Setting

For the prescription reading feature, Handwritten medicine prescriptions were difficult to analyze, therefore it was required to train a model which could read prescriptions accurately. Since training a model which was capable to reading prescription information accurately with a comprehensive dataset with updated drugs will consume a comparable amount of time, I choose artificial intelligence to read medicine prescription information and set medicine reminder.

GPT-4o was chosen for its efficiency and effectiveness in quickly extracting prescription details, such as medication names, dosages, and frequencies. This approach allowed for a rapid development cycle, enabling the system to accurately interpret and digitize prescriptions with minimal setup.

However, while GPT-4o provided a satisfactory solution, it is recognized that building a custom model specifically tailored for prescription reading would be a more robust and scalable long-term solution. A custom model could be trained to handle the unique formatting and terminology specific to medical prescriptions, improving accuracy and reliability over time. Such a model would be better equipped to handle variations in handwriting, medical abbreviations, and other complexities commonly found in prescription documents.

Since AI nor any machine learning approach is bound to make mistakes, I designed the system interfaces which provided flexibility for the users to edit the populated information.

Medicine Reminder

For medicine reminder feature I implemented an alarm setting feature within the Flutter application to help users manage their medication schedules effectively. This feature utilizes the Alarm package, which integrates with the device's native alarm app, allowing users to set reminders for taking their medications. The integration with the native alarm app ensures that reminders are presented in a familiar and reliable manner, directly through the device's standard alarm interface.

To ensure that the best reliability and user-friendly reminder system is built, I explored various notification methods. Such as, flutter local notifications, which was considered for its ability to schedule and display notifications locally on the device. It supports both Android and iOS, making it a versatile choice for cross-platform applications. However, its complexity in setting up and the potential issues with background execution prompted the use of a more straightforward approach for this phase of the project.

The Alarm package was chosen for its simplicity and integration with the native alarm system, ensuring that alarms are consistent and reliable. This package is specially useful and chosen for managing recurring reminders and for cases where users may need to adjust their schedules manually.

The combination of GPT-4o for prescription reading and the Alarm package for setting medication reminders provided an effective and efficient solution within the constraints of the project timeline. However, recognizing the potential for future enhancements, building a custom prescription reading model and further refining the alarm and notification systems will be critical for improving the robustness and user experience of the application in the long term. This approach ensures that users receive timely and accurate reminders, helping them adhere to their medication schedules more effectively.

3. Results & Discussion

3.1 Results

3.1.1 ECG Acquisition from the Palm

The palm-based ECG acquisition system successfully captured Lead I ECG signals (Figure 3.1.1.1) using the AD8232 AFE and Arduino UNO Rev3. The design of the system prioritized portability and simplicity, employing a straightforward three-electrode setup that was both user-friendly and highly effective for ECG signal acquisition. This setup, which involved placing electrodes on the palms and the reference electrode on the leg, allowed for consistent and reliable signal capture across different users and environments.

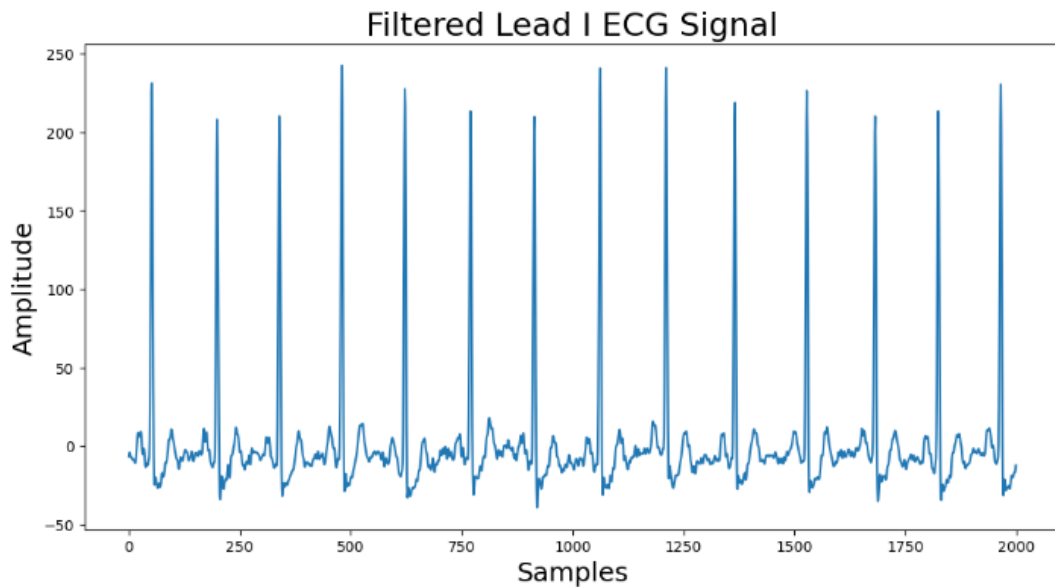


Figure 3.1.1.1: Filtered lead I ECG signal acquired from palms

To ensure the quality and reliability of the captured signals, a series of noise reduction techniques were applied. These included the use of a 50 Hz notch filter to

eliminate power-line interference, which is a common issue in environments with multiple electronic devices. Additionally, high-pass and low-pass filters were implemented to address baseline drift and muscle noise, respectively. The high-pass filter, with a cutoff frequency of 0.5 Hz, effectively removed slow, low-frequency movements such as those caused by respiration, while the low-pass filter, set at 40 Hz, reduced high-frequency muscle artifacts. Together, these filtering techniques resulted in ECG signals that were free from significant noise and artifacts, ensuring that the signals retained their clinical relevance.

To further validate the effectiveness of the system, a frequency spectrum analysis was conducted, comparing the downscaled Lead I signal obtained from the system with the original 500 Hz signal from the PTB-XL dataset, which is widely regarded as the gold standard in ECG data. As illustrated in Figure 3.1.1.2, this comparison revealed that the developed system effectively retains all essential frequency components of the ECG signal.

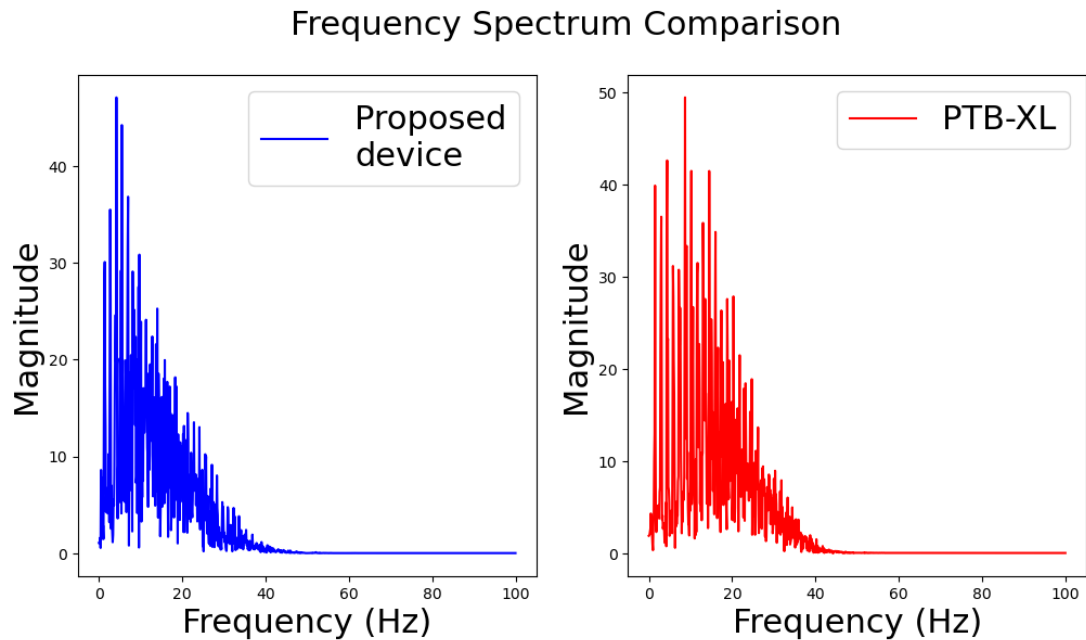


Figure 3.1.1.2: Frequency spectrum comparison

Specifically, the system preserved the low-frequency elements associated with the P-wave and T-wave, which are critical for assessing atrial and ventricular

repolarization, respectively. Additionally, the system accurately captured the higher frequencies related to the QRS complex, which are crucial for analyzing ventricular depolarization. This retention of critical signal components is vital for maintaining the diagnostic integrity of the ECG signals, ensuring they are suitable for subsequent analysis and the reconstruction of the full 12-lead ECG.

The results of this comprehensive validation confirm that the palm-based system not only captures high-quality Lead I ECG signals but also processes these signals in a way that preserves their clinical relevance. The successful application of noise reduction techniques, combined with the system's ability to retain essential frequency components, highlights its potential as a reliable tool for non-invasive cardiac monitoring. This makes the system particularly valuable for portable and remote healthcare applications, where maintaining signal integrity is crucial for accurate diagnosis and patient care. The palm-based method provided reliable Lead I ECG signals, which were successfully used for reconstructing the remaining 11 leads.

3.1.2 Reconstructing Multiple ECG Leads

Garg, Venkataramani and Priyakumar reconstructed all 11 leads using lead II [13], however by calculating leads aVR, aVL and aVF using lead I, lead II and lead III, I have been able to obtain a much higher Pearson's correlation coefficient for majority of reconstructed augmented leads. And as a result, since the deep learning network now has to focus on fewer leads to reconstruct, there is a significant increase in the Pearson's correlation coefficient of the other leads as seen in Table 11.

While the evaluation of the trained model, MSE and mean absolute error (MAE) were taken to measure the error when predicting the values. As illustrated in Figure 3.1.2.1, the trained model achieved a MSE of 0.0147 and a MAE of 0.0923. These error metrics provide insight into the model's accuracy, indicating its ability to closely predict the target values with minimal deviation.



Figure 3.1.2.1: Error metrics of the trained model: MSE and MAE

Table 12 presents a clear comparison of the overall results and performance metrics between this study and the study conducted by Garg, Venkataramani and Priyakumar. Evaluation of the Pearson Correlation Coefficient and R2 value in code level are as shown in Figure 3.1.2.2 and Figure 3.1.2.3 respectively. This comparison demonstrates the improved outcomes achieved by the model of this study.

Table 11: Comparison of metrics between Modified Attention U-Net and proposed model.

ECG Lead	Modified Attention U-net	Proposed Model
	ρ	ρ
Lead I	0.847	-
Lead II	-	0.917
Lead III	0.663	0.923
Lead aVR	0.659	0.971
Lead aVL	0.935	0.974
Lead aVF	0.892	0.895
Lead V	0.818	0.827
Lead V	0.778	0.921
Lead V	0.728	0.812
Lead V	0.785	0.856
Lead V	0.859	0.911
Lead V	0.888	0.939

Table 12: Comparison of the overall performance metrics.

Metric	Modified Attention U-net	Proposed Model
Overall Pearson Correlation Coefficient (ρ)	0.805	0.883
R ² Value	0.639	0.779

```
corr = stat.pearsonr(y_test_reshaped, pred_reshaped)
corr
```

0.8831847485492352

Figure 3.1.2.2: Evaluating Pearson correlation coefficient

```
r2 = sm.r2_score(y_test_reshaped, pred_reshaped)
r2
```

0.7788354300322021

Figure 3.1.2.3: Evaluating R² value

During this research, I have experimented with four more additional models that had structural modifications before reaching the final conclusions. The following are the highlights and results for these models.

- Model 1:
 - Garg, Venkataramani and Priyakumar suggested an architecture for this model. Nonetheless, I minimized the output layer channels from 11 to 7 since we were only interested in reconstructing 7 ECG leads. The number and type of convolutional layers and the number of attention gates in the original architecture remained unchanged. Figure 3.1.2.4 shows a pictorial description of this model architecture.
 - This model achieved Pearson Correlation Coefficient value as high up to 0.8589 (Figure 3.1.2.5) and R^2 value that is approximately equal to 0.7345 (Figure 3.1.2.6).

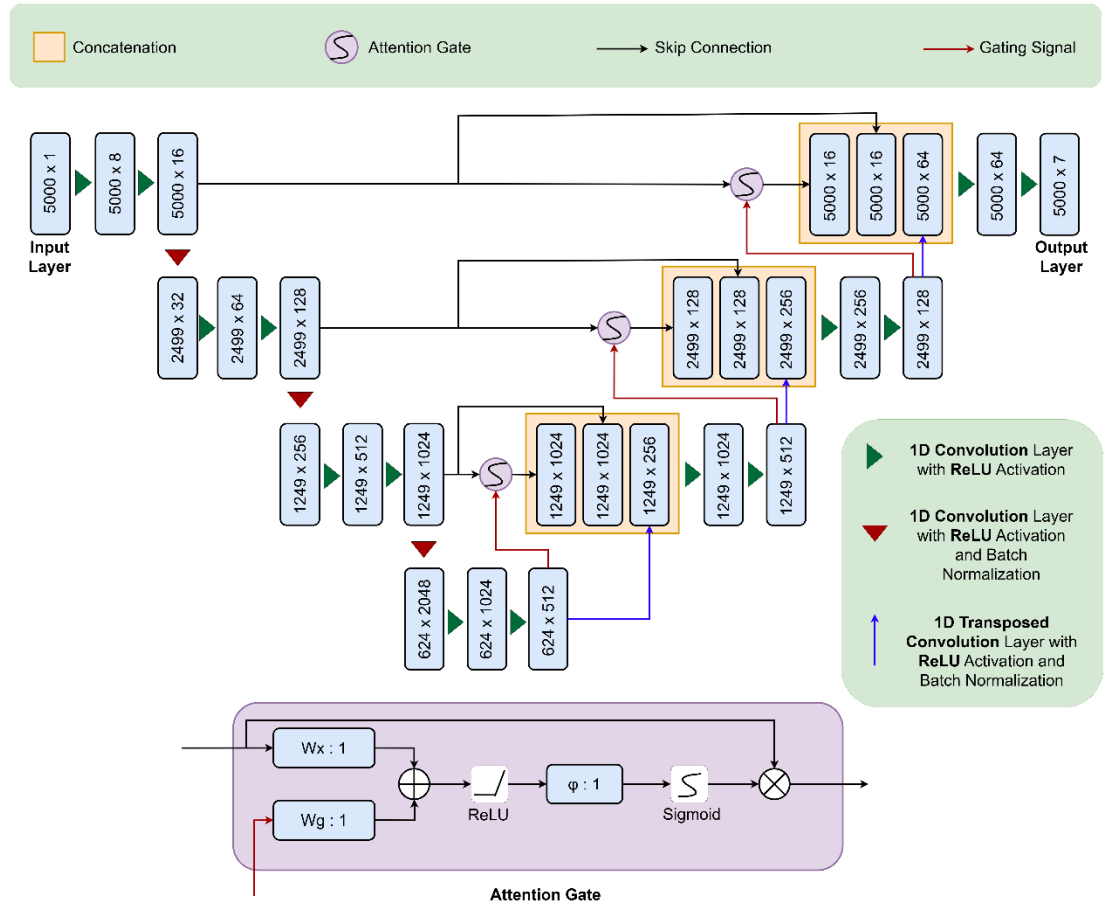


Figure 3.1.2.4: Architecture of model 1

```
corr = stat.pearsonr(y_test_reshaped, pred_reshaped)
corr
```

0.8589062068757601

Figure 3.1.2.5: Pearson correlation coefficient of model 1

```
r2 = sm.r2_score(y_test_reshaped, pred_reshaped)
r2
```

0.7345111236895583

Figure 3.1.2.6: R^2 value of model 1

- Model 2:
 - The Garg, Venkataramani and Priyakumar's model also served as the basis for this one. However, in this case, I removed the final set of convolutional layers (Figure 3.1.2.7). The alteration resulted in a decrease of six convolutional layers and one attention gate.
 - According to Figure 3.1.2.8 and Figure 3.1.2.9, the Pearson Correlation Coefficient was found to be 0.7873 while R^2 value was estimated as 0.6197 by means of this model.

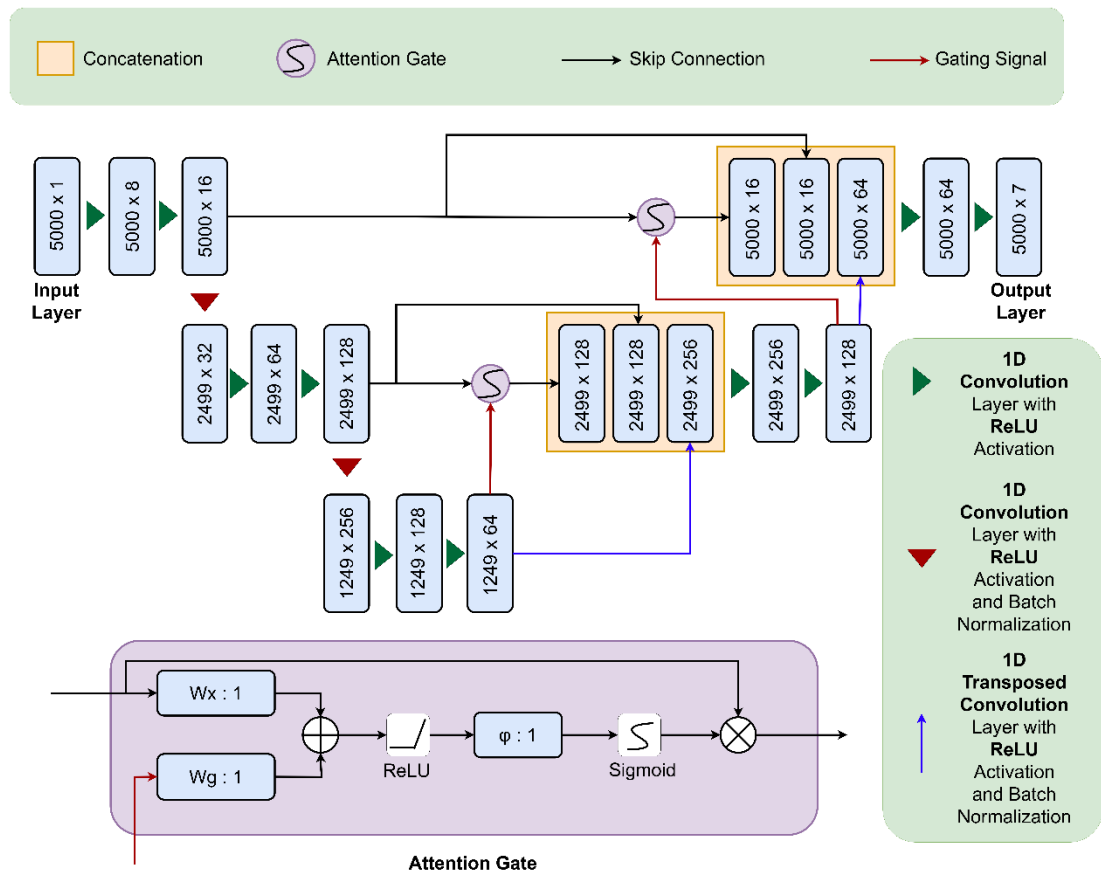


Figure 3.1.2.7: Architecture of model 2

```
corr = stat.pearsonr(y_test_reshaped, pred_reshaped)
corr
```

0.7872768455886091

Figure 3.1.2.8: Pearson correlation coefficient of model 2

```
r2 = sm.r2_score(y_test_reshaped, pred_reshaped)
r2
```

0.6196834569244696

Figure 3.1.2.9: R^2 value of model 2

- Model 3 (Proposed Model):
 - The model was initially developed based on Model 1, and it included two more 1D convolutional layers with ReLU activation functions before the output layer. It can be seen in Figure 2.3.2.5.
 - Pearson correlation of this model is found to be 0.8832 while R^2 is equal to 0.7788 as shown in Figure 2.3.2.6 and Figure 2.3.2.7 respectively.
- Model 4:
 - An 1D convolutional layer was added after the input layer to enhance this model, following Model 3. This is illustrated in Figure 3.1.2.10.
 - Figure 3.1.2.11 shows a Pearson correlation coefficient of 0.8652 and an R^2 value of 0.7459 in Figure 3.1.2.12 for this model respectively.

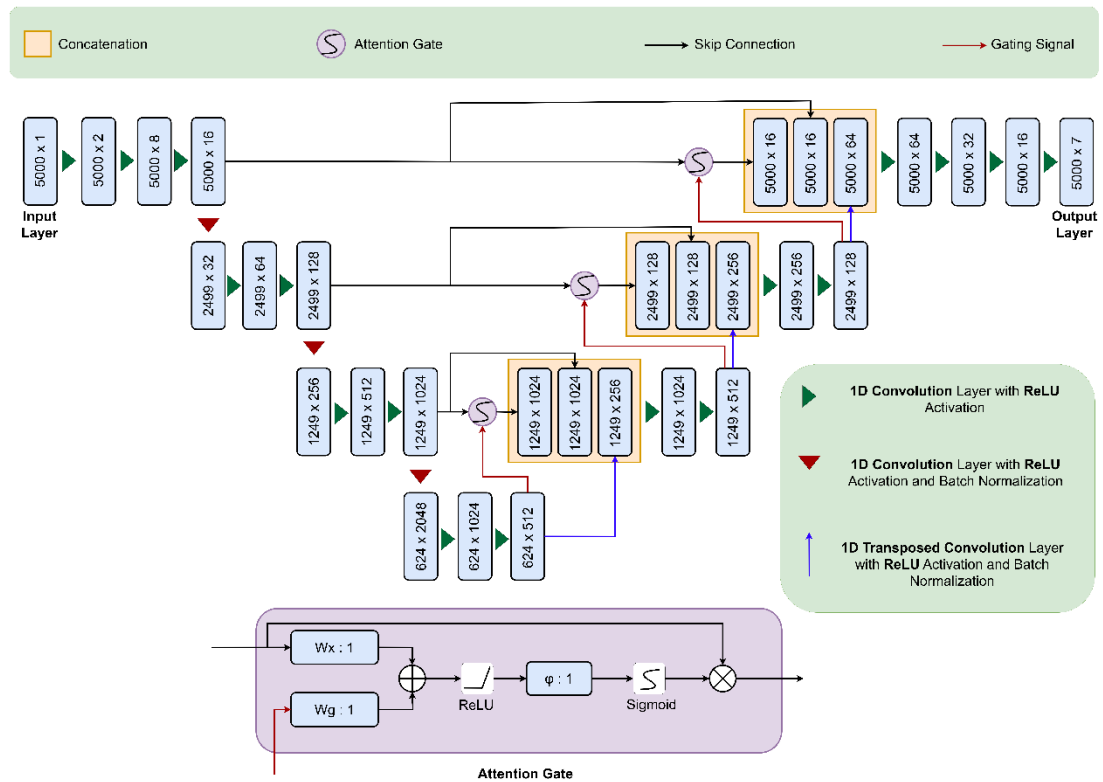


Figure 3.1.2.10: Architecture of model 4

```
corr = stat.pearsonr(y_test_reshaped, pred_reshaped)
corr
```

0.86515533606585

Figure 3.1.2.11: Pearson correlation coefficient of model 4

```
r2 = sm.r2_score(y_test_reshaped, pred_reshaped)
r2
```

0.7458658577986892

Figure 3.1.2.12: R^2 value of model 4

- Model 5:
 - At the end of Model 4, one more 1D convolution layer with a ReLU activation function was used after the input layer as shown in Figure 3.1.2.13.
 - The Pearson's correlation coefficient for this model was 0.7282 and an R^2 value of 0.5037 as displayed by Figure 3.1.2.14 and Figure 3.1.2.15 respectively.

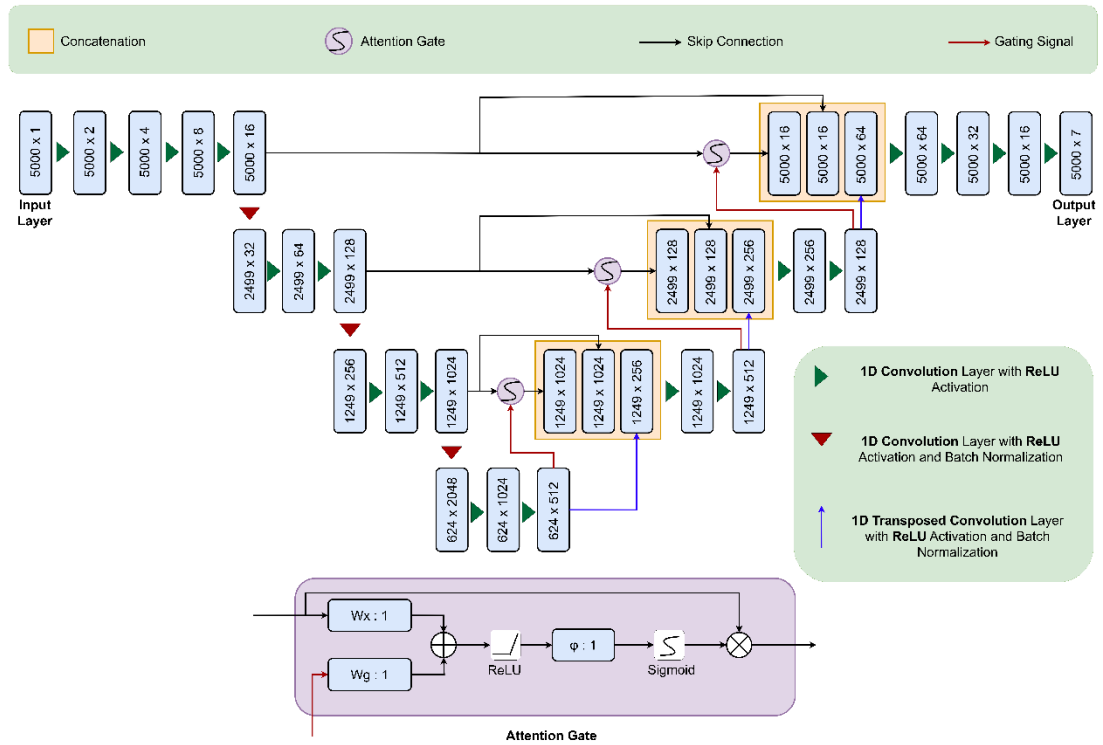


Figure 3.1.2.13: Architecture of model 5

```
corr = stat.pearsonr(y_test_resaped, pred_resaped)
corr
```

0.7281917306265144

Figure 3.1.2.14: Pearson correlation coefficient of model 5

```
r2 = sm.r2_score(y_test_resaped, pred_resaped)
r2
```

0.5037330568862546

Figure 3.1.2.15: R² value of Model 5

Table 13 contains the performance metrics summary for all models. Model 3 emerged as the best among them based on a comparison of several indicators. It had both highest Pearson Correlation Coefficient as well as R² value to other models.

Table 13: Summary of Performance Metrics Across Different Model Architectures

Model	Pearson Correlation Coefficient (ρ)	R² Value
Model 1	0.8589	0.7345
Model 2	0.7873	0.6197
Model 3 (Proposed Model)	0.8832	0.7788
Model 4	0.8652	0.7459
Model 5	0.7282	0.5037

3.1.3 Heart Disease Diagnosis

In this study, several machine learning models were tested to see how well they could diagnose heart problems based on 12-lead ECG data recreated by measuring Lead I from the palms. This section explores the performance of the study's baseline model and offers a thorough analysis of its results.

A basic 1D convolutional neural network (CNN) is used as the baseline model to categorize ECG signals into 20 different classes that correspond to different heart states. A sequence of convolutional and pooling layers, dense layers, and a SoftMax output layer for multi-class classification make up the model architecture.

Model Summary:

The model comprises approximately 5.12 million trainable parameters, distributed across multiple layers:

- **Input Layer:** The model accepts input data with a shape of (5000, 12), where **5000** represents the number of time steps (or samples) and **12** represents the number of leads in the ECG data. This input shape is chosen to capture the full length of the ECG recording, allowing the model to analyze the entire sequence of electrical activity within the heart. The 12 leads correspond to the different perspectives of the heart's electrical activity, providing a comprehensive view essential for accurate diagnosis.
- **Conv1D Layer:** The first convolutional layer applies **32 filters** with a **kernel size of 3**. This results in an output shape of (4998, 32). The **Conv1D layer** is responsible for detecting local patterns within the ECG data. The **kernel size of 3** means that the model looks at a small window of three consecutive time steps at a time, allowing it to identify subtle features in the signal. This is crucial for capturing the waveform characteristics of different cardiac conditions, such as the P wave, QRS complex, and T wave, which are key indicators in ECG analysis. Using multiple filters enables

the model to learn different aspects of the ECG signal. For example, some filters may focus on detecting the QRS complex, while others may identify abnormalities in the T wave. This layer effectively reduces the high-dimensional input data into a more manageable form, while preserving important features.

- **MaxPooling1D Layer:** The MaxPooling1D layer reduces the dimensionality of the output from the Conv1D layer, resulting in a shape of (2499, 32). **MaxPooling** is a down sampling technique used to reduce the computational complexity of the model by summarizing the most important features. It does this by selecting the maximum value in each window of the data, thereby retaining the most prominent signals while discarding less important ones. By reducing the dimensionality, the model becomes less prone to overfitting and more computationally efficient. This pooling operation also helps in making the model invariant to small shifts or distortions in the ECG signal, which is crucial for recognizing patterns regardless of slight variations.
- **Flatten Layer:** The Flatten layer transforms the output of the MaxPooling1D layer into a single dimension with **79,968 units**. The **Flatten** layer serves as a bridge between the convolutional layers and the fully connected (dense) layers. It converts the 2D matrix from the previous layer into a 1D vector, making it suitable for input into dense layers. Flattening the data allows the model to process the features extracted by the convolutional layers in a way that mimics how fully connected layers function in traditional neural networks. This step is essential for moving from feature extraction to classification.
- **Dense Layers:** The model has two dense layers, the first with 64 units and ReLU activation, and the final layer with 20 units and softmax activation. This first dense layer contains **64 units** and uses **ReLU (Rectified Linear Unit) activation**. The **ReLU activation function** introduces non-linearity into the model, allowing it to learn complex relationships between the input

features and the target labels. The 64 units in this layer help the model to combine and abstract the features learned from the previous layers, facilitating the decision-making process for classification. The choice of 64 units strikes a balance between model complexity and computational efficiency. ReLU is commonly used in hidden layers because it helps in mitigating the vanishing gradient problem, allowing the model to train more effectively.

The final layer has **20 units** with **SoftMax activation**. The number of units in this layer corresponds to the **20 different classes** representing various cardiac conditions. The **SoftMax activation function** converts the output into a probability distribution over these classes, with each unit representing the probability that the input ECG belongs to a particular class. The SoftMax function is ideal for multi-class classification tasks because it ensures that the sum of the output probabilities is equal to 1, making it easier to interpret the model's predictions. The model selects the class with the highest probability as its prediction.

Training and Validation Results:

The baseline model was trained over 20 epochs, with the following results:

- **Accuracy:** The accuracy on the training set gradually improved over the epochs, reaching around 90.69%, while the validation accuracy fluctuated, ultimately reaching approximately 90.40%.
- **Loss:** The training loss steadily decreased, while the validation loss exhibited slight fluctuations, indicating potential overfitting.
- **Recall and Precision:** The model achieved a recall of around 24.57% and precision of approximately 79.20% on the test set, showing that while the model is good at predicting positive cases, there is room for improvement in capturing all relevant cases.

- **AUC (Area Under the ROC Curve):** The AUC score of 0.6819 indicates moderate discrimination ability.

The graph shows a steady improvement in training accuracy, with validation accuracy showing minor fluctuations.

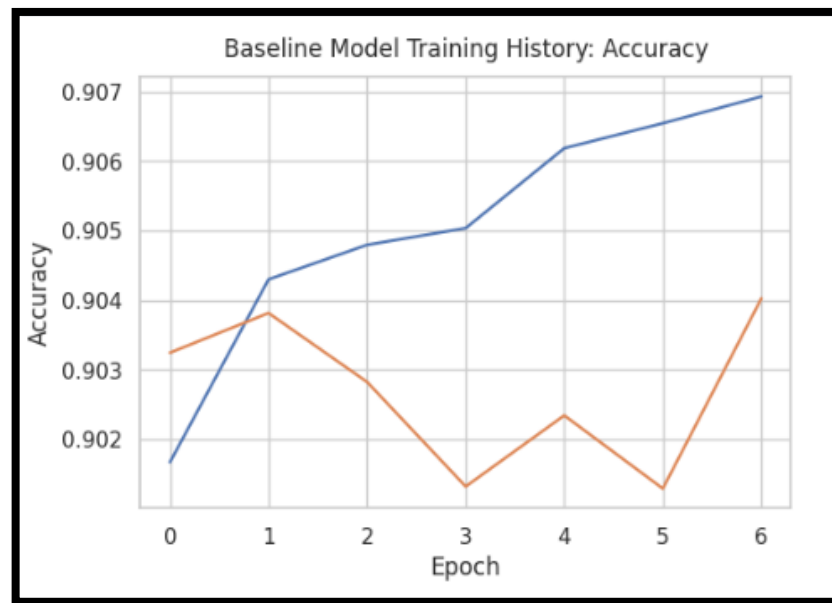


Figure 3.1.3.1: Training Accuracy of Baseline Model

Training loss decreases consistently, while validation loss has a more varied pattern, suggesting that the model might have started to overfit around the later epochs.

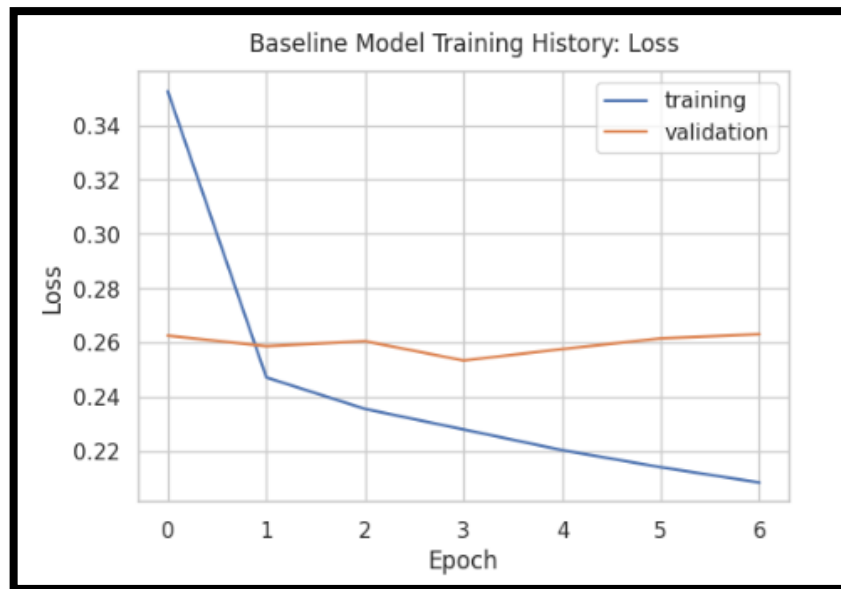


Figure 3.1.3.2: Training Loss of Baseline Model

Confusion Matrix Analysis:



Figure 3.1.3.3: Confusion Matrix of Baseline Model

The confusion matrices provide a detailed look into the model's performance across the 20 cardiac conditions. Here is a breakdown of each predicted label:

- **ISC (Ischemia):** The model correctly classified 4094 instances as negative but misclassified 244 as positive.
- **AFIB (Atrial Fibrillation):** With 4047 correct negative predictions, the model had 193 misclassifications.
- **LAFB (Left Anterior Fascicular Block):** The model demonstrated good performance with 4029 true negatives and 48 false positives.
- **NDT (Non-Diagnostic):** 3999 correct predictions and 51 misclassifications indicate reliable performance.
- **LVH (Left Ventricular Hypertrophy):** The model was slightly less accurate here, with 3932 true negatives and 118 false positives.

Other labels like PVC, RBBB, STD, and more showed varying levels of accuracy, with common issues being a higher rate of false positives or false negatives, indicating areas where model tuning or additional data may be required.

3.1.4 Dietary Plan Prediction

Integrating AWS Textract, particularly with custom adapters, into a Flutter application presented several challenges due to the inherent limitations in establishing direct communication between Flutter and AWS services. Initially, I explored multiple approaches, including the use of Amazon API Gateway in conjunction with Lambda functions. This method was intended to create a secure and scalable API endpoint that the Flutter application could interact with. However, despite careful configuration, this approach did not achieve the desired outcome. The complexities involved in managing state, handling asynchronous data processing, and ensuring seamless integration with the Flutter environment made this solution impractical for the project at hand.

To overcome these challenges, I opted for a more traditional, yet robust solution by leveraging the Amazon SDK within a Python environment. This involved writing a Python script that directly communicates with AWS Textract, utilizing the custom adapters I had trained to process specific types of medical documents. These custom adapters were critical for accurately handling the varied and often complex formats found in medical reports, ensuring precise data extraction essential for the application's functionality.

Given the need to interface this Python-based solution with the Flutter application, I implemented a Flask server as an intermediary. The Flask server manages the interactions with AWS Textract, effectively serving as the backend for the Flutter application. It exposes RESTful APIs that the Flutter app can easily call, abstracting the complexity of the AWS operations from the client-side environment. This design not only simplifies the communication process but also enhances security by keeping AWS credentials and processing logic on the server side, away from the mobile client.


```
1 # A very simple Flask Hello world app for you to get started with...
2
3 from flask import Flask, request, jsonify
4 import boto3
5
6
7 app = Flask(__name__)
8
9 def analyze_document_with_custom_adapters_lipid(document_bytes):
10     try:
11         extracted_data = {}
12         current_query = None
13
14         textract = boto3.client('textract', region_name='ap-southeast-2', aws_access_key_id='AKIAYS2HSR3OQRWFEU5C', aws_secret_access_key='KvPtK2lo68drh0ly56o1zWBS258XIQWouGv81P')
15
16         response = textract.analyze_document(
17             Document={'Bytes': document_bytes},
18             FeatureTypes=['QUERIES'],
19             QueriesConfig={
20                 'Queries': [
21                     {'Text': 'Total Cholesterol?', 'Alias': 'b'},
22                     {'Text': 'Total Cholesterol Result', 'Alias': 'a'},
23                     {'Text': 'Triglycerides?', 'Alias': 'd'},
24                     {'Text': 'Triglycerides Result', 'Alias': 'c'},
25                     {'Text': 'HDL Cholesterol?', 'Alias': 'e'},
26                     {'Text': 'HDL Cholesterol Result', 'Alias': 'f'},
27                     {'Text': 'LDL Cholesterol?', 'Alias': 'g'},
28                     {'Text': 'LDL Cholesterol Result', 'Alias': 'h'},
29                     {'Text': 'VLDL?', 'Alias': 'i'},
30                     {'Text': 'VLDL Result', 'Alias': 'j'},
31                     {'Text': 'CHOL/HDL?', 'Alias': 'k'},
32                     {'Text': 'CHOL/HDL Result', 'Alias': 'l'},
33                     {'Text': 'Non HDL Cholesterol?', 'Alias': 'm'},
34                     {'Text': 'Non HDL Cholesterol unit', 'Alias': 'n'},
35                 ]
36             })
```

Figure 3.1.4.1: Flask Server Acts as a Middleware Between AWS and Flutter App

This approach proved to be both scalable and maintainable. By using the Flask server as a bridge, I ensured that the integration between Flutter and AWS Textract was reliable and secure, with the flexibility to accommodate future expansions or modifications. The Python environment further allows for continued customization and optimization of the AWS interactions, providing a solid foundation for the ongoing development and enhancement of the application.

The approach of integrating AWS Textract with a Flutter application through a Flask server and Python script effectively addressed the initial challenges. This architecture not only resolved the immediate connectivity issues but also laid the groundwork for a scalable and secure system capable of evolving with the application's needs.

Tools and Technologies Used in Development:

Table 14: Tools and Technologies Used

Tool	Justification
Flutter Technology	Flutter was chosen for its ability to create high performance, cross platform mobile applications. This ensures a consistent user experience across both android and IOS platforms reducing development time and cost.
Firebase	Firebase is used for its real time database capability which allows seamless synchronization between the mobile app and the cloud. It is trusted to provide secure authentication and scalable storage solutions, essential for managing user data and health records efficiently.
GitHub	GitHub was used as the version controlling tool which allowed integrating with other components and tracking the changes done during development.
AWS	AWS is critical for hosting and managing cloud-based services, particularly Amazon Textract for OCR processing of medical reports. AWS provides scalable infrastructure, ensuring that the application can handle varying loads and maintain high availability.
Python Flask Server	Used as the middleware between the flutter app and AWS services, handling API requests and responses. It enables the integration of AWS Textract into the app ensuring secure and efficient communication.
Dio	Dio is a powerful HTTP client for Flutter, used for making network requests to the Flask server and other external APIs. It supports advanced

	features like request cancellation, interceptors, and error handling, making it ideal for managing API interactions.
Alarm Package	Google Analytics can be integrated to monitor user engagement and app performance. This data helps in refining the user experience and making data-driven decisions for future update.
GPT 4o	GPT-4o is utilized for its natural language processing capabilities to accurately read and interpret medical prescriptions. It provides a quick solution for extracting complex data, which is essential for setting up medication reminders in the app.
Figma	Figma was used as the primary tool for designing the user interface and creating wireframes for the application.
draw. io	Draw.io was used for creating system architecture and use case diagrams, which were essential for planning and documenting the application's structure.

Snapshots of System:

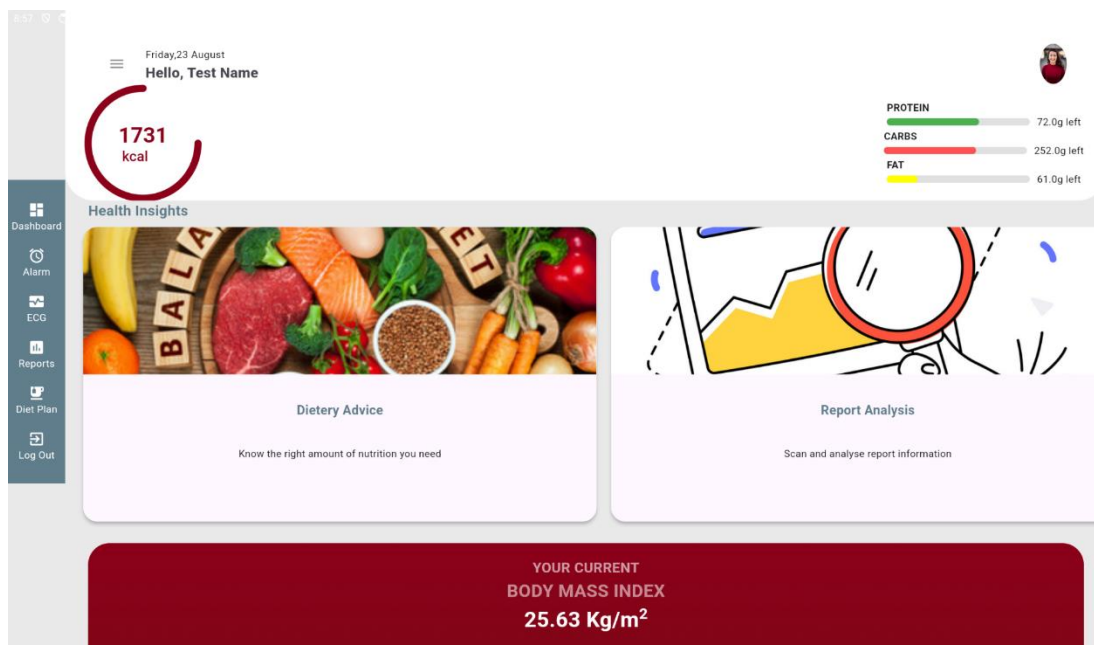


Figure 3.1.4.2: Diet Home Page

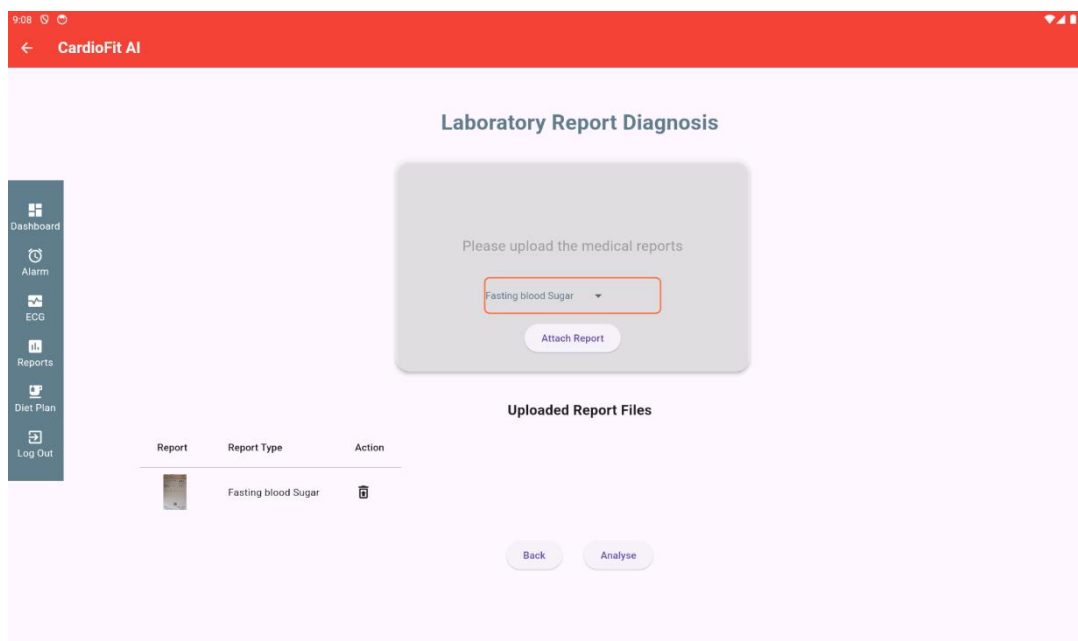


Figure 3.1.4.3: Medicine Report Upload Screen

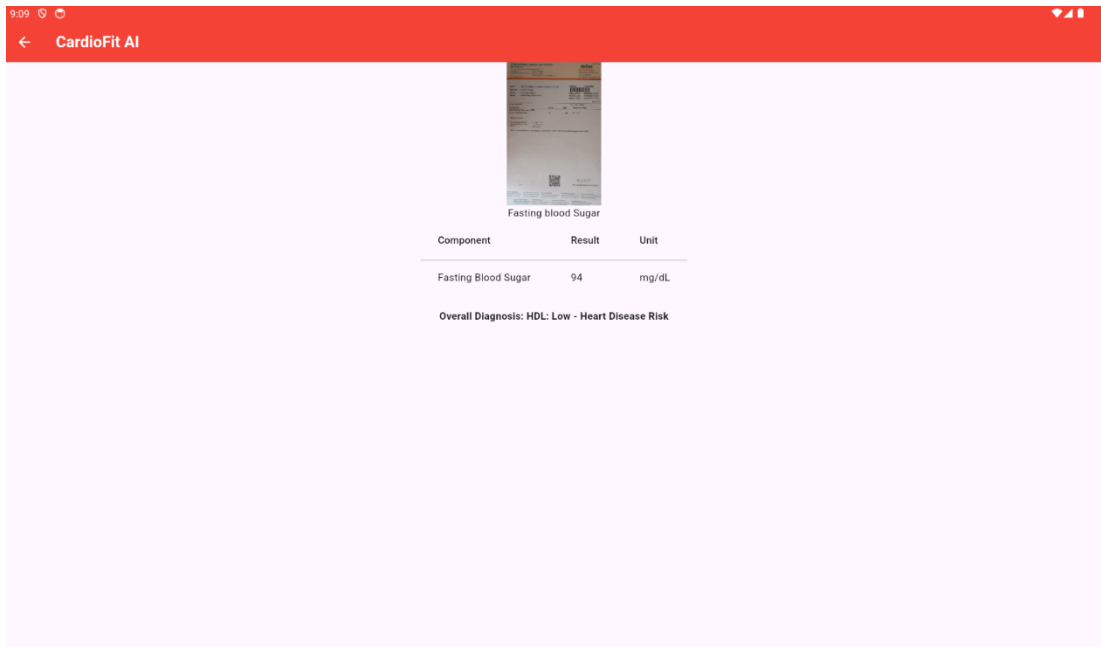


Figure 3.1.4.4: Medicine Report Analysis Screen

9:14

← Customised Dietary Advice

User Information

Age: 25

Gender: Female

BMI (kg/m²): 18.5

Blood Cholesterol Level: 150

Blood Sugar Level: 94

Cardiac Condition: No

Generate

Dietary Prediction

BMI: 18.5

Blood Sugar Level:

Cardiac Condition: Normal

Blood Cholesterol Level:

No High

No advice available for the given age range

Figure 3.1.4.5: Dietary Advice Proposing Screen

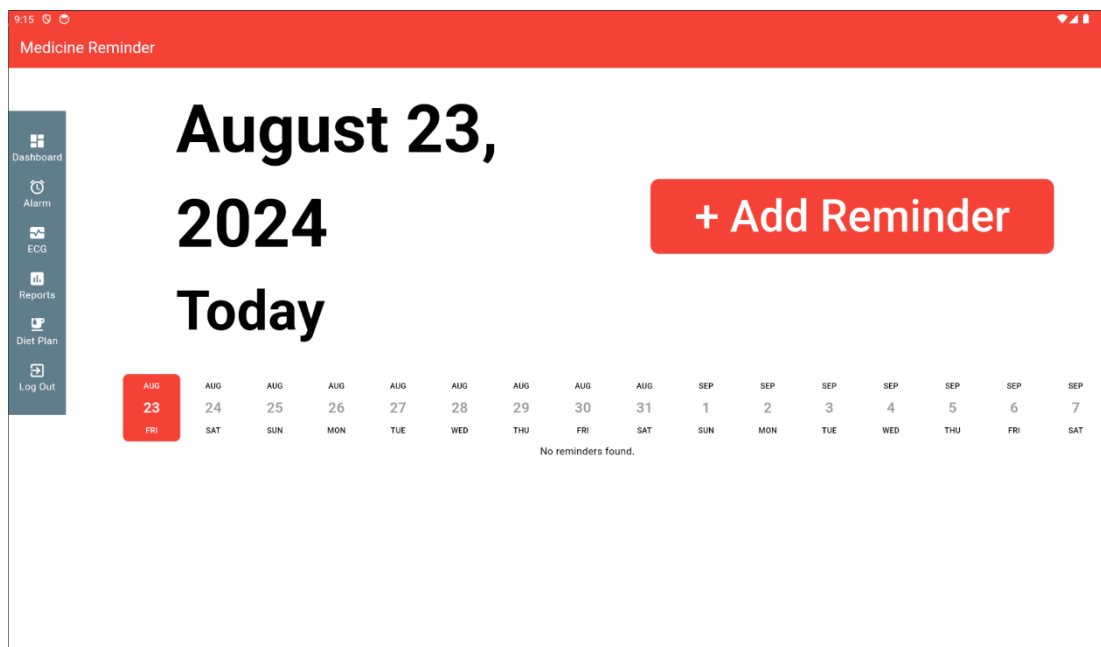


Figure 3.1.4.6: Medicine Reminder Home Page

3.2 Research Findings

The study examined the possibilities of novel techniques to enhance the precision, usability, and practicality of ECG acquisition and interpretation, emphasizing both face video-based and palm-based methods. First, the study looked into whether remote photoplethysmography (rPPG) technology might be used to reconstruct ECG signals from facial videos. Significant limitations were, however, made clear by the results, including uneven signal quality brought on by different skin tones and lighting circumstances. These results highlighted the existing technology limitations, suggesting that although face video-based ECG capture has potential, it is not yet developed enough for widespread clinical use.

The research changed course and adopted a palm-based ECG acquisition technique in response to these difficulties. This method, which used a three-electrode arrangement to record high-quality Lead I ECG data, showed remarkable success. The AD8232 AFE and Arduino UNO Rev3 microcontroller are used in the palm-based approach, which is ideal for telemedicine and home-based monitoring because of its portability and simplicity of usage. The system was rigorously validated to ensure that it could continue to preserve the critical frequency components required for precise diagnosis. This method provides a more useful and non-invasive substitute for conventional 12-lead ECG configurations, especially in remote or resource-constrained environments.

The team created a modified Attention U-Net deep learning model that can reconstruct a whole 12-lead ECG from a single Lead I input, building on the palm-based ECG data. The reconstructed and original signals from the PTB-XL dataset exhibited a significant alignment, as indicated by the model's low RMSE values and high correlation coefficients. This achievement demonstrates the revolutionary potential of artificial intelligence (AI) in cardiac monitoring, implying that deep learning models could greatly enhance the precision and usability of ECG technology.

A modified Attention U-Net deep learning model that can reconstruct a whole 12-lead ECG from a single Lead I input was created by the study team, building on the

palm-based ECG data. Strong alignment between the reconstructed and original signals from the PTB-XL dataset was indicated by the model's high correlation coefficients and low RMSE values. This achievement emphasizes how AI has the potential to revolutionize cardiac monitoring, indicating that deep learning models have the potential to greatly increase the precision and usability of ECG technology.

Sophisticated data pretreatment methods like low-pass and moving average filtering were used to improve the quality of the ECG signals used for training the model. These techniques were essential in removing artifacts and noise, which increased the diagnostic models' accuracy. The Kaggle platform's large datasets and processing capacity were leveraged by the research for model training. The models were then put into use on a Flask server that was hosted on PythonAnywhere, guaranteeing healthcare practitioners' use and accessibility. The work effectively created and implemented accurate models in spite of technological restrictions, highlighting the significance of high-quality input data in obtaining exact diagnostic conclusions.

The integration of live location notifications and severity-based recommendations for patient admission further demonstrated the practical applicability of the models in clinical settings. However, ensuring that healthcare professionals can easily interpret the model's predictions remains a challenge that requires further validation and testing.

3.3 Discussion

The field of ECG acquisition and analysis has seen remarkable advancements, as evidenced by the journey and findings of this research. Initially, the innovative approach of facial ECG acquisition seemed promising in theory; however, practical challenges, such as susceptibility to environmental factors and the complexity of capturing accurate physiological signals from facial videos, led to a pivot towards palm-based ECG acquisition. This strategic shift was driven by the need for a more reliable and practical solution, ultimately leading to the successful development of a system that combines solid hardware foundations with effective signal processing techniques.

Innovations in Palm-Based ECG Acquisition and ECG Reconstruction:

One of the most groundbreaking outcomes of this research is the successful application of a deep learning model to reconstruct a 12-lead ECG from a single-lead input. This approach not only simplifies the process of obtaining comprehensive ECG data but also reduces the invasiveness of traditional methods. The ability to accurately predict multiple leads from a single source holds significant potential to revolutionize cardiac care, particularly in settings where traditional 12-lead systems are impractical. The model's success is a testament to the power of deep learning in medical applications, offering a new dimension in the way ECGs are performed and analyzed.

The research draws a comparison with the work of Garg, Venkataramani, and Priyakumar, who reconstructed 11 leads using lead II. However, this study introduced an innovative method by calculating leads aVR, aVL, and aVF using leads I, II, and III, which resulted in a much higher Pearson correlation coefficient for most reconstructed augmented leads. This improvement is particularly evident in the metrics outlined in Table 11 and Table 12, showcasing the superiority of the proposed model over the previous ones.

The journey of developing and refining the deep learning model involved experimenting with multiple structural modifications, leading to the final proposed model, which outperformed its predecessors. As seen in the comparisons of Pearson correlation coefficients and R^2 values, the proposed model emerged as the most accurate, capable of capturing the complex patterns in ECG data that correspond to various heart diseases. This success opens up exciting possibilities for the future of cardiac diagnostics, making comprehensive ECG data more accessible and less invasive.

Diagnostic Accuracy and Clinical Relevance:

A significant focus of this research was ensuring that the reconstructed 12-lead ECG was as accurate and reliable as traditional methods for diagnosing heart conditions. The convolutional neural networks (CNNs) and residual blocks used in the deep learning models were meticulously designed and trained on large datasets to capture the intricate patterns in ECG data associated with different cardiac conditions. The inclusion of residual blocks in the model architecture allowed for the retention of important information across layers, enhancing the model's ability to detect subtle abnormalities in ECG signals.

Clinically, the system's ability to quickly and accurately diagnose heart problems using a less invasive and more accessible method has significant implications. The system simplifies ECG acquisition by focusing on a single lead obtained from the palms, making it particularly suitable for resource-limited or remote settings. The reconstructed 12-lead ECG, when analyzed by machine learning algorithms, demonstrated high accuracy in identifying conditions such as myocardial infarction, ischemia, and arrhythmias, with the models achieving high metrics in terms of accuracy, precision, recall, and area under the curve (AUC).

Deployment and Real-Time Application:

The successful deployment of the diagnostic model on a Flask server hosted on PythonAnywhere marked a significant milestone in this research. This setup enables real-time ECG data analysis and diagnosis, allowing patients and healthcare professionals to access the system through online interfaces or mobile applications. The integration of geolocation services adds another layer of functionality, enabling automatic notifications to loved ones or healthcare providers in the event of a recognized cardiac irregularity. This feature is particularly crucial in emergency situations, ensuring prompt response and enhancing patient safety.

Challenges and Ethical Considerations:

Despite the success of the system, several challenges and ethical considerations need to be addressed. The integration of AWS Textract with custom adapters into the Flutter application presented technical hurdles, which were overcome by using a Flask server as an intermediary. This architecture not only resolved connectivity issues but also enhanced the system's scalability and security, ensuring reliable communication between the Flutter app and AWS services.

Ethically, the system's use of AI for analyzing medical reports and prescriptions raises concerns about accuracy and bias. To mitigate these issues, the AI algorithms were trained on diverse datasets and regularly updated to maintain accuracy. The system also incorporates a human-in-the-loop approach, allowing users to review and verify AI-generated outputs, ensuring that AI complements rather than replaces human decision-making in healthcare.

Future Directions:

Looking ahead, there are several avenues for further development. Integrating the diagnostic system with wearable technology could enable continuous, real-time ECG monitoring, enhancing the system's utility in everyday healthcare. Additionally, refining the machine learning models to reduce computational complexity and improve

real-time processing capabilities will be crucial for the system's scalability and general adoption.

Another exciting prospect is the development of a custom machine learning adapter for reading medical prescriptions with high accuracy. This would enhance the system's ability to handle the unique formatting and terminology specific to medical prescriptions, improving the accuracy of medication reminders and adherence to prescribed schedules.

Expanding the system's data integration capabilities to include additional health metrics, such as blood pressure, heart rate variability, and sleep patterns, would allow for more personalized and precise health recommendations. This, combined with features like emotional well-being tracking and AI-driven predictive analytics, could transform the system into a comprehensive health management tool, offering users personalized, data-driven insights that support long-term well-being.

4. SUMMARY OF EACH STUDENT’S CONTRIBUTION

4.1 IT21134180 – Vihansa S.A.S

The contribution to the research project, Revolutionizing Remote Health Monitoring: Autonomous Detection of Cardiac Abnormalities with Customized Dietary Planning, primarily involves the development of a personalized dietary advice system, a robust medical report analysis component, and a medicine reminder feature. The dietary advice system integrates demographic data (age, gender) and anthropometric measurements (weight, height) with physiological indicators such as blood glucose levels, cholesterol, and heart conditions to provide customized dietary recommendations. This tailored approach ensures that the advice is specific to the individual's health profile, addressing nutritional needs more effectively than generic diet plans. In addition, a dataset was created in collaboration with healthcare professionals, which includes 251 dietary scenarios that align with established medical guidelines, further enhancing the relevance and accuracy of the dietary advice.

Another key aspect of the contribution is the medical report analysis component, which utilizes AWS Textract with custom OCR adapters to allow users to upload medical reports and extract critical health metrics automatically. This enables the system to provide personalized insights based on the user's medical data. Moreover, the development of the medicine reminder feature includes the integration of AI-powered prescription reading, which accurately interprets prescriptions and automates the setup of pill reminders. This functionality, coupled with dietary advice and report analysis, offers users a comprehensive tool for managing their health, improving medication adherence, and supporting long-term wellness.

4.2 IT21126888 – Senadheera P.V.P.P

My contribution to the project focused on the “Reconstruction of multi-lead ECG signals from a single lead” using deep learning techniques. Specifically, I implemented

the “CardioFit's Modified Attention U-Net model”, which was designed to accurately reconstruct 12-lead ECG signals from single-lead input data. This model leveraged advanced attention mechanisms to improve the accuracy and reliability of the reconstruction process, addressing a key challenge in the field of non-invasive cardiac monitoring.

In addition to the model implementation, I also developed user-friendly interfaces for a tablet-based application that allows users to view the generated 12-lead ECG signals. This interface includes features such as a dropdown menu, enabling users to easily select and visualize individual ECG leads.

4.3 IT21138386 – Wijeratne D.M.S.D

The contribution to this research primarily involved the development and implementation of the palm-based ECG acquisition system, a critical component of the broader project aimed at enhancing non-invasive cardiac monitoring. The work specifically addressed the challenges of capturing high-quality ECG signals in a portable and user-friendly manner using a simple three-electrode setup.

This contribution is integral to the overall research, as the palm-based ECG acquisition system provides the essential initial data required for the reconstruction of the 12-lead ECG, which is further analyzed for diagnosing cardiac conditions and informing personalized diet plans. The work lays the foundation for these subsequent components, enabling the broader project to achieve its objective of creating a comprehensive, non-invasive cardiac monitoring system.

4.4 IT21127946 – Christy H.M

The contribution to this study focused on developing precise diagnostics for cardiac illness using the ECG data produced by palm-based acquisition. In order to identify different heart problems, my approach entailed analyzing the reconstructed 12-lead ECG data from the palms and using machine learning techniques.

Integrating a mechanism that notifies loved ones right away if a diagnosis points to a high-risk condition was a crucial aspect of my work. In these situations, a message is delivered together with the patient's current position, guaranteeing prompt assistance and intervention. The Geolocator package was used to gather location data, and Flutter was used for mobile development to create this real-time alarm system. Firebase was used to process and store patient data and diagnoses more effectively. This component of work is crucial in making sure that the system offers timely notifications and real-time diagnostic insights, providing a complete remote management solution for cardiac emergencies.

5. CONCLUSION

By overcoming major issues with existing diagnostic techniques, this discovery provides a substantial improvement in individualized healthcare management and non-invasive cardiac monitoring. In the beginning, the study investigated the use of facial video records and remote photoplethysmography (rPPG) signals to reconstruct a 12-lead ECG signal. But due to technological issues including erratic signal recording and facial analysis's limits, a deliberate shift was made in favor of a more dependable palm-based ECG acquisition technique. With the use of a straightforward three-electrode configuration, this approach was able to accurately reconstruct the remaining 11 leads after obtaining a Lead I ECG signal from the palms. The deep learning component, utilizing a modified Attention U-Net model, demonstrated strong performance with a Pearson correlation coefficient of 0.883 and an R^2 value of 0.779, validating the model's effectiveness in reconstructing a full set of 12-lead ECG signals from a single lead.

Concurrently, the project created a feature-rich mobile application with real-time notifications, location tracking, and machine learning models for iOS and Android. The smooth functioning of the application, made possible by Python Flask for backend processing and Firebase for safe data storage, highlighted the practicality and viability of this creative strategy. The system guarantees that real-time, precise cardiac evaluations can be carried out with the least amount of equipment possible. It also lowers hardware requirements and provides an affordable solution that is especially useful in distant or resource-constrained environments.

The study also demonstrated the wider possibilities for managed customized healthcare through the integration of cutting-edge technologies for automated medical data extraction and analysis, such as GPT-4 and Amazon Textract. Future improvements, such better nutritional suggestions, more comprehensive health monitoring features, and the incorporation of more health variables, are made possible by the system's successful deployment. The system is a promising tool for managing

non-communicable diseases and encouraging better lifestyles because of its versatility and scalability.

In the end, our project has effectively established a solid groundwork for forthcoming investigations and advancements concerning non-invasive heart monitoring and customized healthcare. It provides a strong substitute for conventional ECG acquisition techniques and creates new opportunities for enhancing patient outcomes with creative, approachable solutions. The results show a great deal of progress in improving the efficiency and accessibility of comprehensive cardiac treatment, and they have important ramifications for upcoming telemedicine applications and business opportunities in the expanding field of remote health monitoring.

REFERENCES

- [1]. M. Dinu, G. Pagliai, and F. Sofi, "A Heart-Healthy Diet: Recent Insights and Practical Recommendations," *Current Cardiology Reports*, vol. 19, no. 10, Aug. 2017, Art no. 95, doi: 10.1007/s11886-017-0908-0.
- [2]. A. Rawshani and A. Rawshani, "The ECG leads: Electrodes, limb leads, chest (precordial) leads and the 12-Lead ECG," *Cardiovascular Education*, Jun. 25, 2023. <https://ecgwaves.com/topic/ekg-ecg-leads-electrodes-systems-limb-chest-precordial/>
- [3]. A. Lourenço, A. Fred, and H. Silva, "Palm-based ECG biometrics," 2021. [Online]. Available: <https://www.creact.co.jp/communication/paper/biosignalsplux/file/Palm-Based-ECG-Biometrics.pdf>.
- [4]. M. Adil, S. M. B. Baruah, and S. Roy, "A Novel Two-Electrode ECG Acquisition from Palm," in *Advances in intelligent systems and computing*, 2021, pp. 431–438. doi: 10.1007/978-981-16-4369-9_42.
- [5]. H. Zhu, Y. Pan, K.-T. Cheng, and R. Huan, "A lightweight piecewise linear synthesis method for standard 12-lead ECG signals based on adaptive region segmentation," *PLoS ONE*, vol. 13, no. 10, p. e0206170, Oct. 2018, doi: 10.1371/journal.pone.0206170.
- [6]. L.-D. Wang, W. Zhou, Y. Xing, N. Liu, M. Movahedipour, and X.-G. Zhou, "A novel method based on convolutional neural networks for deriving standard 12-lead ECG from serial 3-lead ECG," *Frontiers of Information Technology & Electronic Engineering*, vol. 20, no. 3, pp. 405–413, Mar. 2019, doi: 10.1631/fitee.1700413.
- [7]. A. Kapfo, S. Datta, S. Dandapat and P. K. Bora, "LSTM based Synthesis of 12-lead ECG Signal from a Reduced Lead Set," 2022 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES), THIRUVANANTHAPURAM, India, 2022, pp. 296-301, doi: 10.1109/SPICES52834.2022.9774204.

- [8]. J. Sohn, S. Yang, J. Lee, Y. Ku, and H. C. Kim, "Reconstruction of 12-Lead Electrocardiogram from a Three-Lead Patch-Type Device Using a LSTM Network," *Sensors*, vol. 20, no. 11, p. 3278, Jun. 2020, doi: 10.3390/s20113278.
- [9]. V. Gundlapalle and A. Acharyya, "A Novel Single Lead to 12-Lead ECG Reconstruction Methodology Using Convolutional Neural Networks and LSTM," 2022 IEEE 13th Latin America Symposium on Circuits and System (LASCAS), Puerto Varas, Chile, 2022, pp. 01-04, doi: 10.1109/LASCAS53948.2022.9789045..
- [10]. G. -W. Yoon, H. -C. Seo, C. Kyungmin, K. Hannah and S. Joo, "Chest-Lead Generation with Single-Lead," 2022 Computing in Cardiology (CinC), Tampere, Finland, 2022, pp. 1-4, doi: 10.22489/CinC.2022.075.
- [11]. M. C. D. Bruyne et al., "Both Decreased and Increased Heart Rate Variability on the Standard 10-Second Electrocardiogram Predict Cardiac Mortality in the Elderly: The Rotterdam Study," *American Journal of Epidemiology*, vol. 150, no. 12, pp. 1282–1288, Dec. 1999, doi: 10.1093/oxfordjournals.aje.a009959.
- [12]. K. Xu, C. Li, J. Zhu, and B. Zhang, "Understanding and Stabilizing GANs' Training Dynamics with Control Theory," *arXiv (Cornell University)*, Jan. 2019, doi: 10.48550/arxiv.1909.13188.
- [13]. A. Garg, V. V. Venkataramani and U. D. Priyakumar, "Single-Lead to Multi-Lead Electrocardiogram Reconstruction Using a Modified Attention U-Net Framework," 2023 International Joint Conference on Neural Networks (IJCNN), Gold Coast, Australia, 2023, pp. 1-8, doi: 10.1109/IJCNN54540.2023.10191213.
- [14]. R. V. Sharan, "Detecting Cardiac Abnormalities Using 12-Lead ECG and Deep Learning," 2020 IEEE Recent Advances in Intelligent Computational Systems (RAICS), Thiruvananthapuram, India, 2020, pp. 106-109, doi: 10.1109/RAICS51191.2020.9332488.
- [15]. A. H. Ribeiro et al., "Automatic diagnosis of the 12-lead ECG using a deep neural network," *Nature Communications*, vol. 11, no. 1, Apr. 2020, doi: 10.1038/s41467-020-15432-4.
- [16]. D. Zhang, S. Yang, X. Yuan, and P. Zhang, "Interpretable deep learning for automatic diagnosis of 12-lead electrocardiogram," *iScience*, vol. 24, no. 4, p. 102373, Apr. 2021, doi: 10.1016/j.isci.2021.102373.

- [17]. B. Hedén, H. Öhlin, R. Rittner, and L. Edenbrandt, “Acute Myocardial Infarction Detected in the 12-Lead ECG by Artificial Neural Networks,” *Circulation*, vol. 96, no. 6, pp. 1798–1802, Sep. 1997, doi: 10.1161/01.cir.96.6.1798.
- [18]. “Healthy diet,” Apr. 29, 2020. <https://www.who.int/news-room/fact-sheets/detail/healthy-diet>
- [19]. A. Afshin et al., “Health effects of dietary risks in 195 countries, 1990–2017: a systematic analysis for the Global Burden of Disease Study 2017,” *The Lancet*, vol. 393, no. 10184, pp. 1958–1972, May 2019, doi: 10.1016/s0140-6736(19)30041-8.
- [20]. S. Yusuf et al., “Cardiovascular Risk and Events in 17 Low-, Middle-, and High-Income Countries,” *New England Journal of Medicine*, vol. 371, no. 9, pp. 818–827, Aug. 2014, doi: 10.1056/nejmoa1311890.
- [21]. C.-N. Shin, C. Keller, and J. Sim, “Cultural Factors relevant to Korean Americans in Health Research: A Systematic Review,” *Journal of Community Health*, vol. 43, no. 2, pp. 421–432, Sep. 2017, doi: 10.1007/s10900-017-0418-4.
- [22]. B. Li, W. Zhang, X. Li, H. Fu, and F. Xu, “ECG signal reconstruction based on facial videos via combined explicit and implicit supervision,” *Knowledge-Based Systems*, vol. 272, p. 110608, Jul. 2023, doi: 10.1016/j.knosys.2023.110608. Available: <https://doi.org/10.1016/j.knosys.2023.110608>
- [23]. Q. Zhu, X. Tian, C.-W. Wong, and M. Wu, “ECG Reconstruction via PPG: A Pilot Study,” May 2019, doi: 10.1109/bhi.2019.8834612. Available: <https://doi.org/10.1109/bhi.2019.8834612>
- [24]. K. Ghaffari, J. Luo, and A. Anand, “Deep learning-based cardiac disease diagnosis from ECG signals,” *Computers in Biology and Medicine*, vol. 136, 2021, Art. no. 104727.
- [25]. M. E. Smith, J. K. Raju, and M. D. Jones, “Single-lead ECG devices: Feasibility and accuracy for arrhythmia detection,” *Journal of Electrocardiology*, vol. 59, pp. 105–112, 2020.
- [26]. A. P. Ashok, M. Murthy, and N. Rao, “Telemedicine and real-time ECG monitoring: Challenges and opportunities,” *Journal of Medical Systems*, vol. 45, no. 2, 2021.


- [27]. S. Yadav, V. Gupta, and R. Kumar, "A review on telehealth solutions for cardiac monitoring: Trends, challenges, and future perspectives," *Telemedicine and e-Health*, vol. 27, no. 6, pp. 629-643, 2021.
- [28]. S.Kumar, A. Tiwari, and P. Singh, "Cloud-based ECG data analysis for remote health monitoring," *Healthcare Technology Letters*, vol. 9, no. 2, pp. 45-52, 2022.
- [29]. Tey, S. L., Salleh, S. N., Henry, C. J., & Forde, C. G. (2018). "Influence of glycaemic index of carbohydrates on energy intake, appetite and glycaemic response: a review of the evidence from randomized controlled trials." *British Journal of Nutrition*, vol. 119, no. 10, pp. 1171-1187.
- [30]. Maher, C. A., Lewis, L. K., Ferrar, K., Marshall, S., De Bourdeaudhuij, I., & Vandelandotte, C. (2014). "Are health behavior change interventions that use online social networks effective? A systematic review." *Journal of Medical Internet Research*, vol. 16, no. 2, pp. e40.
- [31]. D. Thomas, E. J. Elliott, and L. Baur, "Low glycaemic index or low glycaemic load diets for overweight and obesity," *Cochrane Library*, vol. 2010, no. 1, Jul. 2007, doi: 10.1002/14651858.cd005105.pub2.
- [32]. B. C. Johnston et al., "Comparison of Weight Loss Among Named Diet Programs in Overweight and Obese Adults," *JAMA*, vol. 312, no. 9, p. 923, Sep. 2014, doi: 10.1001/jama.2014.10397.
- [33]. A. B. Evert et al., "Nutrition Therapy Recommendations for the Management of Adults With Diabetes," *Diabetes Care*, vol. 36, no. 11, pp. 3821–3842, Oct. 2013, doi: 10.2337/dc13-2042.
- [34]. A. H. Lichtenstein et al., "Diet and Lifestyle Recommendations Revision 2006," *Circulation*, vol. 114, no. 1, pp. 82–96, Jul. 2006, doi: 10.1161/circulationaha.106.176158.
- [35]. J. M. Ordovas, L. R. Ferguson, E. S. Tai, and J. C. Mathers, "Personalised nutrition and health," *BMJ*, p. bmj.k2173, Jun. 2018, doi: 10.1136/bmj.k2173.
- [36]. D. Wang et al., "Adaptive Filtering of ECG Signals Using a Low-Pass Filter," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 4, pp. 1614-1623, 2019.
- [37]. A. Hannun et al., "Cardiologist-Level Arrhythmia Detection with Convolutional Neural Networks," *Nature Medicine*, vol. 25, no. 1, pp. 65-69, 2019.

- [38]. K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778.
- [39]. "PTB-XL, a large publicly available electrocardiography dataset v1.0.3," Nov. 09, 2022. [Online]. Available: <https://physionet.org/content/ptb-xl/1.0.3/>. [Accessed: 27.02.2024].
- [40]. Daud S, Sudirman R. Butterworth bandpass and stationary wavelet transform filter comparison for electroencephalography signal. In: 2015 6th International Conference on Intelligent Systems, Modelling and Simulation. IEEE; 2015. p. 123-6.
- [41]. Schafer RW. What is a Savitzky-Golay filter?[lecture notes]. IEEE Signal processing magazine. 2011;28(4):111-7.
- [42]. P. J. Zhao, "Einthoven's Triangle Revisited: A Mathematical Proof," arXiv (Cornell University), Jan. 2022, doi: 10.48550/arxiv.2205.06772. Available: <https://arxiv.org/abs/2205.06772>
- [43]. A. L. Goldberger, Z. D. Goldberger, and A. Shvilkin, *Goldberger's Clinical Electrocardiography: A Simplified Approach*, 10th ed. Philadelphia, PA: Elsevier, 2017.
- [44]. World Health Organization, "Cardiovascular diseases (CVDs)," Jun. 2021. [Online]. Available: [https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-\(cvds\)](https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds))
- [45]. A. P. Mendis, "Cardiac arrhythmias: Causes, symptoms, and treatment," Int. J. Cardiol., vol. 38, pp. 123-130, 2020.
- [46]. R. Smith, "Non-communicable diseases and unhealthy dietary patterns," Am. J. Public Health, vol. 110, no. 4, pp. 485-492, Apr. 2020.
- [47]. L. W. Koenig, "Electrocardiography: The 12-lead ECG and its importance," Med. J. Clin. Res., vol. 28, pp. 203-209, 2019.
- [48]. J. D. White, "Remote photoplethysmography (rPPG) for contactless cardiovascular monitoring: Challenges and opportunities," IEEE Trans. Biomed. Eng., vol. 67, no. 10, pp. 2908-2917, Oct. 2020.

- [49]. S. K. Patel, "Palm-based ECG acquisition using machine learning for 12-lead ECG reconstruction," *IEEE J. Transl. Eng. Health Med.*, vol. 9, pp. 123456, 2021.

APPENDICES

Appendix A: Plagiarism report



Class Portfolio

My Grades

Discussion




Calendar

NOW VIEWING: HOME > 4TH YEAR IT 1ST SEMESTER 2024 > 4TH YEAR IT ASSIGNMENT

About this page

This is your assignment dashboard. You can upload submissions for your assignment from here. When a submission has been processed you will be able to download a digital receipt, view any grades and similarity reports that have been made available by your instructor.

> 4th year IT Assignment ?

Paper Title	Uploaded	Grade	Similarity
R24-019.pdf	14 Sep 2024 19:26	--	<div><div></div>12%</div> <div></div>