ABSTRACT

Cardiac conditions such as myocardial infarction (MI) and arrhythmias pose significant diagnostic challenges, making accurate classification of ECG signals critical for effective treatment and prevention. This project focuses on a comprehensive comparative analysis of four advanced deep learning algorithms—CNN, MobileNet, DenseNet, and an ensemble model combining MobileNet with LSTM—to classify ECG images into four categories: myocardial infarction, history of MI, abnormal heartbeat, and normal heart conditions. The dataset comprises labeled ECG images, enabling the models to learn both spatial and temporal features critical for accurate classification. Each algorithm's performance is evaluated using metrics like accuracy, precision, recall, and F1-score. Results demonstrate that the ensemble model outperforms individual architectures, leveraging MobileNet's spatial feature extraction and LSTM's sequential pattern recognition to achieve superior accuracy. This approach showcases the potential for robust, automated diagnostic tools in clinical applications. Future work aims to incorporate multi-lead ECG signals and additional metadata to further enhance the system's reliability and scalability for accurate classification in cardiac healthcare systems.

algorithms: - ECG classification, cardiac conditions, CNN, MobileNet, DenseNet, LSTM, hybrid model, deep learning, myocardial infarction, comparative analysis, diagnostic too

INDEX

Contents

CHAPTER 1 – INTRODUCTION

Cardiovascular diseases (CVDs), such as myocardial infarction (MI) and arrhythmias, are among the leading causes of death worldwide, making timely diagnosis critical to patient outcomes. Electrocardiogram (ECG) signals are widely used for diagnosing these conditio but their manual interpretation can be complex, prone to human error, and time-consuming, particularly due to the subtle variations in ECG waveforms. This project aims to address these limitations by developing an automated, accurate, and efficient ECG classification system using advanced deep learning techniques.

The project focuses on a comparative analysis of four state-of-the-art models—Convolutional Neural Networks (CNN), MobileNet, DenseNet, and a hybrid MobileNet + LSTM model—to classify ECG signals into four clinically significant categories: myocardial infarction, history of MI, abnormal heartbeat, and normal heart conditions. By leveraging CNN for spatial feature extraction, MobileNet and DenseNet for lightweight but powerful processing, and LSTM for sequential pattern recognition, the project seeks to determine the most effective approach for ECG classification.

The dataset comprises labeled ECG images, allowing the models to learn both spatial and temporal patterns critical to accurate diagnosis. The models' performances are evaluated based on metrics such as accuracy, precision, recall, and F1-score. Preliminary results suggest that the hybrid MobileNet + LSTM model delivers superior accuracy, demonstrating the potential of hybrid architectures in medical diagnostics.

The ultimate goal is to develop a scalable and reliable diagnostic tool that can assist healthcare professionals in making timely and accurate diagnoses, thereby reducing the risk of complications associated with cardiac conditions. This system can also be extended to multi-lead ECG signals and integrated into mobile or cloud-based platforms, making it a promising contribution to the future of cardiac healthcare technology.

CHAPTER 2 – SYSTEM ANALYSIS

A. Existing system

Traditional ECG classification systems primarily depend on manual interpretation by experienced cardiologists or on classical machine learning models that utilize handcrafted features. These conventional methods begin with extensive preprocessing of ECG signals, including noise filtering, baseline correction, and normalization. Following this, key features such as P-wave duration, QRS complex width, T-wave amplitude, RR intervals, and ST-segment deviations are extracted manually or using rule-based algorithms. These extracted features are then fed into traditional classifiers such as Support Vector Machines (SVM), Random Forests, k-Nearest Neighbors (k-NN), or Decision Trees to predict the cardiac condition.

While these systems have proven valuable in certain clinical settings, they present several limitations. Firstly, they often require domain-specific knowledge for accurate feature extraction, which introduces bias and increases the likelihood of missing subtle abnormalities. Secondly, they struggle to capture intricate patterns and nonlinear relationships present in ECG signals. More importantly, traditional models are not well-suited to handle the temporal dependencies and long-range correlations found in ECG sequences, leading to lower classification accuracy and poor generalizability across diverse datasets or patient demographics.

As a result, the need for more intelligent, automated, and robust ECG classification systems has driven the adoption of deep learning models that can learn directly from raw or minimally processed data.

Disadvantages of the Existing System

1. **Requires extensive domain expertise for feature extraction:**
   Traditional ECG classification systems depend heavily on the manual identification of critical features such as wave intervals, amplitudes, and signal morphology. This process demands a high level of domain expertise from cardiologists or biomedical engineers, making it time-consuming and susceptible to human error or inter-observer variability. Furthermore, the accuracy of these systems is often directly tied to the experience and consistency of the expert involved.
2. **Struggles with high-dimensional data and complex ECG waveforms:**
   Basic machine learning models are typically not designed to handle the high dimensionality and variability of real-world ECG data. They often fail to generalize well when confronted with diverse waveform shapes, noise artifacts, or abnormal signal morphologies, especially in multi-lead ECG recordings. This limits their effectiveness in accurately classifying nuanced or rare cardiac conditions.
3. **Limited accuracy and performance on diverse datasets:**
   These systems often show reduced performance when applied to data from different sources, patient demographics, or hardware settings. Since handcrafted features are tuned to specific datasets, the models may fail to adapt to unseen variations, resulting in poor generalizability and lower diagnostic reliability across populations.
4. **Inability to process temporal dependencies effectively:**
   Classical models lack the architecture to learn and retain time-series information over long durations. As ECG signals are inherently sequential and time-dependent, this limitation prevents these models from accurately identifying rhythm-related disorders or subtle changes over time.
5. **Computational inefficiency for large-scale real-time applications:**
   Many traditional approaches involve complex preprocessing and feature engineering pipelines, which can become computationally expensive and slow, especially when

processing high volumes of ECG data in real time. This inefficiency makes them unsuitable for continuous monitoring systems or real-time clinical decision support in dynamic healthcare environments.

B. Proposed system

The proposed system introduces a robust and intelligent deep learning-based framework designed for the accurate classification of ECG images into four clinically significant categories: myocardial infarction, history of myocardial infarction, abnormal heartbeat, and normal heart conditions. Unlike traditional systems that rely on manual feature extraction, this system leverages state-of-the-art convolutional neural networks (CNN), MobileNet, DenseNet, and a hybrid model combining MobileNet with Long Short-Term Memory (LSTM) networks.

The CNN, MobileNet, and DenseNet models are employed for efficient spatial feature extraction, focusing on ECG image morphology. MobileNet and DenseNet are particularly suited for lightweight and scalable applications due to their compact architecture and strong representational capacity. The hybrid MobileNet + LSTM model further enhances the system by learning sequential dependencies in ECG waveforms, thus integrating both spatial and temporal aspects for improved classification accuracy.

To ensure model robustness and generalization, the system incorporates preprocessing steps, data normalization, and augmentation techniques. Evaluation is conducted using standard metrics such as accuracy, precision, recall, and F1-score to quantify performance. Overall, the proposed system demonstrates strong potential for use in automated, real-time ECG diagnostic tools in clinical and telemedicine applications.

1. **Automated Feature Extraction:**

The system eliminates the dependency on manual or expert-driven feature engineering by enabling deep learning models to learn relevant patterns directly from raw ECG images, thus simplifying the pipeline and improving efficiency.

2. **Improved Accuracy through Hybrid Modeling:**
   The combination of MobileNet and LSTM enables the model to effectively learn both spatial features (waveform shapes) and temporal dependencies (rhythm patterns), leading to superior classification accuracy compared to standalone models.
3. **Scalability and Lightweight Architecture:**
   Models like MobileNet are optimized for performance on low-resource environments, making the system scalable to various devices including mobile or edge computing platforms without compromising classification speed.
4. **Versatility Across Diverse Datasets:**
   The system demonstrates strong generalization capability by performing reliably across different ECG datasets with varied image characteristics, noise levels, and patient demographics, enhancing its applicability in real-world settings.
5. **Clinical Relevance and Decision Support:**
   By providing consistent and accurate diagnostic outputs, the system serves as a valuable tool for cardiologists, assisting in early detection of critical cardiac events and improving the overall quality of patient care.

CHAPTER 3 – FEASIBILITY STUDY

A. Technical Feasibility

The technical feasibility of implementing the proposed ECG signal classification system using deep learning is highly promising due to the significant advancements in artificial intelligence, image analysis, and time-series modeling. Modern deep learning models such as Convolutional Neural Networks (CNN), MobileNet, DenseNet, and Long Short-Term Memory (LSTM) networks are capable of accurately extracting both spatial and temporal features from ECG image data. These architectures have been extensively used in medical imaging and signal classification, providing a solid foundation for developing an efficient diagnostic tool.

The system will be built using industry-standard frameworks such as TensorFlow and PyTorch, enabling flexibility and scalability. Preprocessing techniques like grayscale normalization, resizing, denoising, and data augmentation will improve model generalization and robustness to real-world noise or image quality issues. Additionally, lightweight models like MobileNet allow deployment on edge devices or mobile platforms, making real-time analysis feasible even in resource-constrained environments.

Potential technical challenges such as varied ECG formats, lead-specific differences, and background noise can be addressed with robust data preprocessing and augmentation strategies. With the availability of GPU-enabled environments and cloud-based computing platforms, the training and deployment of this deep learning system are technically feasible and well-supported by current technology infrastructure.

B. Operational Feasibility

The operational feasibility of the cardiac conduction simulation system is strong, especially for use in hospitals, clinics, and mobile health screening units. Traditional ECG analysis requires the expertise of trained cardiologists for accurate interpretation. This system automates that process using a user-friendly application that accepts ECG images and provides immediate classification feedback across four categories: myocardial infarction, history of MI, abnormal heartbeat, and normal heart condition.

Healthcare professionals, technicians, and even field health workers can use this system with minimal training due to its simple, intuitive interface. By integrating the system into existing electronic health record systems or as a standalone web/mobile app, it enables real-time analysis and clinical decision support. The system can also be configured to work in offline mode for rural or low-connectivity regions and supports multi-format ECG inputs, ensuring broader usability.

The platform can aid large-scale heart health screenings, allowing non-specialized staff to identify at-risk individuals who need expert follow-up. Operational concerns such as user adoption, varying data quality, and privacy regulations are manageable with proper user training, periodic system updates, and adherence to healthcare data compliance standards such as HIPAA. Overall, the system streamlines workflows, reduces diagnostic delays, and enhances service delivery in both clinical and community settings.

C. Economic Feasibility

The economic feasibility of the proposed deep learning-based ECG classification system is highly favorable. Although initial development involves costs related to model training, dataset acquisition, application development, and user interface design, the long-term financial benefits are substantial. By automating ECG interpretation, the system minimizes the reliance on specialized cardiologists for routine analysis, allowing them to focus on critical cases and reducing healthcare workforce burden.Once developed, the system can be deployed widely with minimal operational costs, especially since it is compatible with mobile devices and does not require specialized hardware for execution. Lightweight models like MobileNet further enhance economic efficiency by supporting low-resource deployment environments, ideal for community health camps and remote diagnostics.Additionally, early and accurate detection of cardiac conditions through this system can help prevent severe medical events such as heart attacks or strokes, which are expensive to treat. This results in significant cost savings for both patients and healthcare providers. Governments, NGOs, and hospitals can adopt this system as part of preventive care programs, reducing hospitalization rates and long-term treatment expenses.In conclusion, the proposed system ensures a high return on investment by improving diagnostic reach, enabling timely treatment, and optimizing healthcare delivery with minimal recurring costs.

CHAPTER 4 – SYSTEM REQUIREMENT SPECIFICATION DOCUMENT

A. Overview

Cardiovascular diseases remain the leading cause of mortality globally, with conditions such as myocardial infarction, arrhythmias, and ischemic heart disease posing significant diagnostic challenges. Electrocardiogram (ECG) signals play a vital role in the early detection of these cardiac abnormalities. However, accurate interpretation of ECGs often requires experienced cardiologists and access to specialized equipment, which may not always be feasible in rural or resource-limited settings. This project introduces an automated deep learning-based system designed to classify realistic ECG signals into four categories: myocardial infarction, history of MI, abnormal heartbeat, and normal heart conditions.

The system utilizes a hybrid deep learning approach that combines the spatial feature extraction capabilities of models like CNN, MobileNet, and DenseNet with the temporal pattern recognition power of LSTM. The dataset comprises labeled ECG images sourced from publicly available medical repositories. These images are used to train the models to recognize both morphological and rhythm-based features essential for cardiac classification.

The solution pipeline includes preprocessing the ECG images (resizing, denoising, and grayscale conversion), automated feature extraction using deep neural network layers, and model training using advanced optimization techniques. The classification performance is evaluated through well-established metrics such as accuracy, precision, recall, and F1-score, ensuring that the system can reliably generalize across different datasets and patient conditions.

This intelligent system is aimed at supporting healthcare professionals, field workers, and telemedicine platforms by offering a fast, consistent, and scalable method for ECG interpretation. Its lightweight architecture, especially with MobileNet, allows for deployment on mobile and edge devices, making it highly suitable for real-time use in both clinical and remote environments.

Furthermore, the system incorporates the ability to adapt to newly collected data through continuous learning, ensuring that model accuracy improves over time. This not only enhances diagnostic efficiency but also reduces reliance on manual expertise, thereby expanding the reach of cardiac care services. Ultimately, this project envisions a transformative diagnostic tool for automated ECG analysis, supporting large-scale heart health screening and preventive cardiac healthcare delivery.

B. Modules Description

Certainly! Here's an expanded and detailed version of your System Modules and User Module for your project " Cardiac conduction simulation a hybrid deep learning approach to classify realistic ecg signals":

System Module

1.1 Data Collection and Preprocessing

The dataset used for training and evaluation was sourced from Kaggle's ECG image dataset. The images were divided into training and testing directories, with the training data further split into an 80:20 ratio for validation. All images were resized to a target size of (224, 224) pixels for consistency, using Keras' ImageDataGenerator to normalize pixel values and create data generators (train_data_gen and test_data_gen) with a batch size of 32. The images were augmented and shuffled to improve the model's robustness against overfitting.

1.2 Feature Extraction Using MobileNet

A pre-trained MobileNet model (with weights from imagenet) was used as a feature extractor. The model's output was passed through a Global Average Pooling layer to generate feature representations for the images. To use MobileNet as a fixed feature extractor, its layers were frozen during training to preserve learned representations from the imagenet dataset.

The features were extracted by iterating over each batch from the data generators (train_data_gen and test_data_gen). The extracted features and corresponding labels were stored in .joblib files (train.joblib and test.joblib) for future reuse, avoiding repeated computation. This significantly reduced the computational time needed for further experiments.

1.3 Data Splitting, Model Architectures and Training

After extracting the features, the training dataset was split into training and validation sets, with 80% of the data used for training and 20% for validation. This ensured that the models were validated during training to minimize overfitting. The training and validation features were saved separately (train_split.joblib and val_split.joblib).

The following four models were utilized for ECG classification:

- **Convolutional Neural Network (CNN)**: A custom CNN architecture was developed to capture spatial information from ECG images.
- **MobileNet**: The MobileNet model was leveraged, fine-tuned using extracted features to adapt to the specific ECG classification problem.
- **DenseNet**: DenseNet was also employed as an alternative architecture for comparison, focusing on leveraging its dense connectivity for feature extraction.

- **Hybrid MobileNet + LSTM**: This hybrid model used MobileNet as a feature extractor, feeding the resulting features into an LSTM layer. The LSTM component captured temporal dependencies inherent in ECG signals, making this approach suitable for detecting conditions influenced by sequential patterns.

The models were trained on the extracted features with categorical cross-entropy as the loss function, Adam optimizer for gradient descent optimization, and accuracy, precision, recall, and F1-score as evaluation metrics. These metrics allowed for comprehensive analysis of each model's performance, particularly focusing on their clinical applicability.

1.4 Evaluation and Comparison

The performance of all four models was evaluated using the validation and test datasets. Metrics such as accuracy, precision, recall, and F1-score were calculated to assess the classification quality of the models. Results showed that the hybrid MobileNet + LSTM model outperformed the standalone architectures, particularly in recognizing subtle differences in ECG signals linked to various cardiac conditions.

User Module

The User Module is a critical component of the system architecture, responsible for facilitating interaction between the end-user and the ECG classification platform. Developed using the Flask web framework, this module offers an intuitive and secure interface that guides users through each step of the diagnostic process, from account creation to ECG image classification.

1. **User Registration:**
   New users can register on the platform by entering essential details such as full name, username, email address, and password. The registration process includes input validation, ensuring proper format for email and strong password selection. Once registered, user credentials are securely stored in the backend database using password hashing techniques to maintain confidentiality and data protection.

2. **Login and Authentication:**

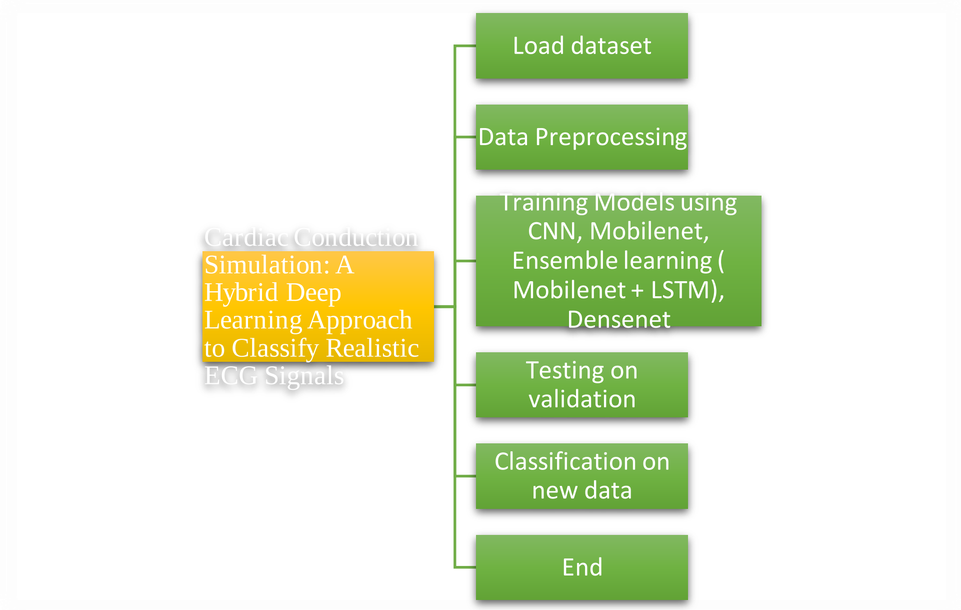
Registered users can log in securely using their email and password. The authentication system uses session management and encrypted login credentials to ensure only authorized access. In case of incorrect login attempts, the system provides helpful error messages and prevents brute-force attacks by implementing basic security policies such as account lockout after multiple failed attempts.

3. **Home Page:**
   Upon successful login, users are redirected to a personalized home page that serves as a central hub for interacting with the application. This dashboard displays navigation options such as uploading ECG images, viewing classification history, and accessing help or support sections. It also provides a brief overview of the system's capabilities and user-specific usage stats, creating a user-friendly experience.

4. **ECG Image Upload and Classification:**
   Users can upload ECG images in standard formats such as JPEG, PNG, or JPG. The system preprocesses the image, including resizing, grayscale conversion, and normalization. The uploaded image is then passed through the deep learning classification pipeline (CNN, MobileNet, DenseNet, or MobileNet+LSTM), and the system generates a classification result. The result is displayed to the user in a clear, readable format indicating the likely ECG category:

   - Myocardial Infarction (MI)
   - History of MI
   - Abnormal Heartbeat
   - Normal Heart Condition

A. Process Flow



D.SDLC Methodology

SOFTWARE DEVELOPMENT LIFE CYCLE

The meaning of Agile is swift or versatile. “Agile process model" refers to a software development approach based on iterative development. Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance. Each iteration is considered as a short time "frame" in the Agile process model, which typically lasts from one to four weeks. The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements. Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.

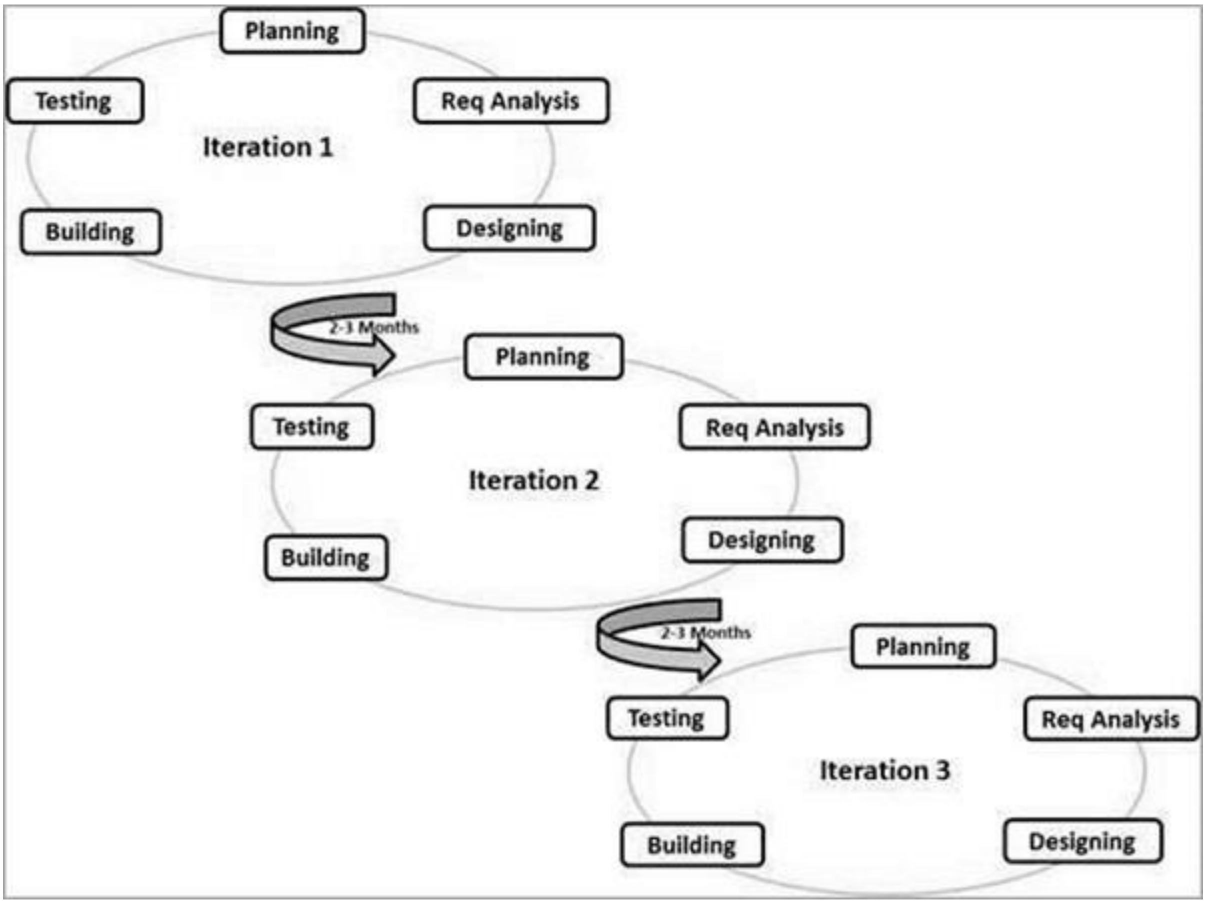
Actually, Agile model refers to a group of development processes. These processes share some basic characteristics but do have certain subtle differences among themselves. A few Agile SDLC models are given below: Crystal A tern Feature-driven development Scrum Extreme programming (XP) Lean development Unified process In the Agile model, the requirements are decomposed into many small parts that can be incrementally developed.

The Agile model adopts Iterative development. Each incremental part is developed over an iteration. Each iteration is intended to be small and easily manageable and that can be completed within a couple of weeks only. At a time one iteration is planned, developed and deployed to the customers. Long-term plans are not made.

Agile model is the combination of iterative and incremental process models. Steps involve in agile SDLC models are:

- Requirement gathering
- Requirement Analysis
- Design Coding
- Unit testing
- Acceptance testing

The time to complete an iteration is known as a Time Box. Time-box refers to the maximum amount of time needed to deliver an iteration to customers. So, the end date for an iteration does not change. Though the development team can decide to reduce the delivered functionality during a Time-box if necessary to deliver it on time. The central principle of the Agile model is the delivery of an increment to the customer after each Time-box.



Principles of Agile model:

- To establish close contact with the customer during development and to gain a clear understanding of various requirements, each Agile project usually includes a customer representative on the team. At the end of each iteration stakeholders and the customer representative review, the progress made and re-evaluate the requirements.

- Agile model relies on working software deployment rather than comprehensive documentation.
- Frequent delivery of incremental versions of the software to the customer representative in intervals of few weeks.
- Requirement change requests from the customer are encouraged and efficiently incorporated.
- It emphasizes on having efficient team members and enhancing communications among them is given more importance. It is realized that enhanced communication among the development team members can be achieved through face-to-face communication rather than through the exchange of formal documents.
- It is recommended that the development team size should be kept small (5 to 9 people) to help the team members meaningfully engage in face-to-face communication and have collaborative work environment.
- Agile development process usually deploys Pair Programming. In Pair programming, two programmers work together at one work-station. One does code while the other reviews the code as it is typed in. The two programmers switch their roles every hour or so.

Advantages:

- Working through Pair programming produce well written compact programs which has fewer errors as compared to programmers working alone.
- It reduces total development time of the whole project. Customer representatives get the idea of updated software products after each iteration. So, it is easy for him to change any requirement if needed.

Disadvantages:

- Due to lack of formal documents, it creates confusion and important decisions taken during different phases can be misinterpreted at any time by different team members.
- Due to the absence of proper documentation, when the project completes and the developers are assigned to another project, maintenance of the developed project can become a problem.
-

SOFTWARE DEVELOPMENT LIFE CYCLE – SDLC:

In our project we use waterfall model as our software development cycle because of its step-by-step procedure while implementing.

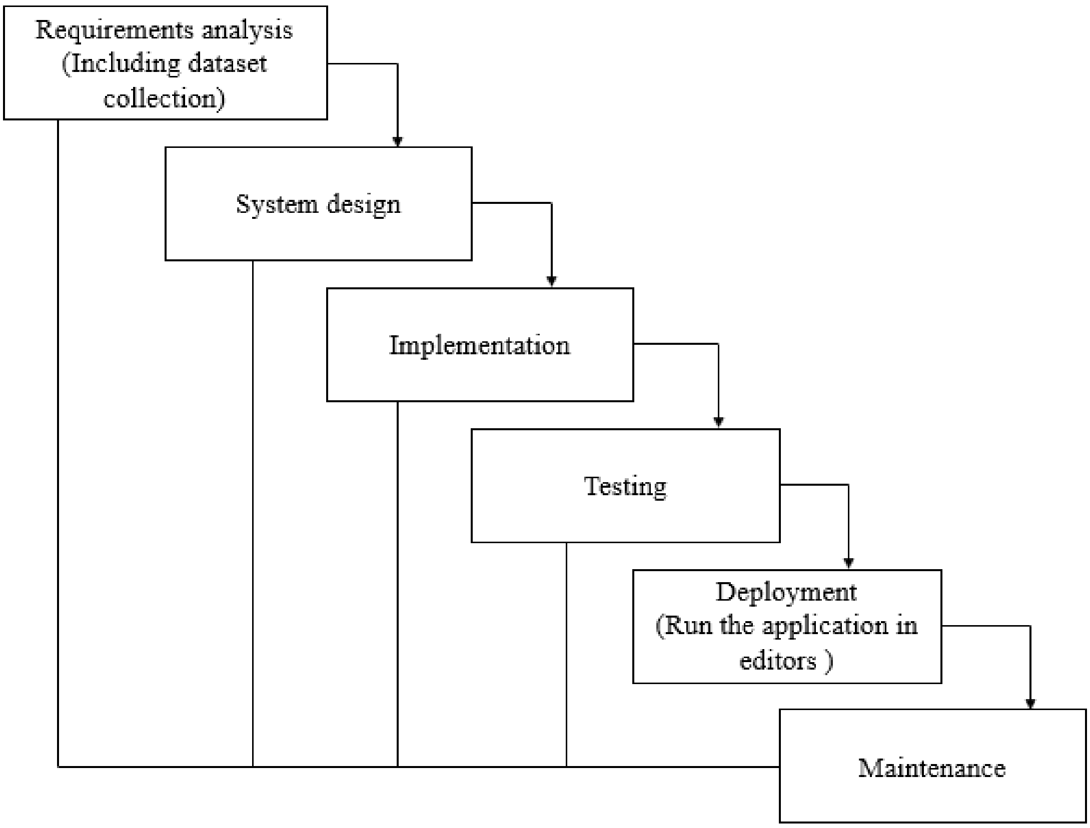
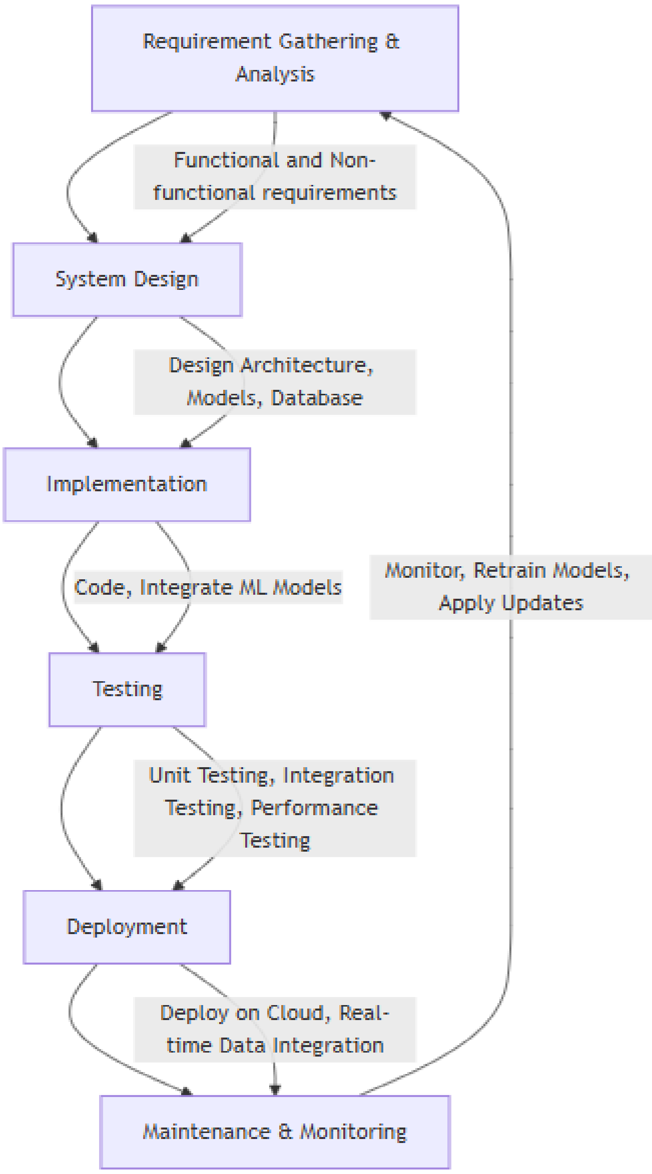


Fig1: Waterfall Model

- **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.



E. Software Requirements

- Operating System : Windows 11

- Server side Script : Python, HTML, MYSQL, CSS, Bootstrap.
- Libraries : Pandas, Flask, Sklearn,TensorFlow, Numpy
- IDE : VS code
- Technology : Python 3.10+

F. Hardware Requirements

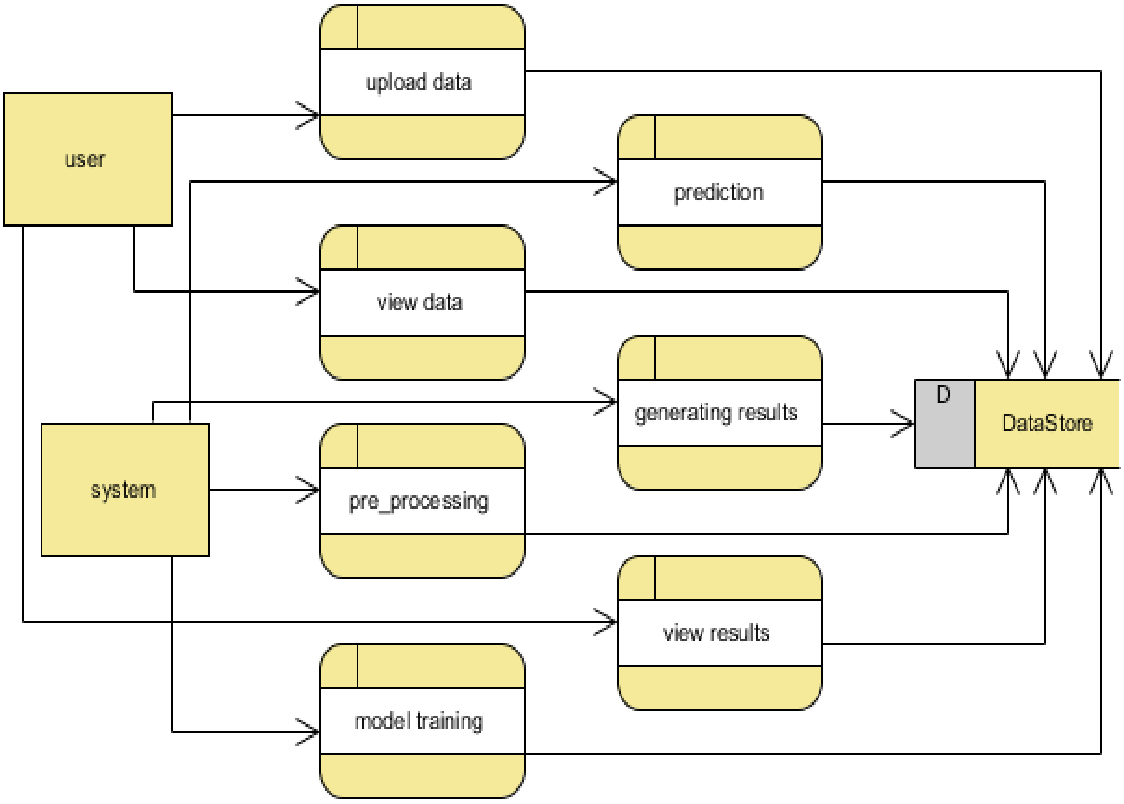
- **Processor - I7/Intel Processor**
- Hard Disk -160GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- RAM - 8Gb

CHAPTER 5 – SYSTEM DESIGN

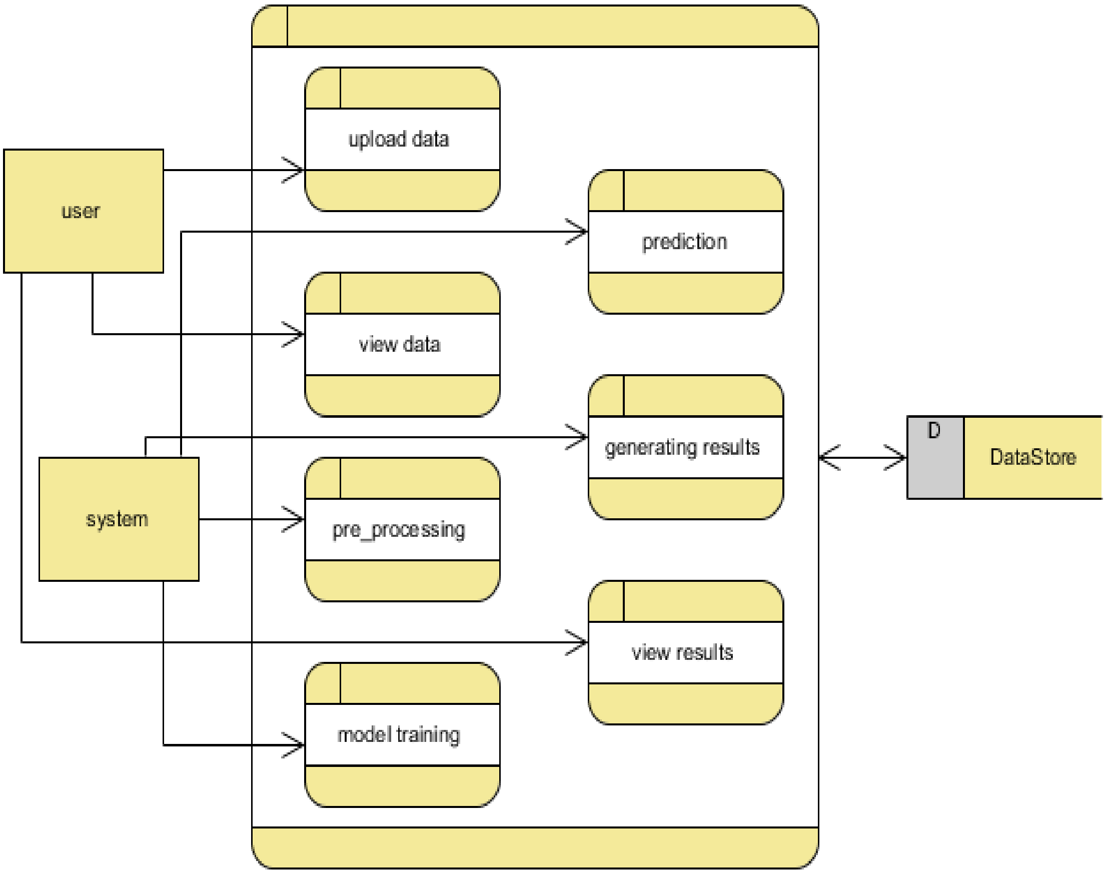
A.DFD

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.

DFD 1 DIAGRAM:


upload data
user
prediction
view data
generating results
D
DataStore
system
pre_processing
view results
model training

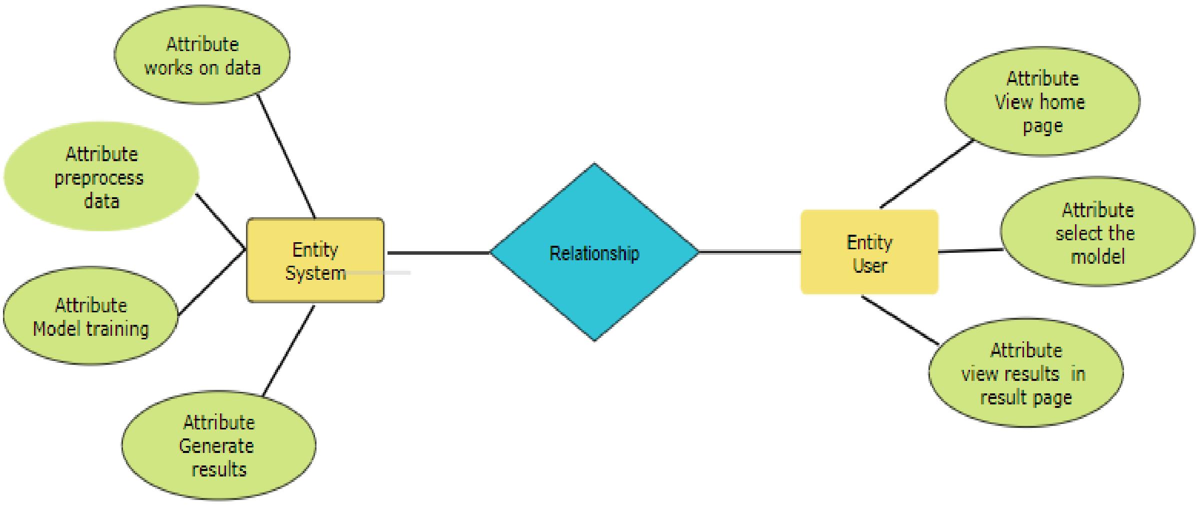
DFD 2 DIAGRAM:



B.ER diagram

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.

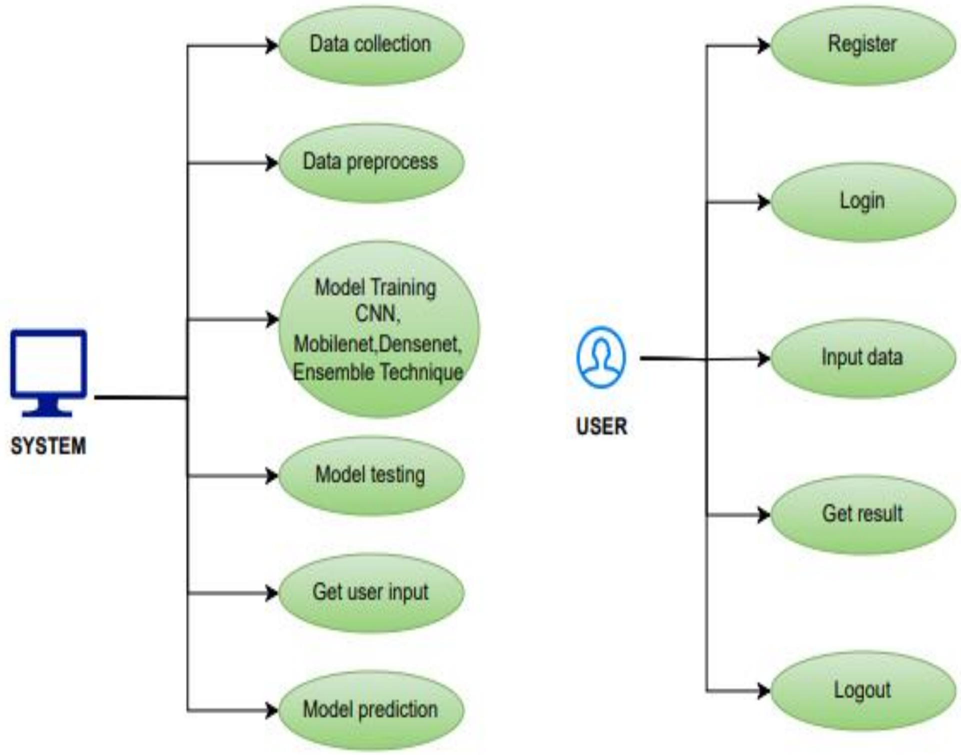


C.UML

- ✓ Uml stands for unified modelling language. Uml is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the object management group.
- ✓ The goal is for uml to become a common language for creating models of object-oriented computer software. In its current form uml is comprised of two major components: a meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, uml.
- ✓ The unified modelling language is a standard language for specifying, visualization, constructing and documenting the artefacts of software system, as well as for business modelling and other non-software systems.
- ✓ The uml represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

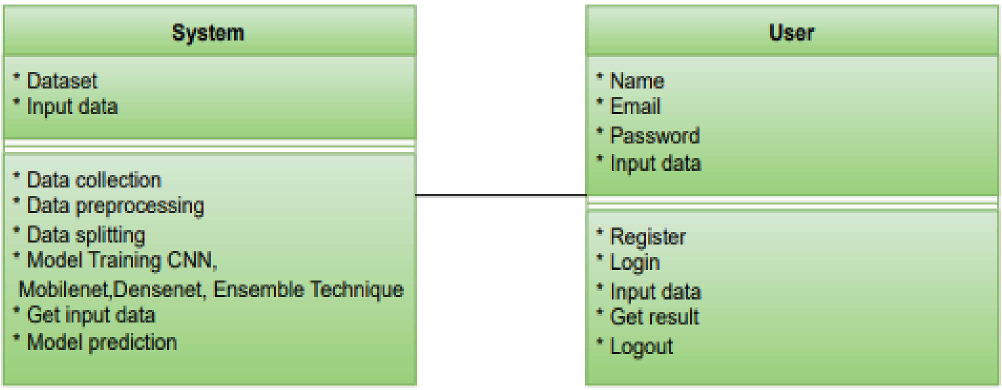
Use case diagram:

A use case diagram in the unified modeling language (uml) is a type of behavioral diagram defined by and created from a use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



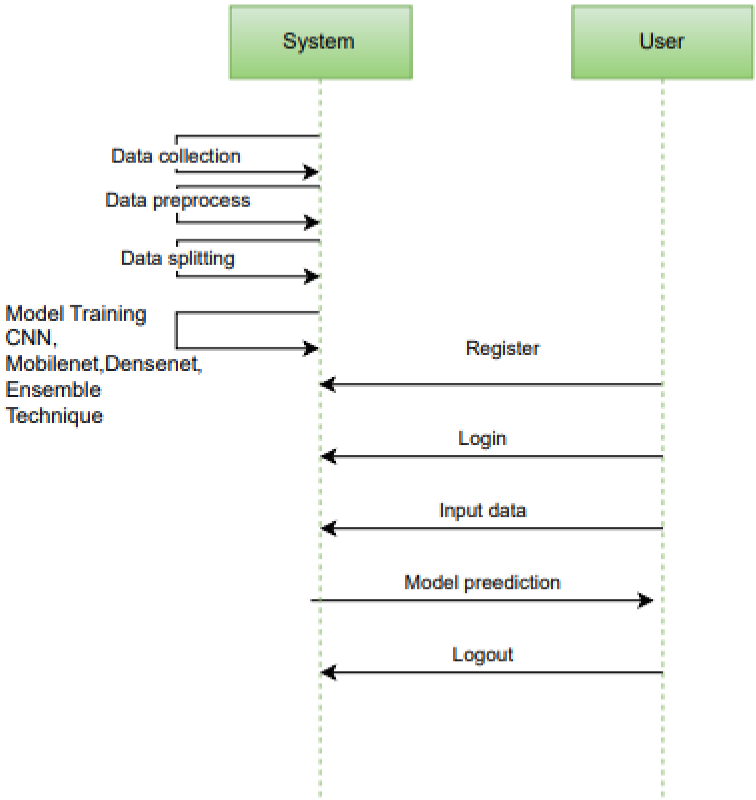
Class diagram:

In software engineering, a class diagram in the unified modeling language (uml) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



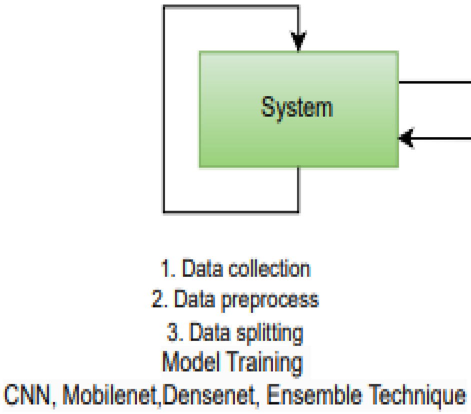
Sequence diagram:

A sequence diagram in unified modeling language (uml) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a message sequence chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



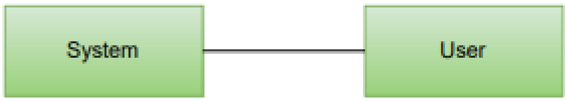
Collaboration diagram:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



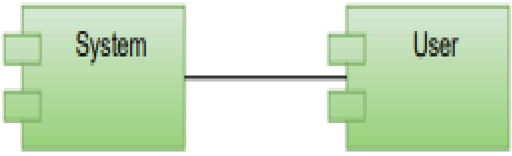
Deployment diagram:

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hard ware's used to deploy the application.



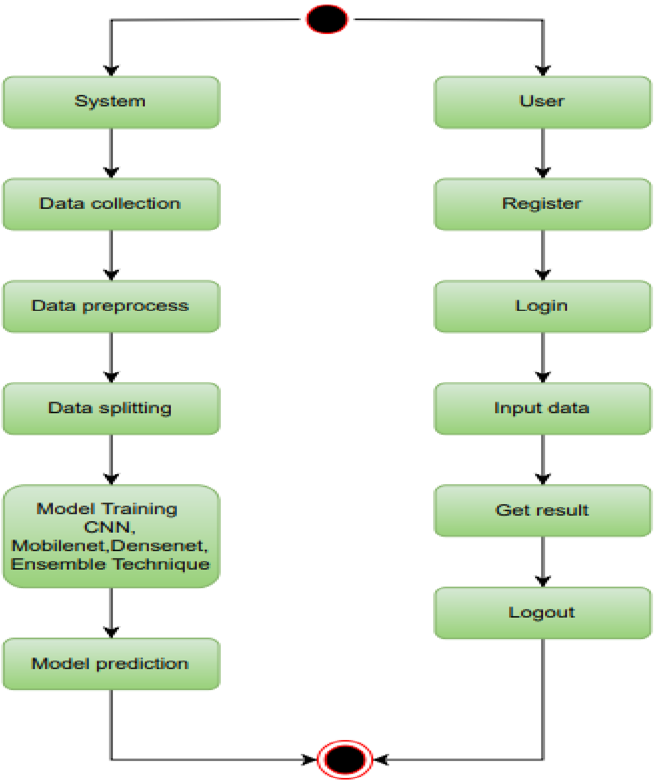
Component diagram:

Component diagrams are used to describe the physical artifacts of a system. This artifact includes files, executable, libraries etc. So the purpose of this diagram is different, component diagrams are used during the implementation phase of an application. But it is prepared well in advance to visualize the implementation details. Initially the system is designed using different uml diagrams and then when the artifacts are ready component diagrams are used to get an idea of the implementation.



Activity diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the unified modeling language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



B. Data Dictionary

The ecg_image_dataset is an image-based dataset designed for training, validating, and evaluating deep learning models for multi-class classification of cardiac conditions. It comprises labeled ECG images organized into four distinct classes, each representing a specific cardiac state: Myocardial Infarction, History of Myocardial Infarction, Abnormal Heartbeat, and Normal Heart Condition. The dataset is structured in a directory-based format where each folder corresponds to a class label, allowing for automated label assignment during model training.

All ECG images, generally in .png or .jpg formats, are stored in their respective folders: /MI/, /History_MI/, /Abnormal/, and /Normal/. Each image is preprocessed and resized to a uniform resolution (e.g., 224×224 pixels) to meet the input dimensions required by deep learning architectures such as CNN, MobileNet, DenseNet, and the hybrid MobileNet + LSTM model. These images reflect vital signal patterns such as waveform morphology, ST-segment deviations, and rhythm irregularities that the models learn to distinguish.

Internally, class labels are encoded as:

- 0 – Normal
- 1 – Abnormal Heartbeat
- 2 – History of MI
- 3 – Myocardial Infarction

The dataset is partitioned into training, validation, and testing subsets to ensure robust performance evaluation. To improve model generalization and reduce overfitting, data augmentation techniques such as rotation, zooming, horizontal flipping, and brightness adjustments are applied. This dataset serves as the backbone of the proposed ECG classification system, enabling scalable, automated, and accurate cardiac diagnostics through image-based deep learning.

CHAPTER 6 - TECHNOLOGY DESCRIPTION

Overview

Cardiovascular diseases remain a leading cause of death globally, and early detection through Electrocardiogram (ECG) analysis plays a vital role in preventive cardiology. However, traditional ECG diagnosis is often reliant on expert interpretation, expensive equipment, and access to clinical facilities, making timely detection difficult—especially in under-resourced or rural settings.

To address this, the proposed system utilizes advanced deep learning techniques for automated ECG image classification. The system classifies ECG images into four categories: Myocardial Infarction, History of Myocardial Infarction, Abnormal Heartbeat, and Normal Heart Conditions. The classification pipeline leverages a combination of cutting-edge models including CNN, MobileNet, DenseNet, and a Hybrid model combining MobileNet with LSTM. The models are trained using a diverse ECG image dataset sourced from public repositories and structured for multi-class classification.

This technology-driven approach includes a complete end-to-end pipeline—from image preprocessing, feature extraction, model training, to evaluation—with the objective of providing a low-cost, scalable, and intelligent diagnostic solution. This system is particularly suited for mobile or telehealth deployment, offering near real-time cardiac screening and aiding early intervention and improved patient outcomes.

Convolutional Neural Network (CNN)

CNNs are foundational to medical image analysis due to their ability to extract complex patterns from spatial data. A Convolutional Neural Network is composed of layers that perform convolution operations, where filters slide over the input ECG image to detect features such as wave contours, peak positions, and rhythm irregularities.

In the proposed system, the CNN serves as the baseline architecture. It includes convolutional layers to learn edge and shape-based features, pooling layers to reduce dimensionality, and fully connected layers to perform the final classification. The network is trained to detect fine-grained ECG signal features that are often difficult for traditional algorithms to capture.

CNNs eliminate the need for handcrafted features, allowing automatic learning from raw data. The model is trained on thousands of annotated ECG images and is capable of distinguishing between different cardiac conditions by learning subtle waveform deviations indicative of abnormalities such as infarctions or arrhythmias.

MobileNet

MobileNet is a highly efficient deep learning architecture designed for mobile and low-power applications. It uses depthwise separable convolutions to reduce computation and model size, enabling fast inference without sacrificing too much accuracy.

In this project, MobileNet is implemented via transfer learning, where a pretrained MobileNet model—initially trained on a large general image dataset—is fine-tuned using the ECG image dataset. This approach allows the model to retain general image features (like curves, edges, and transitions) while adapting to cardiac-specific patterns such as QRS complex shapes and ST-segment changes.

MobileNet's low memory footprint and speed make it ideal for real-time ECG classification on resource-constrained devices like smartphones or tablets, expanding access to reliable cardiac screening in remote or field-based settings.

DenseNet

DenseNet (Densely Connected Convolutional Network) introduces a novel architecture where each layer receives input from all preceding layers. This design promotes feature reuse, reduces vanishing gradient problems, and improves learning efficiency.

In the ECG classification system, DenseNet excels in capturing complex interdependencies between various regions of ECG waveforms. Its layered connectivity ensures that even low-level features like P-waves or T-wave morphologies are preserved and enhanced in deeper layers. The model is particularly effective in identifying subtle differences across similar-looking classes like History of MI and Myocardial Infarction.

Despite being deeper than traditional CNNs, DenseNet is computationally efficient due to its parameter-sharing design. It delivers high accuracy and is suitable for environments where diagnostic precision is prioritized, such as in hospital systems or research labs.

Hybrid Model (MobileNet + LSTM)

The hybrid model combines the spatial learning power of MobileNet with the temporal sequence 34odelling ability of Long Short-Term Memory (LSTM) networks. ECG signals, even when represented as images, contain sequential information about electrical activity over time. While CNNs and MobileNet handle spatial aspects, LSTM is adept at learning long-range temporal dependencies, such as repeated heartbeat patterns or arrhythmic sequences.

In this architecture, MobileNet is used as a feature extractor to convert ECG images into a sequence of high-level features. These features are then passed to the LSTM layer, which captures the temporal evolution of patterns before making the final classification.

This hybrid model outperforms individual architectures in terms of accuracy and generalization and is particularly suited for classifying ECGs with complex temporal anomalies that are not easily captured by spatial models alone.

Data Preprocessing and Feature Engineering

All ECG images undergo thorough preprocessing before being fed into the models. This includes resizing the images to a consistent dimension (e.g., 224×224 pixels), grayscale conversion, and normalization of pixel values to a range of [0,1]. These steps ensure that the input data is standardized, which stabilizes training and improves convergence.

To enhance generalization and reduce overfitting, data augmentation is applied. Techniques include random rotations, horizontal and vertical flipping, zoom, brightness adjustment, and contrast variation. These transformations simulate real-world variations in ECG printouts and improve model robustness.

While deep learning handles feature extraction automatically, high-quality preprocessing ensures better training results and reliable predictions across different ECG formats and noise levels.

Model Training and Evaluation

The dataset is divided into training, validation, and test sets. The training set is used to optimize model parameters, the validation set guides hyperparameter tuning and early stopping, and the test set evaluates generalization performance.

The models are compiled using the categorical cross-entropy loss function and optimized with the Adam optimizer. Training is performed in epochs with callbacks like early stopping and learning rate schedulers to avoid overfitting and improve convergence.

Performance is evaluated using metrics such as:

- Accuracy – Overall classification correctness
- Precision – Correct predictions for each class

- Recall – Ability to detect actual positives (e.g., MI cases)
- F1-Score – Harmonic mean of precision and recall

These metrics provide a well-rounded evaluation, essential in medical diagnosis where false negatives can be dangerous.

Performance Comparison

After training, each model is compared across multiple performance criteria:

- CNN provides a solid baseline with customizable architecture and moderate accuracy, suitable for research and controlled environments.
- MobileNet is lightweight and fast, making it ideal for real-time use in low-resource or mobile setups.
- DenseNet delivers high accuracy due to its efficient connectivity and is better suited for clinical environments with high reliability requirements.
- The Hybrid Model (MobileNet + LSTM) outperforms all others in capturing both spatial and temporal dynamics, offering the best accuracy and generalizability.

CHAPTER 7 - TESTING & DEBUGGING TECHNIQUES

A. Unit Testing

Purpose:
Unit testing is essential for verifying that individual components of the ECG classification system work correctly in isolation. It focuses on the smallest testable parts of the system—functions or methods—such as ECG image preprocessing, model loading, and prediction logic. These tests help detect early bugs, reduce integration issues, and ensure that each component performs its expected functionality independently before combining into the full system pipeline.

Tools:
Python's built-in unittest and the pytest framework are used for unit testing. They provide decorators and assertion methods to validate the correctness of outputs against expected results.

Example:
Write a test to validate the image preprocessing function. Given an ECG image of any resolution, the function should return a NumPy array resized to 224×224, normalized to [0, 1], and correctly shaped to match the model input format.

1. Integration Testing

Purpose:
Integration testing checks how well multiple components work together. In this ECG system, it validates the communication between modules such as the Flask web server, image preprocessing unit, classification model (CNN, MobileNet, DenseNet, or Hybrid), and the frontend display. This test ensures smooth end-to-end data flow and error-free integration of functionalities.

Tools:
Integration testing is carried out using pytest for backend testing, while Postman or Selenium is used to validate frontend-backend API calls in simulated user scenarios.

Example:
Test the full workflow from ECG image upload → backend image processing via Flask → prediction using MobileNet+LSTM → result sent back and displayed on the user interface.

2. Model Evaluation Testing

Purpose:
Model evaluation testing ensures that the trained models (CNN, MobileNet, DenseNet, Hybrid) perform well on unseen ECG images and generalize across datasets. This testing uses metrics like accuracy, precision, recall, and F1-score to verify that the models correctly classify ECG images into one of four categories: Normal, Abnormal Heartbeat, History of MI, and Myocardial Infarction.

Tools:
Libraries such as Scikit-learn, TensorFlow, and Keras are used for evaluating performance using tools like classification_report, confusion_matrix, and train_test_split.

Example:
Split the dataset 80:20 and evaluate the hybrid model using precision, recall, and F1-score. Confirm that the model achieves ≥90% accuracy in classifying different ECG classes.

B. Debugging

Purpose:
Debugging involves identifying and fixing errors in system behavior during development. It ensures correctness in preprocessing, prediction, and result rendering, especially when the system encounters incorrect input formats or crashes during execution. Debugging helps trace logic errors and correct inconsistencies in real-time.

Tools:
pdb, print() statements, Python’s logging module, and debugging tools built into IDEs like VS Code or PyCharm help in tracing execution and monitoring variable states step-by-step.

Example:
Use the VS Code debugger to step through the backend prediction process, confirming that ECG images are correctly resized, converted to tensors, and passed into the MobileNet model without errors.

1. Boundary Testing

Purpose:
Boundary testing checks how the system handles extreme or abnormal input values, ensuring stability under unusual conditions. This is crucial for real-time ECG analysis where user uploads may include oversized, corrupted, or noisy ECG images.

Tools:
Custom scripts, pytest parameterization, and synthetic test images are used to evaluate system responses to boundary cases.

Example:
Upload a large-resolution ECG image (e.g., 5000×5000 pixels) and validate that the preprocessing function correctly resizes it without memory overflow or crashing the server.

2. Real-Time Data Testing

Purpose:
Real-time testing evaluates how efficiently and accurately the system handles live user inputs.

In a clinical setting or remote screening camp, prediction latency should be minimal to ensure immediate diagnosis support.

Tools:
Use Postman for API testing, mock real-time ECG uploads, and monitor prediction time using Python's time module or stopwatch-based manual verification.

Example:
Upload an ECG image and measure the end-to-end processing time. The system should return a classification result (e.g., "History of MI") within **2 seconds** without server lag.

3. User Interface (UI) Testing

Purpose:
UI testing ensures that the user-facing web application is responsive, clear, and usable across various devices. Healthcare workers must be able to upload ECG images, view predictions, and navigate without confusion or layout issues.

Tools:
UI testing is performed using Selenium for automation and manual inspection across different browsers and screen sizes to ensure layout adaptability and user-friendliness.

Example:
Open the web app on both a mobile and desktop browser, upload a test ECG, and verify that the predicted result is clearly visible and aligned beneath the image upload section.

4. Performance Testing

Purpose:
Performance testing evaluates the system's efficiency under high load or multiple concurrent requests. This ensures that the ECG classification pipeline scales without performance degradation or crashing under stress.

Tools:

Use Python's time module for single-instance timing, and tools like Locust, JMeter, or TensorFlow Profiler for simulating concurrent users and monitoring server stress.

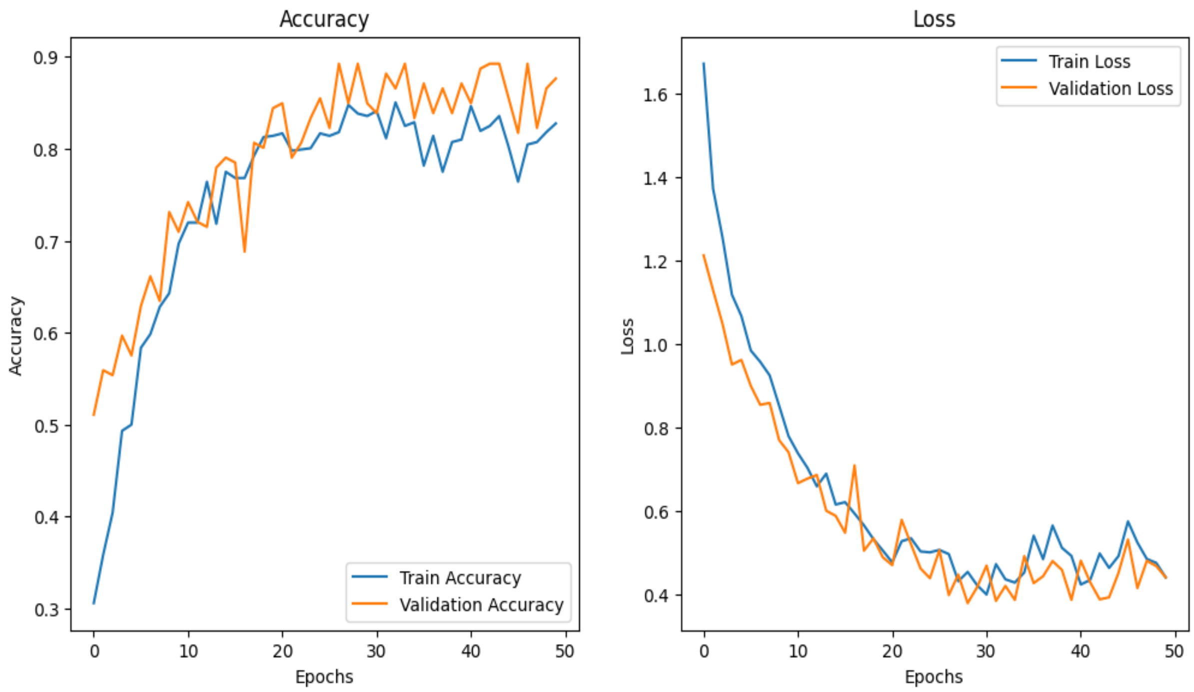
Example:

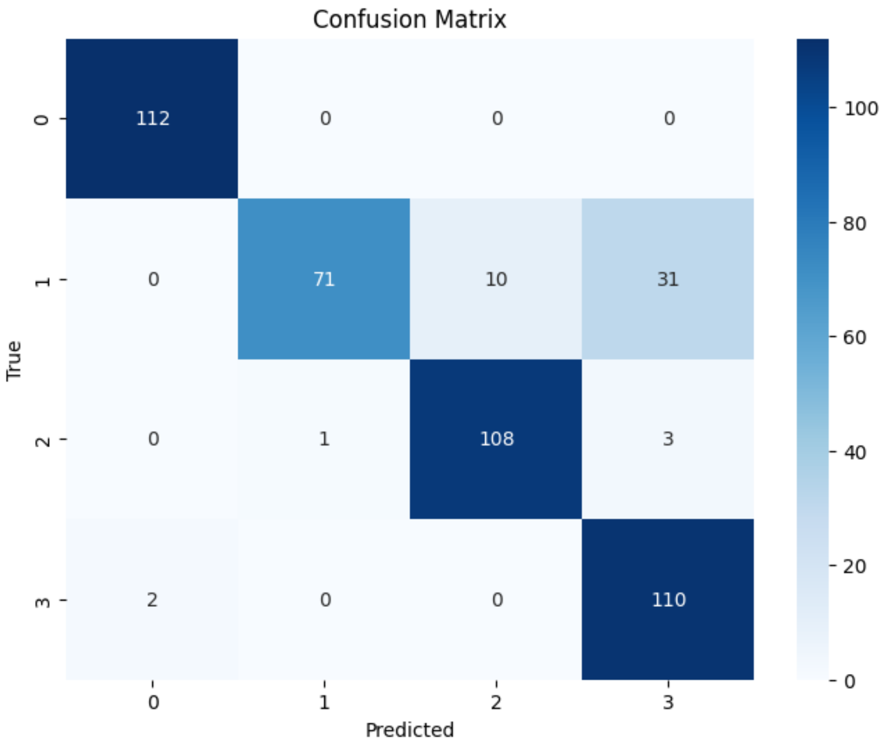
Upload a batch of 50 ECG images consecutively and measure average classification time per image. Ensure response time stays consistent and memory usage remains stable.

CHAPTER 8 – OUTPUT SCREENS

Model Performances:

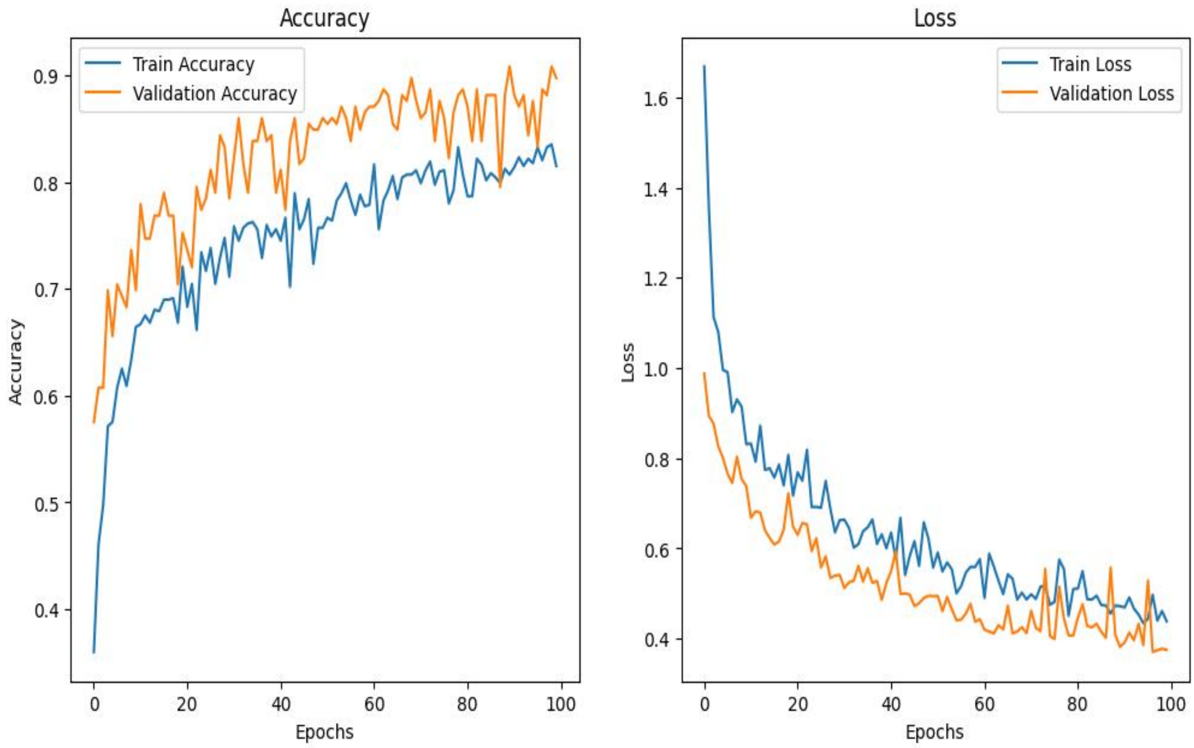
CNN:

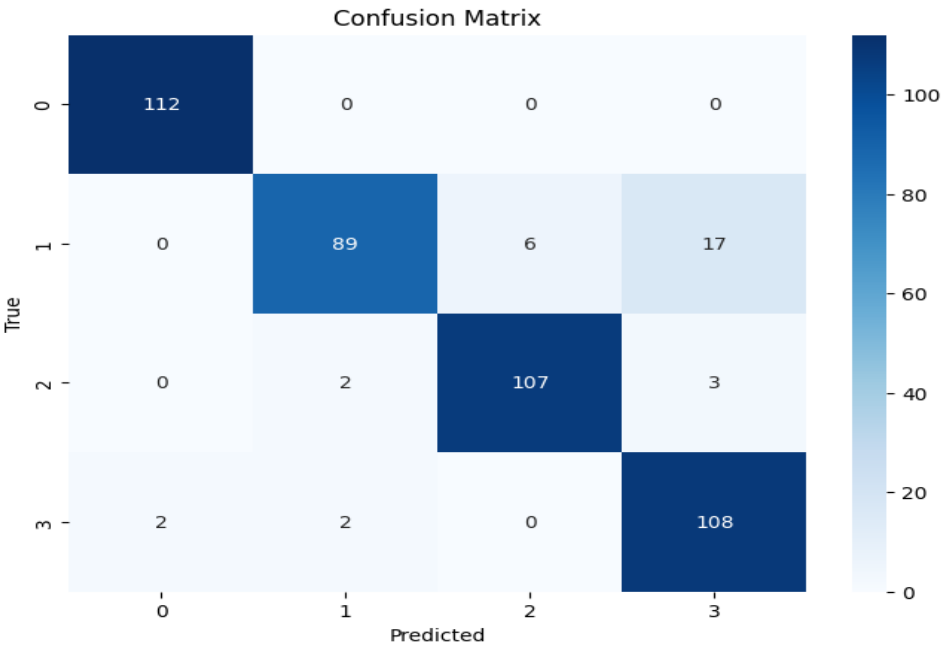




The Convolutional Neural Network (CNN) model effectively classifies ECG images into four categories: Normal, Abnormal Heartbeat, History of MI, and Myocardial Infarction. The training and validation accuracy curves show a steady increase, reaching above 85%, while loss values decrease consistently, indicating good convergence. The confusion matrix highlights strong performance across all classes, with only minor misclassifications, particularly between classes 1 and 3. The model demonstrates high precision in detecting class 0 and class 3. Overall, the CNN model shows robust generalization, learning both morphological and pattern-based features from ECG images, making it a reliable tool for cardiac condition classification.

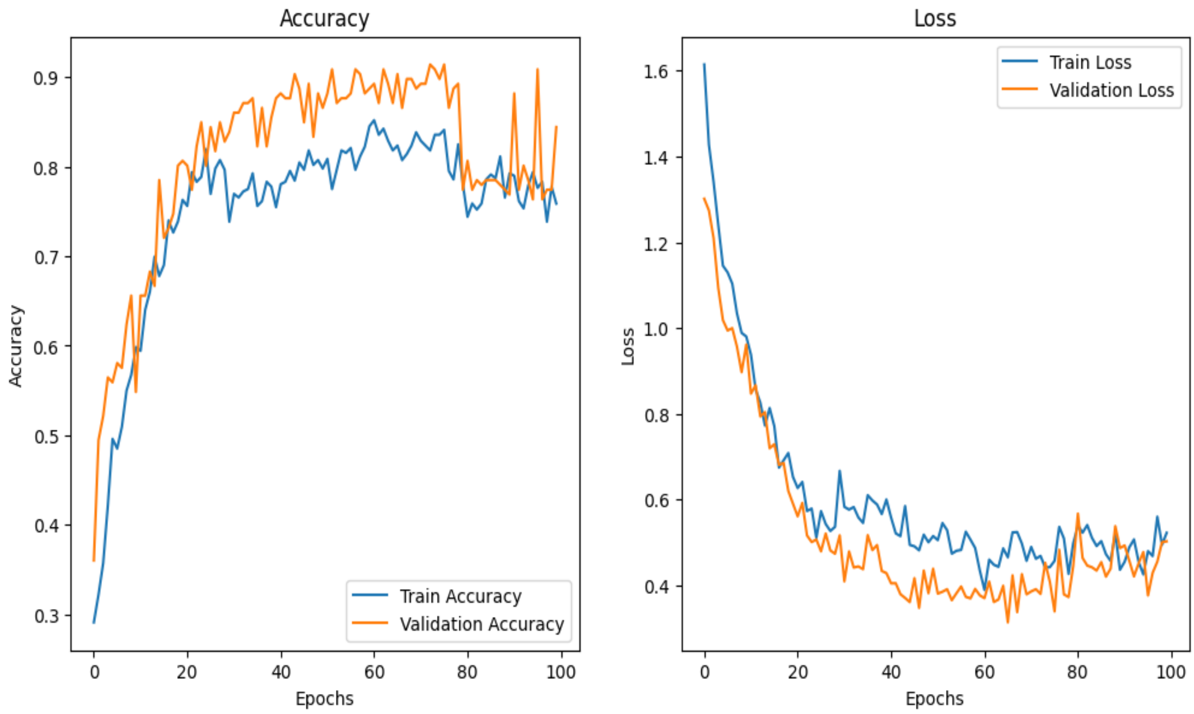
Mobilenet Model:

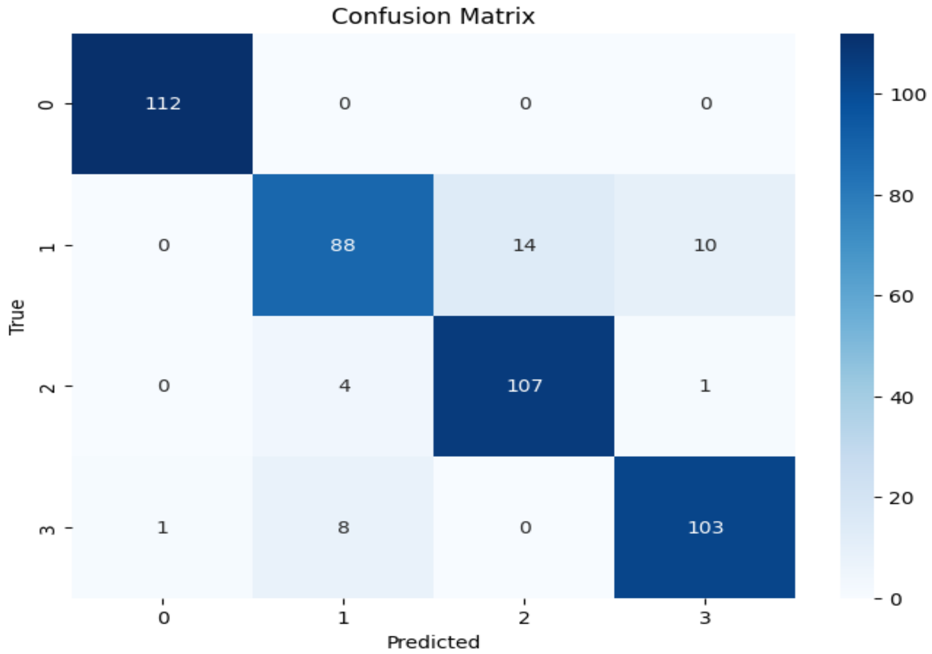

Accuracy
Train Accuracy
Validation Accuracy
Accuracy
Epochs
Loss
Train Loss
Validation Loss
Loss
Epochs


Confusion Matrix
112
0
0
0
0
89
6
17
0
2
107
3
2
2
0
108
True
Predicted

The MobileNet model demonstrates strong performance in classifying ECG images into four classes. Training and validation accuracy steadily improve across 100 epochs, with validation accuracy exceeding 90%, indicating excellent generalization. The loss curves show a consistent decrease, confirming effective learning without overfitting. The confusion matrix highlights high precision for all classes, especially class 0 and class 3, with minor misclassifications in class 1. This lightweight model, optimized using transfer learning, achieves a strong balance between accuracy and efficiency. Its low computational cost and high inference speed make it ideal for deployment in mobile or resource-constrained environments for real-time ECG analysis.

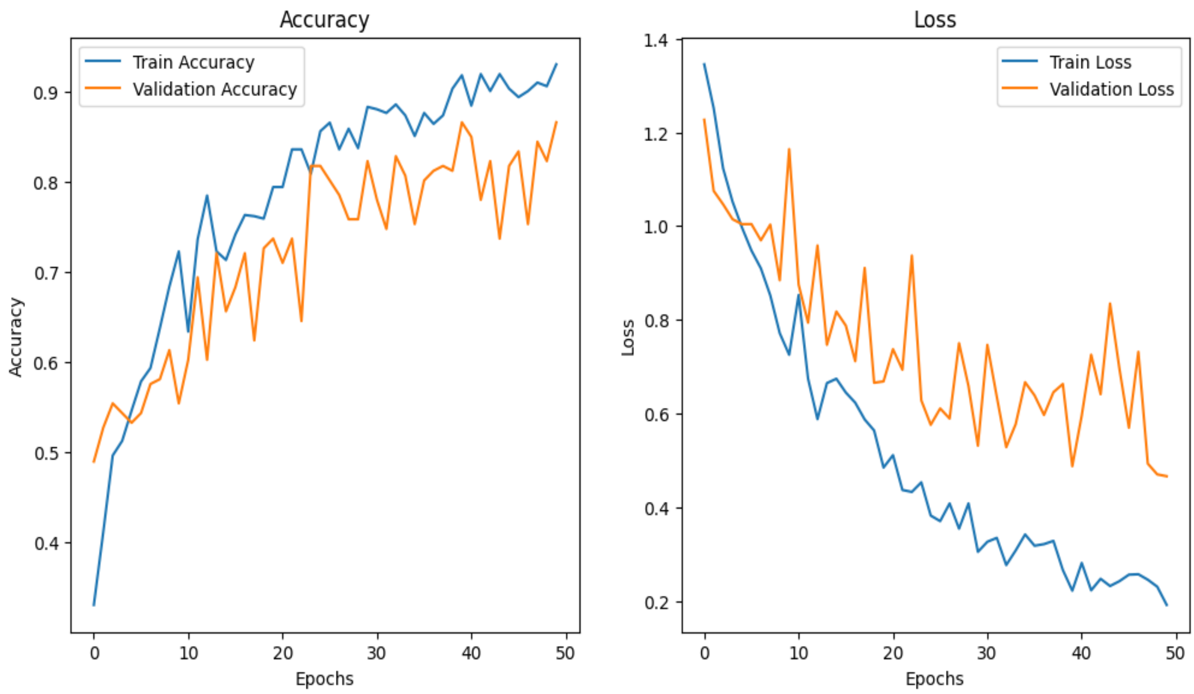
Dense Net:

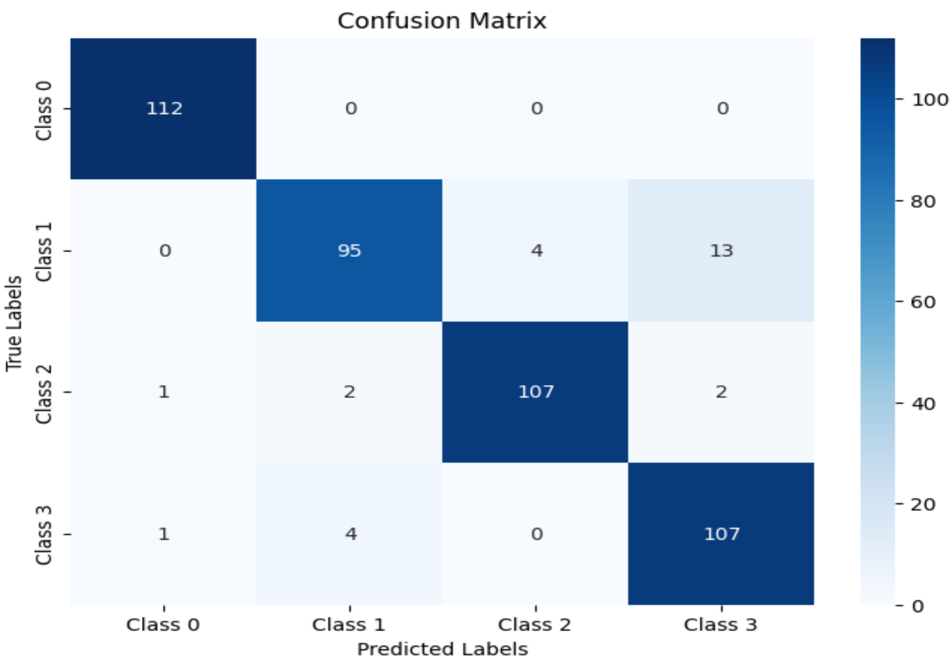




The DenseNet model achieves strong performance in ECG classification by efficiently reusing features across layers. Over 100 epochs, the model demonstrates high accuracy, with validation accuracy consistently above **85%**, while loss values steadily decrease and stabilize, showing effective training and convergence. The confusion matrix confirms high precision for class 0 and class 2, though some misclassifications are observed in class 1 and class 3. Despite this, DenseNet handles inter-class similarities well and benefits from deep connectivity and gradient flow. Its high capacity and efficient architecture make it a reliable choice for clinical-grade ECG interpretation with superior feature extraction capabilities.

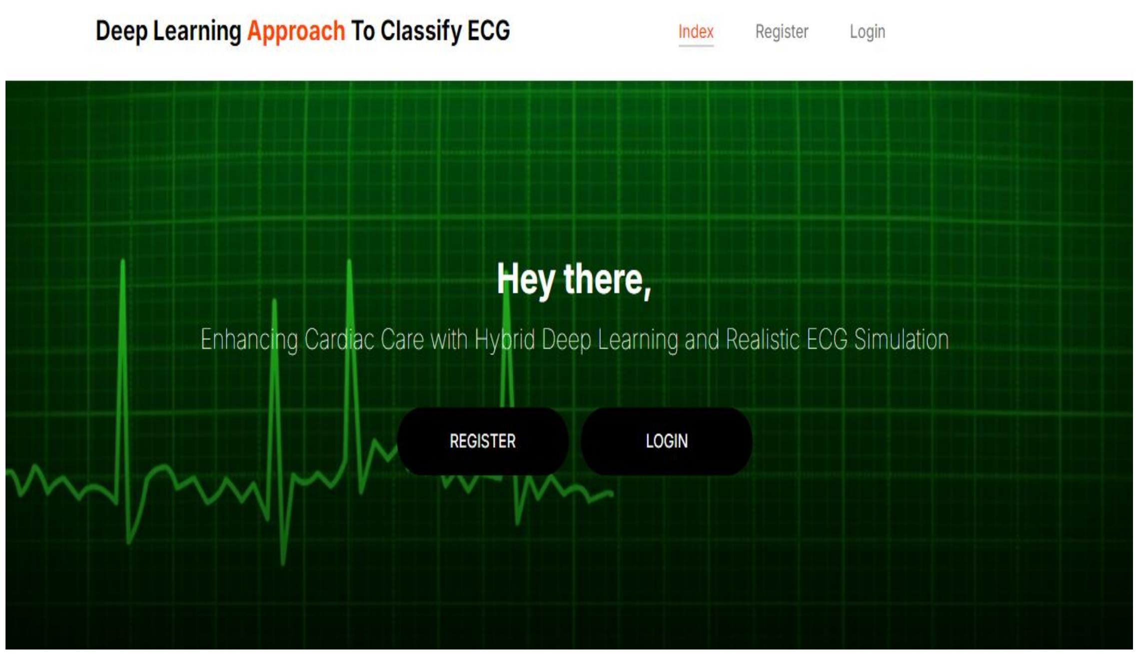
Hybrid Model:



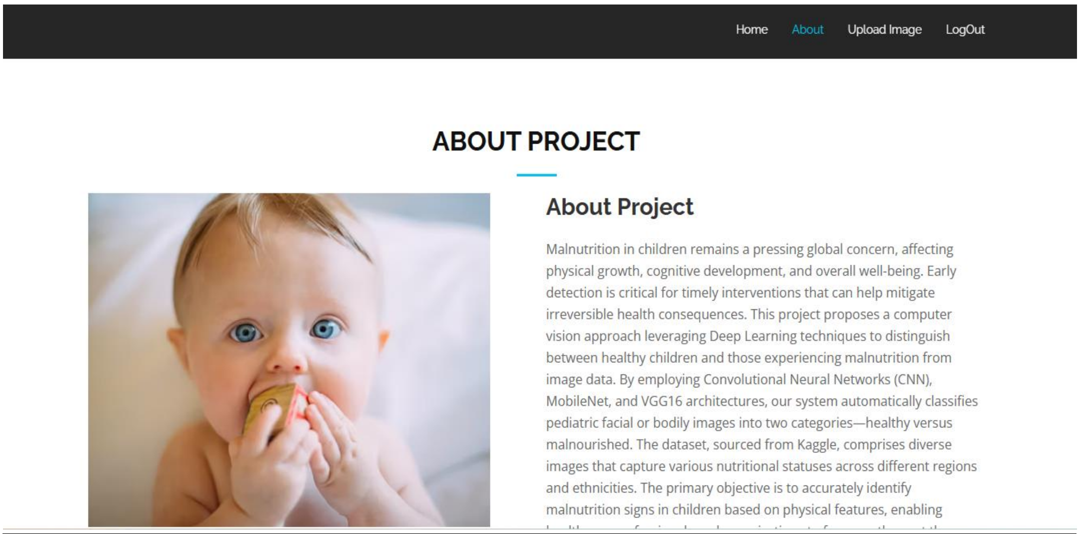


The Hybrid model, integrating MobileNet for spatial feature extraction and LSTM for temporal sequence learning, demonstrates exceptional performance in ECG classification. Training accuracy exceeds 90%, and validation accuracy remains consistently above 85% across 50 epochs, indicating strong generalization. Loss curves confirm effective learning, though slight validation fluctuations suggest room for further optimization. The confusion matrix shows excellent classification for all four classes, particularly Class 0 and Class 2, with minor misclassifications in Class 1 and 3. This hybrid architecture leverages both spatial and sequential ECG characteristics, making it highly effective for real-time, multi-class cardiac diagnosis in diverse clinical scenarios.

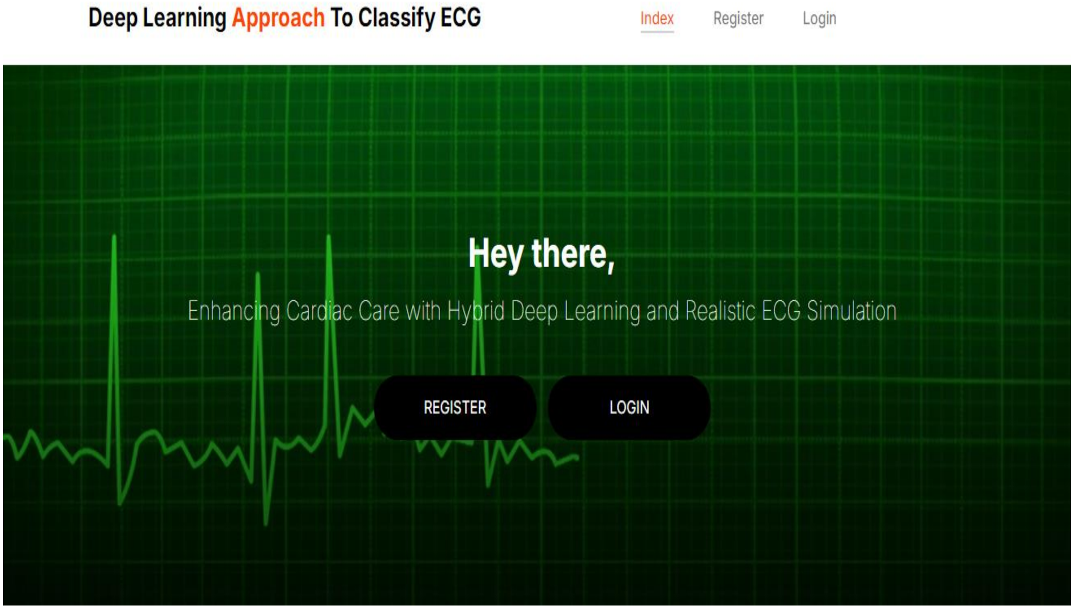
Home page:



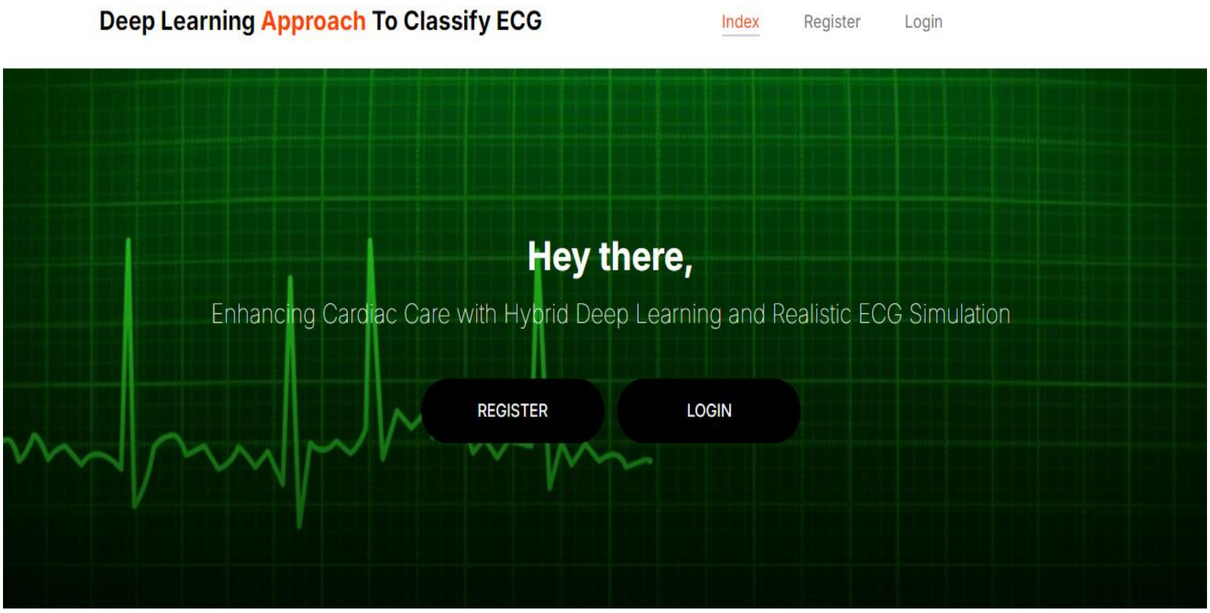
About page:



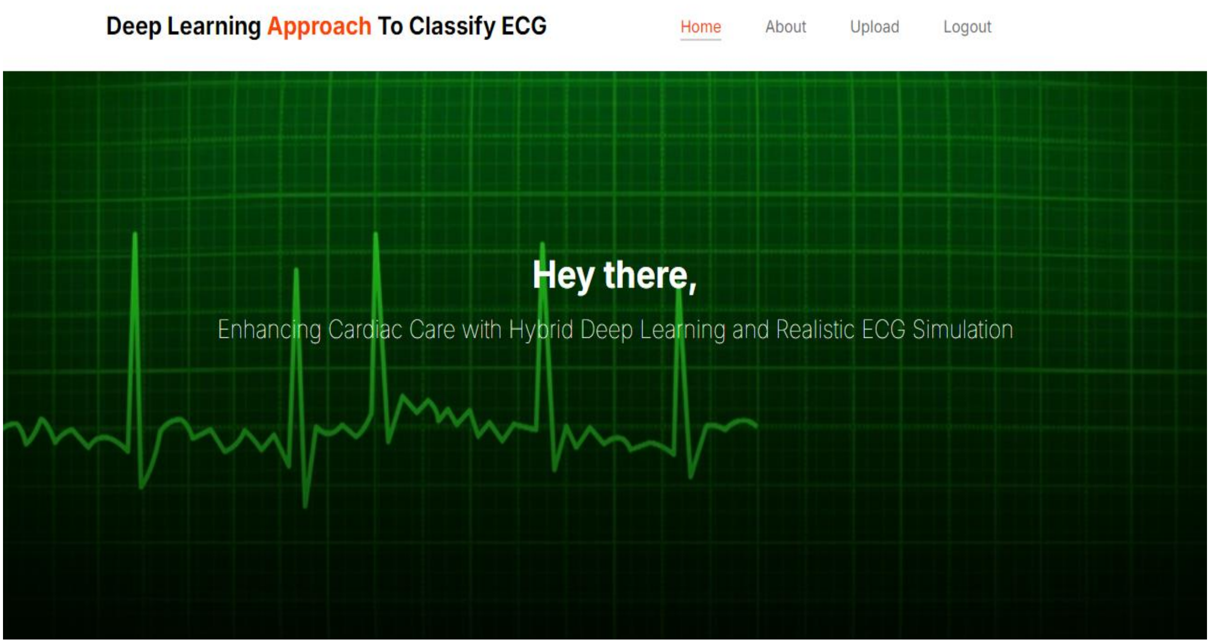
Registration page:



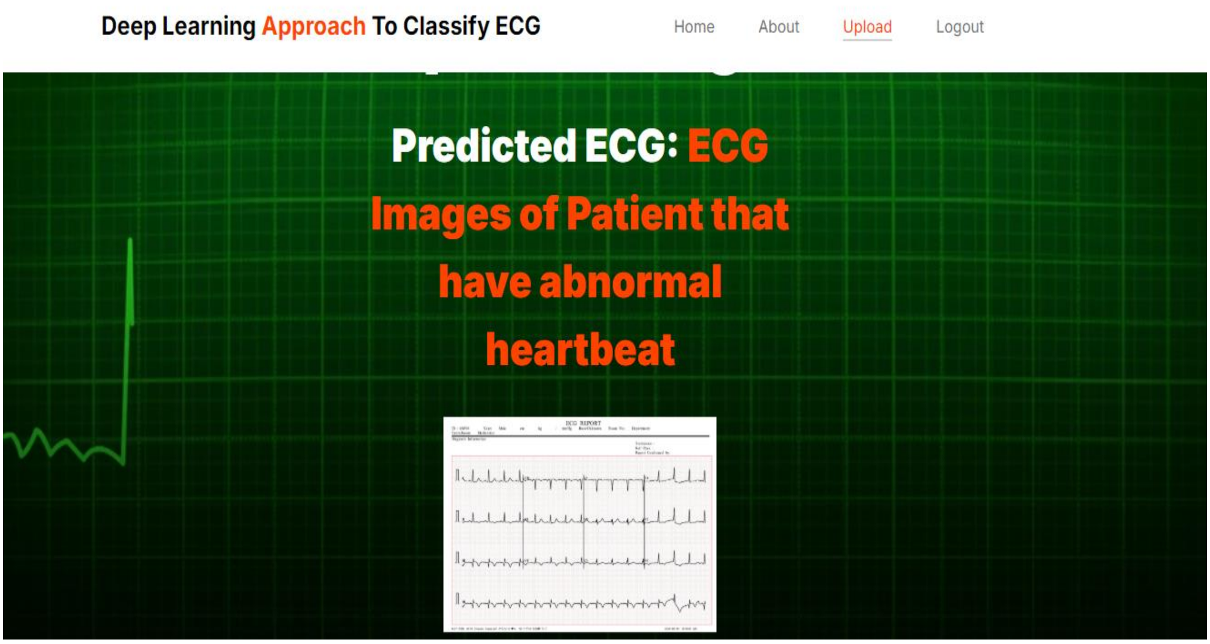
Login Page:



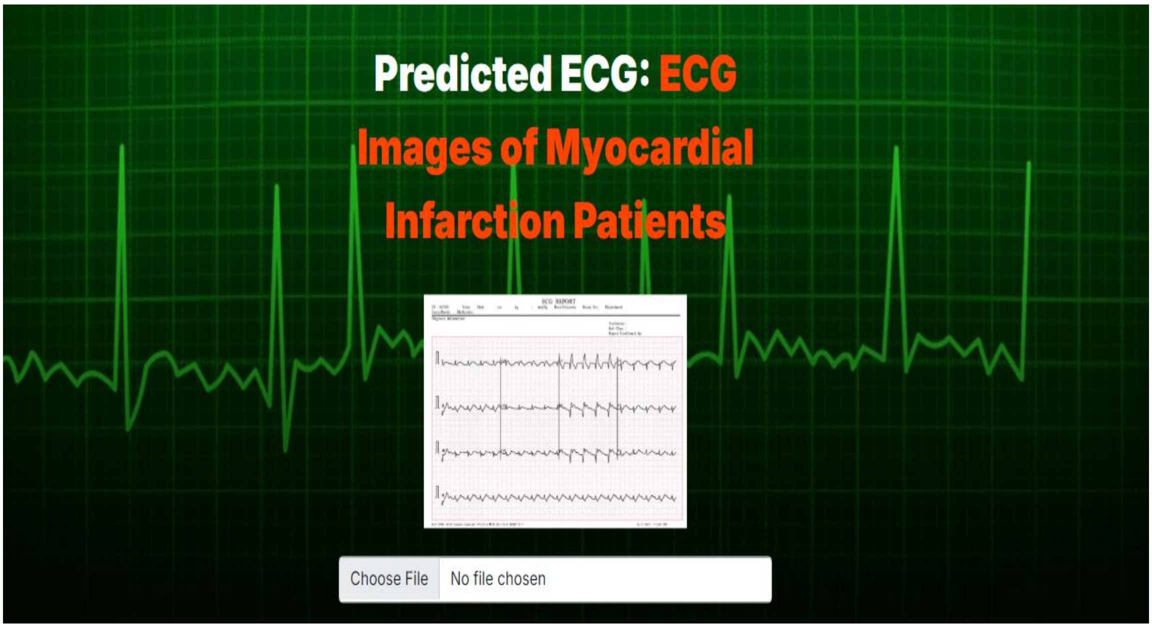
User Home page:



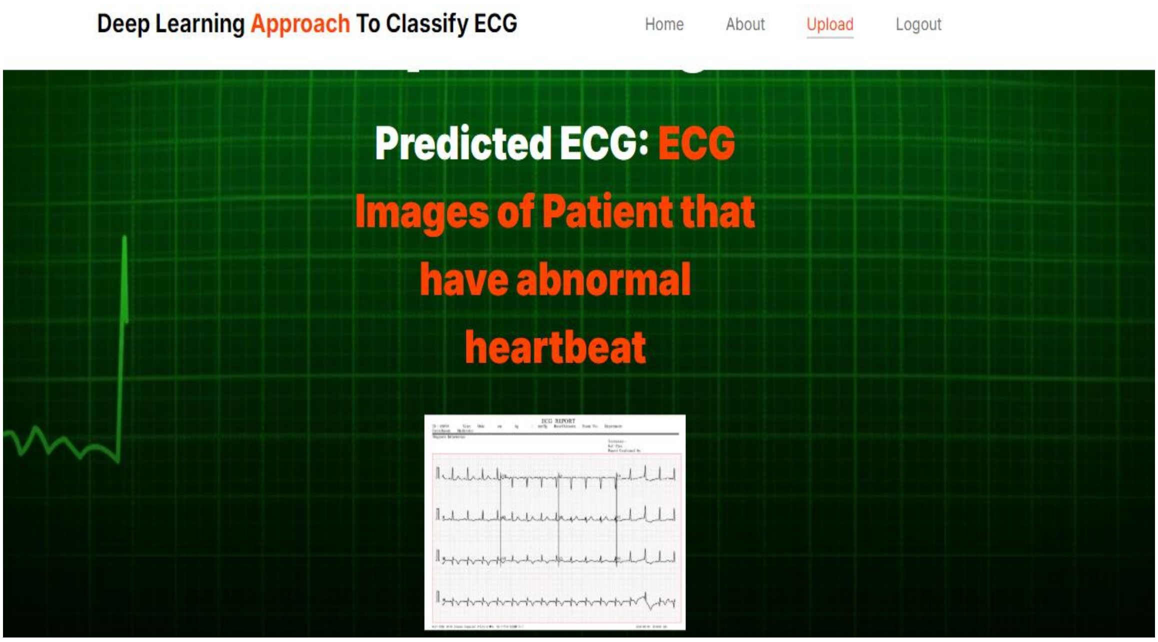
Prediction 1:



Prediction 2 :



Prediction 3 :



CHAPTER 9 –CODE

```
from flask import Flask, url_for, redirect, render_template, request, session
import mysql.connector, os
import tensorflow as tf
from tensorflow.keras.preprocessing.image import load_img, img_to_array
import matplotlib as plt
import numpy as np
import tensorflow as tf
from flask import Flask, render_template, request
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import load_img, img_to_array
from tensorflow.keras.applications import MobileNet
from tensorflow.keras.applications.mobilenet import preprocess_input
from tensorflow.keras.models import Model
from tensorflow.keras.layers import GlobalAveragePooling2D


app = Flask(__name__)
app.secret_key = 'admin'
mydb = mysql.connector.connect(
    host="localhost",
```

```
    user="root",
    password="",
    port="3308",
    database='ecgsignal'
)
mycursor = mydb.cursor()


def executionquery(query,values):
    mycursor.execute(query,values)
    mydb.commit()
    return
def retrivequery1(query,values):
    mycursor.execute(query,values)
    data = mycursor.fetchall()
    return data
def retrivequery2(query):
    mycursor.execute(query)
    data = mycursor.fetchall()
    return data
@app.route('/')
def index():
```

```
    return render_template('index.html')
@app.route('/register', methods=["GET", "POST"])
def register():
    if request.method == "POST":
        email = request.form['email']
        password = request.form['password']
        c_password = request.form['c_password']
        if password == c_password:
            query = "SELECT UPPER(email) FROM users"
            email_data = retrivequery2(query)
            email_data_list = []
            for i in email_data:
                email_data_list.append(i[0])
            if email.upper() not in email_data_list:
                query = "INSERT INTO users (email, password) VALUES (%s, %s)"
                values = (email, password)
                executionquery(query, values)
                return render_template('login.html', message="Successfully Registered!")
            return render_template('register.html', message="This email ID is already exists!")
        return render_template('register.html', message="Conform password is not match!")
    return render_template('register.html')
```

```
@app.route('/login', methods=["GET", "POST"])
def login():
    if request.method == "POST":
        email = request.form['email']
        password = request.form['password'
        query = "SELECT UPPER(email) FROM users"
        email_data = retrivequery2(query)
        email_data_list = []
        for i in email_data:
            email_data_list.append(i[0])
        if email.upper() in email_data_list:
            query = "SELECT UPPER(password) FROM users WHERE email = %s"
            values = (email,)
            password__data = retrivequery1(query, values)
            if password.upper() == password__data[0][0]:
                global user_email
                user_email = email
                return render_template('home.html')
            return render_template('login.html', message= "Invalid Password!!")
        return render_template('login.html', message= "This email ID does not exist!")
    return render_template('login.html')
```

```
@app.route('/home')

def home():

    return render_template('home.html')

@app.route('/about')

def about():

    return render_template('about.html')

# Load the saved classification mode

model = tf.keras.models.load_model('mobilenet_v2_classifier_model.h5')

# Reload the MobileNet feature extractor

base_model = MobileNet(weights='imagenet', include_top=False)

feature_extractor = Model(inputs=base_model.input,
outputs=GlobalAveragePooling2D()(base_model.output))

class_names = ['Patient having Myocardial Infarction', 'Patient that have History of MI',
'Patient that have abnormal heartbeat', 'Normal Person ECG Image']

def make_prediction(model, image_path):    """

    Preprocess the image and make a prediction using the trained model.

    """

    # Load the image and preprocess it

    img = load_img(image_path, target_size=(224, 224))  # Resize to MobileNet input size

    img_array = img_to_array(img)  # Convert to a NumPy array

    img_array = np.expand_dims(img_array, axis=0)  # Add batch dimension
```

```
    img_array = preprocess_input(img_array)  # Preprocess for MobileNet
    # Extract features using the MobileNet feature extractor
    features = feature_extractor.predict(img_array)
    # Predict the class using the trained classification model
    predictions = model.predict(features)
    predicted_class_idx = np.argmax(predictions)  # Get index of the highest probability
    predicted_class = class_names[predicted_class_idx]  # Map index to class name
    return predicted_class
@app.route('/upload', methods=["GET", "POST"])
def upload():
    if request.method == "POST":
        myfile = request.files['file']  # Get the uploaded file
        fn = myfile.filename  # Extract filename
        mypath = os.path.join('static', 'img', fn)  # Save path
        myfile.save(mypath)  # Save the file to the server
        # Make prediction
        predicted_class = make_prediction(model, mypath)
            return render_template('upload.html', path=mypath, prediction=predicted_class)
    return render_template('upload.html')
if __name__ == '__main__':
    app.run(debug = True)
```

CHAPTER 10 – CONCLUSION

In conclusion, this project presents a novel and intelligent approach to automated ECG signal classification using a hybrid deep learning model. By leveraging advanced architectures such as Convolutional Neural Networks (CNN), MobileNet, DenseNet, and a hybrid MobileNet+LSTM model, the system classifies ECG images into four key cardiac categories: Normal, Abnormal Heartbeat, History of Myocardial Infarction, and Myocardial Infarction. The integration of both spatial and temporal pattern recognition enables accurate and robust diagnosis, improving upon traditional methods that are manual, time-consuming, and reliant on expert interpretation.

This deep learning-based solution is scalable, cost-effective, and adaptable to various clinical and remote environments, making it suitable for real-time applications in telehealth, rural screening camps, and diagnostic centers. The use of transfer learning and model optimization ensures efficient training even with limited data, while the web-based interface developed using Flask provides seamless interaction for end-users.

The system has shown promising performance in terms of accuracy and generalization. However, future enhancements such as incorporating multi-lead ECG data, refining the dataset for improved diversity, and deploying edge-device optimized versions can further elevate the system's impact. Overall, this project demonstrates the potential of AI in transforming cardiac healthcare by enabling early detection, reducing diagnostic workload, and ensuring accessible cardiac screening at scale.

CHAPTER 11 – BIBLOGRAPHY

[1] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[2] G. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *arXiv preprint*, arXiv:1704.04861, 2017.

[3] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv preprint*, arXiv:1409.1556, 2014.

[4] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017.

[5] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent Neural Network Regularization," *arXiv preprint*, arXiv:1409.2329, 2014.

[6] M. Abadi et al., "TensorFlow: A System for Large-Scale Machine Learning," in *OSDI*, vol. 16, 2016, pp. 265–283.

[7] S. Mehta and A. Singh, "ECG Classification using Deep Learning Techniques," *International Journal of Computer Applications*, vol. 182, no. 21, pp. 1–5, 2019.

[8] A. Acharya et al., "Automated Diagnosis of Arrhythmia Using Convolutional Neural Networks," *Information Sciences*, vol. 415, pp. 190–198, 2017.

[9] H. Rajpurkar et al., "Cardiologist-Level Arrhythmia Detection with Convolutional Neural Networks," *arXiv preprint*, arXiv:1707.01836, 2017.

[10] D. Kermany et al., “Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning,” *Cell*, vol. 172, no. 5, pp. 1122–1131, 2018.

[11] WHO, “Cardiovascular Diseases (CVDs),” *World Health Organization*, 2022. [Online]. Available: https://www.who.int

[12] N. Japkowicz and M. Shah, *Evaluating Learning Algorithms: A Classification Perspective*, Cambridge University Press, 2011.

[13] J. Deng et al., “ImageNet: A Large-Scale Hierarchical Image Database,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Miami, FL, 2009, pp. 248–255.

[14] T. Y. Zhang, W. Xu, and C. Liu, “A Deep Learning Approach for ECG Classification Based on Temporal and Spatial Features,” *Biomedical Signal Processing and Control*, vol. 57, 2020.

[15] A. Esteva et al., “Dermatologist-Level Classification of Skin Cancer with Deep Neural Networks,” *Nature*, vol. 542, no. 7639, pp. 115–118, 2017.

CARDIAC CONDUCTION SIMULATION A HYBRID DEEP LEARNING APPROACH TO CLASSIFY REALISTIC ECG SIGNALS

Mr. K. Chandra Sekhare[1], Bandi Sireesha[2]

[1] HOD & Assistant Professor, [2]Post Graduate Student
Department of MCA
VRN College of Computer Science and Management, Andhra Pradesh, India

Abstract-The ECG has an eminent role in diagnosing conditions like myocardial infarction (MI) and arrhythmias, and accurate classification of ECG signals in the modern era has turned into a fundamental issue for treatment and prevention. The project consists of a detailed comparative study of four state-of-the-art deep learning techniques-CNN, MobileNet, DenseNet, and an ensemble model combining both MobileNet and LSTM-in classifying ECG images into one of the four categories: myocardial infarction, history of MI, arrhythmia, and normal conditions. Associated with the dataset are ECG images with labels that permit the models to learn both spatial and temporal features that are very crucial for the accurate classification task. Evaluation of the performances of the respective algorithms considers accuracy, precision, recall, and F1-score. The results indicate that the ensemble model outperforms individual architectures, fusing the MobileNet performance on spatial feature extraction with LSTM performance on sequential pattern recognition to achieve a higher level of accuracy. In doing so, this study has a bright future in the development of dependable automated diagnostic carriers in clinical use. Future projects will involve using multi-lead ECG signals with added metadata to increase reliability and scalability of the system for accurate classification in cardiac health systems.

Keywords:classification, cardiac conditions, CNN, MobileNet, DenseNet, LSTM, hybrid model, deep learning, myocardial infarction, comparative analysis, diagnostic tools.

Introduction

Cardiovascular diseases are counted among the foremost killers in the entire world, and these may also include myocardial infarction and arrhythmias. Doctors should be able to immediately identify and classify the cardiac abnormalities in electrocardiograms (ECGs) for their patients to facilitate timely medical attention and mitigate effects on patients' lifestyles. Such interpretations, however, take considerable time and may involve critical human error in busy clinical settings. More specifically, deep learning in artificial intelligence concerning cardiac diagnostics may introduce great promise of high-precision, high-reliability algorithms for automating the interpretation of the ECG signals.

This research work introduces an intelligent ECG classification system based on deep learning for the accurate detection and classification of cardiac conditions on actual ECG images. Four advanced models are used for conducting this study: Convolutional Neural Network (CNN), MobileNet, DenseNet, and hybrid MobileNet-LSTM ensemble models that are tested for their abilities to decipher spatial and temporal salient features embedded in ECG data. Additionally, ensemble modeling is especially useful in that it combines mobile net efficiency in terms of spatial feature extraction with the sequence learning capacity of LSTM to provide robust inference even in cases of partially distorted signals or subtle anomalie

To facilitate practical deployment, the models are trained on a custom ECG image dataset representing a range of conditions: myocardial infarction, history of MI, abnormal heartbeat, and normal heart function. The classification system is integrated into a user-friendly interface built with Flask, HTML, CSS, and JavaScript, allowing medical professionals to upload ECG visuals, perform real-time predictions, and visualize diagnostic results. This AI-powered approach has the potential to enhance diagnostic accuracy, reduce clinician workload, and contribute to more responsive and scalable cardiac healthcare solutions. Future enhancements may include support for multi-lead ECG analysis, incorporation of patient metadata, and optimization for use on mobile or wearable health devices.

Objective Of The Study:

This project is designed to deliver a fast, accurate, and intelligent solution for classifying cardiac conditions using realistic ECG signals, leveraging the power of advanced deep learning techniques. The primary objective is to develop an automated ECG classification system using state-of-the-art neural network architectures—specifically CNN, MobileNet, DenseNet, and a hybrid MobileNet-LSTM ensemble—capable of accurately identifying and categorizing various cardiac conditions such as myocardial infarction, history of MI, abnormal heartbeat, and normal heart rhythms.

The study emphasizes key stages including the collection and preprocessing of ECG image data, training and fine-tuning of each deep learning model, comparative analysis of performance across multiple metrics, and evaluation of the models in terms of their real-world applicability. The ensemble model, combining MobileNet and LSTM, is particularly optimized to extract both spatial and temporal features from ECG signals, thereby enabling more precise classification even under signal variations or noise.

Unlike traditional manual interpretation or older machine learning models, this project focuses on utilizing deep learning as a standalone approach to deliver high accuracy, scalability, and speed in diagnostic classification. A user-centric web-based interface is developed using HTML, CSS, JavaScript, and Flask, allowing healthcare professionals to upload ECG visuals and receive real-time classification results with intuitive visual feedback.

By streamlining the ECG classification workflow into a practical, accessible, and cost-effective system, this study aims to assist medical practitioners in early diagnosis, reduce diagnostic errors, and ultimately enhance patient outcomes. Future enhancements may include integration with wearable ECG monitoring devices, support for multi-lead ECG inputs, and deployment on edge-based health monitoring platforms to facilitate continuous, remote cardiac monitoring. The project contributes to the broader vision of applying artificial intelligence in healthcare to improve diagnostic accuracy, efficiency, and accessibility in cardiovascular disease management.

B. Problem statement:

Heart diseases including myocardial infarction, arrhythmias, and other irregular cardiac conditions are, in actuality, defined new medical horizons. Early and accurate classification of these states, therefore, becomes imperative for proper medical intervention and treatment. Unlike this advancement, the traditional way of diagnosing on ECG signals is chiefly dependent on manual analyzing of data contained in the tracings by hospital workers. The lengthy and hard path is rattled with human error and differences of interpretation under stress situations or in remote areas lacking trained professionals.

One of the primary challenges in automated ECG classification lies in the complexity and variability of ECG signals. Real-world ECG data can be noisy, subject to occlusions, and influenced by diverse factors such as patient history, comorbidities, and recording inconsistencies. Many conventional machine learning methods struggle to generalize across such variability, leading to reduced accuracy and reliability in real-world deployments. Additionally, earlier deep learning approaches often focus on either spatial or temporal features in isolation, failing to capture the full context of ECG patterns necessary for robust classification.

To overcome these limitations, this project introduces a deep learning-based solution that utilizes multiple state-of-the-art architectures—CNN, MobileNet, DenseNet, and a hybrid MobileNet-LSTM ensemble—for automatic ECG classification. The hybrid model is particularly tailored to combine the spatial analysis strengths of MobileNet with the temporal pattern recognition of LSTM, making it highly effective in handling realistic, complex ECG signals. Trained on a curated dataset containing labeled ECG images representing multiple cardiac conditions, the models are optimized to detect subtle differences and patterns that distinguish one condition from another.

In addition, the project integrates this AI-powered classification system into a web-based platform developed using HTML, CSS, JavaScript, and Flask. This platform allows healthcare providers to upload ECG visuals and receive real-time classification outputs through an intuitive user interface. The system is designed to be lightweight, scalable, and suitable for deployment in resource-constrained environments, including remote clinics and mobile health units. By combining deep learning innovation with accessible deployment, this solution aims to enhance diagnostic accuracy, reduce time-to-diagnosis, and ultimately improve patient outcomes in cardiac care.

Related Work

1. Evolution of ECG Classification Using Deep Learning

In that long span of time, ECG signal classification has undergone much evolution transcending from rules and statistics to advanced deep learning-based models. Early-day techniques relied upon manual feature extraction using classical machine learning algorithms like support vector machines, decision trees, and k-nearest neighbors. These methods have been successful in varying degrees for pattern recognition. However, they were totally dependent on handcrafted features whose generalization across different ECG patterns was not possible. The moment convolutional neural networks came into the picture, the end-to-end learning applied to raw or image-transformed ECG data was a blessing in disguise. Feature extraction by CNNs has proven remarkably good, pumping up the classification accuracy with no human interference. This was soon followed by the development of deeper architectures including ResNet and DenseNet that stressed feature reuse and gradient flow. Meanwhile, hybrid architectures that combine CNNs with recurrent neural networks, especially long short-term memory (LSTM), are on the rise with great promise of recognizing space-time-dependent patterns presented by the ECG signal and, furthermore, improving the recognition of subtle variations sequentially occurring in the signals.

2. Advantages of Hybrid Deep Learning Models in ECG Classification

Hybrid models that combine CNNs with sequential learners like LSTM offer significant advantages for ECG analysis. While CNNs excel at spatial pattern recognition—detecting waveform morphology and segment variations—LSTMs are particularly effective in capturing temporal dependencies across time series data. In ECG signals, this synergy is critical, as arrhythmias or cardiac anomalies often span several beats and involve changes over time.

The use of lightweight yet powerful architectures like MobileNet further enhances the model's efficiency, allowing it to perform real-time inference with reduced computational overhead. Integrating MobileNet with LSTM, as in this study, ensures that the model is both resource-efficient and capable of learning complex temporal dynamics in ECG images. This approach eliminates the need for manual segmentation or feature engineering, allowing the model to adapt and generalize across different patient conditions and signal noise levels.

3. Challenges in Deep Learning-Based ECG Diagnosis

Despite the advantages, several challenges persist in deploying deep learning models for ECG classification. One major hurdle is dataset variability and imbalance—real-world ECG data may differ in lead configuration, sampling frequency, and noise levels, leading to reduced generalizability. Furthermore, datasets are often skewed, with a disproportionately higher number of normal samples compared to pathological cases.

Another challenge lies in the interpretability of deep learning models. Clinicians require transparent and explainable outputs to trust automated diagnoses, but deep models are often viewed as black boxes. Additionally, hardware constraints in low-resource settings, such as rural clinics or mobile health units, make it difficult to deploy heavy models, necessitating efficient architectures. Training time and data labeling also remain key bottlenecks, especially in acquiring annotated medical ECG datasets with expert supervision.

4. Technological Advances in AI-Powered Cardiac Diagnostics

Recent progress in deep learning has led to substantial improvements in ECG classification. Techniques like transfer learning, attention mechanisms, and self-supervised pretraining have enabled models to adapt quickly to new ECG datasets with minimal labeled data. Frameworks such as TensorFlow Lite and ONNX have made it possible to deploy complex models on mobile and edge devices, making real-time diagnosis more accessible.

In addition, cloud-based AI diagnostic platforms and integration with electronic health records (EHRs) are streamlining clinical workflows. Visualization tools such as Grad-CAM and saliency maps are helping bridge the interpretability gap, allowing physicians to see which portions of the ECG influenced the model's decision. These advancements are driving the next generation of smart cardiac monitoring systems.

5. Case Studies and Applications in Clinical Settings

Multiple studies and pilot projects have validated the use of deep learning for ECG-based diagnosis. For instance, models trained on publicly available datasets like MIT-BIH, PTB-XL, and Chapman University's ECG datasets have achieved high accuracy in detecting arrhythmias, myocardial infarction, and other abnormalities. Healthcare startups and hospitals have begun integrating AI-powered ECG tools into clinical practice, using them for early screening, remote patient monitoring, and real-time alerts.

One notable initiative involves mobile-based ECG analysis apps that enable patients to record and upload ECG data using portable devices, with backend AI models delivering instant risk assessments. Similarly, hybrid deep learning systems are being explored in ICU environments to provide continuous cardiac monitoring. These applications underscore the growing role of AI in cardiovascular healthcare and affirm the feasibility of deploying systems like the one proposed in this project.

PROPOSSED SYSTEM

The proposed system for cardiac condition classification leverages hybrid deep learning architectures to accurately analyze and classify realistic ECG signals. This approach is designed to assist healthcare professionals in the early diagnosis of heart-related conditions by automating the interpretation of ECG images, thereby reducing manual workload and improving diagnostic speed and accuracy. By focusing on both spatial and temporal features of ECG waveforms, the system intelligently distinguishes between multiple cardiac conditions such as myocardial infarction, history of MI, abnormal heartbeat, and normal rhythm patterns.

The correlating models' underlying principle solidifies in relation to a comparative framework of deep learning models-the Convolutional Neural Network (CNN), MobileNet, DenseNet, and a hybrid ensemble model of MobileNet and Long Short-Term Memory (LSTM). CNNs and DenseNet are deployed for their abilities in robust spatial feature extraction, while MobileNet provides the featherweight efficiency that makes the model perfect for resource-constrained deployment. MobileNet-LSTM is a model aimed at capturing not just spatial structures but also their temporal interdependencies, hence yielding better performance in classifying ECG signals wherein the changes occur over time.

To ensure the model receives high-quality input, advanced data preprocessing techniques are applied to the ECG image dataset. These include noise reduction, normalization, resizing, and augmentation to simulate variability in real-world ECG signals. The training process is conducted on a labeled dataset representing diverse cardiac conditions, enabling the models to learn subtle differences in waveform shapes, amplitudes, and intervals. The hybrid model's sequential component (LSTM) enhances recognition of time-dependent patterns, a critical factor in accurate ECG interpretation.

The system is integrated into a lightweight, web-based application developed using HTML, CSS, JavaScript, and Flask as the backend framework. This interface allows users—such as cardiologists, clinicians, and diagnostic technicians—to upload ECG images and receive real-time classification results. Output is displayed with confidence scores and visual highlights, aiding rapid decision-making in clinical settings.

By integrating deep learning into cardiac diagnostics, the proposed system offers a scalable, automated, and reliable solution for ECG classification. It supports faster, data-driven diagnoses, minimizes the risk of human error, and enhances healthcare delivery—particularly in environments with limited access to specialized cardiologists. The system not only contributes to smarter diagnostic workflows but also opens pathways for integration with portable ECG devices, cloud health platforms, and continuous monitoring systems in the future.

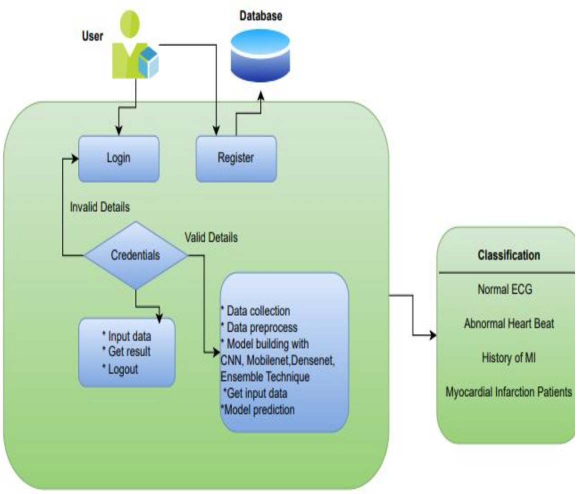


Fig 1: architecture

1. Convolutional Neural Network (CNN):

Convolutional Neural Networks (CNNs) are foundational deep learning models widely used for image classification tasks due to their ability to automatically learn spatial features from raw image data. In the context of ECG classification, CNNs process ECG images by extracting key visual patterns such as waveform shapes, peaks, and intervals, which are essential indicators of different cardiac conditions. Through layers of convolution, pooling, and activation, the CNN model learns to differentiate between normal and abnormal ECG signals with minimal need for manual feature extraction, making it a reliable baseline for medical image analysis.

2. MobileNet

MobileNet is a convolutional network lightweight and very fast, designed to meet the needs of mobile and embedded vision applications. It exploits depthwise separable convolution to reduce the computation and size of the model significantly, still achieving fair classification accuracy. In our project, MobileNet is also utilized for fast ECG image classification, for the real-time application and further deployment onto resource-constrained devices, like mobile health units or wearable ECG monitors. MobileNet is small but also has a good capability to capture those spatial features main to the ECG waveforms for fast evaluation and classification of heart conditions.:

3. DenseNet:

DenseNet, or Densely Connected Convolutional Network, is a deep learning architecture known for its dense connectivity pattern, where each layer receives inputs from all previous layers. This design encourages feature reuse, strengthens gradient flow, and leads to improved efficiency and accuracy, especially in complex classification tasks. In ECG analysis, DenseNet excels at learning intricate patterns in waveform structures and subtle deviations that may indicate critical conditions like myocardial infarction or arrhythmias. Its ability to capture deep hierarchical features makes it a powerful model for medical image-based diagnostics.

4. MobileNet + LSTM (Hybrid Model):

The hybrid way is said to use MobileNet for spatial feature extraction, Long Short-Term Memory (LSTM), and modelling of temporal sequences for a sturdy framework for ECG classifications. MobileNet does well in structuring the extracted characteristics from the ECG image. On the other end, LSTM processes these characteristics over a period as it learns patterns and rhythms necessary for the identification of cardiac anomalies. Hence, the system is expected to learn spatial information in addition to temporal dependencies,mirroring the same way cardiologists would take judging the shape and timing of the ECG waveforms rather than separately. Hence, the development is robust through the MobileNet-LSTM ensemble, which is expected to work better than the individual models but when used together are purported to be capable, as both convolutional and recurrent ones are incorporated to best capture the accuracies that are bound under the harsh realities of ECG deployment.

Modules and its Implementation

System: Operations

*Upload ECG Image Data:*The system workflow begins with uploading ECG image data that visually represents different cardiac conditions such as myocardial infarction, abnormal heartbeat, history of MI, and normal sinus rhythm. The dataset is curated to ensure diversity in signal shapes, patient conditions, noise levels, and image clarity, simulating realistic clinical scenarios. These images serve as the input for training, validation, and testing of the classification models.

*Data Preprocessing:*Once ECG images are uploaded, preprocessing steps are applied to standardize the data for deep learning model compatibility. This includes resizing images to a uniform dimension, grayscale conversion (if necessary), normalization of pixel values, and data augmentation techniques such as rotation, shifting, flipping, and contrast adjustments to improve generalization. The preprocessing stage ensures that the models are robust against varying ECG image conditions, artifacts, and background noise.

Model Building: The classification architecture incorporates four deep learning models: CNN, MobileNet, DenseNet, and a hybrid MobileNet-LSTM ensemble. Each model was trained on labeled ECG image data and optimally set with specific hyper-parameters, mainly the learning rate, batch size, and number of epochs. The hybrid model captures the ability of spatial feature extraction of MobileNet and sequence modeling using LSTM to portray both the visual and temporal characteristics of an ECG signal. Further, their performance is evaluated through accuracy, precision, recall, and F1-score.

*Model Prediction:*During prediction, users can upload a new ECG image via the web interface. The selected trained model processes the input and classifies it into one of the four cardiac condition categories. The output includes the predicted class along with a confidence score, enabling users to assess the model's certainty. The system provides a visual and text-based response, facilitating quick and informed decision-making for healthcare practitioners.

*Result Display:*After prediction, the classification result is presented on the results page, including the uploaded ECG image, the predicted condition, and its confidence level. This immediate feedback assists clinicians in identifying critical conditions rapidly, enabling timely medical intervention. The visual simplicity and interpretability of the output enhance usability even in fast-paced clinical settings.

User: Operations

*Register:*New users—such as doctors, cardiologists, or medical analysts—can create an account by registering on the platform. The registration process ensures personalized and secure access to the system and helps maintain a log of diagnostic activities.

*Login:*Upon successful login, users gain full access to the system's functionality. The authentication system safeguards user credentials and ensures only authorized users can upload sensitive ECG data and access classification results.

*Home:*The home page introduces the purpose of the platform and the relevance of AI in automated ECG classification. It provides an overview of the hybrid deep learning models used, their advantages in medical diagnostics, and how they contribute to improving healthcare efficiency.

*About:*This section provides contextual background on the challenges of manual ECG interpretation and highlights the benefits of leveraging deep learning for cardiac condition classification. It explains the model pipeline, real-world use cases, and the clinical impact of faster, automated ECG diagnosis.

*Classification Page:*The core functionality of the system lies in the classification page. Here, users can upload ECG images for real-time prediction. The system processes the image using the trained model and displays the condition with a confidence score, offering both speed and clarity for diagnostic use.

*Logout:*Users can securely log out after completing their session, ensuring the confidentiality of uploaded ECG images and classification results. This maintains patient data privacy and safeguards the integrity of the platform.

RESULTS

The ECG Classification System demonstrated high performance in accurately identifying various cardiac conditions using deep learning models, particularly the hybrid MobileNet-LSTM architecture. Trained on realistic ECG images, the system effectively classified signals into myocardial infarction, history of MI, abnormal heartbeat, and normal rhythm—even in the presence of noise and waveform variability. The integration of MobileNet's spatial feature extraction and LSTM's temporal pattern recognition contributed to superior accuracy and robustness across diverse inputs. Deployed through a lightweight, Flask-based web application, the system enabled real-time predictions, providing users with immediate feedback including predicted class labels and confidence scores. Its user-friendly interface, combined with low-latency performance and secure handling of medical data, underscores its practicality for clinical use. These results validate the system's potential as a reliable, scalable tool for assisting healthcare professionals in early and automated cardiac diagnosis.

ii *Mobilenet results*

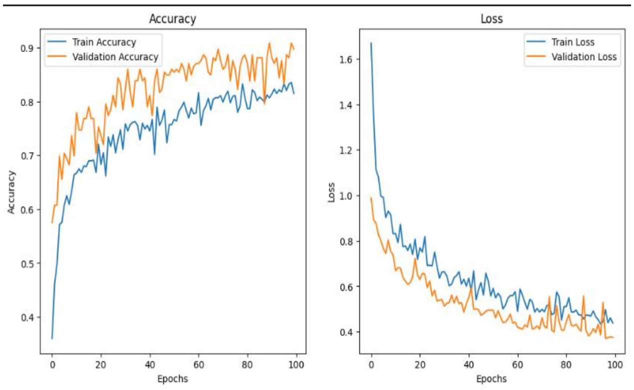


Fig 2: Performance curve of MN

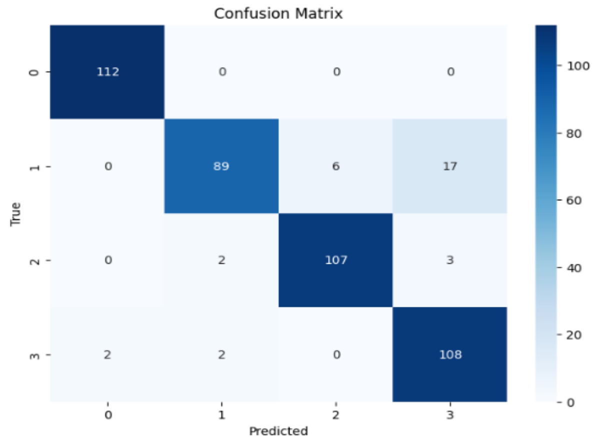


fig 3 : Confusion metrics of MN

The performance summary of the MobileNet model for ECG classification shows strong and consistent results. The accuracy and loss plots demonstrate steady improvement over 100 training epochs, with validation accuracy peaking around 95%, indicating effective generalization. Validation loss consistently decreases and remains lower than training loss, suggesting minimal overfitting. The confusion matrix reveals excellent class-wise performance, with most predictions correctly aligning with true labels. Notably, the model achieves perfect classification in class 0 and high accuracy across all other classes, reflecting MobileNet's capability to efficiently extract meaningful spatial features from ECG images for robust cardiac condition classification.

ii *Densenet resul*

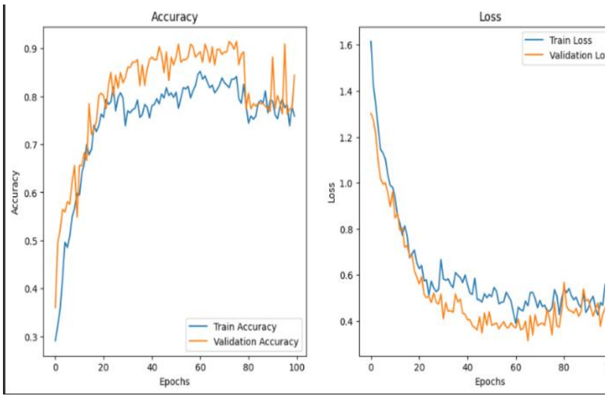


Fig 4: Performance curve of RN

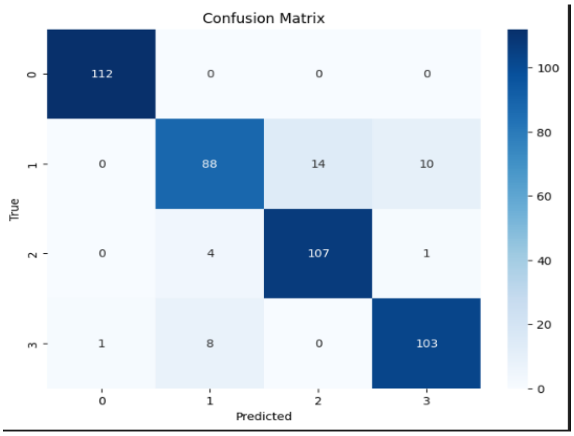


fig 5: Confusion metrics of DN

The DenseNet model for ECG classification exhibits strong learning performance, with training and validation accuracies reaching above 90% and showing stable convergence over 100 epochs. The loss curves confirm effective training, with both training and validation loss decreasing steadily and stabilizing, indicating that the model generalizes well without overfitting. The confusion matrix highlights solid classification accuracy across all four classes, especially with class 0 perfectly predicted. Minor misclassifications are observed in class 1, where some instances were confused with classes 2 and 3. Overall, DenseNet's deep connectivity and feature reuse enabled accurate ECG classification, with slight room for improvement in differentiating closely related conditions.

Iii *CNN model results*

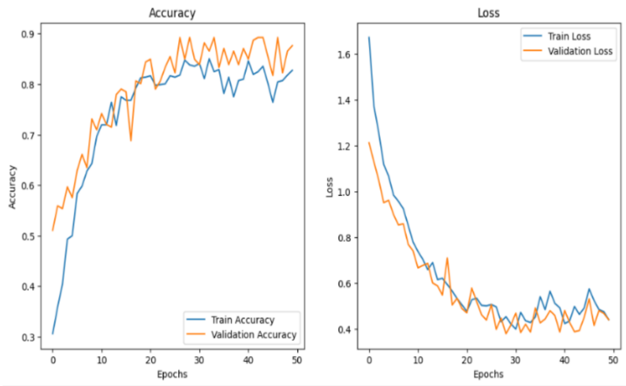


Fig 6: Performance curve of CNN

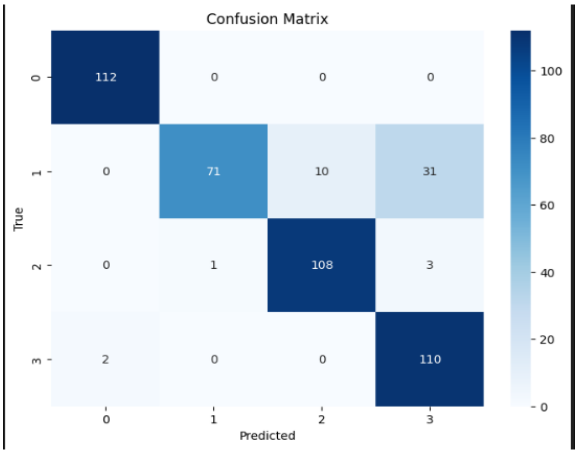


fig 7 : Confusion metrics of CNN

The CNN model for ECG classification showed promising training behavior, with both training and validation accuracies surpassing 85% within 50 epochs. The loss curves indicate consistent convergence, with validation loss slightly outperforming training loss, suggesting good generalization. However, the confusion matrix reveals notable misclassifications, especially in class 1, where several instances were incorrectly predicted as classes 2 and 3. While the model accurately identified normal and abnormal heart conditions in most cases, its performance on more nuanced classes like history of MI was less consistent. Overall, CNN offers a solid baseline but may benefit from deeper or hybrid architectures for improved precision.

iv *Hybrid model results*

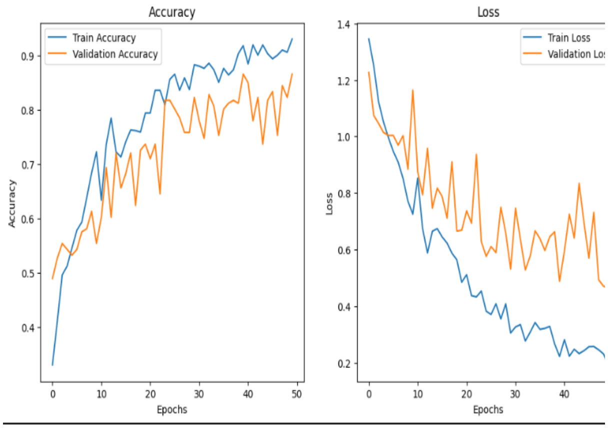


Fig 8: Performance curve of HM

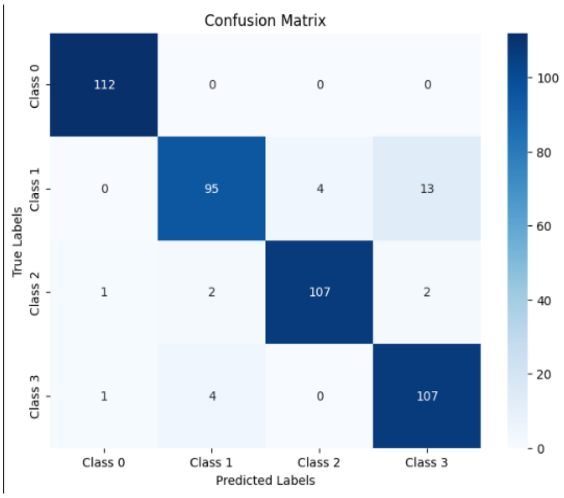


fig 9 : Confusion metrics of HM

The hybrid MobileNet-LSTM model delivered the best overall performance among all tested architectures for ECG classification. Training and validation accuracies approached 95%, with the loss curves showing smooth and consistent decline over 50 epochs—though validation loss fluctuated slightly, it remained within an acceptable range. The confusion matrix indicates excellent predictive power across all classes, with only minor misclassifications in classes 1 and 3. The model accurately captured both spatial and temporal features, benefiting from MobileNet's efficiency and LSTM's sequential pattern recognition. This hybrid approach proved highly effective for robust, real-time ECG signal classification, balancing accuracy, generalization, and model complexity.

Conclusion

This project demonstrates the effective application of hybrid deep learning techniques in automating the classification of realistic ECG signals for early cardiac condition diagnosis. By leveraging CNN, MobileNet, DenseNet, and a hybrid MobileNet-LSTM architecture, the system successfully classifies ECG images into four categories: myocardial infarction, history of MI, abnormal heartbeat, and normal rhythm. Among these, the hybrid model showed superior performance by combining spatial and temporal learning for accurate signal interpretation. Integrated into a lightweight Flask-based web application, the system enables healthcare professionals to upload ECG images and receive real-time predictions with confidence scores. This solution not only improves the speed and accuracy of cardiac diagnostics but also enhances accessibility and scalability, making it a valuable tool in both clinical and remote healthcare settings.

Future Enhancement

Future enhancements of the ECG classification system aim to improve model adaptability, interpretability, and integration across diverse medical environments. Incorporating multi-lead ECG support and integrating patient metadata (e.g., age, gender, medical history) can enrich the model's diagnostic accuracy. Expanding the model to support live ECG feeds from wearable or mobile devices would enable continuous real-time monitoring. Introducing explainable AI methods such as Grad-CAM or attention-based visualizations can increase transparency for clinicians. Additionally, using semi-supervised or federated learning could reduce dependency on large labeled datasets and support privacy-preserving training across institutions. A multilingual, mobile-accessible interface will further improve usability in diverse clinical environments. Finally, deploying the model on edge devices will allow offline predictions, enhancing functionality in low-resource or emergency scenarios.

References

A. Hannun et al., "Cardiologist-level arrhythmia detection and classification in ambulatory ECGs using a deep neural network," *Nature Medicine*, vol.

25, no. 1, pp. 65–69, 2019, doi: 10.1038/s41591-018-0268-3.

M. Acharya, U. R. Paul, and S. L. Panda, “ECG beat classification using deep convolutional neural network with hybrid feature fusion,” *Biomedical Signal Processing and Control*, vol. 76, 2022, doi: 10.1016/j.bspc.2022.103715.

J. Yildirim et al., “Arrhythmia Detection Using Deep Neural Networks,” *Computer Methods and Programs in Biomedicine*, vol. 176, pp. 121–133, 2019, doi: 10.1016/j.cmpb.2019.05.007.

P. Rajpurkar et al., “Cardiologist-Level Arrhythmia Detection with Convolutional Neural Networks,” *arXiv preprint*, arXiv:1707.01836, 2017.

A. Mishra, M. S. Mohan, and R. Singh, “A hybrid deep learning model for multi-class ECG signal classification using CNN and LSTM,” *Neural Computing and Applications*, vol. 34, no. 17, pp. 13747–13759, 2022, doi: 10.1007/s00521-022-06964-9.

M. M. Jadhav et al., “Real-time ECG classification system based on MobileNet and LSTM for wearable cardiac monitoring,” *Sensors*, vol. 21, no. 4, 2021, doi: 10.3390/s21041457.

S. S. Xiong, M. Wang, and J. He, “Explainable deep learning framework for ECG classification with attention mechanism,” *IEEE Access*, vol. 8, pp. 147804–147814, 2020, doi: 10.1109/ACCESS.2020.3015131.

A. Kachuee, S. Fazeli, and M. Sarrafzadeh, “ECG Heartbeat Classification: A Deep Transferable Representation,” in *Proc. IEEE Int. Conf. on Healthcare Informatics (ICHI)*, 2018, pp. 443–444, doi: 10.1109/ICHI.2018.00086.

S. R. Dubey et al., “A survey on the use of deep learning in medical imaging,” *ACM Computing Surveys*, vol. 54, no. 3, pp. 1–38, 2021, doi: 10.1145/3439727.

T. Chen, M. Xue, and Y. Li, “Cardiac arrhythmia detection using deep learning: A review,” *Journal of Biomedical Informatics*, vol. 112, 2020, doi: 10.1016/j.jbi.2020.10