FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)

DEPARTMENT OF COMPUTER APPLICATIONS

MAIN PROJECT

SCRUM BOOK

Name of the Student: ANJALYKRISHNA A S **Roll No: 22** **Batch: S4 A**

Email ID: anjalykrishna522002@gmail.com

Name of the Guide: Ms.Joice T

Name of the Scrum Master: Dr.Rakhi Venugopal

Project Title: CNN-LSTM Based Model for ECG Arrhythmias and Myocardial Infarction Classification

GitHub ID: https://github.com/anjalykrishna522002/Arrthymia

Sprint Release 2:

Date: 12-03-25

Description of Work:
In the second Scrum release, I focused on implementing classification using the hybrid CNN-LSTM model for ECG signal analysis. Building upon the foundation established in the first release, I fine-tuned the model for real-time ECG classification, ensuring optimal performance in identifying Fibrillation, Myocardial Infarction, Normal, Block, Supraventricular, Ventricular with high accuracy. enhance usability, I developed a web-based interface that allows users to upload ECG recordings and view classification results instantly. This interface provides a seamless user experience, enabling healthcare professionals to access model predictions conveniently. roughout this sprint, I focused on optimizing model deployment, ensuring smooth integration between the backend (model processing) and frontend (user interface). Furthermore, I conducted rigorous testing and performance evaluation to validate the effectiveness of the web-based system.
Remarks:

Name and Signature of the Guide **Name and Signature of the Scrum Master**

Date: 12-03-25

PRODUCT BACKLOG

1. Data Preprocessing & Management

- Collect ECG dataset from Kaggle.
- Preprocess raw ECG images (convert to grayscale, resize to 48×48).
- Normalize image pixel values (scale from 0 to 1).
- Implement dataset splitting (50% training, 50% testing).
- Convert labels to categorical format (one-hot encoding).

2. Model Development (CNN-LSTM Architecture)

- Build CNN layers for spatial feature extraction.
- Add pooling layers (MaxPooling & AveragePooling) to reduce dimensionality.
- Integrate LSTM layer to capture temporal dependencies in ECG signals.
- Add fully connected layers for classification.
- Use Dropout to prevent overfitting.
- Implement Softmax activation for multi-class classification (6 categories)

3. Model Training & Optimization

- Implement ImageDataGenerator for data augmentation.
- Optimize model using Adam optimizer & Categorical Crossentropy.
- Train model for 100 epochs with a batch size of 45.
- Introduce early stopping to avoid unnecessary training.
- Save the trained model as "cnnlstmmodel1.h5".

4. Model Evaluation & Testing

- Load the trained model for inference.
- Perform prediction on test data.
- Compute performance metrics (Accuracy, Precision, Recall, F1-Score, AUC-ROC).
- Generate Confusion Matrix to analyze misclassified cases.

5. Deployment & Real-Time Prediction

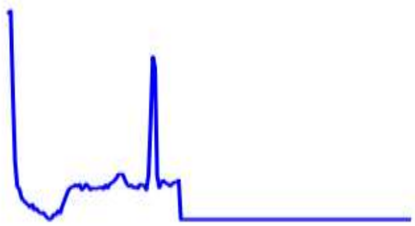
- Develop a Flask-based interface for real-time ECG classification.
- Load and preprocess user-uploaded ECG images.
- Predict class probabilities using the trained model.
- Return classification results to the user.

Name and Signature of the Guide **Name and Signature of the Scrum Master**

Date: 12-03-25

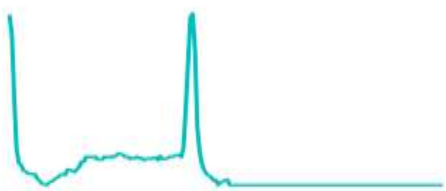
DATABASE & UI DESIGN

ECG image dataset (Kaggle)



Fibrillation

Myocardial Infarction



Normal

Block

Supraventricular

ventricular

Name and Signature of the Guide: **Name and Signature of the Scrum Mast**

Date: 12-03-25

DATABASE & UI DESIGN

User Registration

Column Name	Data Type	Description
id	INT (Primary Key)	Unique user ID (auto-increment).
first_name	VARCHAR(50)	User's first name.
last_name	VARCHAR(50)	User's last name.
house_name	VARCHAR(100)	House name/address of the user.
place	VARCHAR(100)	Location of the user.
phone	VARCHAR(10)	User's phone number (10 digits).
email	VARCHAR(100)	Unique email for authentication.
username	VARCHAR(100)	Unique username (usually the email).
password	VARCHAR(255)	Hashed password for security.
created_at	TIMESTAMP	Stores the registration date and time.

Doctor Registration

Column Name	Data Type	Description
id	INT (Primary Key)	Unique doctor ID (auto-increment).
first_name	VARCHAR(50)	Doctor's first name.
last_name	VARCHAR(50)	Doctor's last name.
house_name	VARCHAR(100)	House name/address of the doctor.
place	VARCHAR(100)	Location of the doctor.
landmark	VARCHAR(100)	Landmark near the doctor's location.
department	VARCHAR(100)	Doctor's department/qualification.
phone	VARCHAR(10)	Doctor's phone number (10 digits, unique).
email	VARCHAR(100)	Unique email for authentication.
profile_image	VARCHAR(255)	Path to uploaded profile image.
username	VARCHAR(100)	Unique username (usually the email).
password	VARCHAR(255)	Hashed password for security.
created_at	TIMESTAMP	Stores the registration date and time.

Upload ecg image

Column Name	Data Type	Description
id	INT (Primary Key)	Unique upload ID (auto-increment).
doctor_id	INT (Foreign Key)	Links uploaded file to the respective doctor.
file_name	VARCHAR(255)	Original name of the uploaded ECG file.
file_path	VARCHAR(255)	Path to the stored file on the server.
uploaded_at	TIMESTAMP	Stores the upload timestamp.

view result

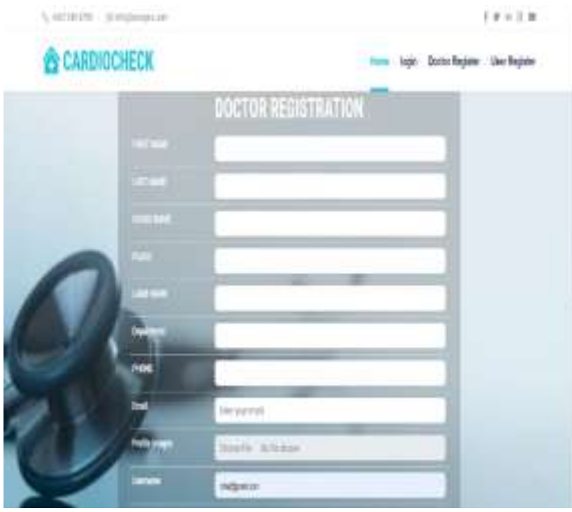
Column Name	Data Type	Description
id	INT (Primary Key)	Unique result ID (auto-increment).
doctor_id	INT (Foreign Key)	Links result to the respective doctor.
ecg_img	VARCHAR(255)	Path to the uploaded ECG image.
plot_image_path	VARCHAR(255)	Path to the generated **graph plot**.
classification_result	VARCHAR(50)	Model's classification result (**Normal, Myocardial Infarction, Fibrillation, etc.**).
ecg_details	TEXT	Additional details (confidence score, model output, etc.).
date	TIMESTAMP	Stores the date of prediction.

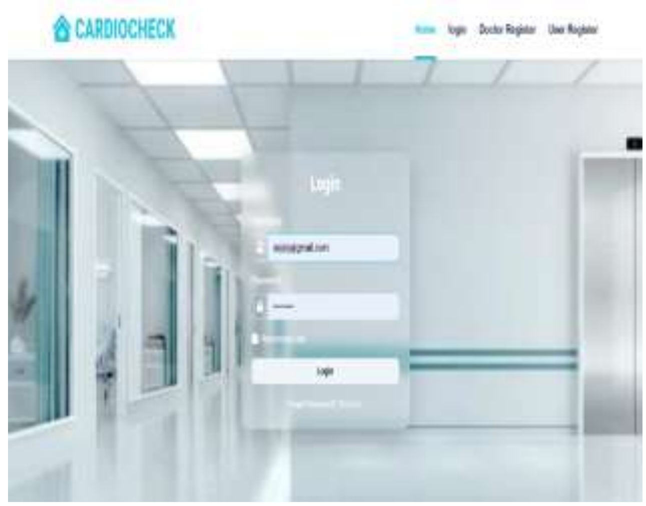
Name and Signature of the Guide:

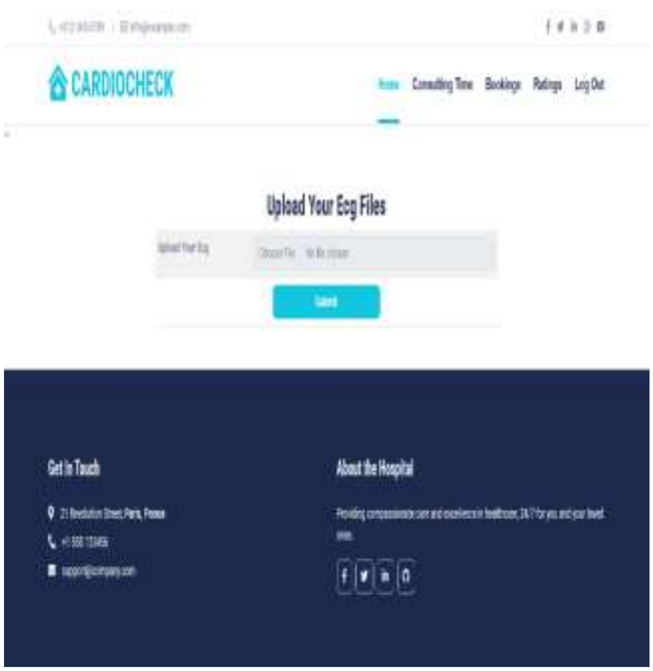
Name and Signature of the Scrum Master

Date: 12-03-25

UI DESIGN







Name and Signature of the Guide: **Name and Signature of the Scrum Maste**

Date: 12-03-25

TESTING AND VALIDATION

1. Unit Testing

Objective: Ensure individual components function correctly in isolation.

Components to Test

- Data Preprocessing:
    - Validate proper handling of missing ECG data and signal normalization.
    - Ensure segmentation of ECG signals into uniform lengths for consistent input.
- Model Functions (CNN-LSTM):
    - Test CNN's ability to extract spatial ECG features correctly.
    - Verify LSTM's capability to capture temporal dependencies.
    - Ensure classification outputs (Fibrillation, Myocardial Infarction, Normal, Block, Supraventricular, and Ventricular) align with expected labels.
- Database Functions
    - Confirm ECG records are correctly stored, retrieved, and deleted.
    - Validate query performance for large datasets.
- Classification Output:
    - Ensure model predictions match ground truth test cases with high precision.
    - Validate AUC-ROC, Precision, Recall, and F1-score results.
- Flask API & UI Components:
    - Test API endpoints for model inference and real-time predictions.
    - Verify UI elements (buttons, input fields, graphs) for usability and interactivity.

2. Integration Testing

Objective: Ensure different modules work together seamlessly.

Testing Scope

- Data Flow Validation:
    - Test the complete flow from raw ECG data input → preprocessing → model inference → prediction output in UI.
- Model Performance on Different ECG Types:
    - Validate classification consistency across Fibrillation, Myocardial Infarction, Normal, Block, Supraventricular, and Ventricular cases.
- Real-Time Classification:
    - Ensure real-time ECG inputs flow correctly to the backend model.
    - Validate the classification updates dynamically in the UI.
- Database & UI Integration (if applicable):
    - Ensure that classified ECG records are correctly retrieved and displayed.
    - Validate synchronization between the backend and frontend.

Name and Signature of the Guide: **Name and Signature of the Scrum Master**

Date: 12-03-25

TESTING AND VALIDATION

3. System Testing

Objective: Evaluate the overall system functionality, performance, and robustness.

Testing Scope

- Overall Functionality:
    - Ensure users can upload ECG signals, receive classifications, and view history without issues.
- Load Testing:
    - Simulate high ECG signal processing and measure response times.
- Edge Case Testing:
    - Input ECG signals with missing waveforms or noise and test model robustness.
    - Provide borderline cases (e.g., minor arrhythmic patterns) and check classification accuracy.
- Compatibility Testing:
    - Run tests on different devices (desktop, tablet, mobile) and browsers (Chrome, Edge, Firefox).

4. User Acceptance Testing (UAT)

Objective: Collect feedback on model accuracy, UI usability, and overall user experience.

Testing Scope

- Real-World ECG Data Testing:
    - Validate accuracy with cardiologist-labeled ECG signals.
- UI Feedback:
    - Gather user feedback on result interpretation, ease of navigation, and response times.
- Overall Satisfaction:
    - Ensure medical professionals find the tool useful and accurate for assisting diagnoses.

Name and Signature of the Guide: **Name and Signature of the Scrum Master**

Date: 12-03-25

TESTING AND VALIDATION

5. Validation

Objective: Ensure the system meets functional, performance, security, and usability standards.

Functional Validation

- Classification Accuracy:
    - Validate Precision, Recall, and F1-score for Myocardial Infarction and Arrhythmias.

Performance Validation

- Response Time:
    - Ensure classification predictions are generated within milliseconds for real-time medical use.
- Concurrent Requests:
    - Test system performance under multiple simultaneous ECG classification requests.

Security Validation

- Input Sanitization:
    - Prevent unauthorized modifications and ensure secure model interaction.
- Data Privacy:
    - Ensure sensitive patient data remains protected and is not stored unnecessarily.

Usability Validation

- Cross-Device Responsiveness:
    - Verify the UI is responsive across different devices (laptops, tablets, mobile phones).
- User Experience Testing:
    - Ensure the Flask-based UI is intuitive and users can easily interpret ECG classification results.

Name and Signature of the Guide: **Name and Signature of the Scrum Master**

Date: 12-03-25

DETAILS OF VERSIONS

V2.0 – [07-02-25]
Data Preprocessing & Augmentation

- Preprocessed images:
    - Converted them to grayscale for uniformity.
    - Resized all images to 48×48 for CNN input compatibility.
    - Normalized pixel values (scaling between 0 and 1).
- Implemented data augmentation (rotation, zoom, flipping) to enhance model generalization.
- Split data into training (80%) and testing (20%) sets.

V3.0 – [14-02-25]
Model Development & Training

- Implemented CNN, LSTM, and CNN-LSTM hybrid models for ECG image classification.
- Defined convolutional layers to extract spatial features from ECG images.
- Used LSTM layers in the hybrid model to capture sequential dependencies.
- Trained models using categorical cross-entropy loss and Adam optimizer.

V4.0 – [21-02-25]
Model Evaluation & Performance Analysis

- Evaluated models using accuracy, precision, recall, F1-score, and AUC-ROC.
- Compared CNN, LSTM, and CNN-LSTM models to find the best one.
- Visualized prediction probabilities for better understanding.

V5.0 – [28-02-25]
Web Interface Enhancement & Finalization

- Improved the web interface for better user experience.
- Optimized the ECG image upload and prediction process.
- Ensured real-time result display after image upload.
- Deployed the web application for user access.

Name and Signature of the Guide **Name and Signature of the Scrum Master**