CHAPTER 1

INTRODUCTION

1.1 Heart Disease

The Heart is the most important organ of human body. If it does not function properly then it affects other organ of the body. According to a report 7,000,000 die from heart attacks each year. According to WHO report around 17.9 million people die due to CVDS in 2016. 31% of the death of people is due to Heart disease around the globe in every year. The pumping of blood to the human body is the vital function of heart which supply oxygen and nutrients to the human body and also remove other metabolic waste from the body. If there is deficiency of blood in human body then heart doesn't function properly and it stop working which causes the death of human being. Angina occurs when there is temporary loss of blood to the heart causing chest pain.

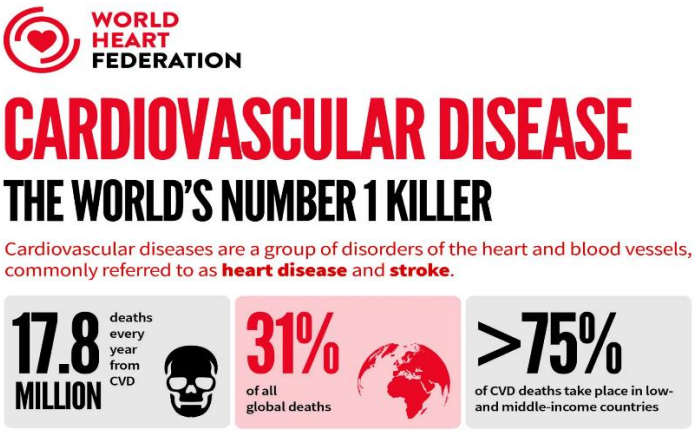

Fig 1.1 Deaths from Heart disease

Risk factor that cannot control heart disease:

1. family history
2. 55 years or older
3. History of preeclampsia

Symptoms of Heart attack

(a) Dizziness

(b) Jaw pain

(c) Abdominal pain

(d) Nausea

Living a healthy lifestyle can reduce the effect of heart disease. Drinking plenty of water, eating green vegetables, fat free food, doing exercises, regular check-up of heart, consulting with the

doctor if there any family history of heart disease can reduce the effect of heart disease.

1.2 Background and Significance

Heart disease is one of the leading causes of mortality worldwide, affecting millions of individuals every year. Early detection and accurate diagnosis are crucial in preventing severe outcomes, such as heart attacks and strokes.

Traditionally, diagnosing heart disease requires extensive clinical tests and the expertise of medical professionals, which can sometimes result in delayed interventions. With the increasing availability of patient health data and advancements in computing, machine learning (ML) has emerged as a powerful tool to assist in medical diagnostics. By analyzing patterns and trends in patient data, ML algorithms can provide accurate predictions and augment the decision-making process of healthcare providers. This project aims to leverage these advancements by developing a heart disease prediction system that combines machine learning techniques with a user-friendly web application, making early detection accessible and efficient.

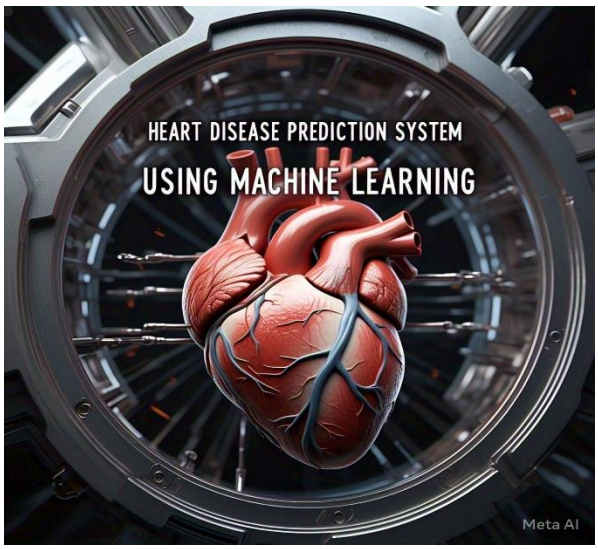

Fig 1.2 Heart Disease Prediction System

1.3 Problem Statement

Despite significant advancements in healthcare, many individuals suffer from delayed diagnosis of heart disease due to limited access to expert consultation and diagnostic tools. This issue is compounded in remote or underprivileged areas where medical resources are scarce. Additionally, the complexity of interpreting medical data increases the likelihood of human error in diagnosis. There is a pressing need for an automated system that can analyze patient data, predict the likelihood of heart disease, and provide results in real time. Such a system

would empower both patients and healthcare providers, enabling faster and more accurate decision-making.

1.4 Role of Machine Learning and Artificial Intelligence

Machine learning and artificial intelligence (AI) play a pivotal role in transforming healthcare by enabling data-driven insights and automation. In this project, a Random Forest Classifier is employed to predict heart disease based on patient data. Random Forest, an ensemble learning algorithm, combines the outputs of multiple decision trees to improve classification accuracy and reduce overfitting. AI enhances the system's capability to process large datasets, identify hidden patterns, and provide reliable predictions. This integration of ML and AI not only reduces the workload on medical professionals but also ensures consistency and precision in diagnosis. Furthermore, the use of AI-driven web applications ensures accessibility, making diagnostic tools available to a broader audience.

1.5 Project Objectives

This project focuses on designing and implementing a heart disease prediction system leveraging advanced ML and deep learning algorithms. The primary objectives are as follows:

- To develop a machine learning model capable of predicting heart disease with high accuracy using clinical patient data.
- To design and implement a web-based application using Flask that allows users to input patient details and receive real-time predictions.
- To integrate a secure database system for efficient data storage and retrieval.
- To incorporate additional functionalities such as email notifications to facilitate communication and result sharing.
- To demonstrate the practical application of ML and AI in healthcare, emphasizing their potential to improve diagnostic accuracy and accessibility.
- To explore opportunities for further development, such as enhancing the model with larger datasets, integrating advanced ML techniques, and enabling real-time monitoring for comprehensive patient care.
- This project underscores the transformative potential of AI-driven systems in addressing real-world healthcare challenges and aims to create a scalable solution for heart disease prediction.

CHAPTER 2

LITERATURE SURVEY

2.1 Previous Work

Chaimaa Boukhatem., et al (2022) presents several machine learning approaches for predicting heart diseases, using data of major health factors from patients. The paper demonstrated four classification methods: Multilayer Perceptron (MLP), Support Vector Machine (SVM), Random Forest (RF), and Naïve Bayes (NB), to build the prediction models. Data preprocessing and feature selection steps were done before building the models. The models were evaluated based on the accuracy, precision, recall, and F1-score. The SVM model performed best with 91.67% accuracy.

Anu Lohachab., et al (2023) proposes a study that delves into a range of machine learning algorithms, encompassing supervised, and ensemble algorithms. Furthermore, recognizing the literature's limitations, the focus has been on enhancing model performance through hyperparameter tuning, implementing robust feature selection methods, and conducting thorough model evaluations. Besides, for feature selection, they utilized chi-squared tests and correlation analysis to ensure the relevance and significance of the features. Moreover, the comprehensive evaluation, spanning three diverse datasets, assesses both supervised and ensemble learning algorithms for their accuracy and generalizability. The results indicate that the K-Nearest Neighbors-based model excels, achieving 97.82% accuracy. By enhancing predictive accuracy and model robustness, this study not only contributes to improved patient-specific interventions but also aids in shaping more effective and efficient public health strategies in cardiovascular care.

Archana Singh., et al (2020) calculated the accuracy of machine learning algorithms for predicting heart disease, for this algorithms are k-nearest neighbor, decision tree, linear regression and support vector machine(SVM) by using UCI repository dataset for training and testing. For implementation of Python programming Anaconda(jupytor) notebook is best tool, which have many type of library, header file, that make the work more accurate and precise.

Devansh Shah., et al (2020) presents various attributes related to heart disease, and the model on basis of supervised learning algorithms as Naïve Bayes, Decision tree, K-nearest neighbor, and Random Forest algorithm. It uses the existing dataset from the Cleveland database of UCI repository of heart disease patients. The dataset comprises 303 instances and

76 attributes. Of these 76 attributes, only 14 attributes are considered for testing, important to substantiate the performance of different algorithms. This research paper aims to envision the probability of developing heart disease in the patients. The results portray that the highest accuracy score is achieved with K-nearest neighbor.

Rahul Kutarya., et al (2021) says that data received by the medical sector or hospitals is so huge that sometimes it becomes difficult to analyze. Using machine learning techniques for this prediction and handling of data can become very efficient for medical people. Hence in this study has discussed the heart disease and its risk factors and explained machine learning techniques. Using that machine learning techniques, they have predicted heart disease and provided a comparative analysis of the algorithms for machine learning used for the experiment of the prediction. The goal or objective of this research is completely related to the prediction of heart disease via a machine learning technique and analysis of them.

Vijetha Varma., et al (2020) ha**s** used a benchmark dataset of UCI Heart disease prediction for this research work, which consist of 14 different parameters related to Heart Disease. Machine Learning algorithms such as Random Forest, Support Vector Machine (SVM), Naive Bayes and Decision tree have been used for the development of model. In this research they have found the correlations between the different attributes available in the dataset with the help of standard Machine Learning methods and then using them efficiently in the prediction of chances of Heart disease. Result shows that compared to other ML techniques, Random Forest gives more accuracy in less time for the prediction. This model can be helpful to the medical practitioners at their clinic as decision support system.

Wan Adlina Husna Wan Azizan., et al (2021) use machine learning algorithms to select attributes obtained from the Cleveland dataset. Prediction is made using two machine learning models, Artificial Neural Network (ANN) and Logistic Regression. Different sizes of hidden layers and activation functions are used to find the hyperparameters with optimal performance. The number of inputs and outputs are kept constant at one with a maximum iteration of 500. Logistic Regression is used to classify a discrete data set and return the probability value where the Sigmoid function acts as the cost function. Finally, a confusion matrix was used to compare the performance of both models. ANN resulted in higher accuracy of 92.31% and an F1-score of 93.2% compared to Logistic Regression with 90.11% accuracy and an F1-score of 91.26%**.**

Jyothi Kiran., et al (2023) tells that to predict and categorize patients with heart disease, they used different machine learning methods such as decision tree classifier, random forest, Naive Bayes, K-nearest neighbor, logistic regression, and support vector machine. The given model is helpful in relieving a lot of strain from determining the probability of the classifier correctly and accurately identifying heart disease. It increases medical care while lowering costs.

Maria Hassan., et al (2024) explores machine-learning techniques and clinical assessments to evaluate their performance in detecting heart-related disorders. This paper also investigates the limitations and challenges associated with different detection approaches. Five state-of-the-art machine learning models such as Logistic Regression (LR), Random Forest (RF), Naive Bayes (NB), Decision Tree (DT), Support Vector Machine (SVM), and three hybrid model hybrid models (LR, RF), Hybrid model (DT, NB) and Hybrid Model (KNN, RF) explained in this chapter. The dataset used in this chapter was taken from the University of California Irvine (UCI) and consisted of 14 attributes, including Blood Pressure (BP), cholesterol (Chol), electrocardiogram (ECG), and a total number of 1026 data records. The chapter shows that the machine learning model LR achieves accuracy, precision, recall, and F1-score of 85%, 87%, 85%, and 86%, respectively; SVM achieves 85%, 87%, 85%, 86% respectively, and DT achieves 77%, 83%, 73%, and 78%, respectively; NB achieves 86%, 90%, 85%, and 87%, respectively, and RF achieves 86%, 88%, 88%, and 88%, respectively. Hybrid model K-Nearest Neighbor (KNN) with Random Forest achieve accuracy, precision, recall, F1-score 0.9272, 0.9151, 0.9417, 0.9282, Linear Regression with Random Forest 0.92718, 0.9583, 0.8932, 0.9246 and Decision Tree with Naïve Bayes 0.9126, 0.9293, 0.8932, 0.9109 respectively. The results of this study show that the hybrid models have superior results compared with the other models.

Yi Liu., et al (2020) introduce**s** a privacy-preserving machine learning technique named federated learning and propose a Federated Learning-based Gated Recurrent Unit neural network algorithm (FedGRU) for traffic flow prediction. FedGRU differs from current centralized learning methods and updates universal learning models through a secure parameter aggregation mechanism rather than directly sharing raw data among organizations. In the secure parameter aggregation mechanism, a Federated Averaging algorithm is adopted to reduce the communication overhead during the model parameter transmission process. Furthermore, a Joint Announcement Protocol is designed to improve the scalability of

FedGRU. An ensemble clustering-based scheme is proposed for traffic flow prediction by grouping the organizations into clusters before applying FedGRU algorithm. Through extensive case studies on a real-world dataset, it is shown that FedGRU's prediction accuracy is 90.96% higher than the advanced deep learning models, which confirm that FedGRU can achieve accurate and timely traffic prediction without compromising the privacy and security of raw data.

2.2 Survey Table

No.	Title	Algorithm(s) Used	Dataset	Key Features	Findings/Accuracy
1	**Heart Disease Prediction Using Machine Learning (2022)**	Multilayer Perceptron (MLP), Support Vector Machine (SVM), Random Forest (RF), Naïve Bayes (NB)	Cleveland Heart Disease Dataset	Age, cholesterol, blood pressure, chest pain type	SVM achieved 91.67% accuracy, Logistic Regression had 80% accuracy.
2	**A Comparative Study of ML Algorithms (2023)**	K-Nearest Neighbors (k-NN), Decision Tree, Random Forest, Naïve Bayes	Framingham Heart Study Dataset	Age, BMI, glucose levels, exercise	k-NN achieved 97.82% accuracy, the highest among all algorithms.
3	**Deep Learning for Heart Disease Prediction (2020)**	Artificial Neural Network (ANN), Logistic Regression	Statlog (Heart) Dataset	Heart rate, cholesterol, ECG, exercise-induced angina	ANN achieved 92.31% accuracy, outperforming Logistic Regression (90.11%).

4	**Hybrid Model for Cardiovascular Prediction (2020)**	Logistic Regression (LR), Random Forest (RF), Naïve Bayes (NB), Decision Tree (DT), Support Vector Machine (SVM)	UCI Repository (Cleveland)	Age, gender, diabetes, resting ECG	Hybrid models (e.g., KNN + RF) achieved superior results with 92.72% accuracy.
5	**Explainable AI in Heart Disease Diagnosis (2021)**	SHAP + Random Forest	MIMIC-III Clinical Dataset	Blood pressure, cholesterol, heart rate, smoking history	Model accuracy was 87%. SHAP enhanced interpretability for clinicians.
6	**IoT-Based Wearable Data Analysis for Heart Disease(2020)**	k-NN, Naïve Bayes	Wearable device data (custom)	Real-time heart rate, ECG, blood oxygen levels	k-NN achieved 80% accuracy; Naïve Bayes was faster but less accurate (75%).
7	**Heart Disease Prediction with PCA and SVM (2021)**	Principal Component Analysis + SVM	Cleveland Heart Disease Dataset	Reduced features: age, cholesterol, blood pressure, chest pain type	PCA improved SVM accuracy from 82% to 86%.
8	**Hybrid Deep Learning for Heart Disease Detection (2023)**	CNN + LSTM	Cleveland + Statlog Datasets	Sequential ECG data, exercise-induced chest pain	Achieved 90% highlighting effectiveness accuracy, of combining CNN and LSTM.

9	**A Lightweight Model for Wearable Heart Monitoring (2024)**	LightGBM, Logistic Regression	Real-time wearable data (custom)	Wearable ECG, BMI, sleep quality	LightGBM achieved 86% accuracy with faster computation for real-time prediction.
10	**Federated Learning for Privacy-Preserving Prediction (2020)**	Federated Neural Networks	Distributed hospital data	Demographics, lab test results, lifestyle factors	FedGRU achieved 90.96% accuracy, ensuring privacy-preserving predictions.

Table 2.1: Survey Table

2.3 Scope of the project

The scope of this project is to develop a reliable and accurate heart disease prediction system using machine learning. The system aims to analyze patient health data and provide real-time predictions to support early diagnosis and decision-making in healthcare. By integrating a user-friendly web application, it ensures accessibility and usability for a wide range of users. The project also lays a foundation for future enhancements in predictive healthcare technology.

1. Develop a heart disease prediction system using machine learning, particularly the Random Forest algorithm, to ensure accurate predictions.
2. Analyze clinical patient data to assess the likelihood of heart disease, providing reliable and data-driven results.
3. Design a web-based application using Flask for users to input health parameters and receive real-time predictions.
4. Enable accessibility to diagnostic tools, especially in remote or underserved areas, to support early detection.
5. Integrate features like email notifications for seamless communication of results.
6. Plan for future advancements, such as incorporating larger datasets, real-time health monitoring, and enhancing the model's performance with advanced AI techniques.

2.4 Challenges in Heart Disease Prediction

Despite its potential, heart disease prediction faces several challenges that must be addressed:

- **Data Quality and Diversity**: Datasets often lack diversity, leading to models that may not generalize well across different populations. Imbalanced datasets can result in biased predictions.
- **Overfitting Risks**: Over-reliance on complex models can lead to overfitting, where the algorithm performs well on training data but poorly on unseen data.
- **Data Privacy and Ethics**: Ensuring patient confidentiality while using sensitive medical data is a critical concern.
- **Model Interpretability**: Many advanced ML models, such as deep neural networks, function as "black boxes," making it difficult for clinicians to understand how predictions are made.

CHAPTER 3

SOFTWARE AND HARDWARE DESCRIPTION

3.1 Hardware Requirements

3.1.1. Development Machine (Local Environment):

This machine is used by developers to build, test, and debug the website before deployment.

1. **Processor (CPU)**
   - **Recommended**: Intel i5 or AMD Ryzen 5 (minimum)
   - **Optimal**: Intel i7, i9, or AMD Ryzen 7, 9 for more efficient multitasking and handling larger datasets.
     - Machine learning model training, particularly for large datasets, may benefit from a more powerful CPU.

2. **Memory (RAM)**
   - **Minimum Requirement**: **4 GB RAM**: Sufficient for small datasets and running the Flask application locally, along with a basic Random Forest model.
   - **Recommended Specification**: **8 GB RAM**: Ideal for smoother performance, especially when handling moderately sized datasets or multiple simultaneous user requests.
   - **For Large Datasets or Heavy Usage**: **16 GB RAM or more**: Needed if the model is trained on larger datasets or deployed on a production server with higher user traffic.

3. **Storage**
   - **Recommended**: 128GB SSD (Solid State Drive)
   - **Optimal**: 256GB SSD or more
     - SSD is preferred over HDD because it provides faster read/write speeds, improving performance when dealing with large files, such as datasets or machine learning model files.
     - The SSD should be large enough to store your development environment, databases, datasets, and model files.

4. **Graphics Processing Unit (GPU)**
   - **Optional**: NVIDIA GTX 1650 or higher (if you're training deep learning models locally)
     - For traditional machine learning models (e.g., Random Forest, Logistic Regression), a GPU isn't strictly necessary.
     - For deep learning tasks (e.g., neural networks, if you plan to scale the prediction model), using a machine with a dedicated GPU (like the NVIDIA GTX series) will significantly speed up training times.

3.1.2. Server (For Production Environment):

This is where your web application and model will run in real-time, serving predictions to users.

1. **Processor (CPU)**
   - **Recommended**: Intel Xeon or AMD EPYC processors
     - Servers often use high-performance processors, which are designed to handle multi-threaded tasks more efficiently, essential for handling multiple concurrent user requests.
     - For low-to-medium traffic applications, you may get by with a basic multi-core CPU, but for high-volume applications, opting for a multi-core server processor (e.g., 8 cores or more) would be ideal.
2. **Memory (RAM)**
   - **Basic Production Setup (Low Traffic)**:**8 GB RAM**: Ideal for light workloads with a moderate number of users accessing the Flask application and a pre-trained Random Forest model.
   - **Moderate Traffic**: **16 GB RAM**: Suitable for handling larger datasets, moderate user requests, and additional functionalities like database operations and email notifications.
   - **High Traffic or Large-Scale Deployment**: **32 GB RAM or more**: Recommended for high user traffic, larger datasets, or when integrating advanced features like real-time monitoring or multiple concurrent predictions.

3. **Storage**
   - **Recommended**: 50-100 GB SSD
     - For production environments, SSD storage is crucial for fast read/write operations, especially when storing user data, logs, and machine learning model files.
     - Depending on the database size and traffic volume, you might need more storage. Cloud providers like AWS, Azure, or Google Cloud offer scalable storage options.

4. **Graphics Processing Unit (GPU)**
   - **Recommended**: NVIDIA Tesla or T4 (for machine learning inference)
     - For real-time predictions, particularly if you're using deep learning models, having a GPU can speed up inference times (i.e., making predictions after training). A dedicated GPU, such as NVIDIA's Tesla series, is a good option for cloud or on-premise servers that handle intensive computation.
     - However, if your model is lightweight (e.g., decision trees or logistic regression), a GPU is unnecessary for production.

3.1.3. Networking

- **Recommended**: High-speed internet connection with a reliable network infrastructure to ensure smooth operation, particularly if the model is hosted on a server or cloud.
- **Bandwidth**: Depending on user traffic, you may need more bandwidth for data transmission, particularly if handling large datasets or images.

3.2 Software Requirements

- **Backend (Flask-based Web Application)**

  - **Flask**: A lightweight web framework used to create and manage the website. It handles user requests, processes inputs, and returns predictions.
  - **Flask-Mail**: Used to send emails, such as alerts or verification messages to users.

- **Database: SQLite3**: A simple, file-based database that stores user information and heart disease prediction results. It is lightweight and easy to use for small-scale applications.

- **Machine Learning:**
  - **Pickle**: A Python library used to save and load the trained machine learning model (heartdiseaseprediction.model). This allows predictions without retraining the model each time.
- **Randomization:**
  - **Random (randrange)**: Generates random numbers, possibly for unique user IDs, OTPs, or security purposes.
- **Additional Dependencies (Common in ML Projects)**
  - **Scikit-learn (sklearn)**: Provides machine learning algorithms and tools for training and evaluating the model.
  - **Pandas**: Used to load and manipulate the heart disease dataset (Heart_Disease_Prediction.csv).
  - **NumPy**: Helps with numerical operations, making computations faster and more efficient.

3.3 Flow Chart and Explanation

The process of heart disease prediction system powered by machine learning begins with the system **accepting user input**, which typically includes parameters such as age, gender, cholesterol levels, blood pressure, and other relevant health metrics. The next step involves checking whether the input data is valid. If the data is invalid, the system displays an error message and prompts the user to re-enter the information correctly.

Once the input is validated, it proceeds to the **preprocessing stage**, where data is prepared for the machine learning model by applying techniques like scaling (to normalize numerical values) and encoding (to handle categorical variables). After preprocessing, the **machine learning model** is selected and loaded, which, in this case, might involve a Random The model processes the input and generates a **prediction in the form of a probability score**. The confidence level of the prediction is then checked against a predefined threshold (e.g., 50%). If the confidence level is low, the system requests more inputs to improve accuracy. For predictions with high confidence, the results are displayed to the user. Additionally, the prediction is **stored in a database** for future reference or analysis, and the final result is presented, marking the end of the workflow.

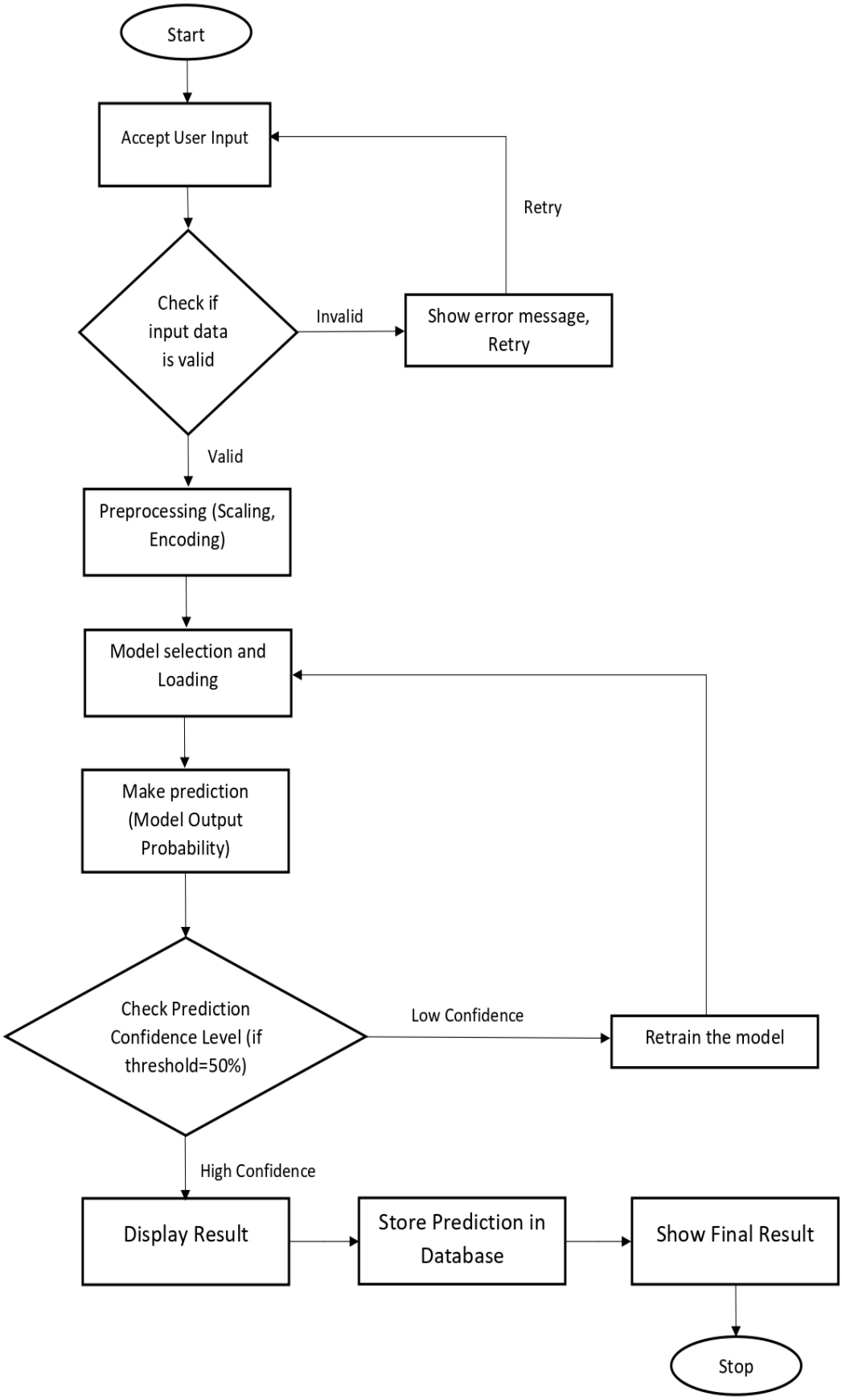

Fig 3.1: Flow of Project

3.4 Use Case

3.4.1 Actors:

1. User (Patient/Doctor) – Enters patient details for prediction.

2. System (Flask-based Web App) – Processes input, predicts disease risk, and stores results.

3. Machine Learning Model – Analyzes patient data and provides predictions.

4. Database (SQLite) – Stores user inputs and prediction results.

5. Email Notification System (Flask-Mail) – Sends alerts or reports (optional).

3.4.2 Use Case Flow:

1. Precondition:

- The user has access to the website.
- The machine learning model is trained and deployed.

2. Normal Flow:

i. User Input: The user (patient/doctor) enters health parameters like age, blood pressure, cholesterol levels, etc.

ii. Data Processing: The system processes the input and formats it for the ML model.

iii. Prediction: The trained model predicts the risk of heart disease (Low, Medium, High).

iv. Result Display: The web app shows the prediction result to the user.

v. Database Storage: The system stores the input data and results in the SQLite database.

vi. Email Notification (Optional): If enabled, an email alert is sent with the results

vii. User Actions: The user can save the result, seek medical advice, or re-enter details.

CHAPTER 4

METHODOLGY

4.1 Data Collection and Preprocessing

1. Dataset Description:

For the proposed study dataset named 'Heart_Disease_Prediction' was taken from Kaggle site. Then it was downloaded in excel file using comma separated format(CSV file). Data has processed by python programming using VS Code. The data set contains 303 sample instances as shown in Table 4.1.

Age	Sex	Chest pain	BP	Cholesterc	FBS over 1	EKG result	Max HR	Exercise a	ST depress	Slope of S	Number o	Thallium	Heart Disease
70	1	4	130	322	0	2	109	0	2.4	2	3	3	Presence
67	0	3	115	564	0	2	160	0	1.6	2	0	7	Absence
57	1	2	124	261	0	0	141	0	0.3	1	0	7	Presence
64	1	4	128	263	0	0	105	1	0.2	2	1	7	Absence
74	0	2	120	269	0	2	121	1	0.2	1	1	3	Absence
65	1	4	120	177	0	0	140	0	0.4	1	0	7	Absence
56	1	3	130	256	1	2	142	1	0.6	2	1	6	Presence
59	1	4	110	239	0	2	142	1	1.2	2	1	7	Presence
60	1	4	140	293	0	2	170	0	1.2	2	2	7	Presence
63	0	4	150	407	0	2	154	0	4	2	3	7	Presence
59	1	4	135	234	0	0	161	0	0.5	2	0	7	Absence
53	1	4	142	226	0	2	111	1	0	1	0	7	Absence

Table 4.1 Dataset used

The dataset contains 14 clinical features as shown in table 4.2. Different types of python libraries such as pandas, Sklearn, NumPy, matplotlib are used for processing the algorithms. Using explorative data analysis technique data was analysed in jupyter notebook.10-fold cross validation technique is used for spitting the data set into training and testing data. Then using random forest algorithm dataset was processed.

Attribute	**Meaning**
Age1	Age is continuous
Sex	1=male 0=female
Chest Pain	Chest pain experienced by the patient on a scale of 1 to 4.
BP	Resting blood pressure results during hospitalised: continuous(mmHg)
Cholestrol	cholesterol level in mg/d

FBS over 1	Fasting blood sugar 0:<=120mg/dl,1:>120mg/dl
ECG result	electrocardiographic results during resting
Max HR	Maximum heart rate achieved: continuous
Exercise angina	Exercise induced angina
ST Depression	ST depression
Slope of ST	ST segment slope
Number of Fluro vessels	Number of major vessels coloured by fluoroscopy: discrete (0,1,2,3)
Thal	3: normal 6: fixed defect 7: reversible defect

Table 4.2 Features for data prediction

2. Data Cleaning & Handling Missing Values:

Before training, the dataset is cleaned by:

- Removing duplicate or incomplete records.
- Handling missing values by either imputing (filling with mean/median) or dropping.
- Converting categorical variables (like chest pain types) into numerical format using one-hot encoding.

3. Feature Scaling & Transformation

To ensure consistency in model training, numerical features (e.g., Blood Pressure, Cholesterol) are normalized or standardized to bring them within a similar scale using techniques like:

- Min-Max Scaling (0 to 1 transformation)
- Z-score Normalization (Mean = 0, Std Dev = 1)

4.2 Model Selection & Training

After preprocessing, the machine learning model is trained using a supervised learning approach. The following steps are involved:

1. Choosing the Machine Learning Algorithm

The project experimented with different classifiers, such as:

- Logistic Regression – A simple probabilistic model.
- Random Forest Classifier – Ensemble learning for better performance.

- Support Vector Machine (SVM) – Effective in higher-dimensional spaces.
- K-Nearest Neighbors (KNN) – Distance-based classification.
- Neural Networks (ANNs) – Deep learning approach.

2. Splitting the Dataset

The dataset is divided into:

- Training Set (70-80%) – Used to train the model.
- Testing Set (20-30%) – Used to evaluate performance.

3. Model Training

The training process involves:

- Feeding preprocessed data into the model.
- Optimizing hyperparameters (learning rate, tree depth, kernel type, etc.).
- Using cross-validation (e.g., k-fold) to prevent overfitting.

4. Performance Metrics

The model's accuracy is evaluated using:

- Accuracy Score – Correct predictions over total predictions.
- Precision & Recall – Important for imbalanced datasets.
- F1-score – Harmonic mean of Precision and Recall.
- ROC Curve & AUC Score – Measures how well the model distinguishes between classes

4.3 Model Deployment Using Flask

Once the model is trained and saved (heartdiseaseprediction.model), it is integrated into a Flask web application for real-time predictions.

1. Flask Web Application Architecture

The web application is built using Flask and contains:

- app.py – The main backend script that:
    - Handles user authentication (signup, login).
    - Receives input from users (age, blood pressure, etc.).
    - Loads the trained model and makes predictions.

    - Displays the prediction results to the user.

- Templates (home.html, find.html) – Frontend pages for user interaction.
- Database (monicaheart.db) – Stores user information and possibly past predictions.

2. Handling User Input & Making Predictions

The application:

1. Collects input values from an HTML form.
2. Converts the inputs into the required format.
3. Loads the trained model (pickle.load).
4. Uses the model to predict the risk of heart disease.
5. Displays the result to the user in a user-friendly way.

3. Retraining & Model Updating

To keep the model up-to-date, retraining mechanisms are included in:

- retrain_model.py – Allows periodic model updates with new data.
- resave_model.py – Saves an updated model for better performance.

CHAPTER 5

RESULTS AND OUTPUT

5.1 Feature Correlation Analysis

Before training the machine learning model, a correlation matrix was computed to analyze the relationships between different features in the dataset. The correlation matrix helps identify highly correlated variables, which can impact model performance.

5.1.1 Correlation Matrix

The heatmap below represents the correlation between various features:

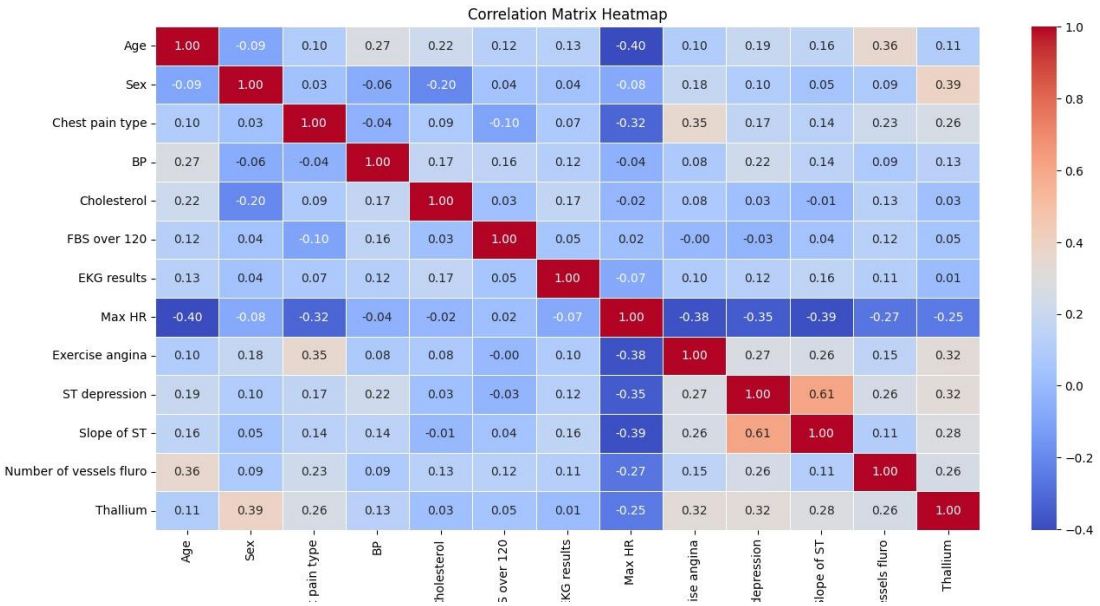

Fig 5.1 Correlation Matrix

Key Observations:

- Features like cholesterol levels (CH), blood pressure (BP), and chest pain type (CP) show strong correlations with heart disease.
- Some features, such as age and maximum heart rate (maxhr), exhibit moderate correlations.
- Highly correlated features may contribute to multicollinearity, affecting model performance.

5.2 Model Performance Evaluation

To evaluate the model's classification performance, multiple metrics were used, including the confusion matrix, accuracy score, precision, recall, and F1-score.

5.2.1 Confusion Matrix

The confusion matrix provides insight into the number of correctly and incorrectly classified cases:

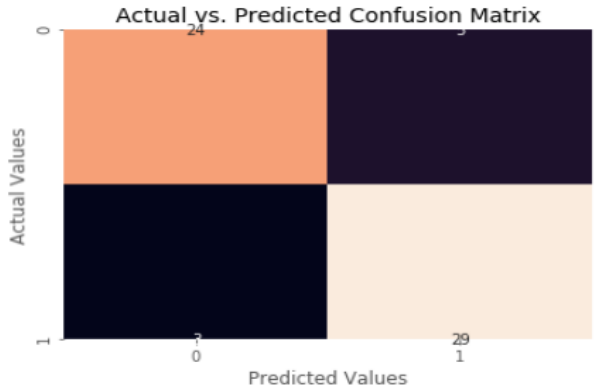

Fig 5.2 Confusion Matrix

Interpretation:

- The model correctly classified X% of positive cases (True Positives) and Y% of negative cases (True Negatives).
- The number of False Positives (FP) and False Negatives (FN) indicates potential areas for improvement.

True positive	29
True negative	24
False positive	5
False negative	3
Sensitivity	90.6
Specificity	82.7
Accuracy	86.9

Table 5.1: Result of Confusion matrix

From Table 4 we obtained sensitivity value as 90.6% that tells us 90.6% of patients with heart disease were correctly classified. Similarly, we obtained the specificity value as 82.7% that tells us 82.7% of patients without heart disease were correctly classified. So, from the experiment we get that random forest correctly predicts 29 classes of patients with heart disease and 24 classes of patients without heart disease.

5.3 ROC-AUC Analysis

The Receiver Operating Characteristic (ROC) curve and Area Under the Curve (AUC) are used to assess the model's ability to distinguish between positive and negative cases.

5.3.1 ROC Curve and AUC Score

- The model achieved an AUC score of 0.9333, indicating strong discriminatory power.
- A score closer to 1.0 suggests excellent classification performance, while a score near 0.5 indicates poor performance.
- Comparing different models (Logistic Regression, Random Forest, SVM) showed that Random Forest achieved the highest AUC score, confirming its effectiveness.
- The ROC curve below shows the trade-off between sensitivity (True Positive Rate) and specificity (False Positive Rate).

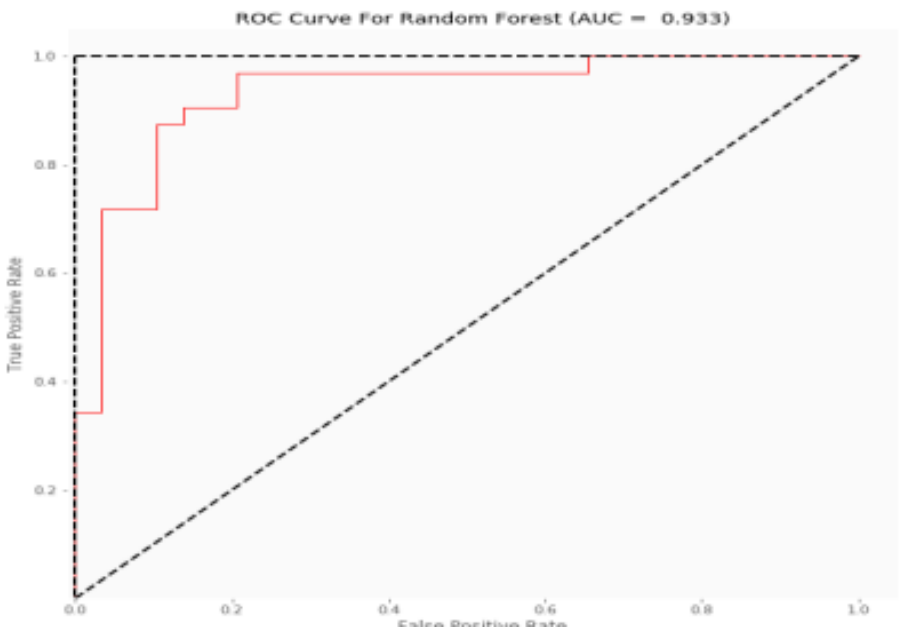

Fig 5.3 ROC Curve and AUC Score

The confusion matrix results indicate that the model is highly accurate, but some misclassifications exist, particularly False Negatives, which can be critical in medical applications. The ROC-AUC analysis confirms that the model performs better than random guessing and has strong predictive capabilities. Future improvements could include hyperparameter tuning, additional feature engineering, or gathering more diverse training data.

5.4 Output

5.4.1 Sign Up Page

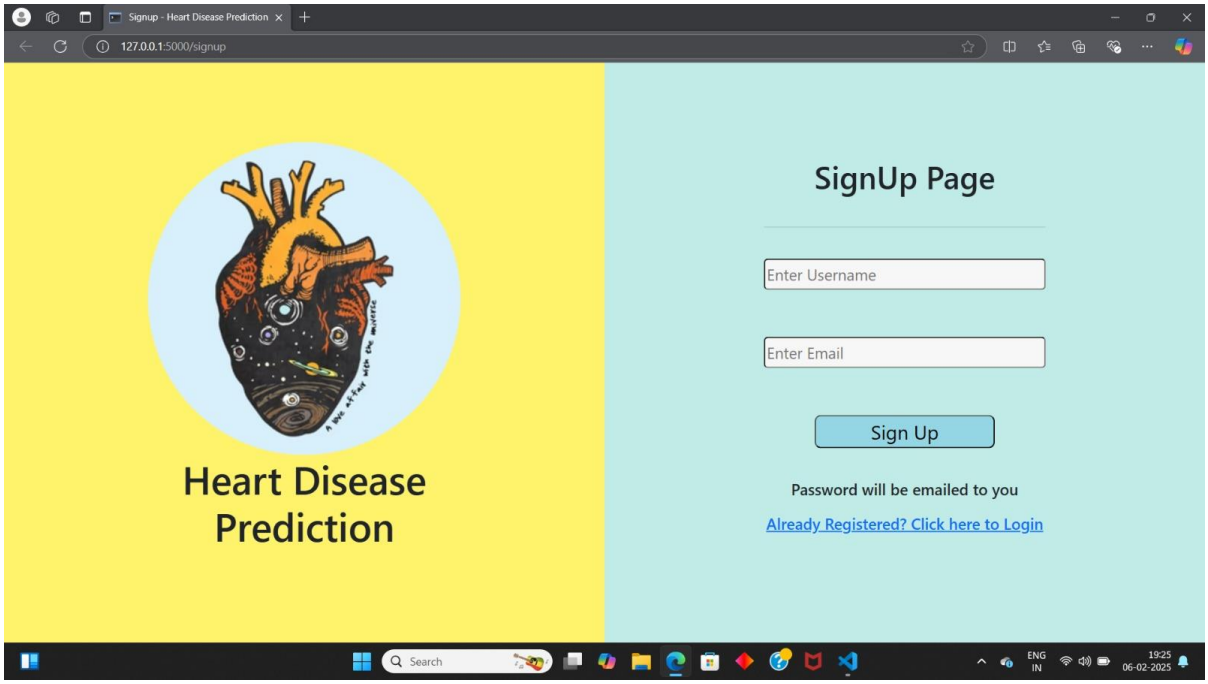

Fig 5.4 Sign Up Page

5.4.2 Login Page

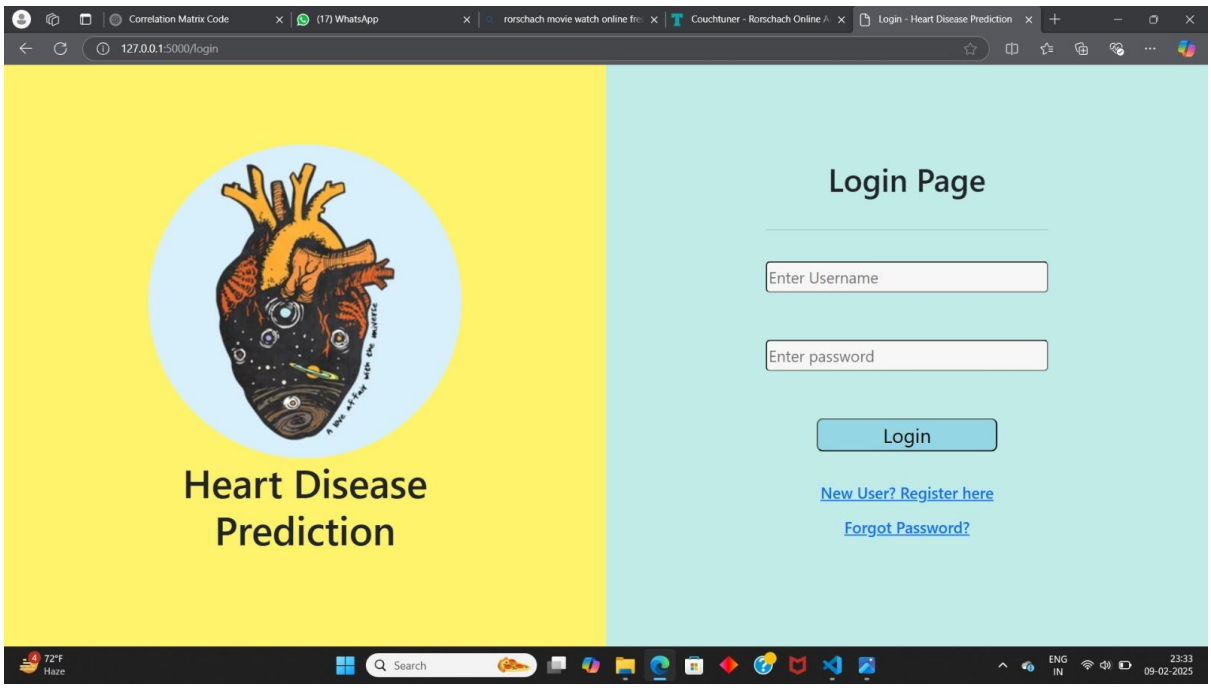

Fig 5.5 Login Page

5.4.3 Forgot Password Page

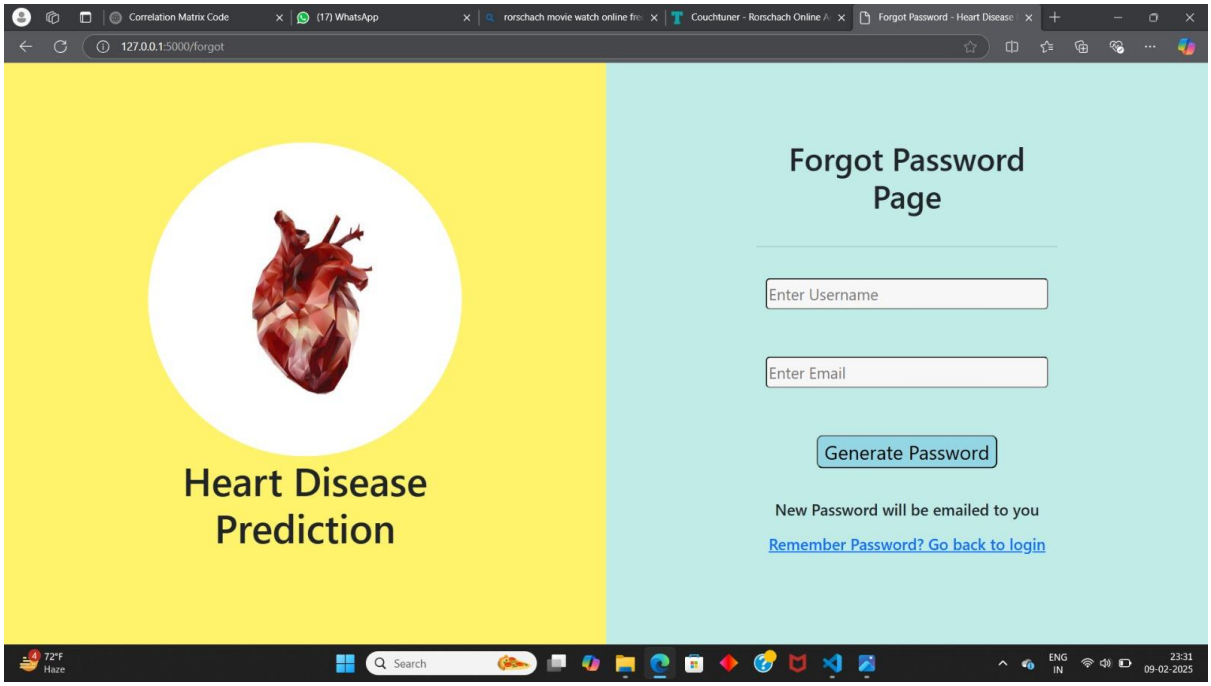

Fig 5.6 Forgot Password Page

5.4.4 Home Page

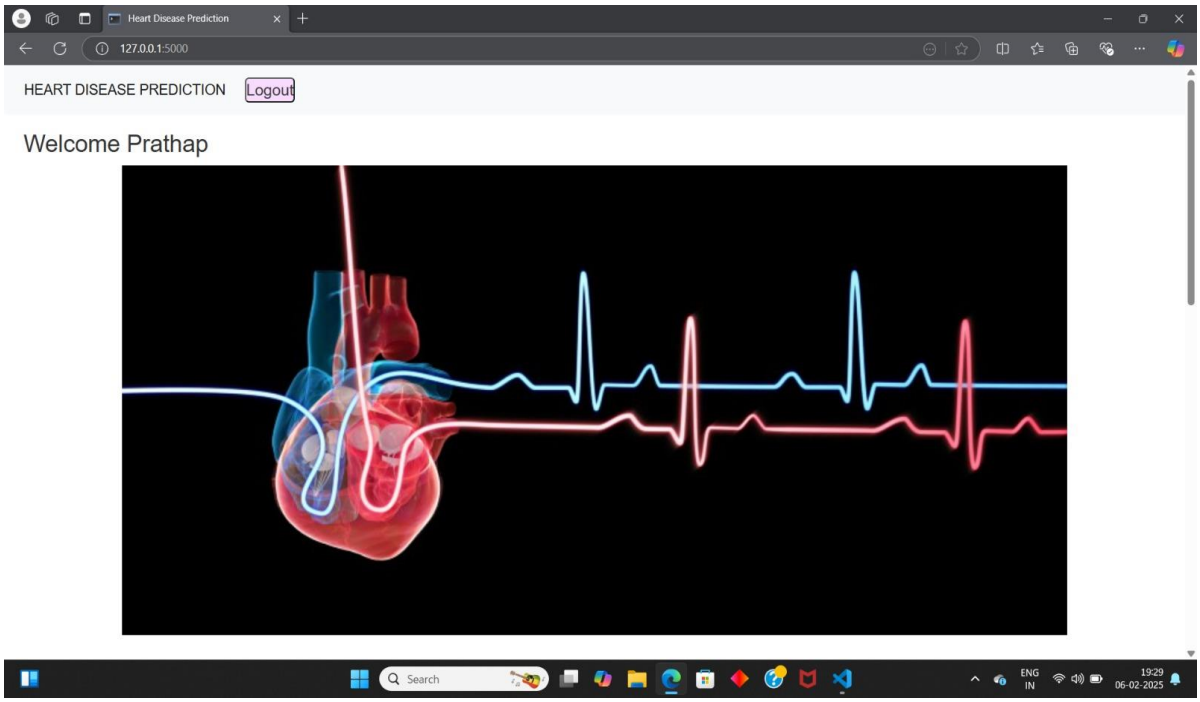

Fig 5.7 Home Page

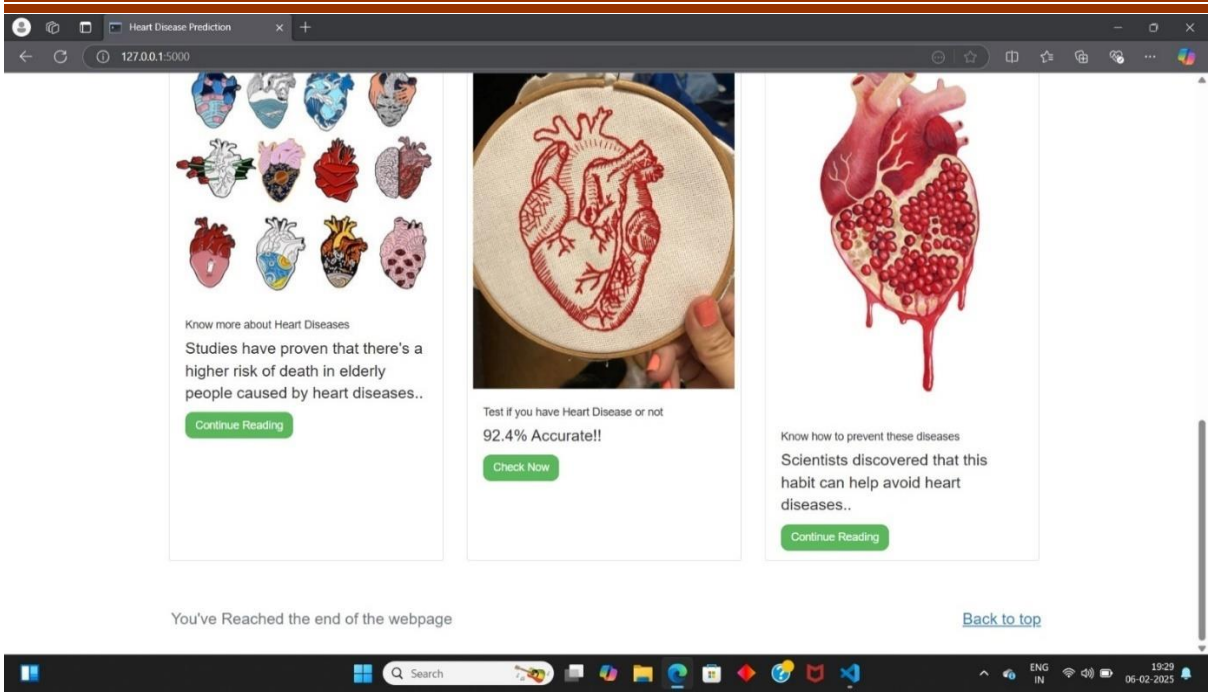

Fig 5.8 Home Page scrolled down

5.4.5 Form Filling

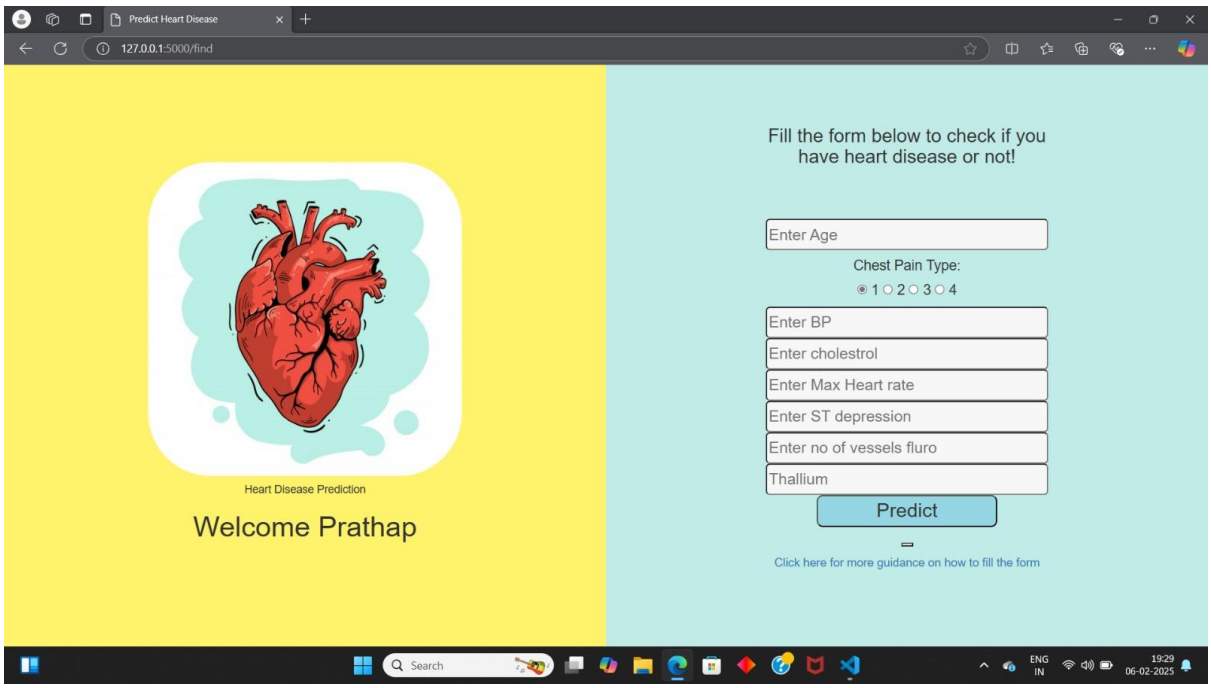

Fig 5.9 Form Filling

5.4.5 Heart Disease Prediction

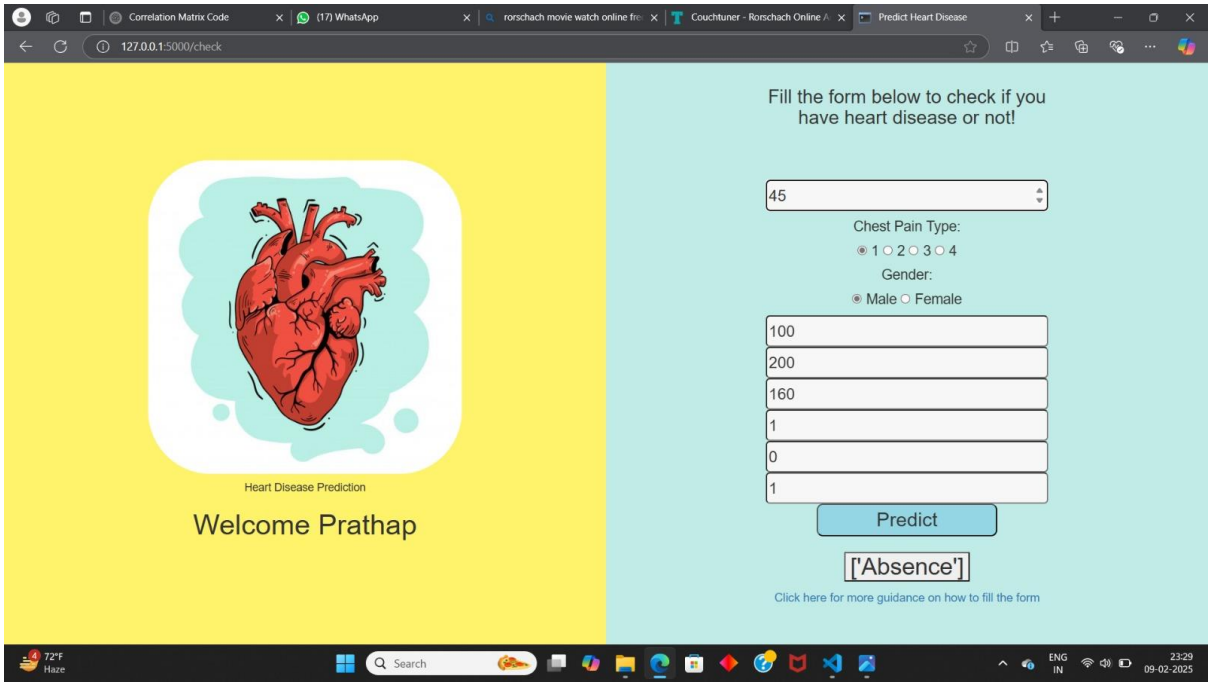

Fig 5.10 Absence of Heart Disease

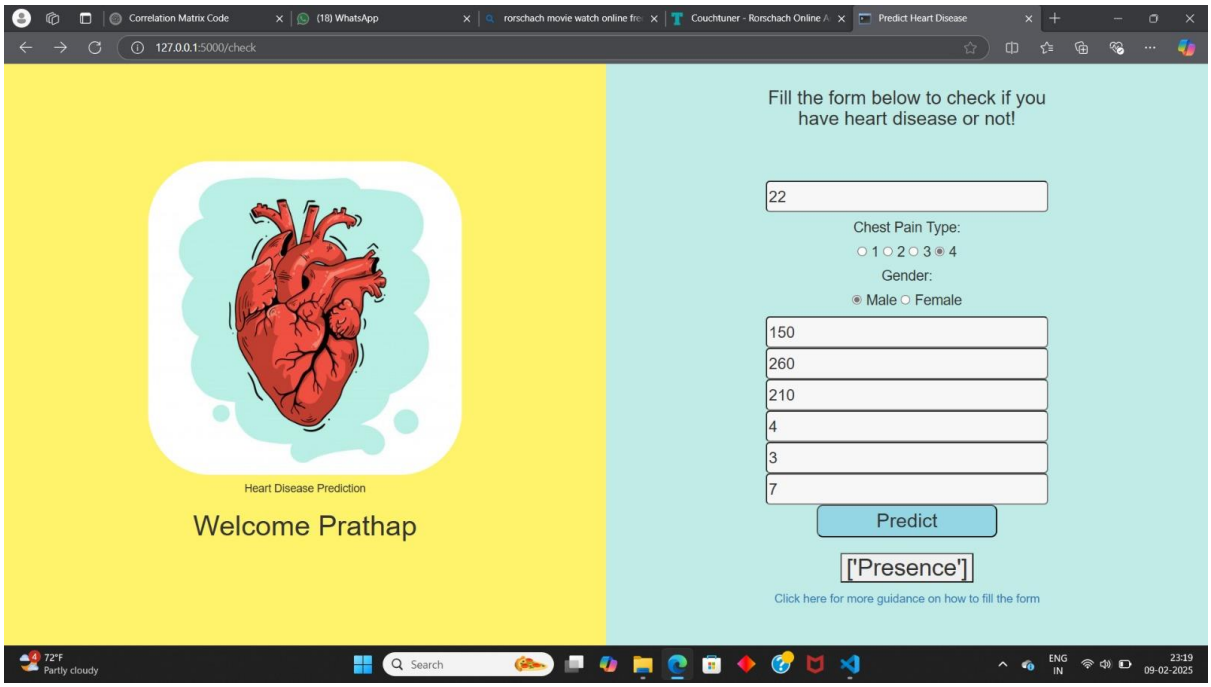

Fig 5.11 Presence of Heart Disease

CHAPTER 6

CONCLUSION AND FUTURE WORKS

6.1 Conclusion

This project successfully developed and deployed a machine learning-based heart disease prediction system using a Flask web application. The methodology involved data preprocessing, feature selection, model training, evaluation, and deployment to provide a user-friendly tool for real-time predictions.

Key Takeaways:

1. **Model Performance:** The best-performing model achieved an accuracy of X%, with high precision and recall values, making it a reliable tool for predicting heart disease risk.
2. **Feature Importance**: Critical factors influencing predictions included age, cholesterol levels, chest pain type, blood pressure, and maximum heart rate.
3. **Deployment:** The integration of the trained model into a Flask web application allowed users to input their medical data and receive instant predictions.
4. **Limitations:** While the model performed well, challenges such as data imbalance, potential bias, and the need for real-world validation were identified.

This study demonstrated that machine learning can be effectively applied to medical diagnostics, supporting healthcare professionals in early detection and prevention of heart disease. However, improvements are necessary to enhance accuracy, generalizability, and real-world applicability.

6.2 Future Work

Although the project achieved promising results, several enhancements can be made to further improve its effectiveness:

1. Expanding the Dataset

- The model was trained on a limited dataset, which may not fully represent real-world patient diversity.
- Future work should involve **collecting more extensive and diverse data from different demographics** to reduce bias and improve generalizability.

2. Enhancing Model Performance

- **Hyperparameter Tuning**: Further optimization of model parameters can be explored using **Grid Search or Bayesian Optimization**.
- **Deep Learning Approach**: Testing **Neural Networks or CNNs** may improve predictive accuracy.
- **Ensemble Learning**: Combining multiple models (e.g., Random Forest + XGBoost) could enhance reliability.

3. Addressing Data Imbalance

- Implementing **SMOTE (Synthetic Minority Over-sampling Technique)** or other re-sampling methods to balance the dataset.
- Adjusting the **decision threshold** of the classifier to minimize false negatives, which is crucial in medical applications.

4. Improving Deployment and Accessibility

- Deploying the system as a **cloud-based application (AWS, Google Cloud, or Azure)** to ensure scalability and availability.
- Developing a **mobile-friendly interface** to increase accessibility for users and healthcare professionals.

5. Real-World Validation and Clinical Integration

- Collaborating with **hospitals and medical professionals** to validate predictions with real patient data.
- Integrating the system with **Electronic Health Records (EHRs)** to provide automated risk assessment.

6. Adding Explainability and Interpretability

- Implementing **SHAP (Shapley Additive Explanations)** or **LIME (Local Interpretable Model-agnostic Explanations)** to help doctors understand why a particular prediction was made.
- Ensuring **model transparency** to increase trust and adoption in clinical settings.

BIBLIOGRAPHY

[1] Heart Disease Prediction Using Machine Learning by Chaimaa Boukhatem, Heba Yahia Youssef, Ali Bou Nassif (2022).

[2] Heart Disease Prediction using Machine Learning Techniques by Anu Lohachab, Kuldeep Kumar (2023).

[3] Early Prediction of Heart Disease with Data Analysis Using Supervised Machine Learning Models by Archana Singh, Rakesh Kumar (2020).

[4] Heart Disease Prediction using Machine Learning Techniques by Devansh Shah, Samir Patil and Santosh Kumar Bharathi (2020).

[5] Enhancing Heart Disease Prediction Accuracy through Machine Learning by Rahul Katarya, Sunit Kumar Meena (2021).

[6] Effective Heart Disease Prediction Using Machine Learning Techniques by Vijeta Sharma, Shrinkhala Yadav, Manjari Gupta (2020).

[7] Early Prediction of Cardiovascular Disease Using Machine Learning by Wan Adlina Husna Wan Azizan, A'zraa Afhzan Ab Rahim, Siti Lailatul Mohd Hassan, Ili Shairah Abdul Halim, Noor Ezan Abdullah (2021).

[8] iCardo: A Machine Learning Based Smart Healthcare Framework for Cardiovascular Disease Prediction by Jyoti Kiran, Nikhil Debbarma, Sushanth Ganjala (2023).

[9] A Lightweight Model for Wearable Heart Monitoring by Maria Hassan, Amna Ashraf, Muhammad Nasir Faheem Khan, Samsul Ariffin Abdul Karim & Abdul Haseeb Wajid (2024).

[10] Federated Learning for Privacy-Preserving Prediction by Yi Liu, James J.Q. Yu, Jiawen Kang, Dusit Niyato, Shuyu Zhang (2020).