



TRIBHUVAN UNIVERSITY
FACULTY OF HUMANITIES AND SOCIAL SCIENCE

A Project Report
on
“CardioBot: Heart Disease Prediction System using Logistic
Regression”

Submitted To:
Department of Computer Applications
National College of Computer Studies

In partial fulfillment of the requirements for a Bachelor's Degree in
Computer Applications

Submitted By:
Sichu Maharjan
6-2-551-49-2021

Under the supervision of
Teksan Gharti Magar

2025



Tribhuvan University
Faculty of Humanities and Social Sciences
National College of Computer Studies

SUPERVISOR'S RECOMMENDATION

I hereby recommend that this project, prepared under my supervision by **Sichu Maharjan** entitled "**CardioBot: Heart Disease Prediction System**", in partial fulfillment of the requirements for the degree of Bachelor of Computer Application, is recommended for the final evaluation.

.....

SIGNATURE

Teksan Gharti Magar

SUPERVISOR

Faculty Member

Department of Computer Applications

National College of Computer Studies



Tribhuvan University
Faculty of Humanities and Social Sciences
National College of Computer Studies

LETTER OF APPROVAL

This is to certify that the project prepared by **Sichu Maharjan**, entitled "**CardioBot: Heart Disease Prediction System**", in partial fulfillment of the requirements for the degree of Bachelor in Computer Application, has been evaluated. In our opinion, it is satisfactory in scope and quality as a project for the required degree.

<p style="text-align: center;">SIGNATURE of Supervisor</p> <p style="text-align: center;">.....</p> <p style="text-align: center;">Teksan Gharti Magar Faculty Member Department of Computer Applications National College of Computer Studies</p>	<p style="text-align: center;">SIGNATURE of HOD/Coordinator</p> <p style="text-align: center;">.....</p> <p style="text-align: center;">Faculty Member Department of Computer Applications National College of Computer Studies</p>
<p style="text-align: center;">SIGNATURE of Internal Examiner</p>	<p style="text-align: center;">SIGNATURE of External Examiner</p>

ABSTRACT

The Heart Disease Prediction System is an innovative web-based application that leverages machine learning to predict the risk of heart disease. Built with Django, this comprehensive system offers three distinct user interfaces for patients, healthcare providers, and administrators. The core system utilizes a Logistic Regression to analyze critical cardiovascular health parameters to provide binary classification results (Healthy/Unhealthy).

Patients can input their health parameters through an intuitive interface and receive instant predictions. Healthcare providers can monitor patients, access health histories, and make predictions on their behalf. The system features secure user authentication, role-based access control, prediction history tracking, and feedback management.

With its emphasis on data security and privacy, the system serves as a crucial tool in preventive healthcare, enabling early detection of heart disease risk factors and efficient patient monitoring, ultimately contributing to improved cardiovascular health outcomes.

Keywords: *Heart Disease Prediction, Machine Learning, Logistic Regression, Django, Web-based Application, Preventive Healthcare, User Authentication, Role-based Access Control, Health Monitoring, Data Privacy*

ACKNOWLEDGEMENT

I extend my heartfelt gratitude to my supervisor, **Mr. Teksan Gharti Magar**, for his invaluable guidance, encouragement, and insightful suggestions throughout this project. I am also deeply thankful to Mr. Chhetra Bahadur Chhetri for his guidance during the development phase.

Furthermore, I express my sincere appreciation to our Vice Principal, Mr. Santosh Maskey, whose constant monitoring, guidance, and motivation played a pivotal role in every phase of this project. I am grateful for the unwavering support from all supervisors, friends, and family members, whose encouragement and assistance contributed significantly to the successful completion of this endeavor.

Sichu Maharjan

TABLE OF CONTENTS

SUPERVISOR’S RECOMMENDATION	ii
LETTER OF APPROVAL	iii
ABSTRACT.....	iv
ACKNOWLEDGEMENT.....	v
LIST OF FIGURES	viii
LIST OF TABLES.....	ix
LIST OF ABBREVIATIONS	x
CHAPTER 1: INTRODUCTION.....	1
1.1 Introduction.....	1
1.2 Problem Statement.....	1
1.3 Objectives	2
1.4 Scope and Limitations.....	2
1.5 Report Organization.....	3
CHAPTER 2: BACKGROUND STUDY.....	4
2.1 Background Study.....	4
2.1.1 Study of Existing System.....	4
2.1.2 Literature Review.....	4
CHAPTER 3: SYSTEM ANALYSIS AND DESIGN.....	7
3.1 System Analysis.....	7
3.1.1 Requirement Identification	8
3.1.2 Feasibility Study	12
3.1.3 Data Modeling	14
3.1.4 Process Modeling.....	15
3.1.5 Use-Case Diagram	17
3.1.6 Flow Chart	18
3.2 System Design	19
3.2.1 Architectural Design	19
3.2.2 Database Design Schema.....	19
3.3 Algorithm Details.....	20
CHAPTER 4: IMPLEMENTATION AND TESTING.....	23
4.1 Implementation	23
4.1.1 Tools Used.....	23

4.1.2 Implementation Details of Modules.....	23
Module 1: Patient Module	23
Module 2: Doctor Module	29
Module 3: Admin Module.....	31
4.2 Testing.....	33
4.2.1 Testing cases for Unit Testing	33
4.2.2 Testing case for Patient Prediction.....	34
4.2.3 Testing case for Doctor Prediction.....	34
CHAPTER 5: CONCLUSION AND FUTURE RECOMMENDATIONS	36
5.1 Lesson Learnt.....	36
5.2 Conclusion	36
5.3 Future Recommendations	36
REFERENCES.....	37

LIST OF FIGURES

Figure 3.1: Waterfall Model	7
Figure 3.2: ER Diagram of CardioBot	14
Figure 3.3: Level 0 DFD of CardioBot	15
Figure 3.4: Level 1 DFD of CardioBot	16
Figure 3.5: Use-Case Diagram of CardioBot	17
Figure 3.6: Flowchart of CardioBot	18
Figure 3.7: Architectural Design of CardioBot	19
Figure 3.8: Database Schema of CardioBot	19

LIST OF TABLES

Table 4.1: Test cases of unit testing for login and logout operations	33
Table 4.2: Test cases of unit testing for patient prediction	34
Table 4.3: Test cases of unit testing for doctor prediction	34
Table 4.4: Test cases of system testing for the admin dashboard	35

LIST OF ABBREVIATIONS

BCA	Bachelor's in Computer Application
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
DB	Database
DFD	Data Flow Diagram
Django	A high-level Python web framework
ER	Entity Relationship
ERD	Entity Relationship Diagram
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment
JS	JavaScript
ML	Machine Learning
MVC	Model-View-Controller
Python	A high-level programming language
SQLite	Structured Query Language - Lite
UI	User Interface
XAMPP	Cross-platform, Apache, MySQL, PHP, Perl

CHAPTER 1:

INTRODUCTION

1.1 Introduction

CardioBot: Heart Disease Prediction System is a web-enabled application that leverages Machine Learning and Data Mining to forecast the likelihood of heart disease in patients. Heart disease is a significant health concern worldwide, and its treatment can be highly costly. These issues can be addressed through early detection by analyzing primary medical parameters such as age, blood pressure, cholesterol levels, and other vital health factors.

Designed using Django, Python, and modern web technologies, the system provides a simple user interface for medical professionals and patients. It is equipped with features such as a doctor search facility, patient registration, disease prediction, and feedback mechanisms. The backend utilizes advanced machine learning technique (Logistic Regression) to analyze patient data and provide accurate predictions.

This project facilitates access to healthcare and make it more economical by providing an intelligent system for early detection of heart disease. Early detection can lower healthcare costs and improve cure rates through timely intervention.

1.2 Problem Statement

Heart disease remains one of the leading causes of mortality worldwide, with delayed detection often resulting in severe complications or fatal outcomes. While existing heart disease prediction systems exist, they exhibit several critical limitations that reduce their effectiveness in real-world healthcare settings:

1. **Accessibility Barriers:** Current systems often require specialized medical knowledge to interpret results, making them inaccessible to the general public and limiting their preventive healthcare potential.
2. **Data Management Challenges:** Existing platforms typically focus solely on prediction without comprehensive data management capabilities, making it difficult to track patient history and monitor health trends over time.
3. **Security and Privacy Concerns:** Many current solutions lack robust security measures and role-based access control, raising concerns about sensitive medical data protection.

4. **User Experience Issues:** Available systems often feature complex interfaces that discourage regular use by both patients and healthcare providers.

The Heart Disease Prediction System addresses these limitations by providing:

- A role-based platform tailored for patients, doctors, and administrators
- An intuitive interface accessible to both healthcare professionals and the general public
- Comprehensive data management, including historical tracking and visualization
- Enterprise-grade security with robust role-based access control
- Real-time monitoring and automated risk alerts
- Integration of advanced machine learning for accurate predictions

This system is developed to bridge the gap between early detection capabilities and practical healthcare delivery, enabling proactive heart health management through precise prediction and systematic tracking of heart disease risk factors.

1.3 Objectives

- To develop CardioBot: Heart Disease Prediction System utilizing Logistic Regression for accurate prediction of heart disease risk.

1.4 Scope and Limitations

The project aims to predict heart disease risk by analyzing thirteen critical health parameters for healthcare providers, providing immediate health insights to both groups. It serves as a preventive healthcare tool for patients, doctors, and administrators to monitor and track heart health conditions.

Limitations:

- Users are limited to entering only the predefined health parameters; custom parameters are not supported.
- The system provides binary risk predictions (Healthy/Unhealthy) without detailed medical explanations or personalized recommendations.
- There is no integration with external medical devices for real-time health data acquisition.
- The system's predictive capability is limited to heart disease and does not extend to other health conditions.

1.5 Report Organization

Chapter 1 Introduction, introduces the concept of this project. It describes the problems that has been existing and how its objective can tackle it. It also presents the scope and limitations of the project.

Chapter 2 Background Study, includes the background study of fundamental theories, general concepts and literature review of similar projects, theories and results by other researchers.

Chapter 3 System Analysis and Design, provides an overview of all the requirements along with system analysis, feasibility analysis of the system, detailed description of how the system was designed and the implementation of algorithm.

Chapter 4 Implementation and Testing, gives information about how the project has been implemented, what kind of software and tools has been used and the type of testing that the project has gone through.

Chapter 5 Conclusion and Future Recommendations, includes the possible outcome of this project, conclusion and future recommendations.

CHAPTER 2:

BACKGROUND STUDY

2.1 Background Study

The project examines the evolution of heart disease prediction systems, tracing developments from traditional clinical diagnosis methods to contemporary AI-powered solutions. Earlier systems primarily relied on hospital diagnostic tools and electronic health records, often requiring manual analysis by medical professionals. More recent approaches have introduced research-based prediction models utilizing statistical methods and, increasingly, machine learning techniques.

2.1.1 Study of Existing System

In examining existing systems for heart attack detection, various approaches and technologies have been employed to monitor cardiovascular health. Many current solutions rely on datasets collected from sensors such as ECG, heart rate, and blood pressure monitors, which help detect anomalies or predict heart attack risks. Machine learning algorithms, such as decision trees and neural networks, are often trained on historical health data to classify individuals into risk categories.

The CardioBot system advances beyond these approaches by not only utilizing pre-existing datasets to train its machine learning models, but also enabling users to input their real-time health data directly into the system. This allows CardioBot to assess a user's current health status based on live parameters such as age, blood pressure, cholesterol levels, and other vital health indicators. By leveraging both historical and real-time data, the system can predict the likelihood of a heart disease more accurately and provide immediate, personalized insights. This capability distinguishes CardioBot from many existing systems that focus solely on historical data, enabling it to deliver timely and individualized heart attack risk assessments.

2.1.2 Literature Review

To date, different studies have been done on heart disease prediction. Various data mining and machine learning algorithms have been implemented and proposed on the datasets of heart patients, and different results have been achieved for different techniques. But, still today we are facing a lot of problems related to heart disease. Some of the recent research papers are as follows:

In 2010, Authors applied machine learning algorithms such as Naive Bayes, KNN (K- Nearest neighbors), and decision list for heart disease prediction. The Tanagra tool is used to classify the data, and the data is evaluated using 10-fold cross-validation, and the results are compared in Table 4. The data set consists of 3000 instances with 14 different attributes. The dataset is divided into two parts; 70% of the data is used for training, and 30% is used for testing. The results of the comparison are based on 10-fold cross-validation. Comparison is made among these classification algorithms, out of which the Naive Bayes algorithm is considered the best-performing algorithm. Because it takes less time to build a model and also gives the best accuracy as compared to KNN and Decision Trees [1].

The author developed a Decision Support in Heart Disease Prediction System (DSHDPS) using a data mining modeling technique, namely, Naive Bayes. Using heart disease attributes such as chest pain, age, sex, cholesterol, blood pressure, and blood sugar can predict the likelihood of patients getting heart disease. It is implemented as a web-based questionnaire application. The historical data set of heart patients from the Cleveland database of the UCI repository was used to train and test the Decision Support System (DSS). The reasons to prefer the Naive Bayes machine learning algorithm for predicting heart disease are as follows: when the data is high, when the attributes are independent of each other, and when we want to achieve high accuracy as compared to other models. When the dimensionality of the inputs is high, in that case Naïve Bayes classifier technique is particularly suited. Despite its simplicity, Naive Bayes can often outperform more sophisticated classification methods [2].

In 2013, Authors applied data mining and machine learning algorithms, namely Decision Tree (J48 algorithm), Naive Bayes, and Artificial Neural Networks (ANN) for heart disease prediction. A dataset of 7339 instances with 15 attributes has been taken from PGI Chandigarh. The WEKA 3.6.4 tool was used for the experiment. For model training and testing, 10-Fold Cross-Validation techniques are used randomly. The Best First Search method was used to select the best attributes from the already available 15 attributes, and among them, only 8 attributes were selected. Each experiment was done on two different scenarios, the first one containing all 15 attributes and the second case only 8 selected attributes. From all these experiments comparative results has been obtained and from these comparative results it has been found that J48 pruned in selected attributes case has performed best in accuracy with 95.56% and Naive Bayes with all attributes case gives less accuracy 91.96% but takes least time to build a model in the whole experiment [3].

In 2014, Authors applied data mining algorithms such as J48, Naive Bayes, REPTREE, CART, and Bayes Net in this research for predicting heart attacks. The patient dataset is collected from medical practitioners in South Africa. Only 11 attributes, namely Patient Id, Gender, Cardiogram, Age, Chest Pain, Blood Pressure Level, Heart Rate, Cholesterol, Smoking, Alcohol consumption, and Blood Sugar Level from the database, are considered for the predictions required for heart disease. The WEKA data mining tool is used for experiments. The algorithms were applied to the dataset using a 10-fold cross-validation technique to calculate the average accuracy of all folds for each classification technique to predict a class. From the results, it has been seen that J48, REPTREE, and SIMPLE CART algorithm performs best in this data set, while the Bayes Net algorithm outperformed the Naïve Bayes algorithm [4].

In 2016, the Author applied three machine learning algorithms, viz Naïve Bayes, J48, and Artificial Neural Network (ANN), to achieve the best accuracy in heart disease prediction for male patients. A dataset of 210 records with 8 attributes has been used in this experiment. To carry out experiments and implementations, WEKA was used as the data mining tool. From the experiments, comparative results have been drawn in Table 8, and from the comparative results has been found that Naïve Bayes performed best as compared to J48 and ANN to predict heart disease with an accuracy of 79.9043% and takes less time, 0.01 seconds, to build a model [5].

CHAPTER 3:

SYSTEM ANALYSIS AND DESIGN

3.1 System Analysis

The Heart Disease Prediction System was developed using the Waterfall methodology, a linear and sequential approach to software development. This methodology was particularly suitable for this project due to its well-defined requirements and clear progression through distinct phases:

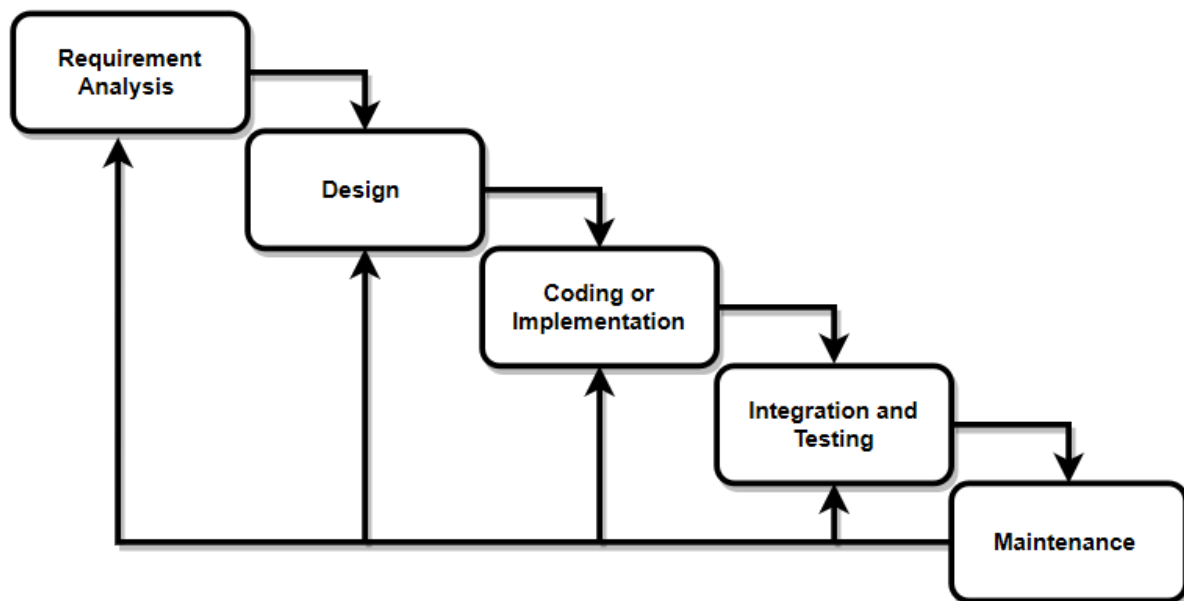


Figure 3.1: Waterfall Model

1. Requirements Analysis

- Gathered comprehensive requirements for the heart disease prediction system
- Identified key medical parameters needed for accurate prediction
- Defined system functionalities including user registration, data input, and prediction display
- Documented specific needs of different user roles (patients, doctors, administrators)

2. System Design

- Designed the system architecture using the Django framework
- Created database schema for storing patient data and medical records
- Designed user interfaces for different stakeholders

- Selected appropriate machine learning algorithms (Logistic Regression)
- Planned the integration of machine learning models with the web application

3. **Implementation**

- Developed the web application using Django, HTML, CSS, and JavaScript
- Implemented machine learning models using scikit-learn
- Created database models and implemented CRUD operations
- Integrated the prediction system with the user interface
- Implemented user authentication and authorization

4. **Testing**

- Conducted unit testing of individual components
- Performed integration testing of the complete system
- Validated machine learning model accuracy
- Tested system performance and security
- Verified user interface functionality across different devices

5. **Deployment**

- Deployed the system on a production server
- Configured necessary security measures
- Set up database and file storage
- Implemented backup and recovery procedures

6. **Maintenance**

- Monitored system performance
- Addressed user feedback and bug reports
- Updated machine learning models as needed

3.1.1 Requirement Identification

Requirements will be gathered through a combination of interviews, Google searches, visits to relevant websites, and friendly suggestions.

A. Functional Requirements

1. User Management

- i. User Registration and Authentication
 - a. Patients can register and create accounts
 - b. Doctors are added by admin

- c. Secure login system for all users (patients, doctors, admin)

2. Health Parameter Management

- i. Patient Data Input
 - a. Simplified health assessment form with:
 - Personal information (name, age, sex)
 - Physical measurements (height, weight)
 - Health history (smoking, hypertension, diabetes, high cholesterol)
 - Family history of heart disease
 - Current symptoms (chest pain, shortness of breath)
 - Lifestyle habits (exercise frequency, diet habits, stress levels)
- ii. Doctor Data Input
 - a. Detailed clinical parameters:
 - Patient information (name, contact)
 - Basic information (age, sex)
 - Vital signs (resting blood pressure, cholesterol, fasting blood sugar, maximum heart rate)
 - Cardiac assessment (chest pain type, resting ECG, exercise induced angina, ST depression)
 - Advanced assessment (slope, number of vessels, thalassemia)
 - Additional notes
- iii. Prediction Model
 - a. Uses Logistic Regression
 - b. Binary classification (Healthy/Unhealthy)
 - c. Provides impact factor for normal users (patients)

3. Prediction History Management

- i. For Patients:
 - a. View their prediction history
 - b. Track changes over time
- ii. For Doctors:
 - a. View predictions they've made
 - b. Access patient prediction histories
- iii. For Admin:
 - a. View all predictions in the system

- b. Delete prediction records
- c. System-wide monitoring

4. Doctor Management

- i. Doctor profile management
- ii. View and edit personal details
- iii. Add new doctors (admin)
- iv. Edit doctor details (admin)
- v. Delete doctor profiles (admin)
- vi. View all registered doctors (admin)

5. Patient Management

- i. Patient profile management
- ii. View and edit personal details
- iii. Delete patient profiles (admin)
- iv. View registered patients (admin)

6. Feedback System

- i. Users can submit feedback
- ii. Admin can view all feedback
- iii. Admin can delete feedback entries

7. Dashboard and Analytics

- i. Admin Dashboard:
 - a. Total number of doctors
 - b. Total number of patients
 - c. Total feedback count
 - d. Total number of predictions
- ii. Doctor Dashboard:
 - a. Recent Predictions
 - b. Prediction accuracy metrics

8. Data Visualization

- i. Display prediction results
- ii. Tabular display of prediction history
- iii. Sortable and searchable data tables

9. Search and Filter Capabilities

- i. Search through prediction history
- ii. Filter predictions by date

- iii. Sort predictions by various parameters

10. Profile Management

- i. Users can view their profiles
- ii. Edit personal information
- iii. Update contact details
- iv. Change profile pictures

11. System Administration

- i. Monitor system usage
- ii. Manage user accounts
- iii. Handle doctor accounts
- iv. System maintenance capabilities

B. Non-Functional Requirements

- i. **Performance:** The system should process and analyze data in real time, providing risk predictions and alerts within seconds to ensure timely intervention.
- ii. **Scalability:** The system should be designed to handle multiple users concurrently without performance degradation, particularly if it scales to accommodate larger user bases.
- iii. **Security & Data Privacy:** The system must adhere to high standards of security to protect user data. This includes encryption for both data storage and transmission to prevent unauthorized access. The system should comply with data privacy regulations (e.g., GDPR) to ensure that personal health data is handled and stored securely and ethically.
- iv. **Reliability:** The system should be highly reliable, with minimal downtime. It should handle errors gracefully and provide accurate results without failure.
- v. **Usability:** The system must be easy to use for individuals of varying technical expertise. The UI should be intuitive, allowing users to easily input data and understand the results.
- vi. **Maintainability:** The system should be easy to update and maintain, ensuring that it can be upgraded with new features or fixes without major disruptions.

3.1.2 Feasibility Study

It involves assessing how effectively the system will operate within specified constraints, encompassing considerations of operational feasibility, economic feasibility, and technical feasibility.

A. Technical Feasibility

This Heart Disease Prediction System is technically feasible, built on a Django web app with user management, Logistic Regression via scikit-learn, and strong data preprocessing. It supports model retraining, prediction history, and cross-platform use. With minimal dependencies and SQLite (easily upgradable), it's easy to deploy. The modular, well-structured code with proper validation and error handling makes it production-ready for healthcare.

1. Technology Stack

- i. Backend: Django (Python)
- ii. Frontend: HTML, CSS, JavaScript
- iii. Database: Django's default SQLite (can be upgraded to PostgreSQL/MySQL)

2. Machine Learning Integration

- i. Using Logistic Regression
- ii. Data Processing

3. System Architecture

- i. MVC Pattern (Django's MTV)
- ii. Security Implementation

4. Performance Considerations

- i. Prediction Speed
- ii. Database Operations

B. Operational Feasibility

The system is operationally feasible, offering a user-friendly interface that fits well into typical workflows. Users can easily register, log in, and make predictions, while administrators can manage data effortlessly, ensuring smooth day-to-day operations.

1. User Management

- i. User Roles
 - a. Clear role separation (Admin, Doctor, Patient)
 - b. Appropriate access controls
 - c. Role-specific features
- ii. User Interface

- a. Intuitive navigation
- b. Clear error messages
- c. Responsive design

2. Data Management

- i. Data Storage
 - a. Structured database schema
 - b. Proper data relationships
 - c. Data integrity maintained
- ii. Data Security
 - a. Protected health information
 - b. Access controls
 - c. Audit trails

3. System Maintenance

- i. Updates and Patches
 - a. Django's update mechanism
 - b. Version control support
 - c. Easy deployment process
- ii. Backup and Recovery
 - a. Database backup capabilities
 - b. Data export options
 - c. System restores points

4. Integration Capabilities

- External Systems
 - a. Standard data formats
 - b. Interoperability support
- Third-party Services
 - a. Easy to integrate additional services
 - b. Modular design
 - c. Extensible architecture

C. Economic Feasibility

The Heart Disease Prediction System is economically viable, using free, open-source technologies like HTML, CSS, JavaScript, PHP, and MySQL. Development tools such as VS

Code, GitHub, and SQLite further cut costs. With affordable hosting, the project requires minimal investment, making it budget-friendly and sustainable.

3.1.3 Data Modeling

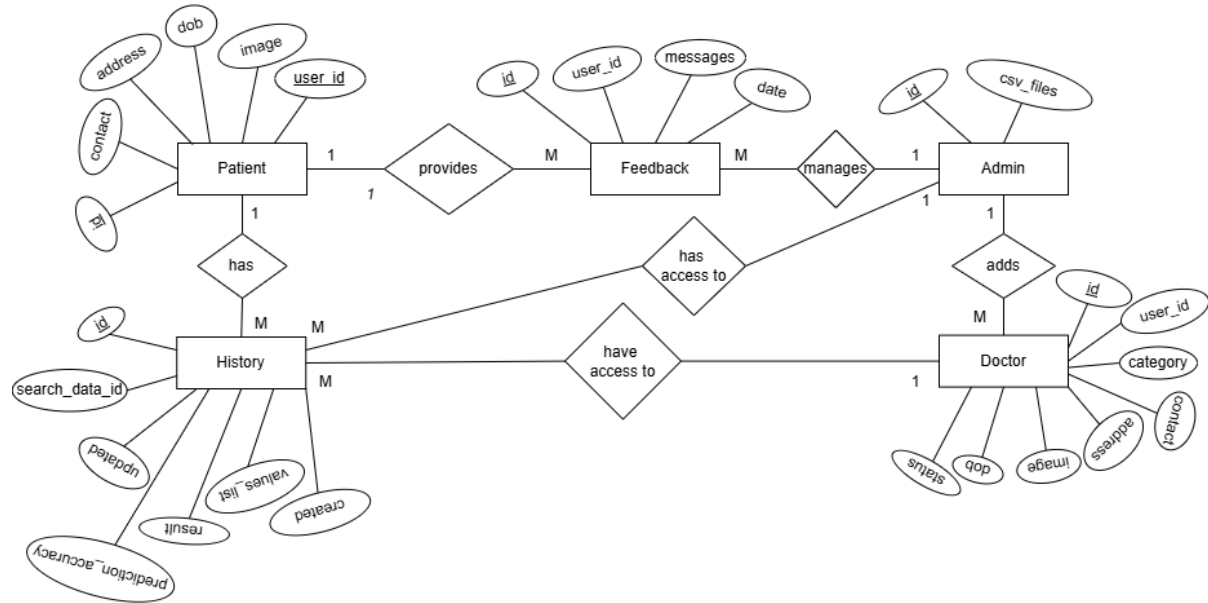


Figure 3.2: ER Diagram of CardioBot

The ER diagram for the Heart Disease Prediction System illustrates how the main components of the Heart Disease Prediction System are connected. Patients use the system to generate heart health predictions, which are stored in the History table. Doctors, who are added by admins, can access these prediction histories to monitor patient health. Patients can also provide feedback, which is managed by admins. Admins oversee the entire system, including managing doctors and feedback. The diagram highlights the relationships between patients, doctors, admins, feedback, and prediction history, illustrating how data flows and is managed within the platform.

3.1.4 Process Modeling

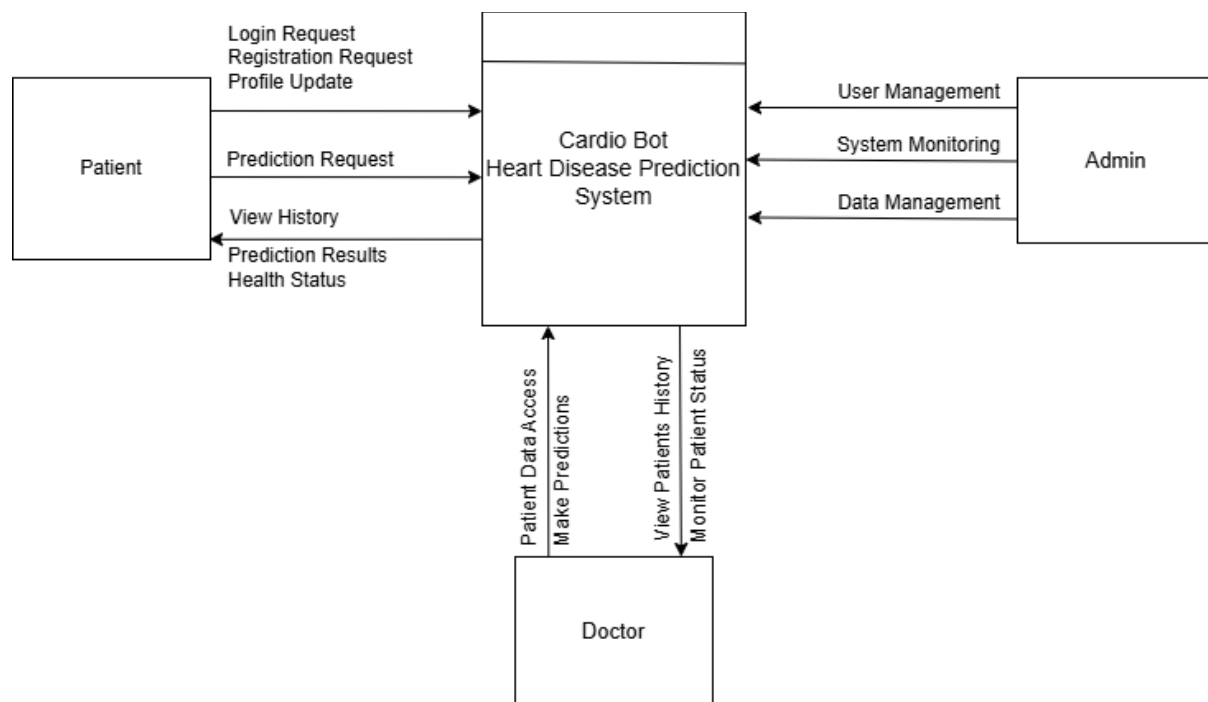


Figure 3.3: Level 0 DFD of CardioBot

Figure 3.3 shows the interaction between three main external entities (Patients, Doctors, and Admin) and the central system. Patients can log in, register, input their health data, and view prediction results. Doctors interact with the system by making predictions, viewing patient data, and monitoring health records. The admin manages the overall system, including doctor approvals, user management, and system monitoring. All data flows through the central Heart Disease Prediction System, which processes requests and returns appropriate responses to each user type. This high-level diagram provides a clear overview of how different users interact with the system and the basic flow of information.

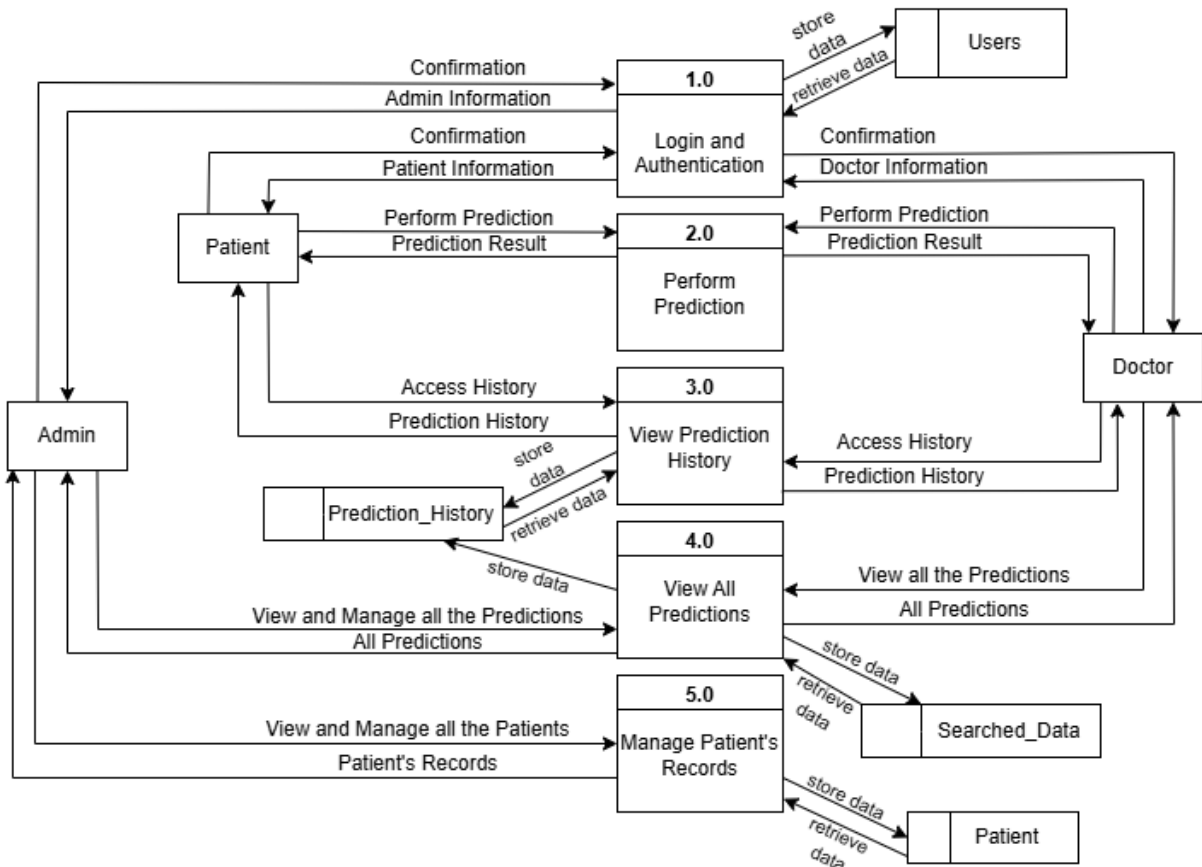


Figure 3.4: Level 1 DFD of CardioBot

Figure 3.4 illustrates the primary processes and data interactions within the system. The main external entities—Patient, Doctor, and Admin—interact with the system through various processes such as Login and Authentication, Perform Prediction, View Prediction History, View All Predictions, and Manage Patient Records. Each process is responsible for handling specific tasks, such as authenticating users, processing heart disease predictions, displaying prediction history, and managing patient records. Data stores like Users, Prediction_History, Searched_Data, and Patient are used to securely store and retrieve relevant information, ensuring data integrity and accessibility. The diagram clearly shows the flow of information between users, processes, and data stores, highlighting how patients and doctors can perform predictions and access histories, while admins oversee and manage all predictions and patient records. This structured approach ensures efficient data management and supports accurate heart disease prediction and record-keeping within the system.

3.1.5 Use-Case Diagram

The User in the CardioBot project is responsible for registering and logging into the system. They input their heart-related data (e.g., heart rate, blood pressure) and view their health status. They can also update their profile and track their health trends over time.

The admin manages the overall functionality of the CardioBot system. They handle user account management (creating, updating, or deleting accounts) and configure system settings (alert thresholds, data input parameters). Admins also review and manage alerts, generate basic reports on system activity, and ensure smooth operation through regular maintenance and troubleshooting.

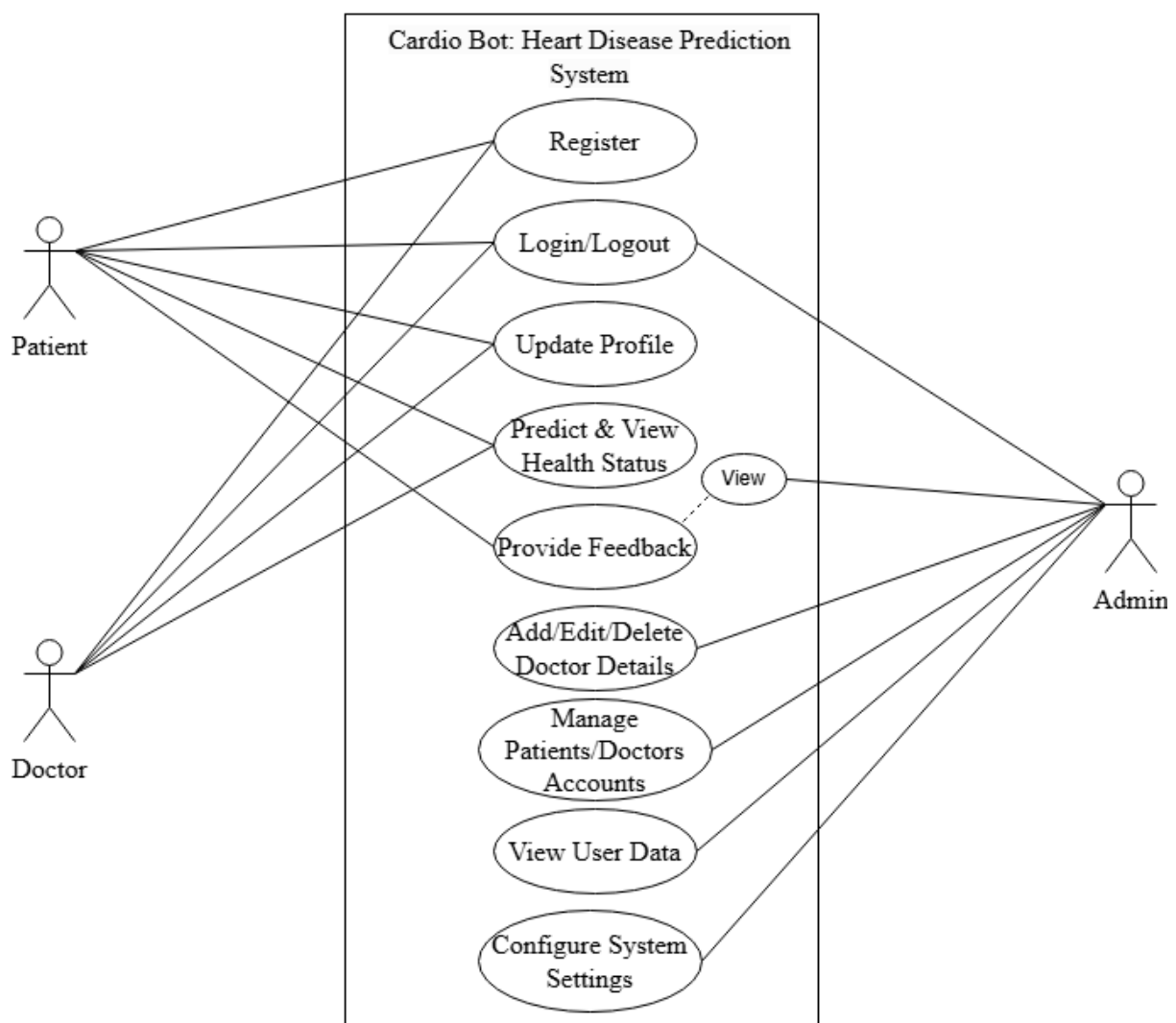


Figure 3.5: Use-Case Diagram of CardioBot

3.1.6 Flow Chart

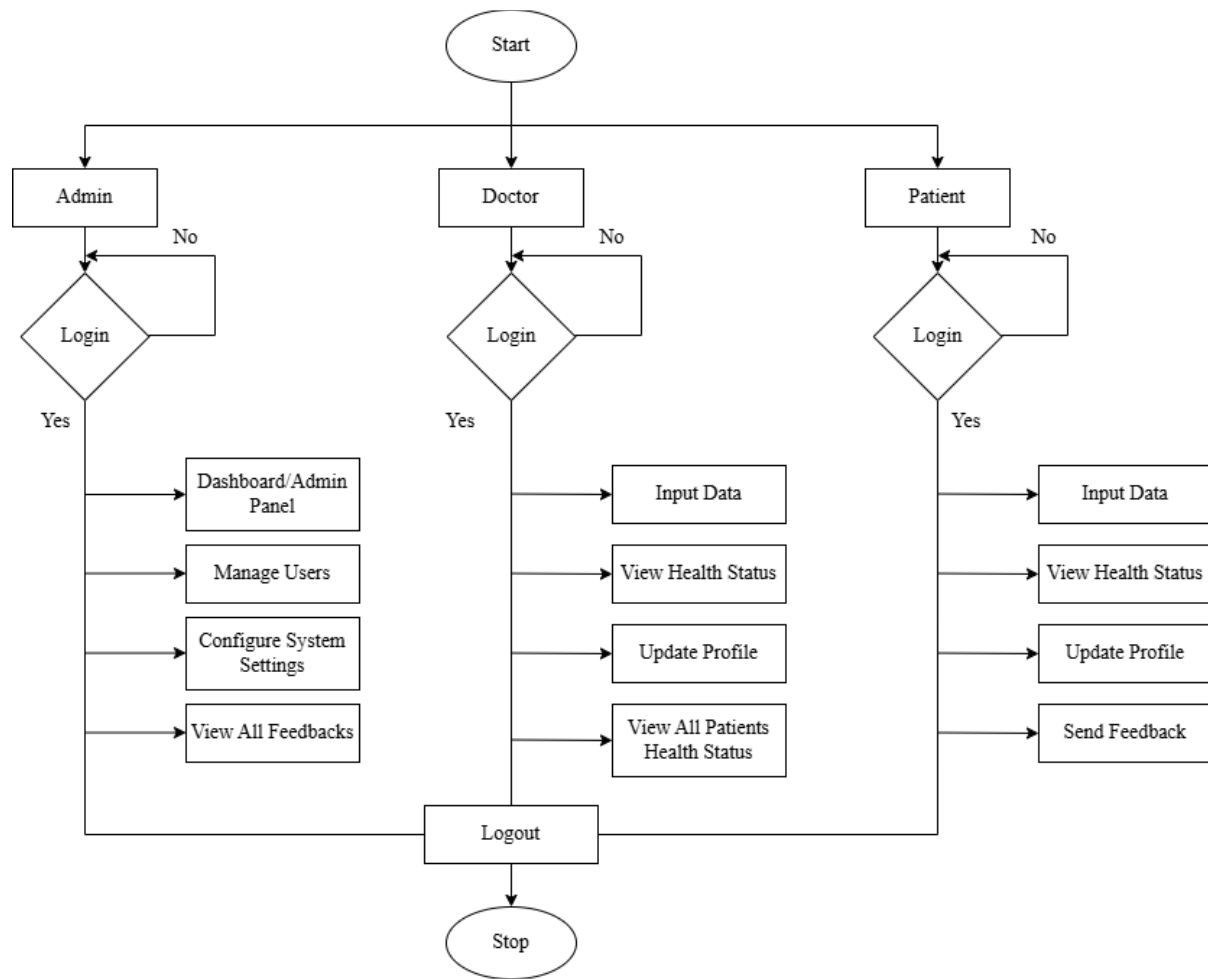


Figure 3.6: Flow Chart of CardioBot

The flowchart of the Heart Disease Prediction System starts with three user types: Admin, Doctor, and Patient. Each user must successfully log in to access their specific functionalities. After logging in, Admins can access their dashboard, manage users, configure system settings, and view all feedback. Doctors can input patient data, view health status, update their profile, and view all patients' health status. Patients can input their health data, view their health status, update their profile, and send feedback. All users have access to a logout function, which returns them to the start. This hierarchical structure ensures proper access control and functionality distribution based on user roles, maintaining the system's security and efficiency.

3.2 System Design

3.2.1 Architectural Design

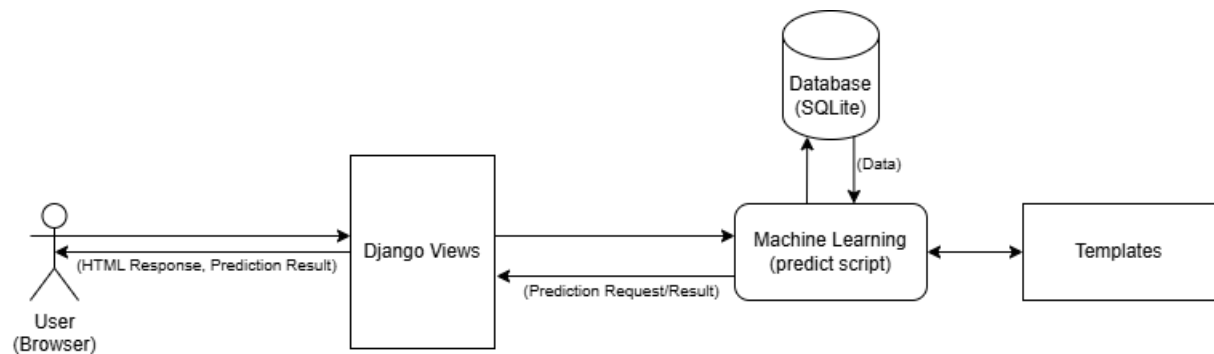


Figure 3.7: Architectural Design of CardioBot

The architectural design of this project follows a layered web application structure implemented using Django. Users interact with the system via a browser interface, submitting data through HTML forms defined in the templates (see the `health/templates/` directory). User requests are processed by Django views (e.g., in `health/views.py`), which handle business logic and communicate with the machine learning prediction script to generate results. The machine learning component accesses and updates data in the SQLite database (`db.sqlite3`) as needed. Finally, prediction results and responses are rendered back to the user through Django templates. This modular separation ensures maintainability and clear data flow throughout the system.

3.2.2 Database Design Schema

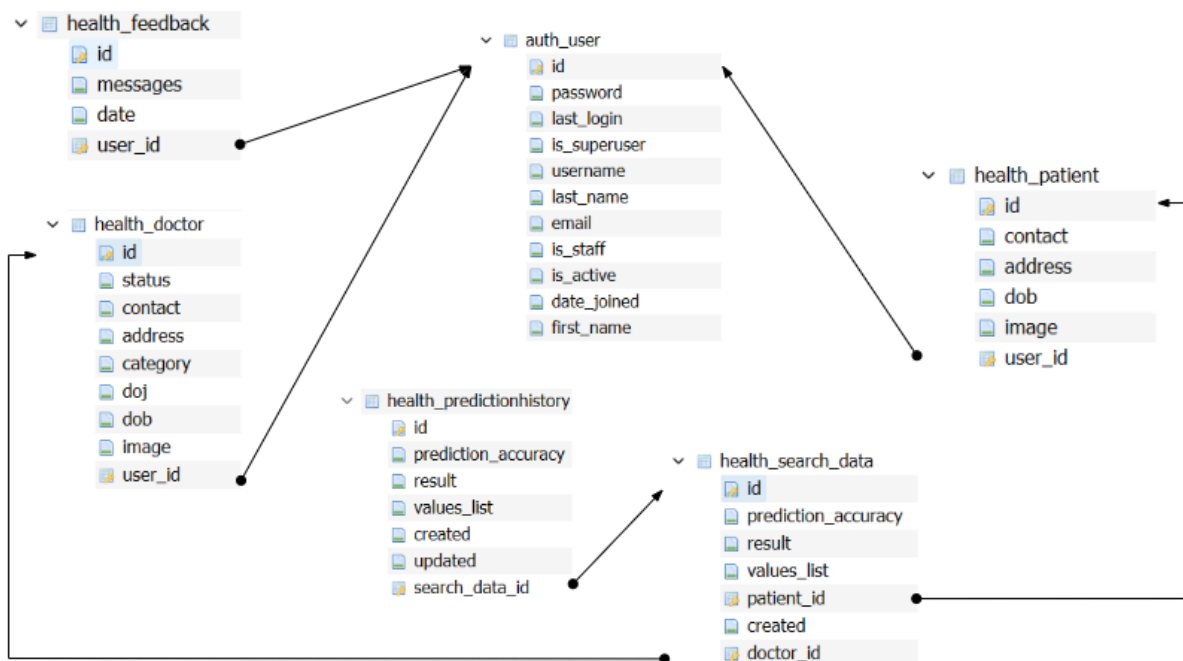


Figure 3.8: Database Schema of CardioBot

The database schema consists of several related tables that organize user and application data efficiently. The `auth_user` table stores core user account information, while `health_patient` and `health_doctor` link additional patient and doctor details to user accounts via foreign keys. The `health_feedback` table records feedback messages from users. Patient interactions and predictions are tracked through the `health_search_data` and `health_predictionhistory` tables, which store prediction results, accuracy, and related metadata. Relationships between tables are established using foreign keys, ensuring data integrity and enabling efficient queries across users, patients, doctors, and their activities. In the ER diagram, a bullet end (●) indicates a foreign key relationship, while an arrow end (→) points to the referenced primary key column (id) in the related table.

3.3 Algorithm Details

Logistic Regression Algorithm

The heart disease prediction system employs a Logistic Regression algorithm as the primary machine learning model for both doctor and patient assessments. The implementation utilizes scikit-learn's Logistic Regression class with balanced class weights to address potential dataset imbalances, ensuring fair predictions across different risk categories. The model processes 13 features for doctor predictions (including age, sex, chest pain type, blood pressure, cholesterol, fasting blood sugar, ECG results, maximum heart rate, exercise-induced angina, ST depression, slope, vessel count, and thalassemia) and 19 features for patient predictions (encompassing demographic data, lifestyle factors, medical history, and symptom severity). Prior to training, all features undergo standardization using `StandardScaler` to normalize the data and improve model convergence. The algorithm is trained on 80% of the dataset with stratified sampling to maintain class distribution, while the remaining 20% serves as the test set for performance evaluation. The model achieves interpretability through coefficient analysis, enabling the system to provide detailed feature impact explanations to users, showing how each health parameter contributes to the final risk prediction. This implementation ensures both high accuracy in heart disease risk assessment and transparency in decision-making, making it suitable for healthcare applications where explainability is crucial.

Step 1: Model Definition

Logistic Regression models the probability that the target variable $y=1$ given input features x :

$$x: [P(y=1 | x) = \sigma(z) = \frac{1}{1+e^{-z}}]$$

Where:

$$[z = w^T x + b]$$

w = weights (coefficients)

b = bias (intercept)

$\sigma(z)$ = sigmoid function

Step 2: Loss Function (Log-Loss / Binary Cross-Entropy)

The loss for a single training example:

$$[L(y, \hat{y}) = - [y \cdot \log(\hat{y}) + (1 - y) \cdot \log(1 - \hat{y})]]$$

Where:

$y \in \{0, 1\}$: true label

$\hat{y} = P(y = 1 | x)$: predicted probability

Step 3: Cost Function (Objective)

The cost function (to minimize) over all (n) samples:

$$[J(w, b) = \frac{1}{n} \sum_{i=1}^n L(y_i, \hat{y}_i)]$$

Step 4: Optimization (Gradient Descent)

Update weights using gradient descent:

$$[w := w - \alpha \cdot \frac{\partial J}{\partial w}] [b := b - \alpha \cdot \frac{\partial J}{\partial b}]$$

Where α is the learning rate.

The gradients are:

$$[\frac{\partial J}{\partial w} = \frac{1}{n} \sum_{i=1}^n L(\hat{y}_i - y_i) x_i]$$

$$[\frac{\partial J}{\partial b} = \frac{1}{n} \sum_{i=1}^n L(\hat{y}_i - y_i)]$$

Step 5: Prediction

After training, make predictions as follows:

Probability output: $[\hat{y} = \sigma(w^T x + b)]$

Class label decision: ($\hat{y} > 0.5$), predict 1; else, predict 0.

Doctor Prediction Model

Accuracy: 80.98%

Model: Logistic Regression

Trained on: 205 samples

Patient Prediction Model

Accuracy: 70.9%

Model: Logistic Regression

Trained on: 4000 samples

CHAPTER 4:

IMPLEMENTATION AND TESTING

4.1 Implementation

4.1.1 Tools Used

Frontend:

- HTML: Creates web page structure, forms for health data input, and result displays
- CSS: Manages styling, responsive design, and visual presentation of the system
- JavaScript: Handles client-side interactivity, form validation, dynamic content updates, and enhances user experience.
- Bootstrap: Framework for modern UI components and responsive layouts

Backend:

- Python: Primary programming language
- Django: Web framework for handling server-side operations, user authentication, and data processing

Database:

- SQLite: Default database for storing user data, predictions, and system information

Machine Learning:

- Pandas: Data manipulation and analysis
- NumPy: Numerical computations and array operations
- scikit-learn: Logistic Regression and StandardScaler

This tech stack ensures efficient data processing, secure user management, and accurate prediction capabilities while maintaining a user-friendly interface.

4.1.2 Implementation Details of Modules

Module 1: Patient Module

- i. **Patient Registration:** The Patient Registration module securely registers users as patients or doctors by collecting personal details, validating unique usernames and email formats, and using Django authentication for password hashing. It creates linked User and Patient records, manages profile images, and sets appropriate permissions to ensure organized, secure healthcare data management.

```

def Signup_User(request):
    error = ""
    if request.method == 'POST':
        f = request.POST['fname']
        l = request.POST['lname']
        u = request.POST['uname']
        e = request.POST['email']
        p = request.POST['pwd']
        d = request.POST['dob']
        con = request.POST['contact']
        add = request.POST['add']
        type = request.POST['type']
        im = request.FILES['image']
        dat = datetime.date.today()
        user = User.objects.create_user(email=e, username=u, password=p, first_name=f,
last_name=l)
        if type == "Patient":
            Patient.objects.create(user=user, contact=con, address=add, image=im, dob=d)
        else:
            Doctor.objects.create(dob=d, image=im, user=user, contact=con, address=add,
status=2)
        error = "create"
        d = {'error': error}
        return render(request, 'register.html', d)

```

- ii. **Patient/Doctor Login:** The Login module securely authenticates users via a username and password form, using Django's authentication system to verify encrypted credentials. Successful logins grant personalized dashboard access with role-based controls. Failed attempts show helpful error messages, ensuring both security and a user-friendly experience. Sessions are securely maintained throughout.

```

def Login_User(request):
    error = ""
    if request.method == "POST":

```

```

form_type = request.POST.get('form_type')

if form_type == "login": # Login request
    u = request.POST['uname']
    p = request.POST['pwd']
    user = authenticate(username=u, password=p)
    sign = ""
    if user:
        try:
            sign = Patient.objects.get(user=user)
        except:
            pass
        if sign:
            login(request, user)
            return redirect('patient_home')
        else:
            pure = False
            try:
                pure = Doctor.objects.get(status=1, user=user)
            except:
                pass
            if pure:
                login(request, user)
                return redirect('doctor_home')
            else:
                login(request, user)
                error = "notmember"
        else:
            error = "not"
    elif form_type == "register": # Registration request
        # ... registration logic ...
    d = {'error': error}
    return render(request, 'login.html', d)

```

- iii. Predict Heart Disease:** The system includes additional lifestyle and hereditary risk factors in the patient prediction form by asking specific questions about smoking habits, consumption of fatty foods, and family history of heart disease. Patients select their responses—such as Yes/No or frequency options—which are then incorporated into the risk prediction for a more comprehensive assessment.

```
def predict_patient_heart_disease(patient_input_data: dict):
```

```
    try:
```

```
        model_path = 'patient_model.pkl'
```

```
        metrics_path = 'patient_model_metrics.json'
```

```
        # Check if model files exist
```

```
        if not os.path.exists(model_path) or not os.path.exists(metrics_path):
```

```
            print("Patient model files not found. Training model...")
```

```
            train_patient_model()
```

```
        # Load model and metrics
```

```
        loaded_data = joblib.load(model_path)
```

```
        model = loaded_data['model']
```

```
        scaler = loaded_data['scaler']
```

```
        # Prepare input feature array (19 features)
```

```
        input_features = [float(patient_input_data[feature]) for feature in  
loaded_data['feature_names']]
```

```
        input_scaled = scaler.transform([input_features])
```

```
        prediction = model.predict(input_scaled)
```

```
        return prediction[0]
```

```
    except Exception as e:
```

```
        print("Prediction error:", e)
```

```
    return None
```

- iv. Edit Profile:** The Patient Profile Editing module lets patients update personal details, such as contact info, address, and profile image, after logging in, while keeping core

credentials like username unchanged for security. All changes are validated before saving, ensuring data integrity and protecting user privacy.

```
@login_required(login_url="login")
```

```
def Edit_My_deatail(request):
    terror = ""
    print("Hii welcome")
    user = User.objects.get(id=request.user.id)
    error = ""
    # type = Type.objects.all()
    try:
        sign = Patient.objects.get(user=user)
        error = "pat"
    except:
        sign = Doctor.objects.get(user=user)
    if request.method == 'POST':
        f = request.POST['fname']
        l = request.POST['lname']
        e = request.POST['email']
        con = request.POST['contact']
        add = request.POST['add']
        try:
            im = request.FILES['image']
            sign.image = im
            sign.save()
        except:
            pass
        to1 = datetime.date.today()
        sign.user.first_name = f
        sign.user.last_name = l
        sign.user.email = e
        sign.contact = con
        if error != "pat":
            cat = request.POST['type']
            sign.category = cat
```

```

        sign.save()
    sign.address = add
    sign.user.save()
    sign.save()
    terror = "create"
    d = {'error': error, 'terror': terror, 'doc': sign}
    return render(request, 'edit_profile.html', d)

```

- v. **Send Feedback:** The Feedback module allows patients to submit their feedback about the system's services and functionality. After logging in, users can write and submit their feedback messages, which are stored in the database with timestamps and user information. The system maintains a record of all feedback for administrative review and system improvement purposes.

```

@login_required(login_url='login')
def sent_feedback(request):
    terror = None
    if request.method == "POST":
        username = request.POST['uname']
        message = request.POST['msg']
        username = User.objects.get(username=username)
        Feedback.objects.create(user=username, messages=message)
        terror = "create"
    return render(request, 'sent_feedback.html', {'terror': terror})

```

- vi. **Prediction History:** The Prediction History module allows patients to view their complete heart disease prediction records. Each prediction entry includes the date and time of prediction, health parameters used, prediction result (Healthy/Unhealthy), and accuracy percentage. The history is displayed in chronological order with the most recent predictions first, helping patients track their heart health over time.

```

@login_required(login_url="login")
def view_prediction_history(request, search_id):
    search_data = Search_Data.objects.get(id=search_id)

```

```

history    = PredictionHistory.objects.filter(search_data=search_data).order_by('-
created')

return render(request, 'prediction_history.html', {
    'search_data': search_data,
    'history': history
})

```

Module 2: Doctor Module

- i. **Predict Heart Disease:** The Doctor Prediction module allows doctors to input 13 key health parameters, which are analyzed using a Logistic Regression to predict heart disease risk (Healthy/Unhealthy) with accuracy. Prediction results are timestamped and stored in the database for historical tracking and future reference.

```

def predict_heart_disease(list_data):
    try:
        model_path = 'heart_model.pkl'
        scaler_path = 'heart_model_scaler.pkl'
        acc_path = 'heart_model_acc.txt'

        # Check if model files exist and are valid
        if not all(os.path.exists(p) for p in [model_path, scaler_path, acc_path]):
            print("Model files not found. Retraining model...")
            model, scaler, accuracy = retrain_heart_model()
        else:
            # Load existing model and scaler
            try:
                model = joblib.load(model_path)
                scaler = joblib.load(scaler_path)
            except Exception as e:
                print("Error loading model/scaler:", e)
                model, scaler, accuracy = retrain_heart_model()

        # Preprocess input and predict
        scaled_data = scaler.transform([list_data])
        prediction = model.predict(scaled_data)

```

```

        return prediction[0]
    except Exception as e:
        print("Prediction error:", e)
    return None

```

- ii. Edit Profile:** The Doctor Profile Editing module enables doctors to securely update their professional details, including contact information, specialization, and profile image, after logging in. Using `@login_required`, it ensures only authenticated access. Changes are validated and saved to both User and Doctor models, supporting accurate, up-to-date information for effective patient-doctor communication.

```
@login_required(login_url="login")
```

```
def Edit_Doctor(request, pid):
```

```
    doc = Doctor.objects.get(id=pid)
```

```
    error = ""
```

```
    if request.method == 'POST':
```

```
        f = request.POST['fname']
```

```
        l = request.POST['lname']
```

```
        e = request.POST['email']
```

```
        con = request.POST['contact']
```

```
        add = request.POST['add']
```

```
        cat = request.POST['type']
```

```
    try:
```

```
        im = request.FILES['image']
```

```
        doc.image = im
```

```
        doc.save()
```

```
    except:
```

```
        pass
```

```
    dat = datetime.date.today()
```

```
    doc.user.first_name = f
```

```
    doc.user.last_name = l
```

```
    doc.user.email = e
```

```
    doc.contact = con
```

```
    doc.category = cat
```

```
    doc.address = add
```



```

        doc.user.save()
    doc.save()
    error = "create"
    d = {'error': error, 'doc': doc, 'type': type}
    return render(request, 'edit_doctor.html', d)

```

Module 3: Admin Module

- i. **Add Doctor:** The Add Doctor module in the admin panel lets administrators securely register new doctors by entering their details, ensuring only vetted professionals interact with patients.

```

@login_required(login_url="login")
def assign_status(request, pid):
    doctor = Doctor.objects.get(id=pid)
    if doctor.status == 1:
        doctor.status = 2
        messages.success(request, 'Selected doctor are successfully withdraw his approval.')
    else:
        doctor.status = 1
        messages.success(request, 'Selected doctor are successfully approved.')
    doctor.save()
    return redirect('view_doctor')

```

- ii. **Patient's List:** The Admin's Patient Management module provides administrators with a comprehensive view of all registered patients. The system allows admins to view, manage, and, if necessary, delete patient records through a user-friendly interface.

```

@login_required(login_url="login")
def View_Patient(request):
    patient = Patient.objects.all()
    d = {'patient': patient}
    return render(request, 'view_patient.html', d)

```

- iii. **All Predictions:** The Admin's Prediction View module allows administrators to access and monitor all heart disease predictions made within the system. The data is ordered by creation date (newest first), and admins can delete prediction records if necessary.

```
@login_required(login_url="login")
def view_search_pat(request):
    try:
        if request.user.doctor_set.all().exists() or request.user.is_staff:
            # For doctors and admins, show all predictions
            data = Search_Data.objects.all().order_by('-created')
        else:
            # For patients, show only their predictions
            patient = Patient.objects.get(user=request.user)
            data = Search_Data.objects.filter(patient=patient).order_by('-created')
    except:
        data = []

    return render(request, 'view_search_pat.html', {'data': data})
```

- iv. **Feedback Management:** The Admin's Feedback Management module provides administrators with access to all feedback submitted by users. The system allows admins to monitor user experiences, delete inappropriate feedback using the delete_feedback function, and track system improvements based on user suggestions.

```
@login_required(login_url="login")
def View_Feedback(request):
    dis = Feedback.objects.all()
    d = {'dis': dis}
    return render(request, 'view_feedback.html', d)

@login_required(login_url="login")
def delete_feedback(request,pid):
    doc = Feedback.objects.get(id=pid)
    doc.delete()
    return redirect('view_feedback')
```

4.2 Testing

4.2.1 Testing cases for Unit Testing

Testing case for unit testing of the login and logout system

Table 4.1: Test cases of unit testing for login and logout operations

Test case(s)	Steps	Expected results	Actual Result	Status
1	1. Open Git Bash 2. Run “python manage.py runserver 8000” 3. Open the browser and type “ http://127.0.0.1:8000 ”	The Home Page must be displayed.	As expected	Pass
2.	1. Select user type 2. Enter a valid email and password 3. Press the login button	Users should be redirected to the respective user type dashboard.	As expected	Pass
3.	1. Enter the valid email and password of the patient or doctor in the admin login panel	An error message, “You are not authorized as admin” should be displayed.	As expected	Pass
4.	1. Enter an invalid email or password	An error message, “Invalid username or password,” should be displayed.	As expected	Pass
5.	1. Do not fill in any field	A validation message, “Please fill out this field” should be displayed.	As expected	Pass
6.	1. Press the logout button	The user should be redirected to the home page.	As expected	Pass

4.2.2 Testing case for Patient Prediction

Table 4.2: Test cases of unit testing for patient prediction

Test case(s)	Steps	Expected results	Actual Result	Status
1	1. Log in using patient credentials	The user should be redirected to the patient's home page.	As expected	Pass
2.	1. Fill in the form using healthy data sets 2. Submit the form	It should display “Healthy” in a green colored font.	Healthy	Pass
3.	1. Click the “View History” button present on the home page.	The user should be able to view all their old predictions and their recent prediction.	As expected	Pass

4.2.3 Testing case for Doctor Prediction

Table 4.3: Test cases of unit testing for doctor prediction

Test case(s)	Steps	Expected results	Actual Result	Status
1	1. Log in using doctor credentials	The user should be redirected to the doctor's home page.	As expected	Pass
2.	1. Fill in the form using healthy data sets 2. Submit the form	It should display “Low Risk.	As expected	Pass
3.	1. Fill in the form using unhealthy data sets 2. Submit the form	It should display “High Risk”.	As expected	Pass
4.	1. Click the “View Predictions” button present on the home page.	The user should be able to view all the predictions and their recent prediction.	As expected	Pass

4.2.4 Testing case for System Testing (Admin)

Table 4.4: Test cases of system testing for the admin dashboard

Test case(s)	Steps	Expected results	Actual Result	Status
1	1. Log in using admin credentials	The user should be redirected to the admin's home page.	As expected	Pass
2.	1. Click on "Total Doctors"	The user should be able to view, edit, and delete the doctor's information.	As expected	Pass
3.	1. Click on "Total Patients"	The user should be able to view and delete the patient's information.	As expected	Pass
4.	1. Click on "Total Feedback"	The user should be able to view and delete the feedback.	As expected	Pass
5.	1. Click on "Prediction History"	The user should be able to view and delete the predictions.	As expected	Pass

CHAPTER 5:

CONCLUSION AND FUTURE RECOMMENDATIONS

5.1 Lesson Learnt

Through this project, I learned the importance of clean data preprocessing, clear role-based access control, and user-friendly design. Implementing and evaluating a machine learning model like Logistic Regression was crucial. Effective debugging and thorough documentation were key to building a reliable, secure, and maintainable heart disease prediction system.

5.2 Conclusion

For the CardioBot project, the goal is to create a system that helps predict heart disease risks by analyzing key health data like age, cholesterol levels, blood pressure, and heart rate. Using machine learning algorithms, the system will offer personalized insights and recommendations to users. It's designed to be easy to use, making health monitoring accessible to anyone who wants to take control of their well-being.

This project is an opportunity to refine skills in programming, quality assurance, and reporting while providing a tool that could make a real difference in people's lives.

5.3 Future Recommendations

For future improvements, I recommend integrating more advanced machine learning models and regularly updating them with new data to enhance accuracy. Adding real-time health monitoring, mobile app support, and multilingual interfaces would improve accessibility. Implementing a secure chat box feature for direct communication between patients and doctors, along with strengthened data privacy, detailed analytics dashboards, and automated alerts for high-risk patients, can further boost the system's impact and reliability in clinical settings.

REFERENCES

- [1] Asha Rajkumar, and Mrs. G. SophiaReena, “Diagnosis of Heart Disease Using Data Mining Algorithm”, Global Journal of Computer Science and Technology, Vol. 10, pp. 38- 43, Sept. 2010.
- [2] G.Subbalakshmi, K. Ramesh, and M.C. Rao, “Decision Support in Heart Disease Prediction System using Naive Bayes”, ndian Journal of omputer Science and Engineering (IJCSE), Vol. 2, No. 2, Apr-May 2011.
- [3] Abhishek Taneja, “Heart Disease Prediction System Using Data Mining Techniques”, Oriental Journal Of Computer Science and Technology, Vol. 6, pp. 457-466, Dec.2013.
- [4] Hlaudi Daniel Masethe, and Mosima Anna Masethe, “Prediction of Heart Disease Using Classification Algorithms”, in Proceedings of the World Congress on Engineering and Computer Science 2014 Vol. II WCECS 2014, 22-24 Oct. 2014, San Francisco, USA.
- [5] K. Gomath, Dr. Shanmugapriyaa, “Heart Disease Prediction Using Data Mining Classification”, International Journal for Research in Applied Science & Engineering Technology (IJRASET), Vol.4, Issue 2, February-2016.
- [6] “Heart Disease Dataset”, <https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset?resource=download>

APPENDICES

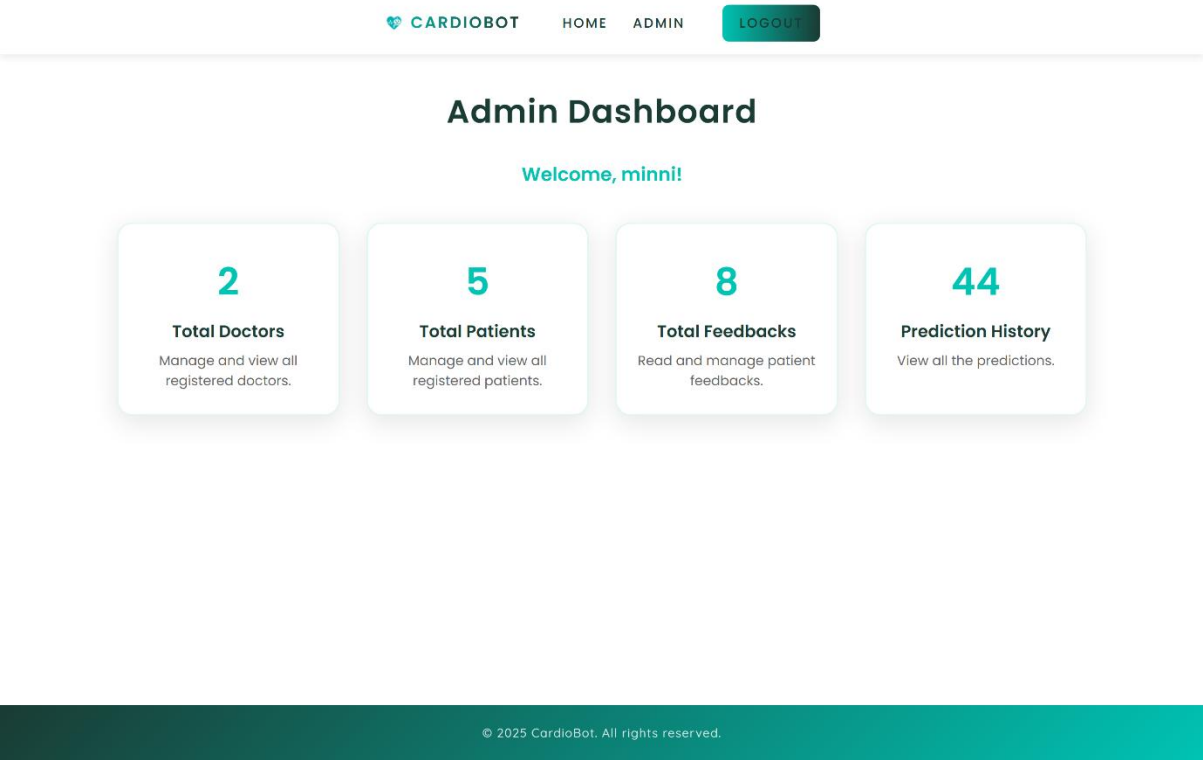


Fig: Admin Home Page

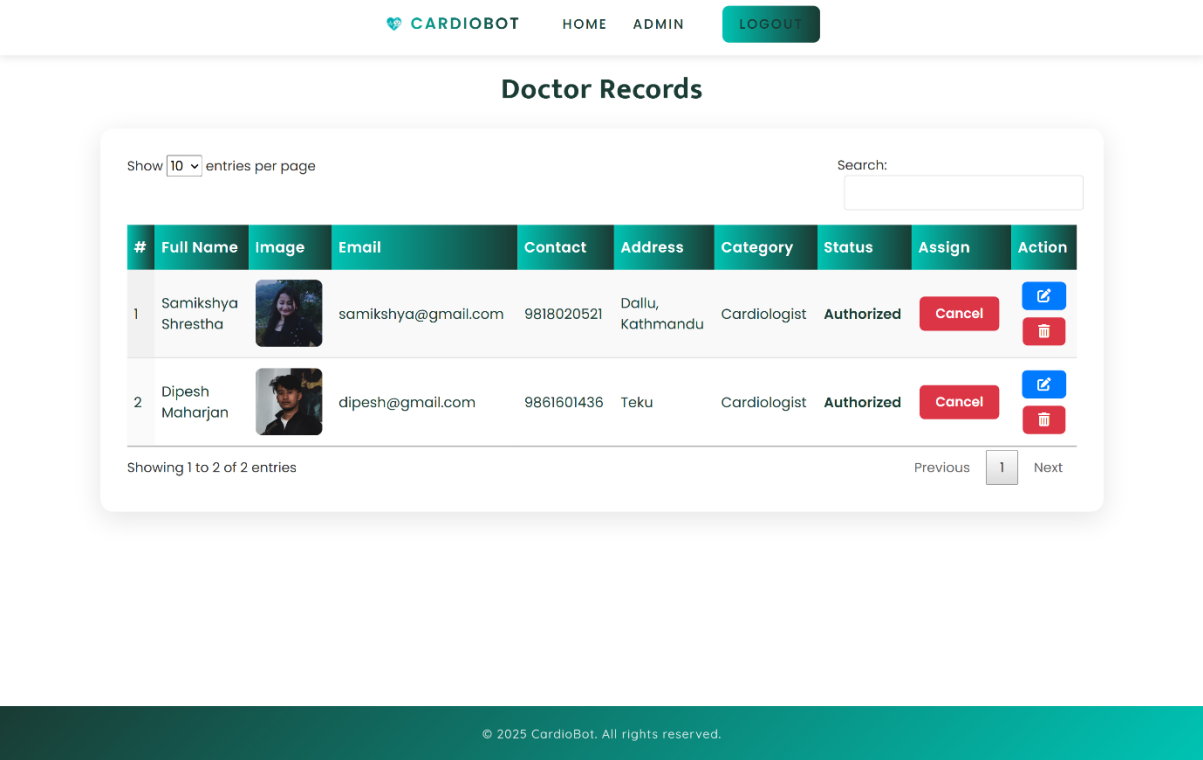










Fig: Doctor Records

Patient Details

Show 10 entries per page

Search:

#	Full Name	Image	Email	Contact	Address	Action
1	Sichu Maharjan		sichu@gmail.com	9861465033	Dekwo Galli, Kathmandu	
2	Shubekshya Lama		subu@gmail.com	9803326740	Tokha	
3	Rusha Manandhar		rusha@gmail.com	9847398065	Dallu Awas	
4	Kristi Prajapati		kristi@gmail.com	9864333083	Jyatha, Kathmandu	

Showing 1 to 4 of 4 entries

Previous 1 Next





© 2025 CardioBot. All rights reserved.

Fig: Patient Details

Feedback Records

Show 10 entries per page

Search:

#	User Info	Email	Contact	Date	Message	Action
1	Samikshya Shrestha	samikshya@gmail.com	9818020521	Jul 29, 2025	hello	
2	kristi	kristi@gmail.com	9864333083	Jul 29, 2025	feedback test 123	
3	Kristi Prajapati	kristi@gmail.com	9864333083	Jul 29, 2025	patient contact test	
4	Dipshika Pandit	pandit@gmail.com	-	Jul 29, 2025	not logged in	

Showing 1 to 4 of 4 entries

Previous 1 Next

© 2025 CardioBot. All rights reserved.

Fig: Feedback Records

Prediction History

Show entries per page

Search:

#	Date	Patient Name	Patient Contact	Result	Action
1	Jul 29, 2025 10:44	Kristi Prajapati	9864333083	Healthy	History Delete
2	Jul 29, 2025 10:20	Kristi Prajapati	9864333083	Unhealthy	History Delete
3	Jul 29, 2025 09:25	Rameshwor KC	9841001122	Unhealthy	History Delete
4	Jul 28, 2025 11:18	Sichu Maharjan	9861465033	Healthy	History Delete
5	Jul 28, 2025 11:17	Dipesh Maharjan	9861601436	Healthy	History Delete
6	Jun 11, 2025 17:39	Rajni Kant	9860123456	Unhealthy	History Delete
7	Jun 11, 2025 17:37	Anil Kapoor	9841002211	Unhealthy	History Delete
8	Jun 11, 2025 17:36	Bibekananda Pun	9818020555	Unhealthy	History Delete
9	Jun 11, 2025 15:11	Raju Kathewada	9827465033	Unhealthy	History Delete
10	Jun 11, 2025 14:42	Ram Prasad Devkota	9841987654	Unhealthy	History Delete

Showing 1 to 10 of 11 entries

[Previous](#) [1](#) [2](#) [Next](#)

Fig: Prediction History

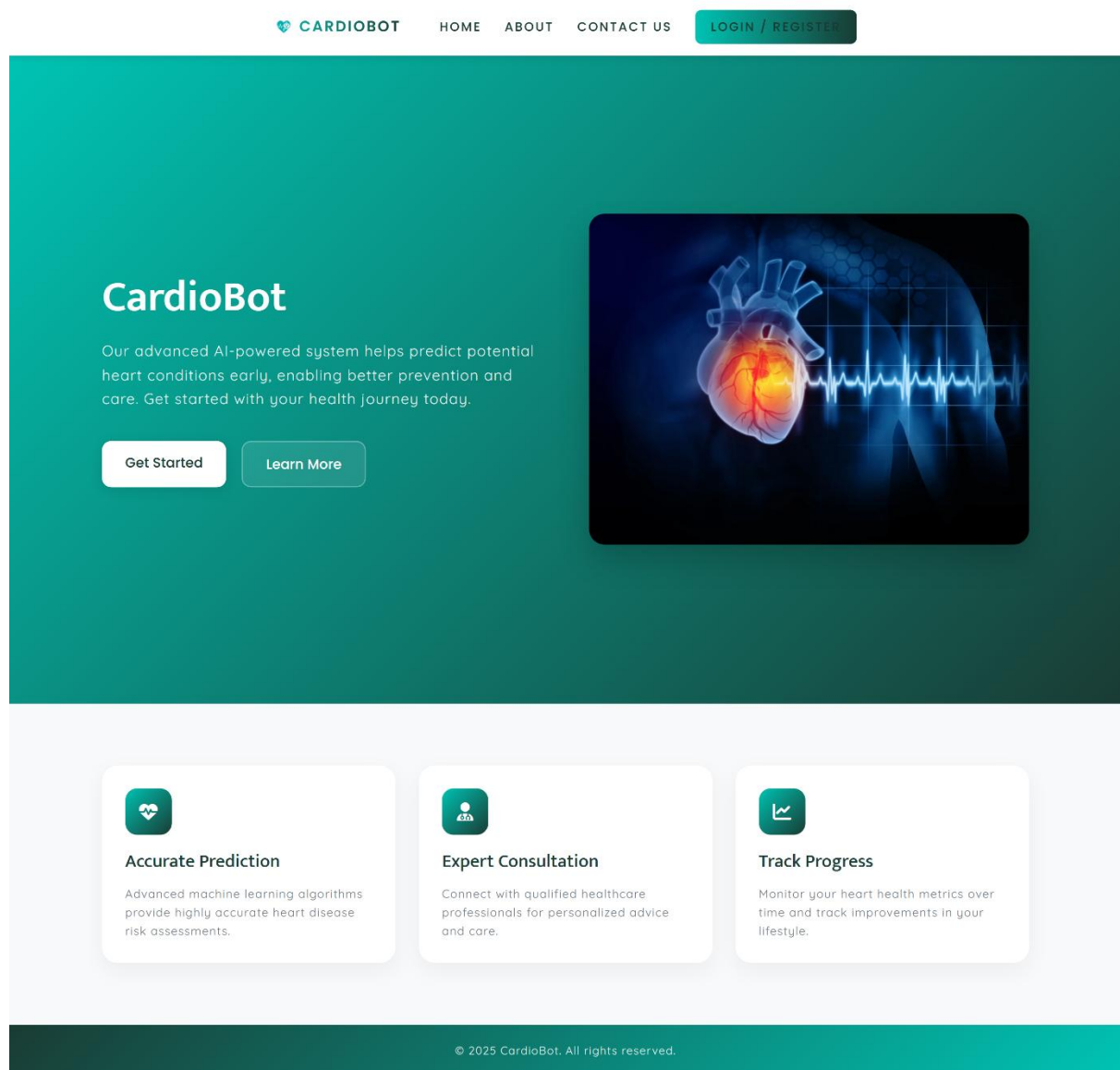


Fig: Home Page

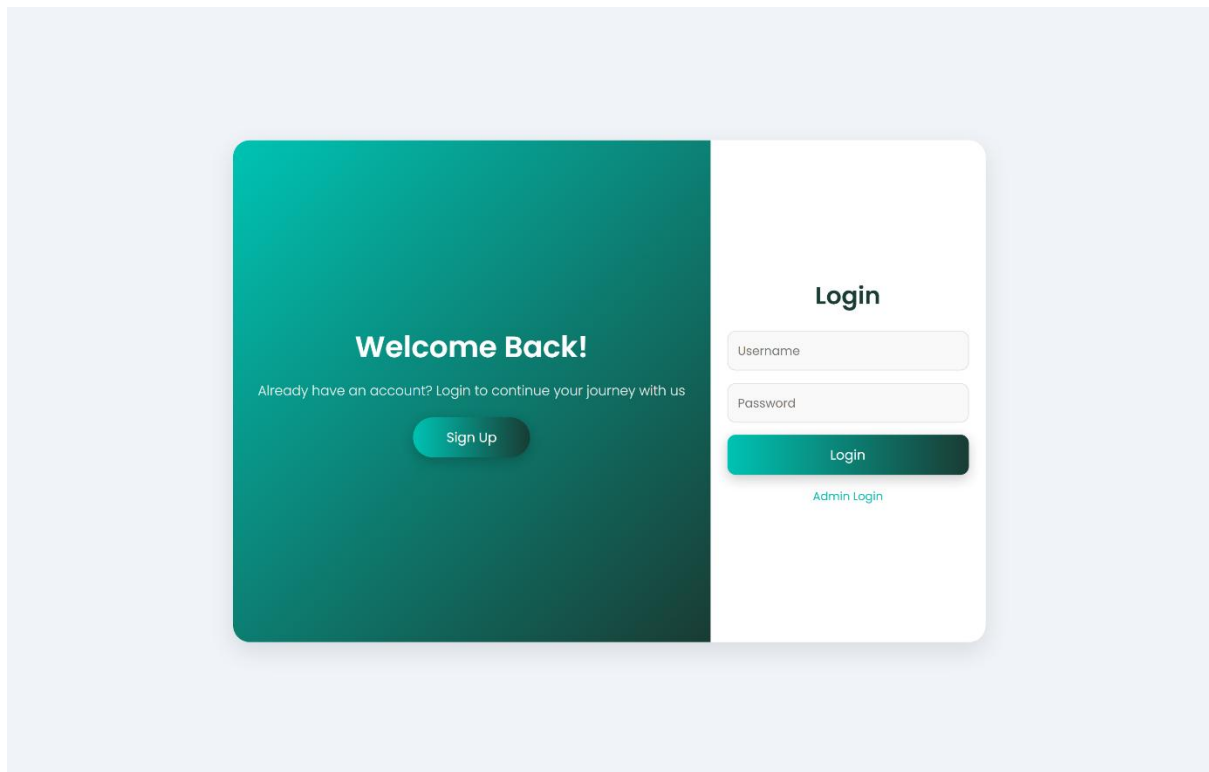


Fig: User Log in Page

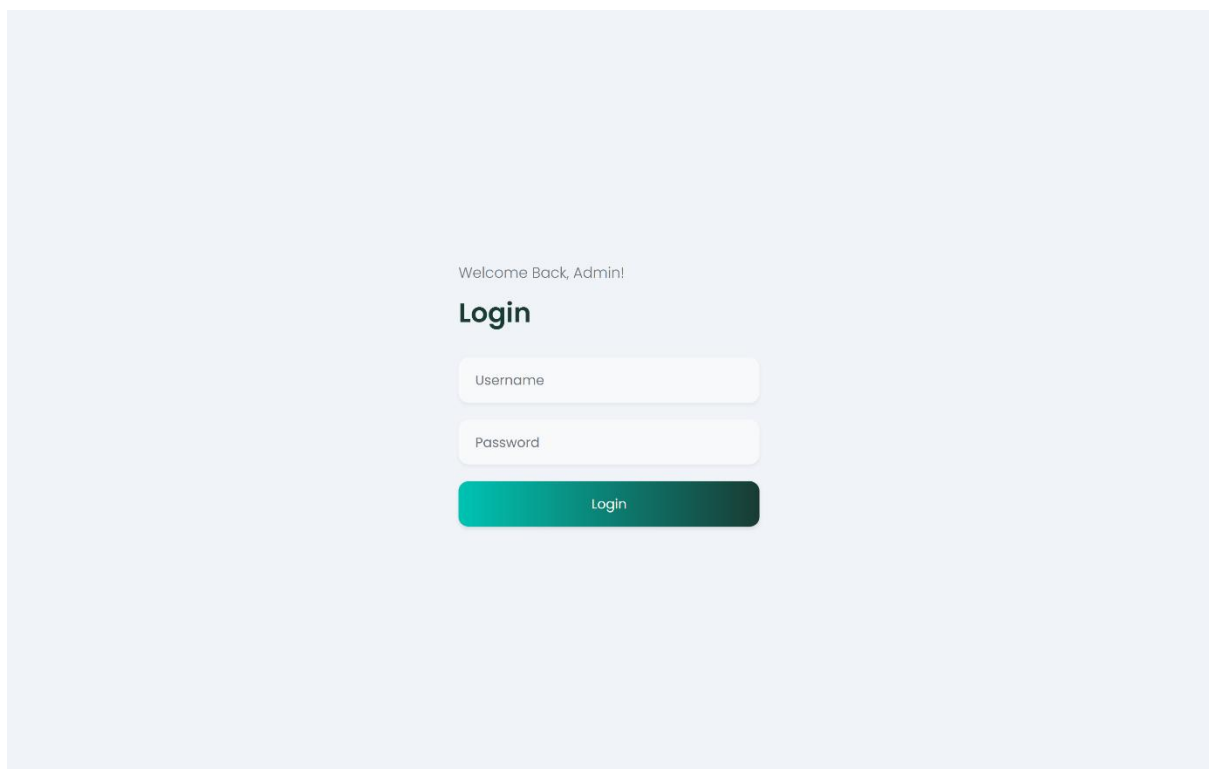
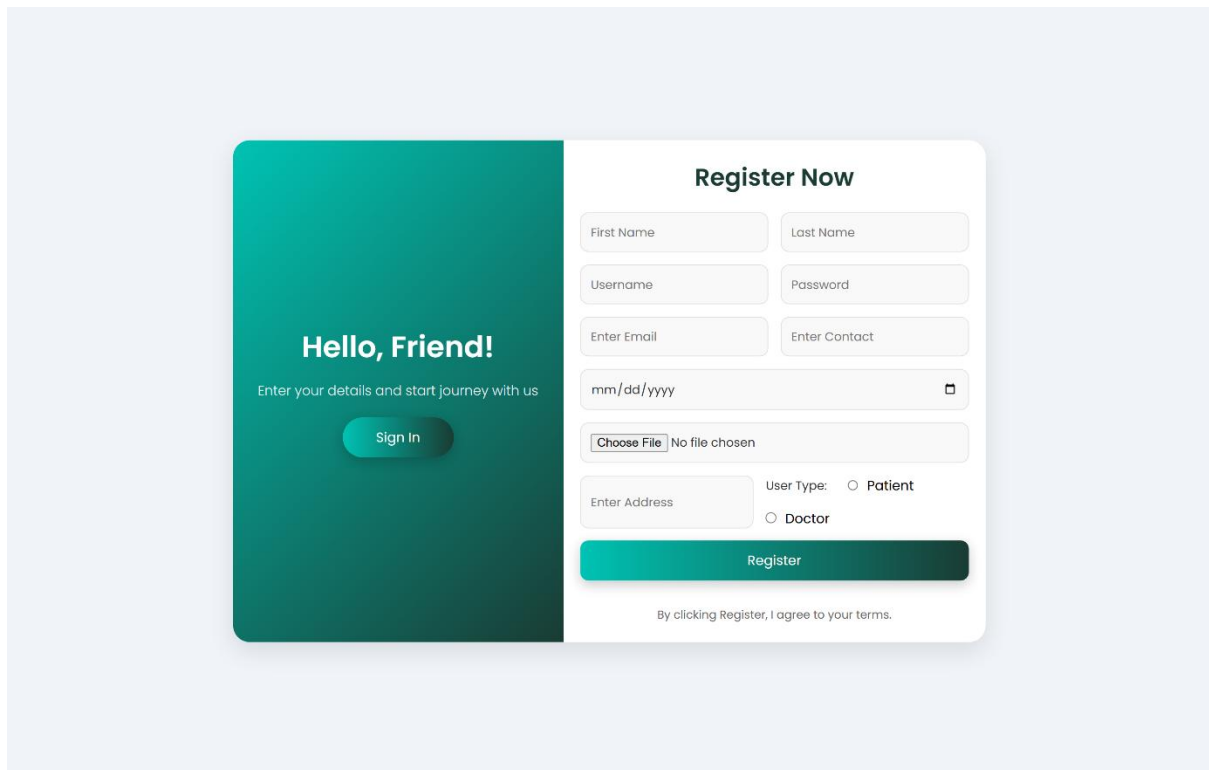
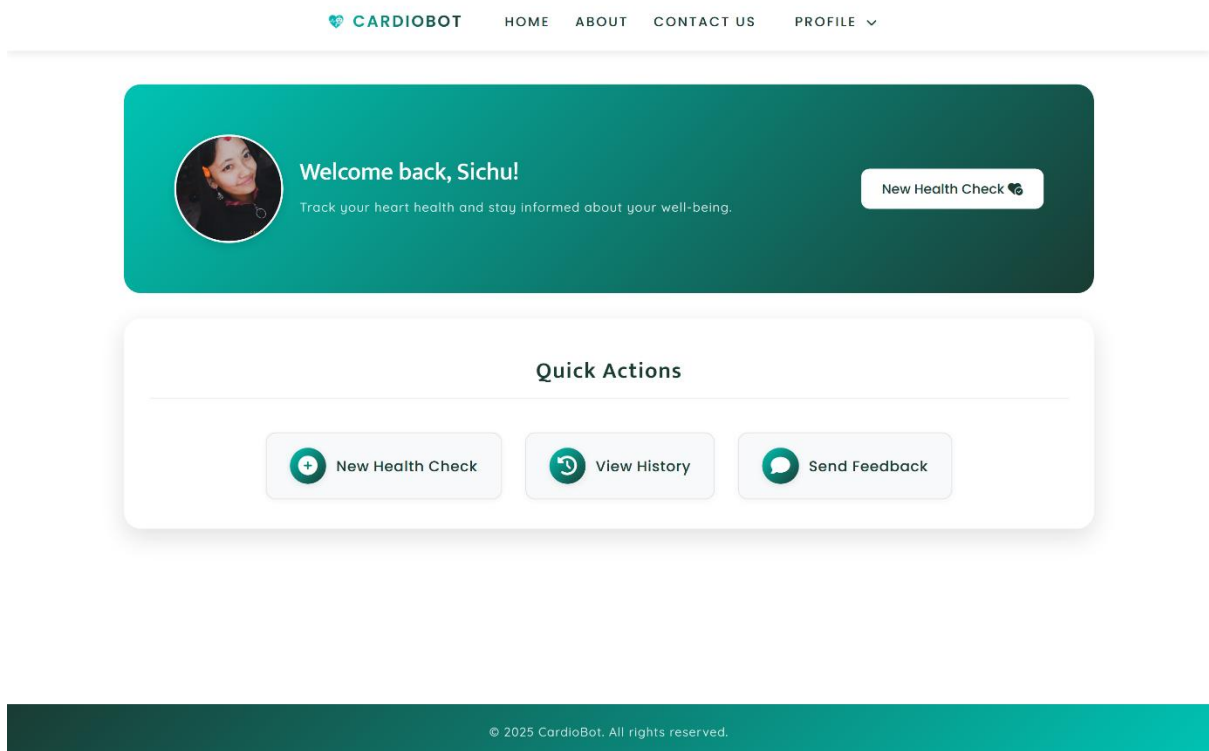


Fig: Admin Log in Page



The registration page features a teal gradient sidebar on the left with the text "Hello, Friend!" and "Enter your details and start journey with us" above a "Sign In" button. The main white area is titled "Register Now" and contains a form with the following fields: First Name, Last Name, Username, Password, Enter Email, Enter Contact, a date picker (mm/dd/yyyy), a file upload button (Choose File) with the text "No file chosen", an address field (Enter Address), and a "User Type" section with radio buttons for "Patient" and "Doctor". A teal "Register" button is at the bottom of the form, with a small disclaimer below it: "By clicking Register, I agree to your terms."

Fig: User Registration Page



The patient home page has a teal header with the "CARDIOBOT" logo and navigation links: HOME, ABOUT, CONTACT US, and PROFILE. Below the header is a teal banner with a circular profile picture of a woman, the text "Welcome back, Sichu!", a subtitle "Track your heart health and stay informed about your well-being.", and a "New Health Check" button. Underneath is a white "Quick Actions" section with three buttons: "New Health Check" (with a plus icon), "View History" (with a clock icon), and "Send Feedback" (with a speech bubble icon). The footer is a teal bar with the copyright notice "© 2025 CardioBot. All rights reserved."

Fig: Patient Home Page

Heart Health Assessment

Please fill out this form to help us assess your heart health risk factors

Personal Information

Full Name

Sichu Maharjan

Age

21

Sex

Select gender

Physical Measurements

Height (cm)

Your height in centimeters

Weight (kg)

Your weight in kilograms

Health History

Do you smoke?

Select option

How long have you been smoking?

In years

How often do you smoke in a day?

In cigarettes

Do you have high blood pressure?

Select option

Do you have diabetes?

Select option

Do you have high cholesterol?

Select option

Family History

Has any close family member had heart disease?

Select option

Parents, siblings, or grandparents

Current Symptoms

Have you experienced chest pain or discomfort recently?

Select frequency

What was the severity of the chest pain?

Select severity

Do you feel short of breath during light activity?

Select frequency

How long do you feel short of breath?

Select minutes

Like walking or climbing stairs

Lifestyle Habits

How often do you exercise?

Select frequency

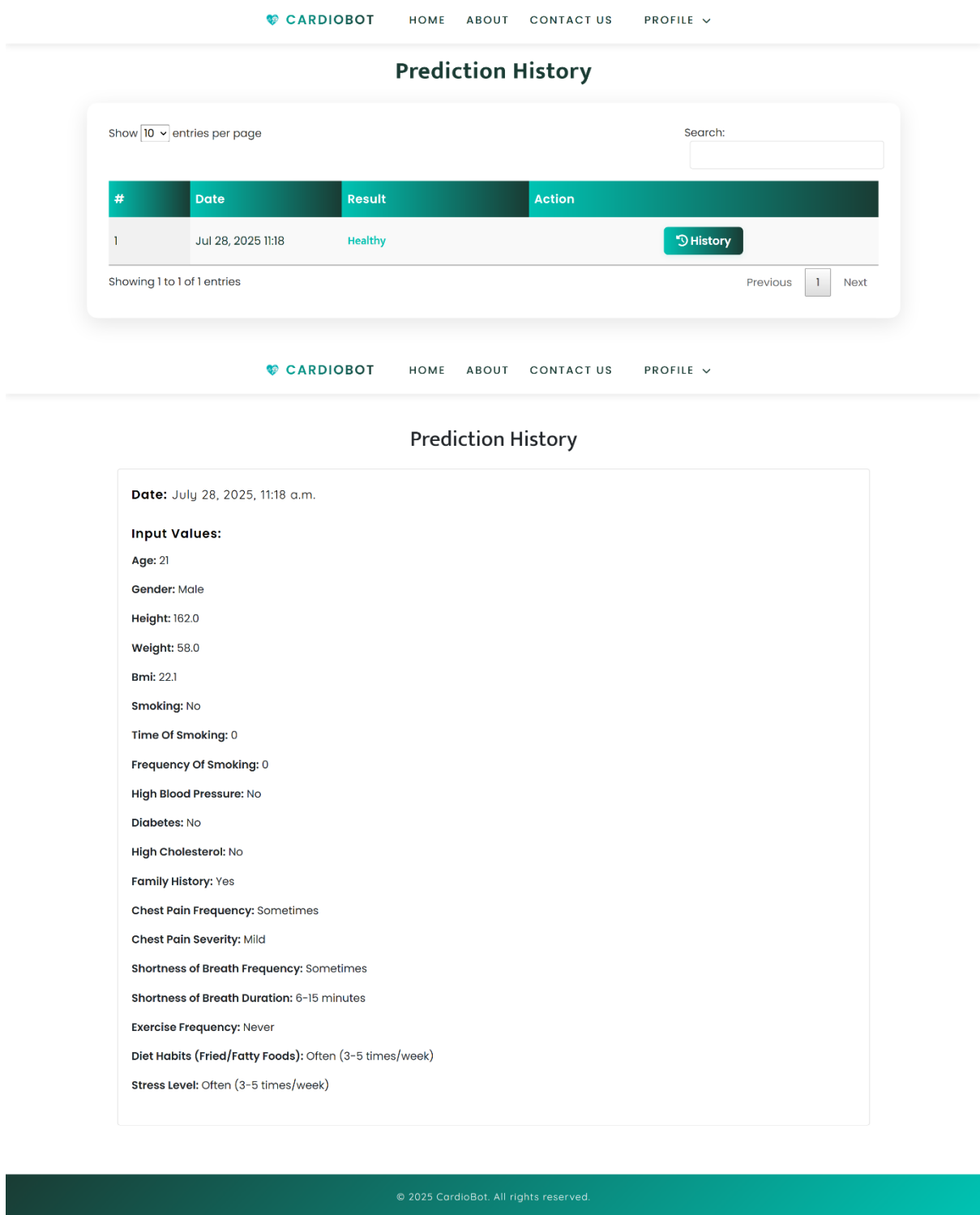
How often do you eat fried or fatty foods?

Select frequency

How often do you feel stressed?

Select frequency

Submit Health Assessment



CARDIOBOT

HOME

ABOUT

CONTACT US

PROFILE

Send Feedback

Username

sicheyyyy

Automatically filled with your account name

Write Message

Your feedback helps us improve our services. Please be specific and constructive.

SEND FEEDBACK

© 2025 CardioBot. All rights reserved.

Fig: Feedback Page

CARDIOBOT

HOME

ABOUT

CONTACT US

PROFILE

My Profile

Full Name

Sichu Maharjan

Email

sichu@gmail.com

Contact

9861465033

City

Dekwo Galli, Kathmandu

Update Profile

Logout

© 2025 CardioBot. All rights reserved.

Fig: Patient Profile Page

CARDIOBOT

HOMEABOUTCONTACT USPROFILE

Update My Detail

First Name

Sichu

Last Name

Maharjan

Email

sichu@gmail.com

Contact

9861465033

City

Dekwo Galli, Kathmandu

Image

Choose File

N...en

Current Photo

Update Detail

© 2025 CardioBot. All rights reserved.

Fig: Patient Profile Editing Page

CARDIOBOT

HOMEABOUTCONTACT USPROFILE

Welcome back, Dr. Samikshya!

Hope you're having a healthy day

New Health Check

View Patient Records

Doctor Mode

Prediction Accuracy

80.98%

Our Logistic Regression model for doctor mode provides reliable predictions.

© 2025 CardioBot. All rights reserved.

Fig: Doctor Home Page

Add Heart Detail

Please fill in all fields carefully. The accuracy of the prediction depends on the accuracy of your input data.

Patient Information

Patient Name

Patient Contact

10-digit phone number

Basic Information

Age (in years)

Patient's age in years

Sex

Patient's biological sex

Vital Signs

Resting Blood Pressure (mm Hg)

Blood pressure while resting (systolic)

Cholesterol (mg/dl)

Serum cholesterol in mg/dl

Fasting Blood Sugar

Blood sugar level after fasting

Max Heart Rate

Maximum heart rate achieved during exercise

Cardiac Assessment

Chest Pain Type

Type of chest pain experienced

Resting ECG

ECG results at rest

Exercise Induced Angina

Angina induced by exercise

ST Depression (Oldpeak)

ST depression induced by exercise relative to rest

Advanced Assessment

Slope

Slope of peak exercise ST segment

Number of Vessels

Number of major vessels colored by fluoroscopy

Thalassemia

Blood disorder type

Additional Notes

Clinical Notes

Optional: Add any relevant clinical notes or observations

Submit Assessment

Fig: Doctor Prediction Page

Prediction History

Show 10 entries per page

Search:

#	Date	Patient Name	Patient Contact	Result	Action
1	Jul 29, 2025 09:25	Rameshwar KC	9841001122	Unhealthy	History
2	Jul 28, 2025 11:17	Dipesh Maharjan	9861601436	Healthy	History
3	Jun 11, 2025 17:39	Rajni Kant	9860123456	Unhealthy	History
4	Jun 11, 2025 17:37	Anil Kapoor	9841002211	Unhealthy	History
5	Jun 11, 2025 17:36	Bibekananda Pun	9818020555	Unhealthy	History
6	Jun 11, 2025 15:11	Raju Kathewada	9827465033	Unhealthy	History
7	Jun 11, 2025 14:42	Ram Prasad Devkota	9841987654	Unhealthy	History
8	Jun 11, 2025 14:38	Priya Sthapit	9810031424	Healthy	History

Showing 1 to 8 of 8 entries

Previous 1 Next

Fig: Doctor Prediction History



My Profile

Full Name Samikshya Shrestha
Email samikshya@gmail.com
Contact 9818020521
City Dallu, Kathmandu
Specialist Cardiologist

Update Profile

Logout

Fig: Doctor Profile Page

CARDIOBOT

HOMEABOUTCONTACT USPROFILE

Update My Detail

First Name

Samikshya

Last Name

Shrestha

Email

samikshya@gmail.com

Contact

9818020521

Specialist

Cardiologist


City

Dallu, Kathmandu

Image

Choose FileN...en

Current Photo



Update Detail

© 2025 CardioBot. All rights reserved.

Fig: Doctor Profile Editing Page