

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JNANA SANGAMA”, BELAGAVI – 590 018



PROJECT REPORT ON

**“MULTIPLE DISEASE PREDICTION SYSTEM USING
MACHINE LEARNING ”**

Submitted in partial fulfilment of the requirement

for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted By

NISARGA M J : 4GH20CS031

RAKSHITHA M N : 4GH20CS042

RAVI VITTHAL TENGALE : 4GH20CS044

VARSHITHA J GOWDA : 4GH20CS055

Under the Guidance of

Dr. Vani V G, BE, M.Tech, PhD

HoD, Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GOVERNMENT ENGINEERING COLLEGE, HASSAN - 573201

2023-24

GOVERNMENT ENGINEERING COLLEGE HASSAN – 573 201

Affiliated to VTU, Belagavi & Approved by AICTE, New Delhi

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



CERTIFICATE

Certified that the **Project work** entitled “**MULTIPLE DISEASE PREDICTION SYSTEM USING MACHINE LEARNING**” carried out by **Ms. Nisarga M J (4GH20CS031)**, **Ms. Rakshitha M N (4GH20CS042)**, **Mr. Ravi Vitthal Tengale (4GH20CS044)** and **Ms. Varshitha J Gowda (4GH20CS055)** a bonafide students of **Government Engineering College Hassan** in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the **Visvesvaraya Technological University, Belagavi** during the year 2023-2024. It is certified that all correction suggestions indicated during internal evaluation has been incorporated in the report. The Project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Dr. Vani V G
Guide & HoD, Dept of CSE

Prof. Kiran M P
Project Co-Ordinator

Dr. Girish D P
Principal

Name of the examiners

1.
2.

Signature with date

DECLARATION

We, **NISARGA M J, RAKSHITHA M N, RAVI VITTHAL TENGALE, VARSHITHA J GOWDA** students of Eighth semester B.E, **GOVERNMENT ENGINEERING COLLEGE, HASSAN** bearing USN **4GH20CS031, 4GH20CS42, 4GH20CS044** and **4GH20CS055** respectively, hereby declare that the Project entitled “**MULTIPLE DISEASE PREDICTION SYSTEM USING MACHINE LEARNING** ” has been carried out by me under the supervision of our Guide, **Dr. Vani V G** BE., M. Tech, Ph.D Department of CS&E, GEC Hassan, have submitted in partial fulfilment of the requirements for the award of the Degree of B.E in CS&E by the Visvesvaraya Technological University, Belagavi during the academic year 2023- 2024. This report has not been submitted to any other Organization/University for the award of degree or certificate.

Date:

Place: Hassan

Project Associates

NISARGA M J

RAKSHITHA M N

RAVI VITTHAL TENGALE

VARSHITHA J GOWDA

ACKNOWLEDGEMENT

We consider it a privilege to whole-heartedly express our gratitude and respect to each and every one who guided and helped us in the successful completion of this Project Report.

We very thankful to the Principal **Dr. D P Girish**, for being kind enough to provide me an opportunity to work on a project in this institution.

We also thankful to **Dr. Vani V G**, HOD, Department of Computer Science, for her co-operation and encouragement at all moments of our approach. as Project Guide for her ideas and co-operation showed on us during our venture and making this Project as a great success.

I am also thankful to **Prof. Kiran M P**, Project Co-Ordinator, Department of Computer Science and Engineering for the Co-Operations and encouragement at all the moments of our approach.

We would also like to thank our parents and well-wishers as well as our dear classmates for their guidance and their kind co-operation.

Finally, it is our pleasure and happiness to the friendly co-operation showed by all the staff members of Computer Science Department, GECH.

Project Associates

Nisarga M J (4GH20CS031)

Rakshitha M N (4GH20CS042)

Ravi Vitthal Tengale (4GH20CS044)

Varshitha J Gowda (4GH20CS055)

TABLE OF CONTENTS

Declaration	i
Acknowledgement	ii
Table Of Content	iii
List Of Figures	v
Abstract	vii
1. INTRODUCTION	1 - 2
1.1 Project Overview	1
1.2 Scope	1
1.3 Challenges	2
1.4 Objectives	2
2. LITERATURE SURVEY	3 – 4
2.1 Literature Review on Heart By Ashir Javeed, Shijie Zhou et al. in 2021	3
2.2 Literature Review on Diabetes By Roshni Saxena, Sanjay Kumar Sharma in 2022	3
2.3 Literature Review on Liver Disease by Bendi Venkata Ramana in 2021	4
3. REQUIREMENT SPECIFICATION	5 – 6
3.1 Functional Requirements	5
3.2 Non Functional Requirements	5
3.3 Hardware Requirements	6
3.4 Software Requirements	6

4. SYSTEM DESIGN	7 – 9
4.1 Anaconda Navigator	7
4.2 Jupyter Notebook	7
4.3 Django	8
4.4 Python	9
5. ARCHITECTURE	10 – 19
5.1 System Architecture	10
5.2 Flowchart of the System	11
5.3 Methodology	13
5.4 ER Diagram	16
5.5 Use Case Diagram	17
5.6 Activity Diagram	18
6. IMPLEMENTATION	20 – 28
6.1 Pseudo Code	20
6.2 Data Collection	22
6.3 Data Preprocessing	23
6.4 Data Analysis and Visualization	24
6.5 Disease Prediction Model	25
7. RESULTS	29-34
7.1 User Module	30
7.2 Admin Module	33
8. CONCLUSION	35 – 36
8.1 Conclusion	35
8.2 Future Scope	36
REFERENCES	37

LIST OF FIGURES

5.1 System Architecture	10
5.2.1 Flowchart of the System	11
5.2.2 Flowchart of the web interface	12
5.4 ER Diagram	16
5.5 Use case Diagram	17
5.6.1 Activity Diagram of User	18
5.6.2 Activity Diagram of Admin	19
6.2.1 Dataset of Liver Disease	22
6.2.2 Dataset of Heart Disease	22
6.2.3 Dataset of Diabetes Disease	23
6.3.1 Summary of the dataset	23
6.3.2 Number of unique values in dataset	24
6.5.1 Accuracy performance of Diabetes disease	25
6.5.2 Accuracy performance of Heart disease	25
6.5.3 Accuracy performance of Liver disease	26
6.6.1: Python code to send a request on JSON	27
6.6.2: Code for API URL	27
6.6.3: Executing web-app on Django local server	28
6.6.4: User Interface Code	28
7.a: Diabetes Disease	29
7.b: Heart Disease	29
7.c: Liver Disease	29
7.1: Home Page	30

7.2: About Us page	30
7.3: User Home page	30
7.4: User Registration Form	31
7.5: User Prediction page	31
7.6: Liver Disease Input Data	31
7.7: Liver Disease Output Result	32
7.8: Heart Disease Output Result	32
7.9 Diabetes Disease Output Result	32
7.10: User Hospital View Page	32
7.11: User Chatbot	33
7.12: User Feedback View Page	33
7.13: Admin Home page	33
7.14: Admin Hospital View Page	34
7.15: Admin User View Page	34
7.16: Admin Feedback View Page	34

ABSTRACT

Machine learning and Artificial Intelligence are playing a huge role in today's world. From self-driving cars to medical fields, we can find them everywhere. The medical industry generates a huge amount of patient data which can be processed in a lot of ways. So, with the help of machine learning, we have created a Prediction System that can detect more than one disease at a time. Many of the existing systems can predict only one disease at a time and that too with lower accuracy. Lower accuracy can seriously put a patient's health in danger. We have considered three diseases for now that are Heart, Liver, and Diabetes and in the future, many more diseases can be added. The user has to enter various parameters of the disease and the system would display the output whether he/she has the disease or not. This project can help a lot of people as one can monitor the persons' condition and take the necessary precautions thus increasing the life expectancy.

CHAPTER 1

INTRODUCTION

In this digital world, data is an asset, and enormous data was generated in all the fields. Data in the healthcare industry consists of all the information related to patients. Here a general architecture has been proposed for predicting the disease in the healthcare industry. Many of the existing models are concentrating on one disease per analysis. Like one analysis for diabetes analysis, one for cancer analysis, one for skin diseases like that. There is no common system present that can analyze more than one disease at a time. Thus, we are concentrating on providing immediate and accurate disease predictions to the users about the symptoms they enter along with the disease predicted. So, we are proposing a system which is used to predict multiple diseases . In this system, we are going to analyze Diabetes, Heart, and Liver disease analysis. Later many more diseases can be included. To implement multiple disease prediction systems we are going to use machine learning algorithms. Python pickling is used to save the behavior of the model. The importance of this system analysis is that while analyzing the diseases all the parameters which cause the disease is included so it is possible to detect the disease efficiently and more accurately. The final model's behavior will be saved as a python pickle file .

1.1 PROJECT OVERVIEW

In “Multiple Disease Prediction”, it is possible to predict more than one disease at a time. So the user doesn’t need to traverse different sites in order to predict the diseases. We are taking three diseases that are Liver, Diabetes, and Heart. As all the three diseases are correlated to each other. To implement multiple disease analyses we are going to use machine learning algorithm. When the user is accessing this website, the user has to give the parameters of the disease , and it will invoke the corresponding model and returns the status of the patient.

1.2 SCOPE

Many of the existing machine learning models for health care analysis are concentrating on one disease per analysis. For example first is for liver analysis, one for cancer analysis, one for lung diseases like that. If a user wants to predict more than one disease, he/she has to go through different sites. There is no common system where one analysis can perform more than one disease prediction. Some of the models have lower

accuracy which can seriously affect patients' health. When an organization wants to analyse their patient's health reports, they have to deploy many models which in turn increases the cost as well as time. Some of the existing systems consider very few parameters which can yield false results. There are many machine learning models we can find on the internet with one specific or two disease predictions without proper solution. For example, let's talk about a patient when he is suffering from any disease; he is not able to think properly and just goes on different websites and every existing disease detection system gives a different solution and he is not able to stick to one solution and that doesn't give a proper solution that affects his health. So here our project works better than existing not only in accuracy but giving a better solution after knowing patients' conditions.

1.3 CHALLENGES

Obtaining comprehensive, high-quality data for multiple diseases is challenging due to privacy concerns and data heterogeneity. Class imbalance in disease datasets can lead to biased models, necessitating careful preprocessing and model adjustments. Interpreting complex ML models for disease prediction is crucial but challenging.

1.4 OBJECTIVES

The main objective of this project was to create a system that would predict more than one disease and do so with high accuracy. The user doesn't need to traverse different websites which saves time as well. Diseases if predicted early can increase your life expectancy as well as save you from financial troubles. For this purpose, we have used various machine learning algorithms like Random Forest, XGBoost, and K nearest neighbor (KNN) to achieve maximum accuracy.

CHAPTER 2

LITERATURE SURVEY

[1] Ashir Javeed, Shijie Zhou et al. (2017) designed “An Intelligent Learning System based on Random Search Algorithm and Optimized Random Forest Model for Improved Heart Disease Detection”

This paper uses random search algorithm (RSA) for factor selection and random forest model for diagnosing the cardiovascular disease. This model is principally optimized for using grid search algorithmic program. Two forms of experiments are used for cardiovascular disease prediction. In the first form, only random forest model is developed and within the second experiment the proposed Random Search Algorithm based random forest model is developed. This methodology is efficient and less complex than conventional random forest model. Comparing to conventional random forest it produces 3.3% higher accuracy. The proposed learning system can help the physicians to improve the quality of heart failure detection. The proposed diagnostic system uses random search algorithm (RSA) for features selection and random forest model for heart failure prediction. The proposed diagnostic system is optimized using grid search algorithm. Two types of experiments are performed to evaluate the precision of the proposed method. In the first experiment, only random forest model is developed while in the second experiment the proposed RSA based random forest model is developed. Experiments are performed using an online heart failure database namely Cleveland dataset. The proposed method is efficient and less complex than conventional random forest model as it produces 3.3% higher accuracy than conventional random forest model while using only 7 features. Moreover, the proposed method shows better performance than five other state of the art machine learning models. In addition, the proposed method achieved classification accuracy of 93.33% while improving the training accuracy as well.

[2] Literature Review on Diabetes By Roshni Saxena, Sanjay Kumar Sharma in 2022

Diabetes is a chronic disease characterized by a high amount of glucose in the blood and can cause too many complications also in the body, such as internal organ failure, retinopathy, and neuropathy. According to the prediction made by WHO, the figure may reach approximately 642 million by 2040, which means in every one in ten may suffer from

diabetes due to unhealthy lifestyle and lack of exercise. Analysis of diabetes data disease is quite challenging because most of the data in the medical field are nonlinear, nonnormal, correlation structured, and complex in nature. Machine Learning based algorithms have been ruled out in the field of healthcare and medical Engineering. Diabetes can effect kidney, eyes, nervous system, blood vessels, and so on. Diabetes is of three types. First is juveline diabetes, which occurs mostly in children and destroys the cells which produce insulin in the pancrease. Second is type 2 diabetes. Which generally happen after 40 years of age because of lack of exercise and unhealthy lifestyle. The third type is gestations, which occurs during pregnancy due to changes in the changes of harmone, and it will generally disappears after the delivery. Qawqzeh have implemented a logistic regression classification technique for the classification of diabetes data. Training data includes 459 patients, and testing data includes 128 patients. Classification accuracy achieved by the authors was 92% using logistic regression. The major disadvantage of model was that it was not compared with the other diabetic patient and hence could not be valuated. Tafa divided the dataset into 50% training set and 50% testing set. The model was proposed by the combination of Naïve Bayes and SVM algorithm and dataset was collected from three different location, it consist of 402 patients and 80 patients are diabetic. Both Naive Bayes and SVM has 97.6% accuracy but the drawback is they have not mentioned any preprocessing technique to filter out an unwanted values from dataset.

[3] Literature Review on Liver Disease by Bendi Venkata Ramana in 2011

This paper based on Modified Rotation Forest, used two dataset as an input UCI liver dataset and Indian liver dataset. And results show that MLP algorithm with random subset gives better accuracy of 94.78% and 73.07% for Indian liver dataset. And in 2012 M.Surendra joined with Bendi gives MLP algorithm with random subset gives accuracy 74.78% than NN with CFS of accuracy 73.07%. MLP algorithm with UCI liver dataset has better accuracy than NN with Indian liver dataset.

CHAPTER 3

REQUIREMENTS SPECIFICATION

A System Requirements Specification is a document or set of documentation that describes the features and behaviour of a system or software application. It includes a variety of elements that attempts to define the intended functionality required by the customer to satisfy their different uses.

3.1 FUNCTIONAL REQUIREMENTS

Functional requirements described the interactions between the system and its environment independent of its implementation. It is a description of the service that the software must offer. It describes a software system or its component. A function is nothing but inputs to the software system, its behaviour, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. Functional Requirements in Software Engineering are also called Functional Specification.

- Acquire the dataset
- Import all the crucial libraries
- Import the dataset
- Identifying and handling the missing values
- Encoding the categorical data
- Splitting the dataset
- Feature Scaling

3.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional Requirements (NFRs) define system attributes such as security, reliability, performance, maintainability, scalability, and usability. They serve as constraints or restrictions on the design of the system across the different backlogs.

- The application should perform the process accurately and precisely to avoid problems.
- The application should be easy to understand and use.
- Execution of prediction operation should be fast and accurate.

- The system will have response time of milliseconds.

3.3 HARDWARE REQUIREMENTS

The hardware requirement includes:

- A laptop or desktop computer (Preferably 64bit)
- Random Access Memory (RAM): 8 Gigabytes Minimum
- Processor: Intel Core i5, 2.4 GHz Minimum

3.4 SOFTWARE REQUIREMENTS

The software requirements for the development of this system include:

- Windows Operating System (8/10)
- Anaconda Navigator (Jupyter Notebook)
- PyCharm Community edition
- Web browser (Preferably Chrome)
- Visual Studio Code

CHAPTER 4

SYSTEM DESIGN

4.1 ANACONDA NAVIGATOR

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage anaconda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

The following applications are available by default in Navigator:

- Jupyter Lab
- Jupyter Notebook
- QtConsole
- Spyder
- Glue
- Orange
- RStudio
- Visual Studio Code

4.1.1 JUPYTER NOTEBOOK

Jupyter Notebook is an open-source, web-based interactive environment, which allows you to create and share documents that contain live code, mathematical equations, graphics, maps, plots, visualizations, and narrative text. It integrates with many programming languages like **Python, PHP, R, Notebook C#,** etc.

The Jupyter is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at Project Jupyter.

Jupyter Notebooks are a spin-off project from the Python project, which used to have an Python Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the Python

kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use.

4.2 DJANGO

Django is a high-level Python web framework that enables developers to build web applications rapidly and efficiently.

Here are some key features of Django:

- **Object-Relational Mapping (ORM):** Django includes a powerful ORM that enables developers to interact with databases using Python classes and objects.
- **Built-in Admin Interface:** Django provides a built-in admin interface that allows developers to manage data models, user accounts, and other aspects of their application with ease.
- **URL Routing:** Django's URL routing system makes it easy to map URLs to specific views and templates, providing a clean and organized structure for web applications.
- **Templating Engine:** Django's templating engine provides a powerful way to render HTML and other content dynamically, with support for variables, loops, conditionals, and other programming constructs.
- **Form Handling:** Django includes built-in support for handling form submissions, including validation, error handling, and security measures to protect against CSRF attacks.
- **User Authentication:** Django provides built-in support for user authentication, with features like login, logout, and password reset functionality.
- **Security:** Django has a strong focus on security, with built-in protections against common web application vulnerabilities like SQL injection, cross-site scripting (XSS), and CSRF attacks.

Overall, Django is a comprehensive and powerful web framework that provides developers with a rich set of features for building web applications quickly and securely. Its ease of use, scalability, and extensibility make it a popular choice for developers of all skill levels.

4.3 PYTHON

Python is a versatile and widely-used high-level programming language renowned for its simplicity, readability, and adaptability. Its clear and concise syntax, emphasizing readability, makes it particularly accessible to beginners and facilitates collaboration among developers. Python's versatility is reflected in its support for multiple programming paradigms, including procedural, object-oriented, and functional programming, allowing developers to choose the approach that best suits their project needs.

One of Python's key strengths lies in its extensive standard library and vast ecosystem of third-party packages and frameworks. These resources cover a wide range of domains, including web development, data analysis, machine learning, scientific computing, automation, and more. Popular libraries like NumPy, pandas, Matplotlib, TensorFlow, and Django provide robust solutions for various tasks, enabling developers to build complex applications efficiently.

Python's cross-platform compatibility ensures that code written on one platform can be easily executed on another, enhancing its portability and interoperability. Moreover, Python's strong integration capabilities enable seamless interaction with other programming languages and technologies, facilitating the incorporation of existing code and libraries into Python projects.

The Python community, known for its inclusivity and collaborative spirit, plays a crucial role in supporting and advancing the language. With a large and active community of developers, users, and contributors, Python benefits from continuous improvement, extensive documentation, and a wealth of educational resources. This vibrant community fosters knowledge-sharing, mentorship, and innovation, making Python an ideal choice for both seasoned developers and newcomers alike.

CHAPTER 5

ARCHITECTURE

5.1 SYSTEM ARCHITECTURE

An architectural design concept is the first part of the project and precedes all other activities in the design process. It's only when the design concept has been developed and finalized can the project begin in earnest. Designers use architectural concepts to respond to the design situations they face.

Fig 5.1 shows the architecture view of the proposed phishing detection system such that a user enters a URL link and the link moves through different trained machine learning and deep neural network models and the best model with the highest accuracy is selected. Thus, the selected model is deployed as an API (Application Programming Interface) which is then integrated into a web application. Hence, a user interacts with the web application which is accessible across different display devices such as computers, tablets, and mobile devices.

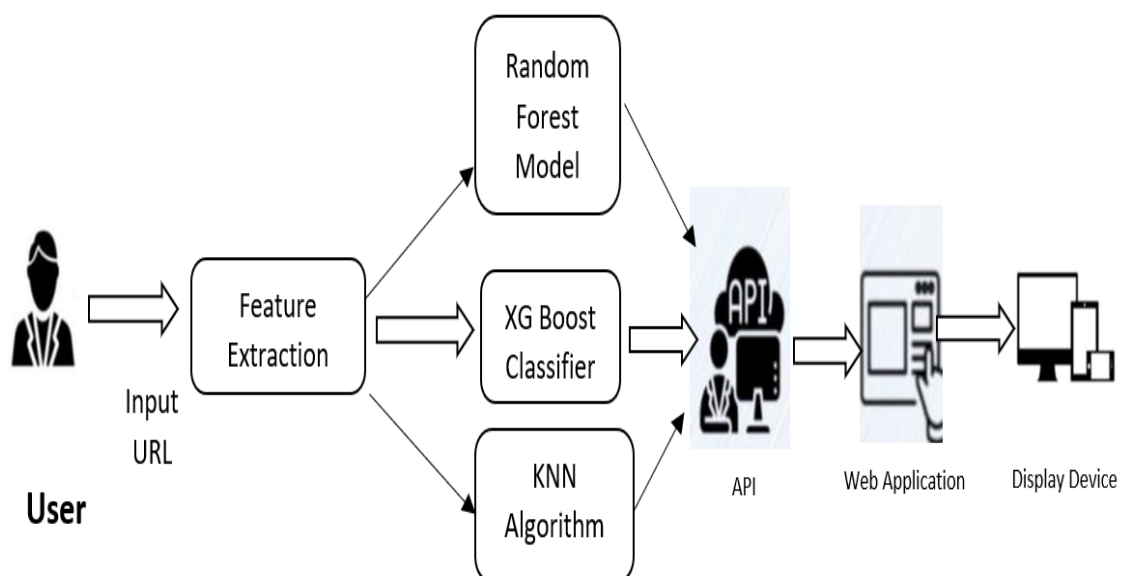


Figure 5.1: Architecture Design Of The proposed System

5.2 FLOWCHART OF THE SYSTEM

A flowchart is a diagram that depicts a process, system, or computer algorithm. It is a graphical representation of the steps that are to be performed in a system, it shows the steps in sequential order. It is used in presenting the flow of algorithms and to communicate complex processes in clear, easy-to-understand diagrams.

Fig 5.2.1 shows the flow of disease detection systems using the machine learning process.

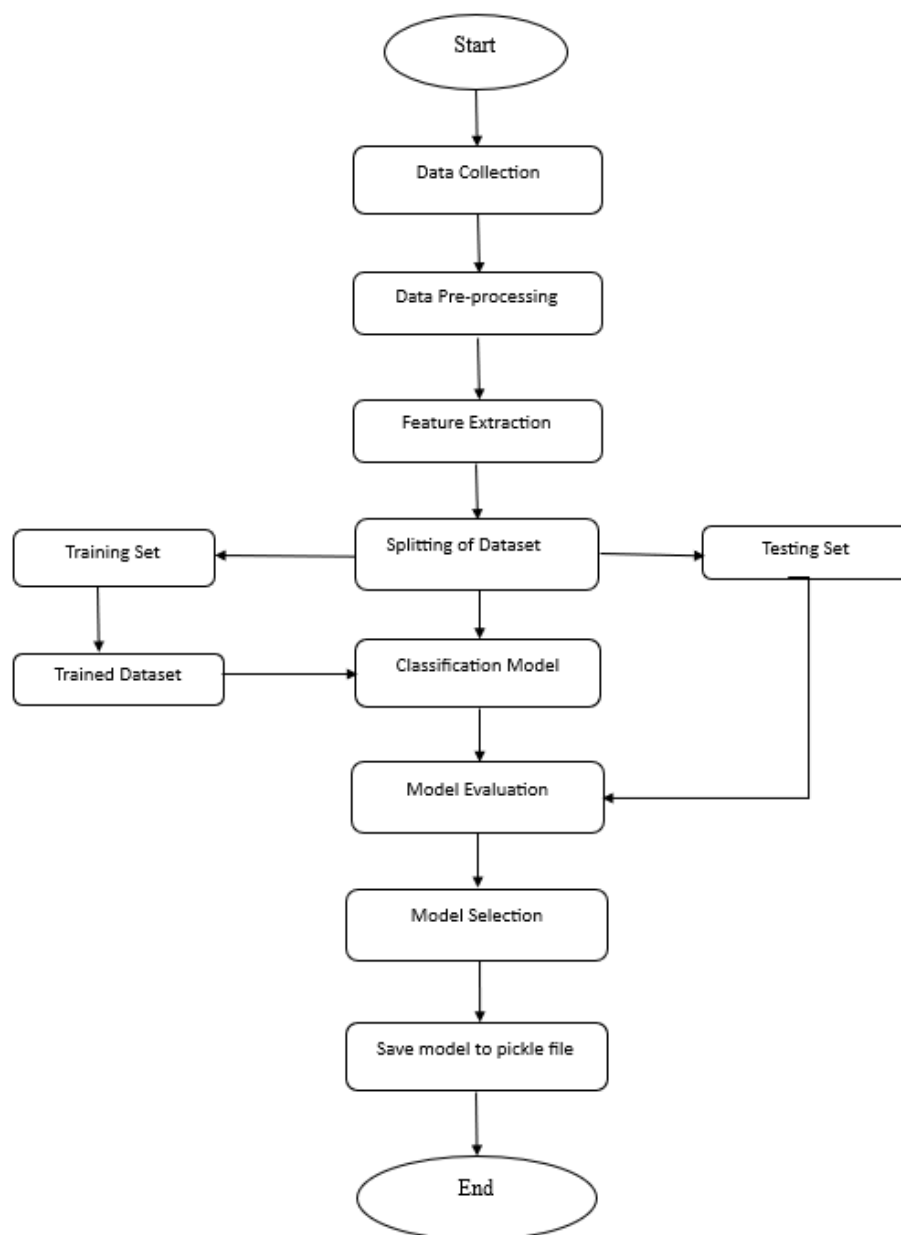


Figure 5.2.1: Flowchart Of The proposed System

Fig 5.2.2 shows the disease detection web interface system. The user inputs a data and the website validates the data and then predicts the disease present or not.

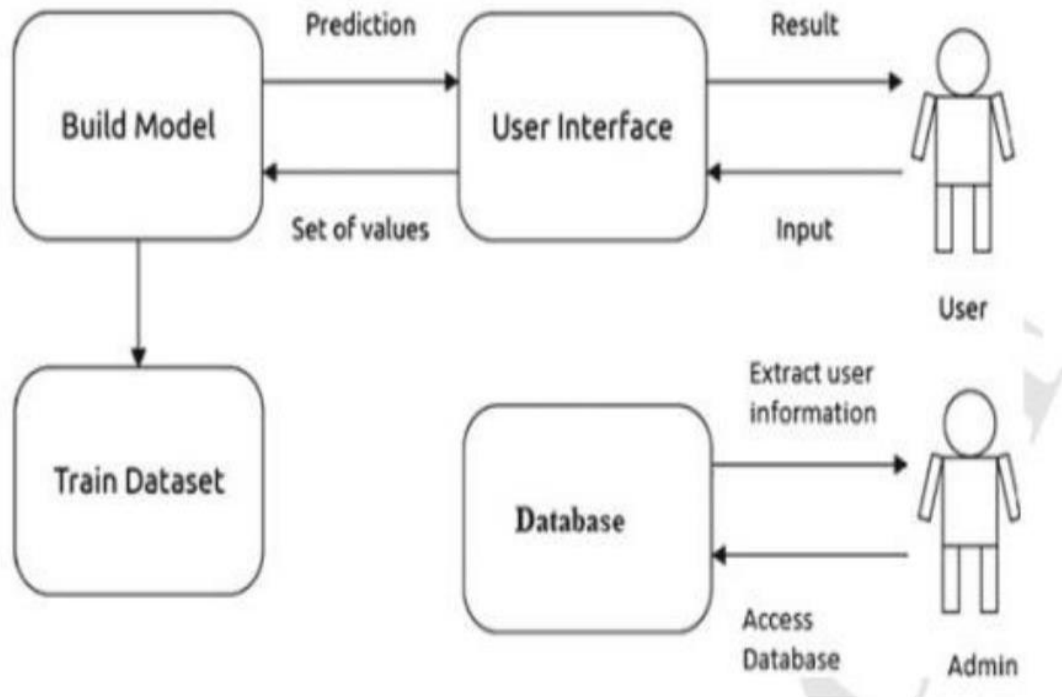


Figure 5.2.2: Flowchart of the web interface

5.3 METHODOLOGY

1. Data Collection

- Identify relevant datasets containing medical records, demographics, symptoms, diagnostic tests, and disease outcomes
- Gather data from diverse sources such as hospitals, research institutions, public health organizations, and online repositories.
- Ensure data integrity, quality, and compliance with privacy regulations.
- Verify the representativeness of the data across different demographics and disease prevalence.

2. Data Preprocessing

- Cleanse the data by handling missing values, outliers, and inconsistencies.
- Normalize or standardize numerical features to ensure uniformity.
- Encode categorical variables into numerical representations using techniques like one-hot encoding or label encoding.
- Feature scaling may be applied to bring all features to a similar scale.
- Perform feature selection to identify relevant features and reduce dimensionality if necessary.

3. Model Selection

- Choose appropriate machine learning algorithms for classification tasks, considering factors like interpretability, scalability, and performance.
- Experiment with a variety of algorithms such as XGBoost, random forests, knn algorithm
- Evaluate the suitability of each algorithm based on the dataset size, complexity, and computational resources available.

4. Training the Model

- Split the preprocessed data into training, validation, and test sets using stratified sampling to maintain class balance.
- Train the selected models using the training data and optimize hyperparameters using techniques like grid search, random search, or Bayesian optimization.
- Validate model performance using the validation set and adjust hyperparameters accordingly to prevent overfitting.
- Cross-validation techniques such as k-fold cross-validation may be employed to assess model generalization.

The datasets are divided into two models:

- training set—a subset to train a model.
- test set—a subset to test the trained model.

Slicing the single data set as follows:**Figure 5.3.1: Slicing a single data set into a training set and test set.****Need of training set:**

- It is an organized form of unorganized data.
- Recognition and Classification of areca.
- Provides a key input to CNN algorithm.
- Validating the machine learning algorithm.

Need of test set:

When training ML and DL models, we often split the entire dataset into training and test sets. This is because we need a separate test set to evaluate your model on unseen data to increase the generalizing capability of the model. We should make sure that our test set meets the following two conditions:

- Is large enough to yield statistically meaningful results.
- Is representative of the data set as a whole. In other words, don't pick a test set with different characteristics than the training set.

Assuming that our test set meets the preceding two conditions, our goal is to create a model that generalizes well to new data. And our test set serves as a proxy for new data. Notice that the model learned for the training data is very simple. This model doesn't do a perfect job - a few predictions are wrong. However, this model does about as well on the test data as it does on the training data. In other words, this simple model does not overfit the training data.

We should never train on test data. If we are seeing surprisingly good results on our evaluation metrics, it might be a sign that we are accidentally training on the test set. For example, high accuracy might indicate that test data has leaked into the training set.

A true positive is an outcome where the model correctly predicts the positive class. Similarly, a true negative is an outcome where the model correctly predicts the negative class.

Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right. Formally, accuracy has the following definition:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total Number of predictions}}$$

5. Evaluation and Testing

- Evaluate the trained models using the test set to assess their performance on unseen data.
- Calculate evaluation metrics such as accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC).
- Conduct statistical significance tests to compare model performance and identify the best-performing model.
- Visualize evaluation results using confusion matrices, ROC curves, or precision-recall curves for better interpretation.

6. Results

- Interpret the trained models to understand the factors influencing disease prediction.
- Analyze feature importance scores to identify the most influential predictors for each disease.
- Use model explanations techniques such as SHAP values, LIME, or partial dependence plots to provide insights into individual predictions.
- Communicate results and findings effectively to stakeholders, including healthcare professionals, researchers, and patients.

5.4 E-R Diagram

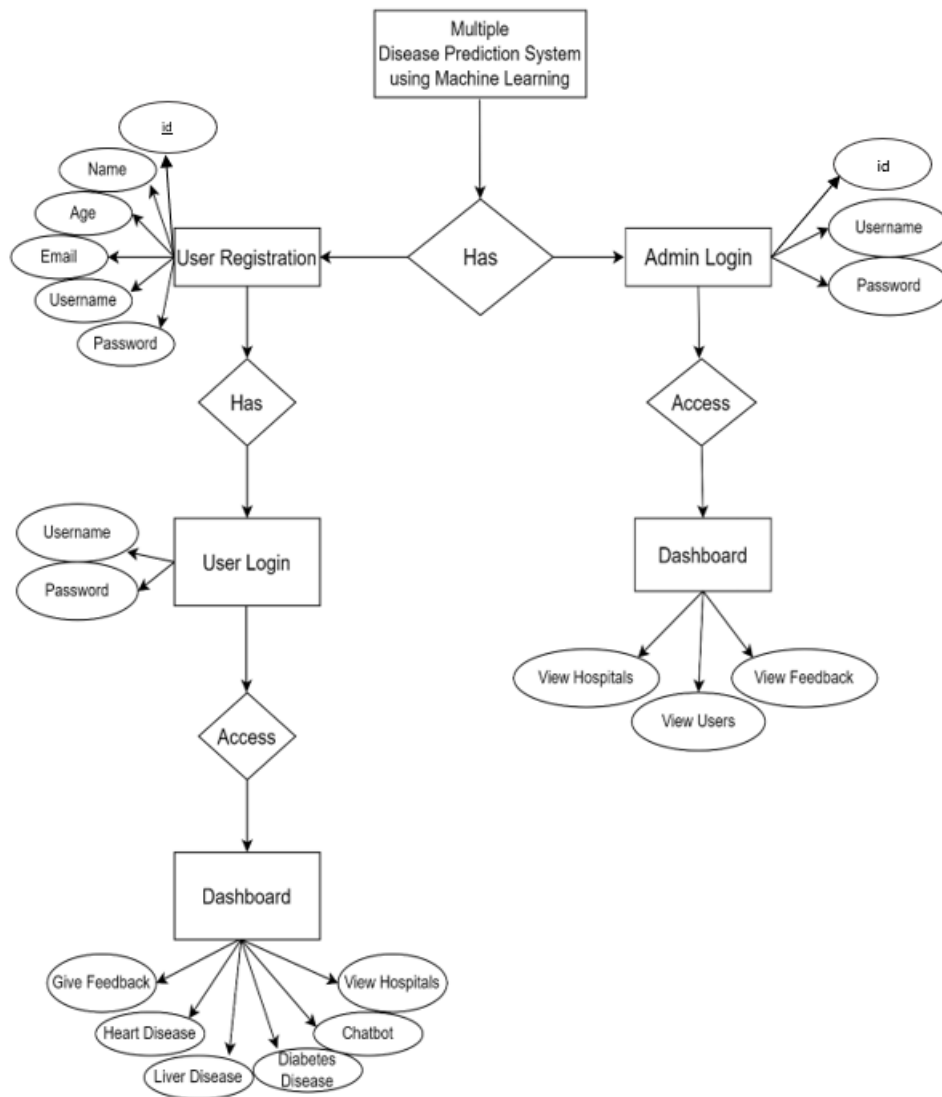


Fig 5.4: E-R Diagram

5.5 Use Case Diagram

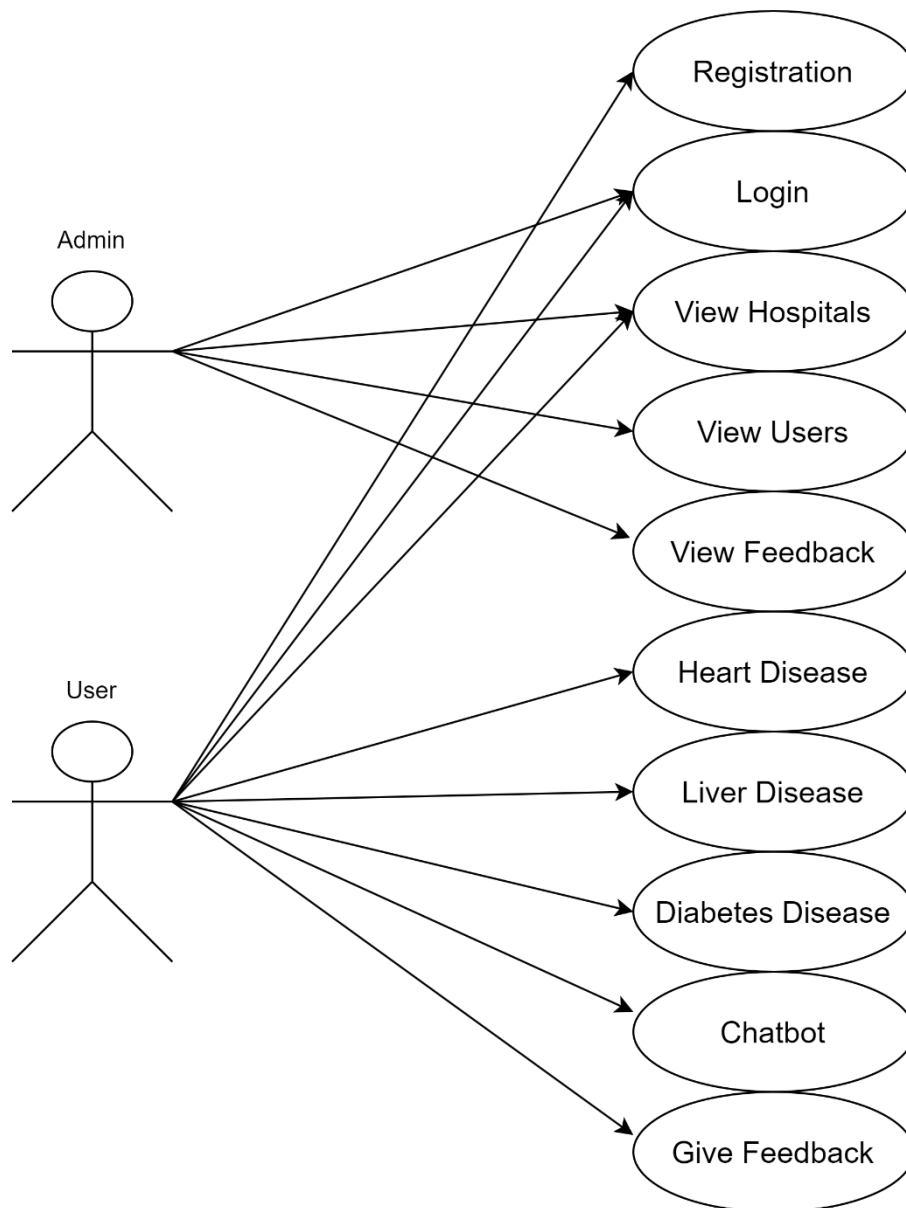


Fig 5.5: Use Case Diagram

5.6 Activity Diagram

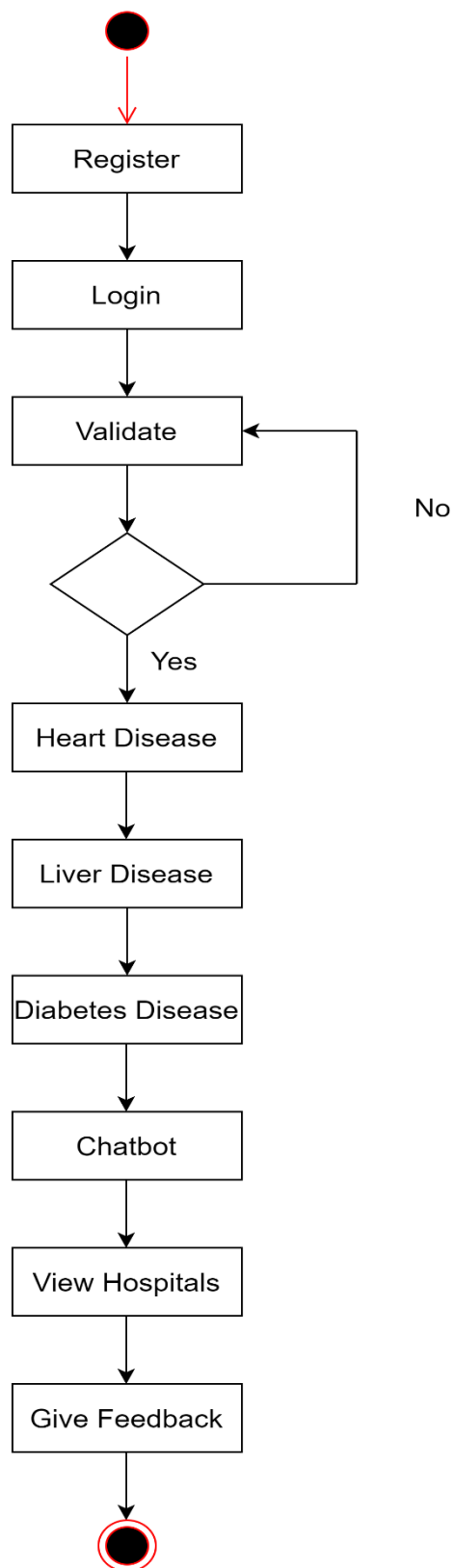


Fig 5.6.1: Activity Diagram of User

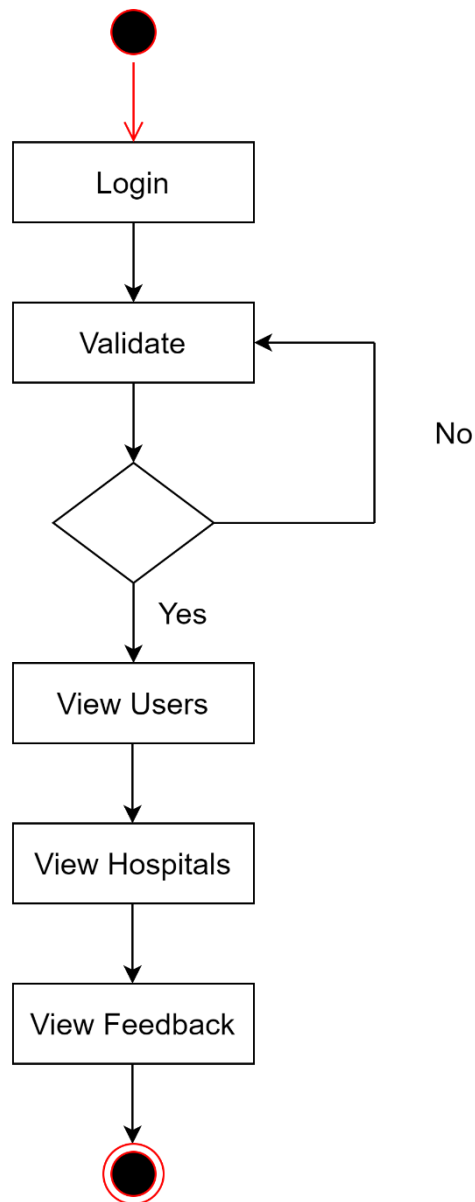


Fig 5.6.2: Activity Diagram of Admin

CHAPTER 6

IMPLEMENTATION

6.1 PSEUDO CODE

Command: `import pandas as pd`

Description: Pandas is an open-source library for data manipulation and analysis. It provides data structures and functions to work with structured data, such as spreadsheets and databases. Pandas is widely used in data science and machine learning projects to clean, preprocess, and transform data.

Command: `from matplotlib import pyplot as plt`

Description: Each Pyplot function makes some change to a figure. For example, a function creates a figure, a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. Make a bar plot.

Command: `import sklearn`

Description: Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. scikit-image includes algorithms for segmentation, geometric transformations, color space manipulation, analysis, filtering, morphology, feature detection, and more characteristics of areca pictures.

Command: `from sklearn.model_selection import train_test_split`

Description: `'train_test_split'` is a function in scikit-learn for dividing datasets into training and testing sets. It takes input arrays and splits them based on specified proportions. This allows for training a model on one subset and testing its performance on another. Key parameters include `'test_size'`, `'random_state'`, and `'stratify'`.

Command: `from sklearn.ensemble import RandomForestClassifier`

Description: Importing `'RandomForestClassifier'` from `'sklearn.ensemble'`, which is used for classification tasks. It's an implementation of the random forest algorithm, a popular ensemble learning technique. Random forests build multiple decision trees during training and output the mode of the classes for classification tasks. They are known for their

robustness and effectiveness in handling complex datasets. This classifier can be fine-tuned using various parameters to optimize performance for specific tasks.

Command: `from sklearn.neighbors import KNeighborsClassifier`

Description: Importing the `KNeighborsClassifier` class from `sklearn.neighbors`. This classifier is based on the k-nearest neighbors algorithm, a simple yet effective supervised learning algorithm used for classification tasks. It assigns the class of a data point based on the classes of its nearest neighbors in the feature space. You can adjust the number of neighbors (`n_neighbors`) and other parameters to customize its behavior.

Command: `import xgboost as xgb`

Description: Importing the XGBoost library using the alias `xgb`. XGBoost is an optimized gradient boosting library that's highly efficient and widely used in machine learning competitions and real-world applications. It's known for its speed, performance, and scalability, particularly in handling large datasets. With XGBoost, you can build and train gradient boosting models for both classification and regression tasks with ease.

Command: `import pickle`

Description: Importing the `pickle` module, which provides a way to serialize and deserialize Python objects. Pickling is the process of converting a Python object into a byte stream, while unpickling is the reverse process of reconstructing a Python object from a byte stream. This module is commonly used for saving and loading trained machine learning models, among other things.

Command: `with open('random_forest_model_heart.pkl', 'wb') as model_file:`

`pickle.dump(random_forest_model, model_file)`

Description: Python provides pickle modules for Serialization and de-Serialization of python objects like lists, dictionaries, tuples, etc. Pickling is also called marshaling or flattening in other languages. Pickling is used to store python objects. Serialization or Pickling: Pickling or Serialization is the process of converting a Python object (lists, dict, tuples, etc.) into byte streams that can be saved to disks or can be transferred over a network.

De-serialization or un pickling: The byte streams saved on file contains the necessary information to reconstruct the original python object. The process of converting byte streams back to python objects is called de-serialization

6.2 DATA COLLECTION

The dataset used for classifying the dataset into liver, heart and diabetes was sourced from open source websites, samples of which are shown below in Fig 6.2.1, 6.2.2 and 6.2.3 respectively

1	Age	Gender	Total_Bilir	Direct_Bili	Alkaline_P	Alamine_A	Aspartate	Total_Prot	Albumin	Albumin_a	Dataset
2	65	Female	0.7	0.1	187	16	18	6.8	3.3	0.9	1
3	62	Male	10.9	5.5	699	64	100	7.5	3.2	0.74	1
4	62	Male	7.3	4.1	490	60	68	7	3.3	0.89	1
5	58	Male	1	0.4	182	14	20	6.8	3.4	1	1
6	72	Male	3.9	2	195	27	59	7.3	2.4	0.4	1
7	46	Male	1.8	0.7	208	19	14	7.6	4.4	1.3	1
8	26	Female	0.9	0.2	154	16	12	7	3.5	1	1
9	29	Female	0.9	0.3	202	14	11	6.7	3.6	1.1	1
10	17	Male	0.9	0.3	202	22	19	7.4	4.1	1.2	2
11	55	Male	0.7	0.2	290	53	58	6.8	3.4	1	1
12	57	Male	0.6	0.1	210	51	59	5.9	2.7	0.8	1
13	72	Male	2.7	1.3	260	31	56	7.4	3	0.6	1
14	64	Male	0.9	0.3	310	61	58	7	3.4	0.9	2
15	74	Female	1.1	0.4	214	22	30	8.1	4.1	1	1
16	61	Male	0.7	0.2	145	53	41	5.8	2.7	0.87	1
17	25	Male	0.6	0.1	183	91	53	5.5	2.3	0.7	2
18	38	Male	1.8	0.8	342	168	441	7.6	4.4	1.3	1
19	33	Male	1.6	0.5	165	15	23	7.3	3.5	0.92	2
20	40	Female	0.9	0.3	293	232	245	6.8	3.1	0.8	1
21	40	Female	0.9	0.3	293	232	245	6.8	3.1	0.8	1
22	51	Male	2.2	1	610	17	28	7.3	2.6	0.55	1
23	51	Male	2.9	1.3	482	22	34	7	2.4	0.5	1
24	62	Male	6.8	3	542	116	66	6.4	3.1	0.9	1
25	40	Male	1.9	1	231	16	55	4.3	1.6	0.6	1
26	63	Male	0.9	0.2	194	52	45	6	3.9	1.85	2
27	34	Male	4.1	2	289	875	731	5	2.7	1.1	1
28	34	Male	4.1	2	289	875	731	5	2.7	1.1	1
29	34	Male	6.2	3	240	1680	850	7.2	4	1.2	1
30	20	Male	1.1	0.5	128	20	30	3.9	1.9	0.95	2
31	84	Female	0.7	0.2	188	13	21	6	3.2	1.1	2

Figure 6.2.1: Dataset of Liver Disease

Source: The Dataset is collected from an open-source service called Kaggle. This dataset consists of 500 random liver disease data which are collected to train the ML models.

	age	sex	chest pain	resting blo	serum cho	fasting blo	resting ele	max heart	exercise in	oldpeak	ST segmen	major vess	thal	heart disease
2	70	1	4	130	322	0	2	109	0	2.4	2	3	3	2
3	67	0	3	115	564	0	2	160	0	1.6	2	0	7	1
4	57	1	2	124	261	0	0	141	0	0.3	1	0	7	2
5	64	1	4	128	263	0	0	105	1	0.2	2	1	7	1
6	74	0	2	120	269	0	2	121	1	0.2	1	1	3	1
7	65	1	4	120	177	0	0	140	0	0.4	1	0	7	1
8	56	1	3	130	256	1	2	142	1	0.6	2	1	6	2
9	59	1	4	110	239	0	2	142	1	1.2	2	1	7	2
0	60	1	4	140	293	0	2	170	0	1.2	2	2	7	2
1	63	0	4	150	407	0	2	154	0	4	2	3	7	2
2	59	1	4	135	234	0	0	161	0	0.5	2	0	7	1
3	53	1	4	142	226	0	2	111	1	0	1	0	7	1
4	44	1	3	140	235	0	2	180	0	0	1	0	3	1
5	61	1	1	134	234	0	0	145	0	2.6	2	2	3	2
6	57	0	4	128	303	0	2	159	0	0	1	1	3	1
7	71	0	4	112	149	0	0	125	0	1.6	2	0	3	1
8	46	1	4	140	311	0	0	120	1	1.8	2	2	7	2
9	53	1	4	140	203	1	2	155	1	3.1	3	0	7	2
0	64	1	1	110	211	0	2	144	1	1.8	2	0	3	1
1	40	1	1	140	199	0	0	178	1	1.4	1	0	7	1
2	67	1	4	120	229	0	2	129	1	2.6	2	2	7	2
3	48	1	2	130	245	0	2	180	0	0.2	2	0	3	1
4	43	1	4	115	303	0	0	181	0	1.2	2	0	3	1
5	47	1	4	112	204	0	0	143	0	0.1	1	0	3	1
6	54	0	2	132	288	1	2	159	1	0	1	1	3	1
7	48	0	3	130	275	0	0	139	0	0.2	1	0	3	1
8	46	0	4	138	243	0	2	152	1	0	2	0	3	1
9	51	0	3	120	295	0	2	157	0	0.6	1	0	3	1
0	58	1	3	112	230	0	2	165	0	2.5	2	1	7	2
1	71	0	3	110	265	1	2	130	0	0	1	1	3	1

Figure 6.2.2: Dataset of Heart Disease

Source: The Dataset is collected from an open-source service called Kaggle. This dataset consists of 500 random heart disease data which are collected to train the ML models.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	Age	Gender	Polyuria	Polydipsia	sudden we	weakness	Polyphagia	Genital thr	visual blur	itching	Irritability	delayed he	partial par	muscle stit	Alopecia	Obesity	class	
2	40	Male	No	Yes	No	Yes	No	No	Yes	No	No	Yes	No	Yes	Yes	Yes	Positive	
3	58	Male	No	No	No	Yes	No	No	Yes	No	No	Yes	No	Yes	No	Yes	Positive	
4	41	Male	Yes	No	No	Yes	Yes	No	No	Yes	No	Yes	No	Yes	Yes	No	Positive	
5	45	Male	No	No	Yes	Yes	Yes	Yes	No	Yes	No	Yes	No	No	No	No	Positive	
6	60	Male	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Positive	
7	55	Male	Yes	Yes	No	Yes	Yes	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Positive	
8	57	Male	Yes	Yes	No	Yes	Yes	Yes	No	No	No	Yes	Yes	No	No	No	Positive	
9	66	Male	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No	No	Positive	
0	67	Male	Yes	Yes	No	Yes	Yes	Yes	No	Yes	Yes	No	Yes	Yes	No	Yes	Positive	
1	70	Male	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No	No	No	Yes	No	Positive	
2	44	Male	Yes	Yes	No	Yes	No	Yes	No	No	Yes	Yes	No	Yes	Yes	No	Positive	
3	38	Male	Yes	Yes	No	No	Yes	Yes	No	Yes	No	Yes	No	Yes	No	No	Positive	
4	35	Male	Yes	No	No	No	Yes	Yes	No	No	Yes	Yes	No	No	Yes	No	Positive	
5	61	Male	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	Yes	Yes	Positive	
6	60	Male	Yes	Yes	No	Yes	Yes	No	Yes	Yes	No	Yes	Yes	No	No	No	Positive	
7	58	Male	Yes	Yes	No	Yes	Yes	No	No	No	No	Yes	Yes	Yes	No	No	Positive	
8	54	Male	Yes	Yes	Yes	Yes	No	Yes	No	No	No	Yes	No	Yes	No	No	Positive	
9	67	Male	No	Yes	No	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Positive	
10	66	Male	Yes	Yes	No	Yes	Yes	No	Yes	No	No	No	Yes	Yes	No	No	Positive	
11	43	Male	Yes	Yes	Yes	Yes	No	Yes	No	No	No	No	No	No	No	No	Positive	
12	62	Male	Yes	Yes	No	Yes	Yes	No	Yes	No	Yes	No	Yes	Yes	No	No	Positive	
13	54	Male	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Yes	No	Positive	
14	39	Male	Yes	No	Yes	No	No	Yes	No	Yes	Yes	No	No	No	Yes	No	Positive	
15	48	Male	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	No	No	No	No	Positive	
16	58	Male	Yes	Yes	Yes	Yes	Yes	No	Yes	No	No	Yes	Yes	Yes	No	Yes	Positive	
17	32	Male	No	No	No	No	No	Yes	No	No	Yes	Yes	No	No	No	Yes	Positive	
18	42	Male	No	No	Yes	Yes	No	No	No	No	Yes	No	No	Yes	No	No	Positive	
19	52	Male	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	Yes	Yes	No	No	Positive	
10	38	Male	No	Yes	No	No	No	Yes	No	No	No	No	No	No	Yes	No	Positive	
11	51	Male	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No	Positive	

Figure 6.2.3: Dataset of Diabetes Disease

Source: The Dataset is collected from an open-source service called Kaggle. This dataset consists of 500 random diabetes disease data which are collected to train the ML models.

6.3 DATA PRE-PROCESSING

In [4]:	1 summary = data.describe()
	2 print(summary)

	Age	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphatase \
count	583.000000	583.000000	583.000000	583.000000
mean	44.746141	3.298799	1.486106	290.576329
std	16.189833	6.209522	2.808498	242.937989
min	4.000000	0.400000	0.100000	63.000000
25%	33.000000	0.800000	0.200000	175.500000
50%	45.000000	1.000000	0.300000	208.000000
75%	58.000000	2.600000	1.300000	298.000000
max	90.000000	75.000000	19.700000	2110.000000

	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens \
count	583.000000	583.000000	583.000000
mean	80.713551	109.910806	6.483190
std	182.620356	288.918529	1.085451
min	10.000000	10.000000	2.700000
25%	23.000000	25.000000	5.800000
50%	35.000000	42.000000	6.600000
75%	60.500000	87.000000	7.200000
max	2000.000000	4929.000000	9.600000

Figure 6.3.1 : Summary of the dataset

The datasets were first cleaned to remove empty entries and fill some entries by applying data pre-processing techniques and transform the data to use in the models. Figure 6.3.1 shows the summary of the dataset while figure 6.3.2 shows the number of unique values in the dataset which all appear to be zero.


```

1 # Loop through each column and print unique values and their counts
2 for column in data.columns:
3     unique_values = data[column].unique()
4     value_counts = data[column].value_counts()
5
6     print(f"Column: {column}")
7     print(f"Unique Values: {unique_values}")
8     print(f"Value Counts:\n{value_counts}\n")

```

Column: Age

Unique Values: [65 62 58 72 46 26 29 17 55 57 64 74 61 25 38 33 40 51 63 34 20 84 52 30
48 47 45 42 50 85 35 21 32 31 54 37 66 60 19 75 68 70 49 14 13 18 39 27
36 24 28 53 15 56 44 41 7 22 8 6 4 43 23 12 69 16 78 11 73 67 10 90]

Value Counts:

Age

60 34

45 24

50 23

42 21

38 21

..

78 1

11 1

67 1

10 1

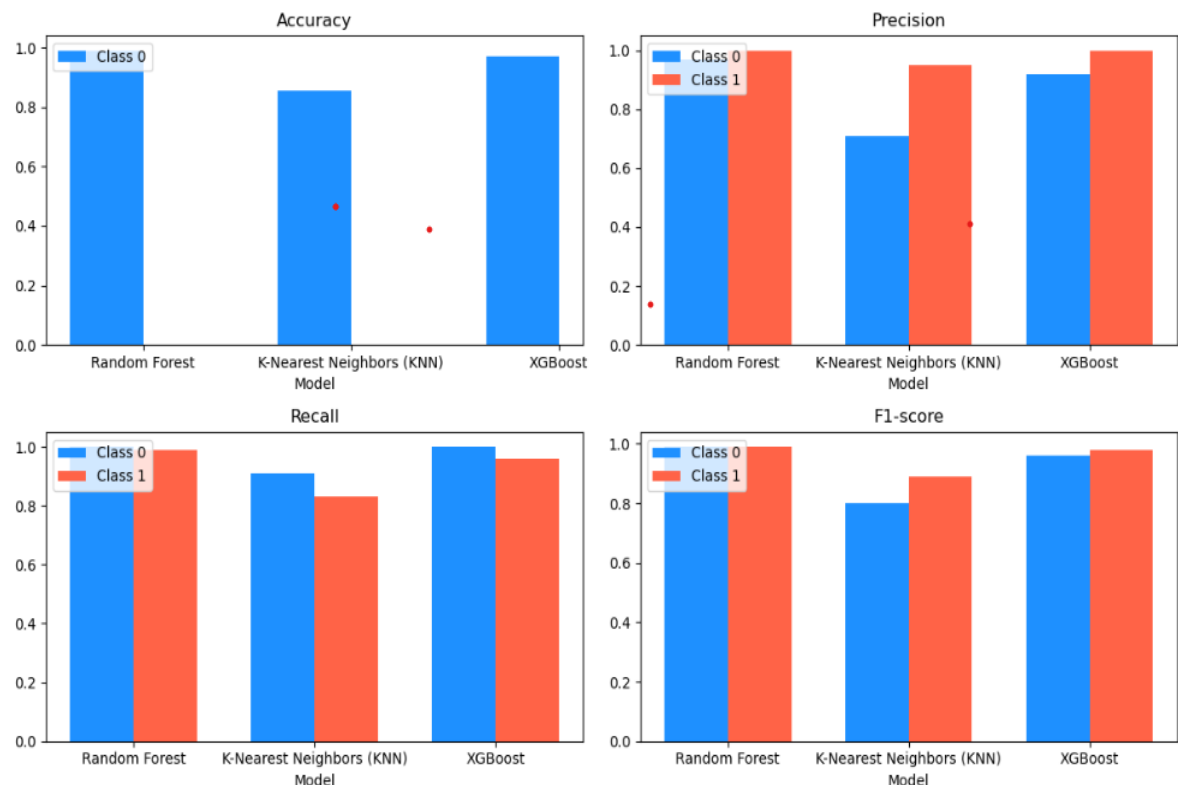
90 1

Name: count, Length: 72, dtype: int64

Figure 6.3.2 : Number of unique values in the dataset

6.4 DATA ANALYSIS AND VISUALIZATION

.



Three machine learning models: Random Forest, K-Nearest Neighbors (KNN), and XGBoost. It appears to be for a binary classification task, where the models are trying to predict one of two classes.

The left side of the graph shows the accuracy and precision of each model. Accuracy is the proportion of correct predictions overall, while precision is the proportion of positive predictions that were actually correct. In the graph, it appears that XGBoost has the highest accuracy and precision, followed by Random Forest and KNN.

The right side of the graph shows the recall and F1-score of each model. Recall is the proportion of actual positive cases that were correctly identified by the model. F1-score is a harmonic mean of precision and recall, which takes both metrics into account. In the graph, it appears that XGBoost also has the highest recall and F1-score, followed by Random Forest and KNN.

6.5 DISEASE PREDICTION MODEL

```
37 print(f'Accuracy: {greatest_accuracy:.4f}')
```

```
38
```

Model with the greatest accuracy: Random Forest
Accuracy: 0.9904

Figure 6.5.1: Accuracy performance of diabetes models

This model employs Random Forest, a versatile ensemble learning technique. It utilizes features such as blood glucose levels, insulin levels, and BMI. Achieving an impressive accuracy of 0.99, it demonstrates exceptional reliability in predicting diabetes.

```
43 print(f'Model with the greatest accuracy: {greatest_accuracy_model}')
44 print(f'Accuracy: {greatest_accuracy:.4f}')
```

```
45
```

Model with the greatest accuracy: Random Forest
Accuracy: 0.9843

Figure 6.5.2: Accuracy performance of heart models

Utilizing Random Forest, an ensemble learning method known for its robustness. Features such as blood pressure, cholesterol levels, and ECG results are likely used. With an accuracy of 0.99, this model exhibits strong predictive capability for heart disease.

```
42 print(f'Model with the greatest accuracy: {greatest_acc}')  
43 print(f'Accuracy: {greatest_accuracy:.4f}')
```

Model with the greatest accuracy: Random Forest
Accuracy: 1.0000

Figure 6.5.3: Accuracy performance of liver models

This model employs Random Forest, a gradient boosting algorithm suitable for structured data. Features such as liver enzyme levels and bilirubin levels are utilized. Despite its lower accuracy of 0.1, it provides valuable insights into liver health and potential disease risks.

6.6 API(Application Programming Interface)

The work of an API here is that it serves as an intermediary between the web server and web application. A python framework called Django was used on the model prediction on the web application. These two ends communicate using a JSON (JavaScript Object Notation) to send a request and receive a response.

Figure 6.6.1 shows how the API was created by using Django python code for communication between the JSON dataset and feature extraction formula for detection phishing URL.

Figure 6.6.2 shows the creation of an API URL.

Figure 6.6.3 shows the execution of the Django local server for the API (Application Programming Interface)

Figure 6.6.4 shows the HTML, CSS and JavaScript code for front end web application.

```

views.py 4 X
MultipleDiseaseDetection > AppMultipleDiseaseDetection > views.py > ...
12 from .gui import *
13 from django.views import View
14 from django.http import JsonResponse
15 from django.db.models import Q
16 import re
17 # User Views
18
19 # Create your views here.
20
21
22 def Home(request):
23     return render(request, "Home.html", {})
24
25
26 def Base(request):
27     return render(request, "Base.html", {})
28
29
30 def about(request):
31     return render(request, 'about.html')
32
33
34 def User_Login(request):
35     if request.method == "POST":
36         C_email = request.POST['email']
37         C_password = request.POST['password']

```

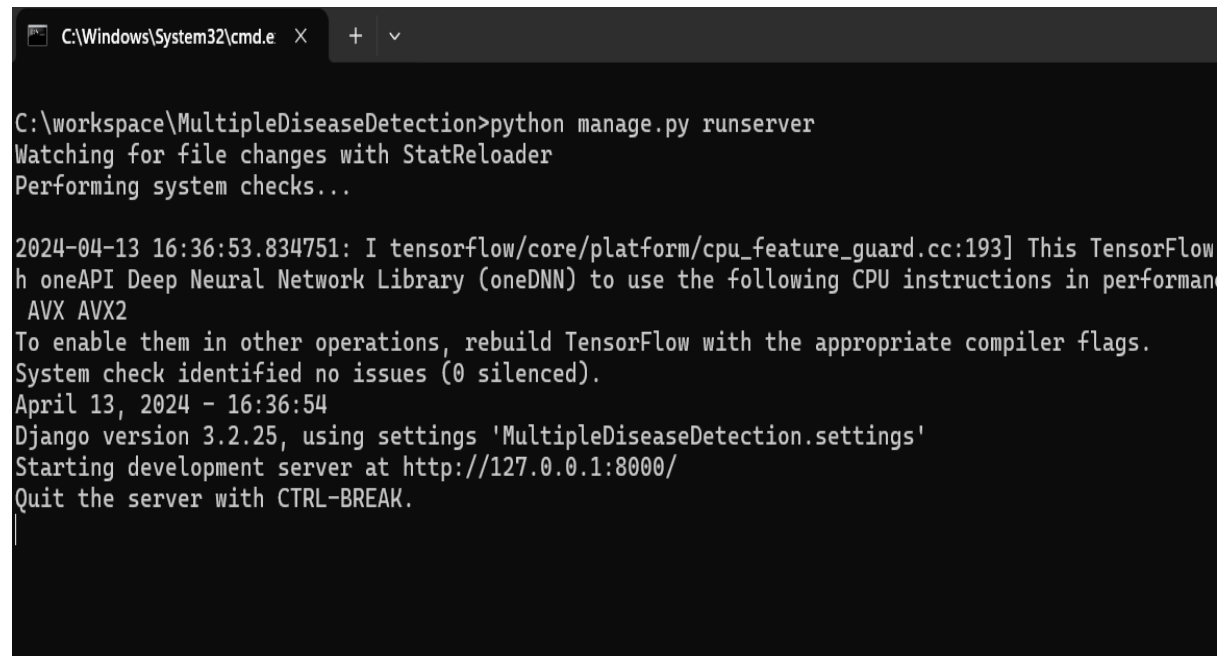
Figure 6.6.1: Python code to send a request on JSON

```

urls.py X
MultipleDiseaseDetection > AppMultipleDiseaseDetection > urls.py > ...
1 from django.urls import path
2 from django.conf import settings
3 from django.conf.urls.static import static
4 from . import views
5 from AppMultipleDiseaseDetection.views import Message
6 urlpatterns = [
7     path('', views.Home, name="Home"),
8     path('Base/', views.Base, name="Base"),
9     path('about/', views.about, name="about"),
10    path('Admin_Login/', views.Admin_Login, name="Admin_Login"),
11    path('User_Login/', views.User_Login, name="User_Login"),
12    path('User_Registration/', views.User_Registration, name="User_Registration"),
13    path('Logout/', views.Logout, name="Logout"),
14    path('view_users/', views.view_users, name="view_users"),
15    path('view_feedback/', views.view_feedback, name="view_feedback"),
16    path('view_hospitals/', views.view_hospitals, name="view_hospitals"),
17    path('give_feedback/', views.give_feedback, name="give_feedback"),
18    path('prediction/', views.prediction, name="prediction"),
19    path('chatbot/', views.chatbot, name="chatbot"),
20    path('heart/', views.heart, name="heart"),
21    path('liver/', views.liver, name="liver"),
22    path('diabetes/', views.diabetes, name="diabetes").

```

Figure 6.6.2: Code for API URL



```

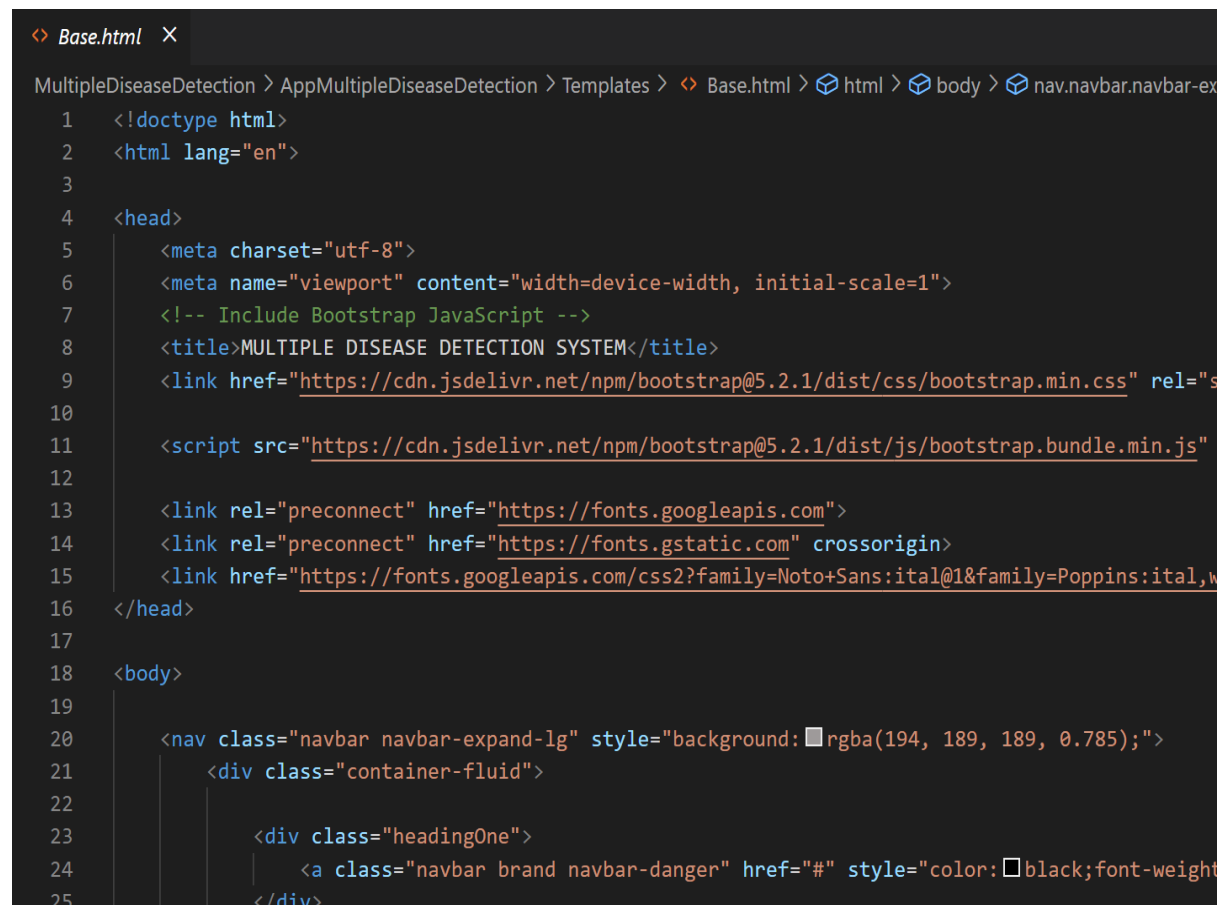
C:\Windows\System32\cmd.e X + v

C:\workspace\MultipleDiseaseDetection>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

2024-04-13 16:36:53.834751: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow
h oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performan
AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
System check identified no issues (0 silenced).
April 13, 2024 - 16:36:54
Django version 3.2.25, using settings 'MultipleDiseaseDetection.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

```

Figure 6.6.3: Executing web-app on Django local server



```

Base.html X
MultipleDiseaseDetection > AppMultipleDiseaseDetection > Templates > Base.html > html > body > nav.navbar.navbar-ex
1  <!doctype html>
2  <html lang="en">
3
4  <head>
5      <meta charset="utf-8">
6      <meta name="viewport" content="width=device-width, initial-scale=1">
7      <!-- Include Bootstrap JavaScript -->
8      <title>MULTIPLE DISEASE DETECTION SYSTEM</title>
9      <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel="s
10
11      <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js"
12
13      <link rel="preconnect" href="https://fonts.googleapis.com">
14      <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
15      <link href="https://fonts.googleapis.com/css2?family=Noto+Sans:ital@1&family=Poppins:ital,w
16  </head>
17
18  <body>
19
20      <nav class="navbar navbar-expand-lg" style="background: rgba(194, 189, 189, 0.785);">
21          <div class="container-fluid">
22
23              <div class="headingOne">
24                  <a class="navbar brand navbar-danger" href="#" style="color: black;font-weight
25              </div>

```

Figure 6.6.4: User Interface Code

CHAPTER 7

RESULTS

In the system diabetes disease prediction model used random forest algorithm, heart disease uses the random forest algorithm and liver uses the random forest algorithm as these gave the best accuracy accordingly. There when the patient adds the parameter according to the disease it will show whether the patient has a disease or not according to the disease selected. The parameters will show the range of the values needed and if the value is not between the range or is not valid or is empty it will show the warning sign that add a correct value.

ACCURACY FOR EACH DISEASE:

Table No 7.a: Diabetes Disease

ALGORITHM	Diabetes
Random Forest	99%
XGBoost	98%

Table No 7.b: Heart Disease

ALGORITHM	Heart
Random Forest	99%
XGBoost	97%
Knn	93%

Table No 7.c: Liver Disease

ALGORITHM	Liver
Random Forest	100%
XGBoost	99%
Knn	97%

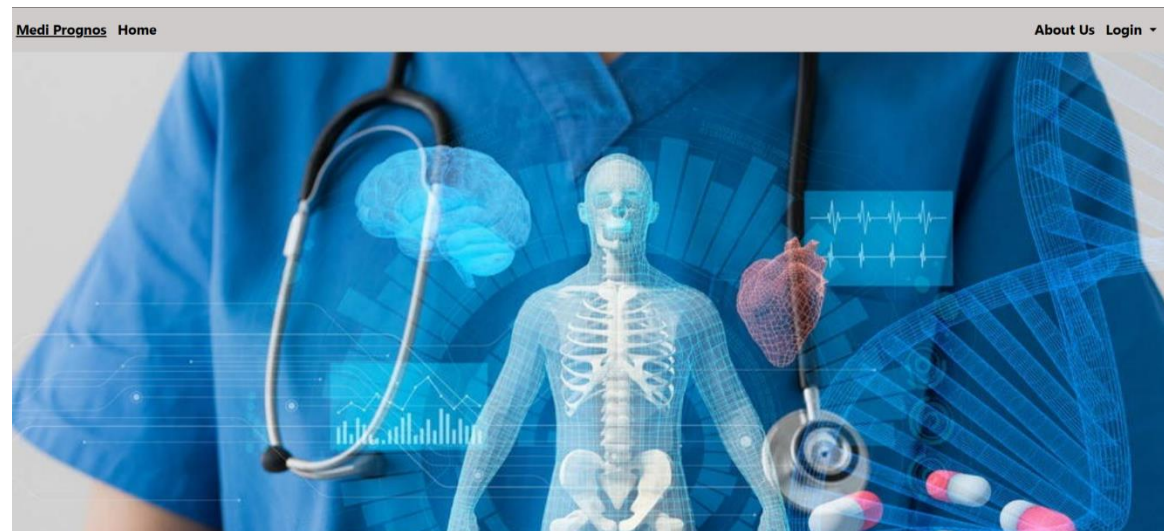


Figure 7.1: Home Page

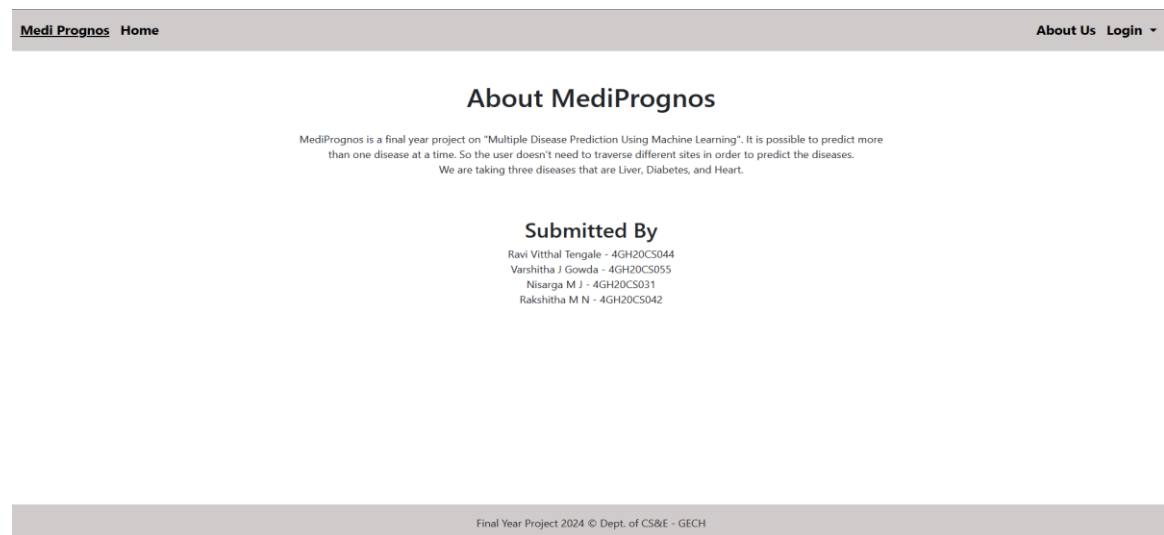


Figure 7.2: About Us page

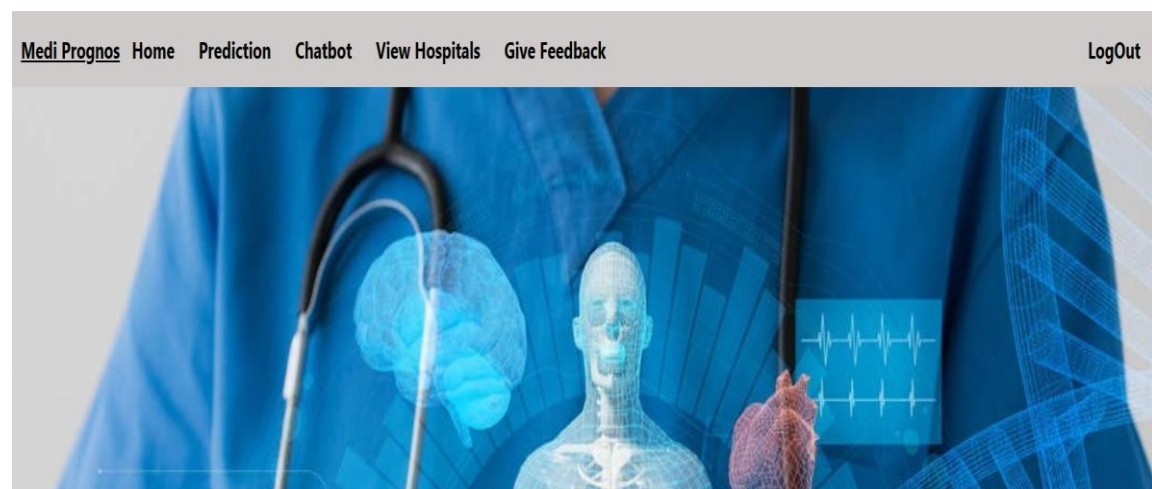
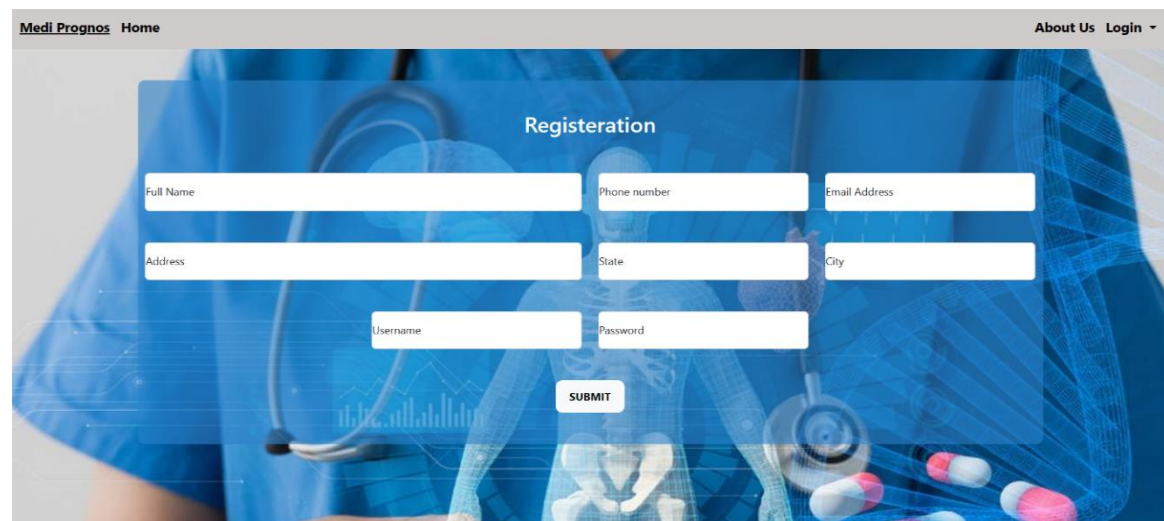
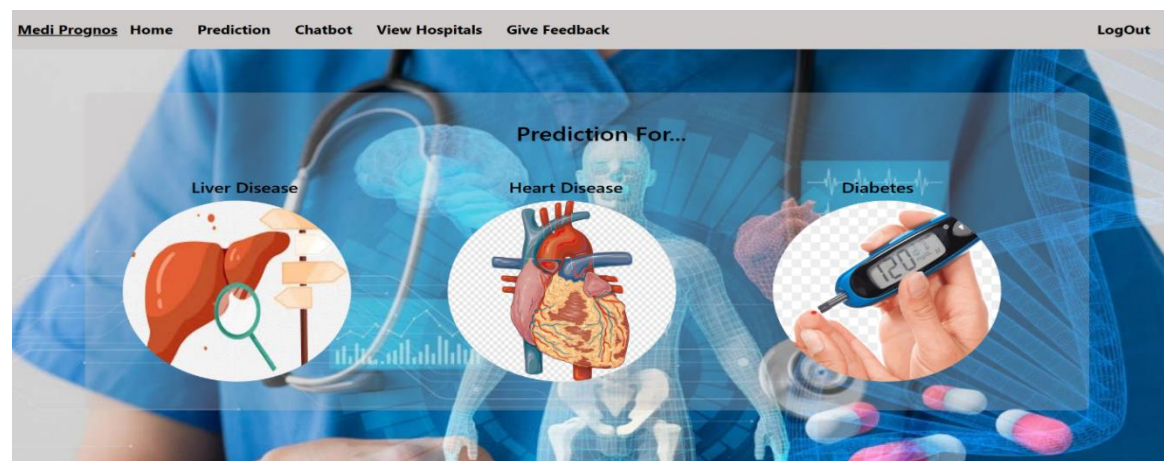
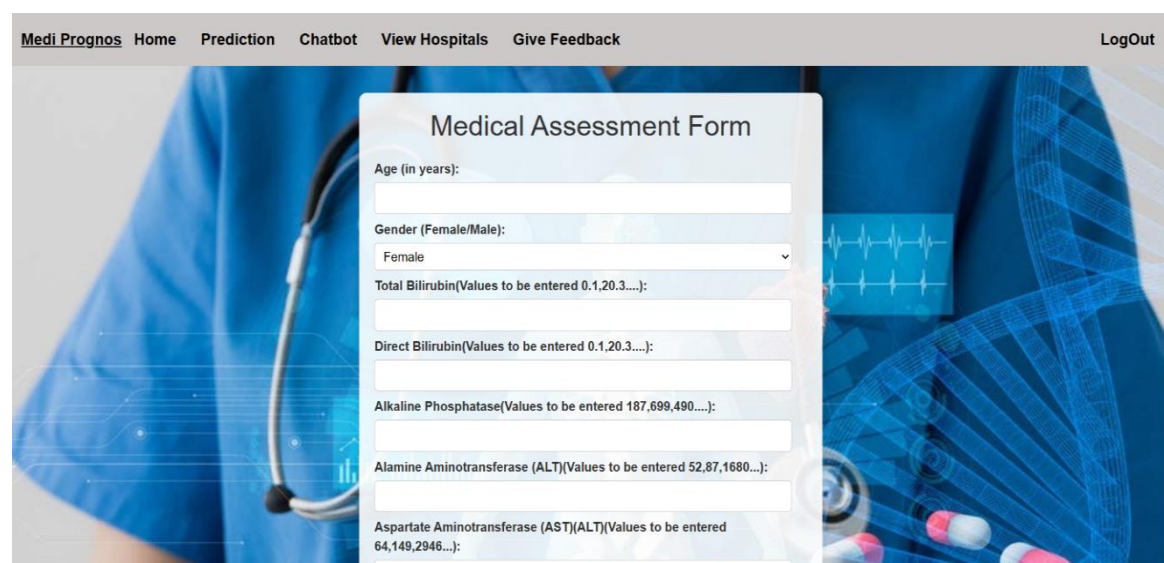


Figure 7.3: User Home page



The registration form is titled "Registration" and is set against a background of a doctor in blue scrubs with a stethoscope. The form includes input fields for "Full Name", "Phone number", "Email Address", "Address", "State", "City", "Username", and "Password". A "SUBMIT" button is located at the bottom center of the form.

Figure 7.4: User Registration Form**Figure 7.5: User Prediction page**

The "Medical Assessment Form" for Liver Disease includes the following fields: "Age (in years):", "Gender (Female/Male):" (with "Female" selected), "Total Bilirubin(Values to be entered 0.1,20.3....):", "Direct Bilirubin(Values to be entered 0.1,20.3....):", "Alkaline Phosphatase(Values to be entered 187,699,490....):", "Alamine Aminotransferase (ALT)(Values to be entered 52,87,1680....):", and "Aspartate Aminotransferase (AST)(ALT)(Values to be entered 64,149,2946....):".

Figure 7.6: Liver Disease Input Data

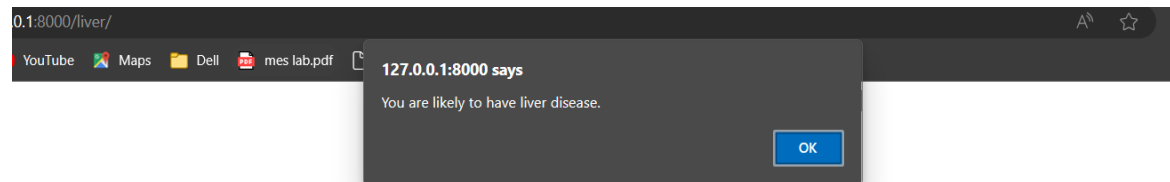


Figure 7.7: Liver Disease Output Result

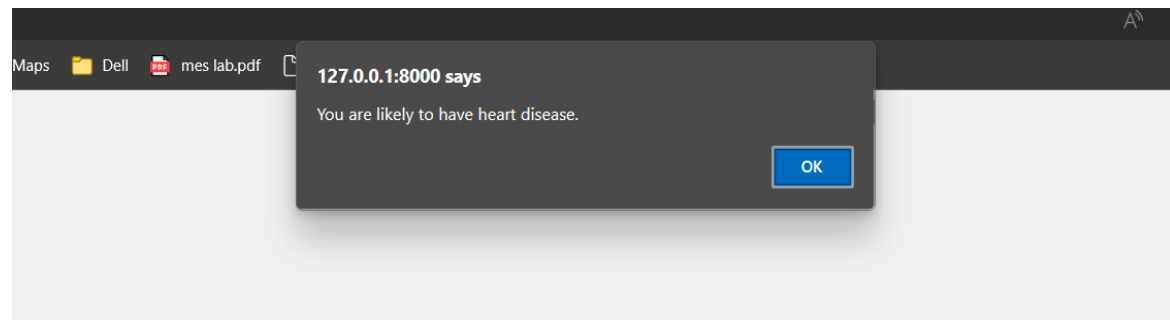


Figure 7.8: Heart Disease Output Result

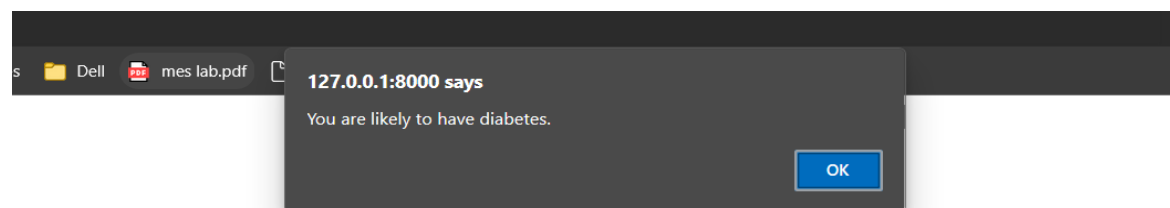


Figure 7.9: Diabetes Disease Output Result

Medi Prognos Home Prediction Chatbot View Hospitals Give Feedback LogOut					
Hospitals					
Hospital ID	Hospital Name	Contact Number	Emergency Number	Specialties	Actions
4	Jeeva Jyothi Hospital	9725392090	9725392091	Heart	View Details
5	Janapriya Hospital	7383059057	7383059058	Liver	View Details
6	Tanya Speciality Hospital	7947141565	7947141566	heart	View Details
7	Karna Multispeciality Hospital	7048877107	7048877108	Diabetes	View Details
8	Sahyaadri Multispeciality Hospital	7383181968	7383181967	liver	View Details
9	Janatha Hospital & Research Centre	7947133113	7947133114	heart	View Details
10	Hasanaba Multispeciality Hospital	9972371879	9972371880	Diabetes	View Details
11	JSS Hospital	7947135236	7947135236	liver	View Details
12	SparsH Hospital	7947135974	7947135975	Diabetes	View Details
13	Mani Super Speciality Hospital	7947141997	7947141998	heart	View Details

Figure 7.10: User Hospital View Page

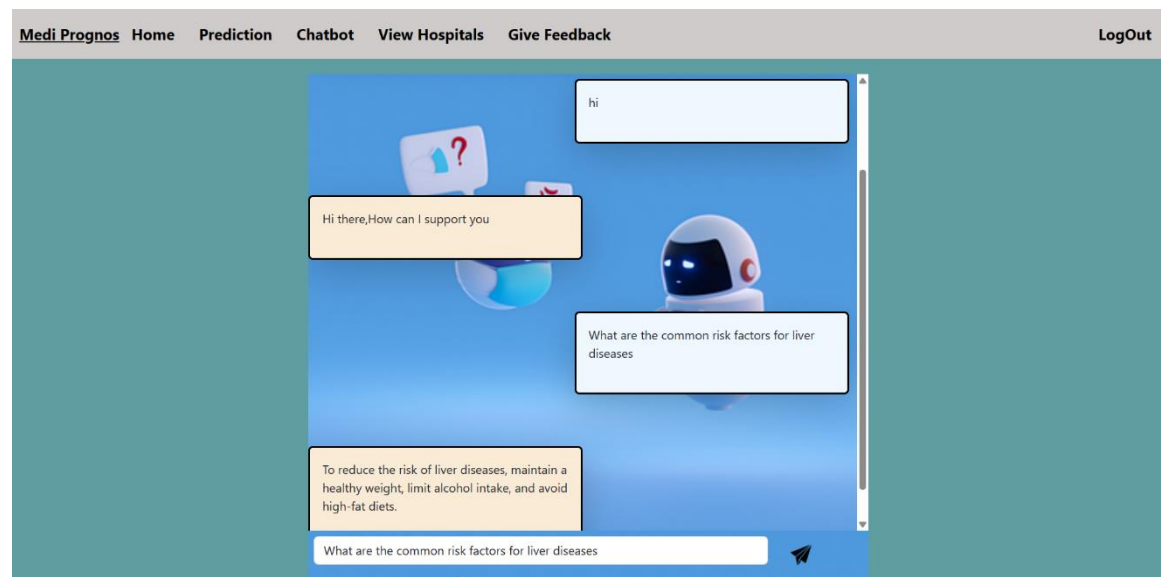


Figure 7.11: User Chatbot

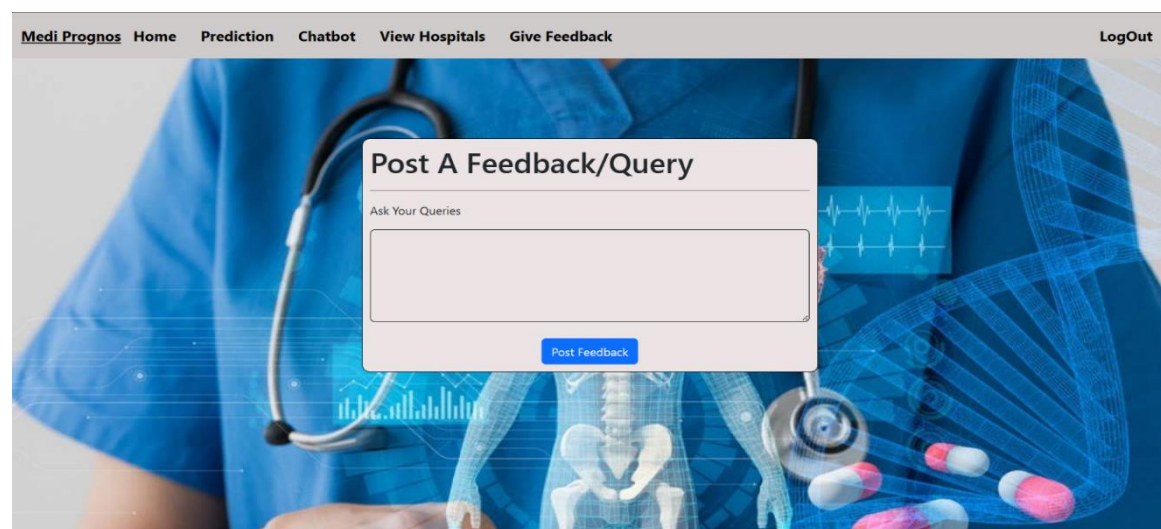


Figure 7.12: User Feedback View Page

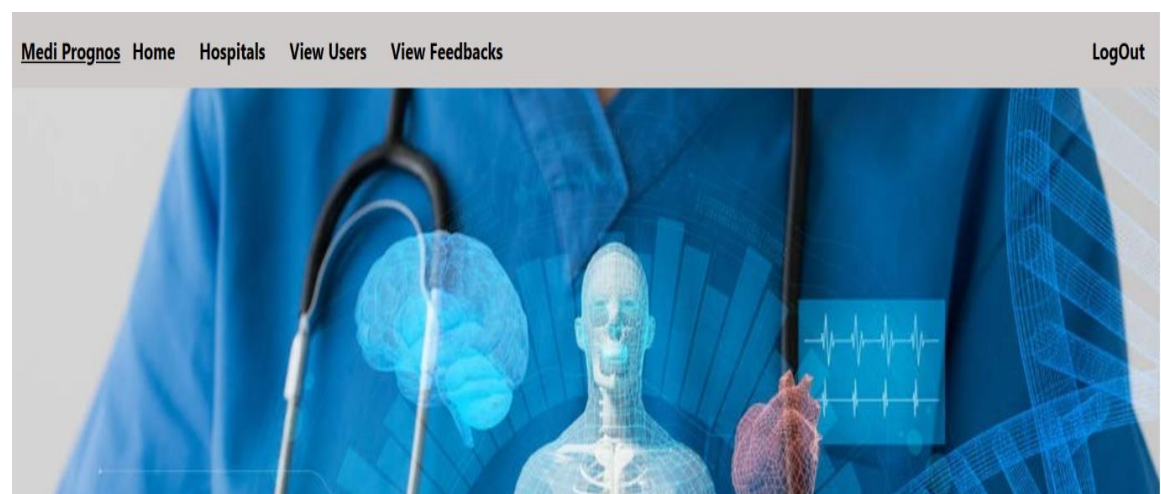


Figure 7.13: Admin Home page

Medi Prognos Home Hospitals View Users View Feedbacks Logout						
Hospital List						
Add Hospitals						
ID	Hospital Name	Contact Number	Emergency Contact	Specialities	Website	Actions
4	Jeeva Jyothi Hospital	9725392090	9725392091	Heart		View Details Update Details Delete
5	Janapriya Hospital	7383059057	7383059058	Liver		View Details Update Details Delete
6	Tanya Speciality Hospital	7947141565	7947141566	heart		View Details Update Details Delete
7	Karna Multispeciality Hospital	7048877107	7048877108	Diabetes		View Details Update Details Delete
8	Sahyaadri Multispeciality Hospital	7383181968	7383181967	liver		View Details Update Details Delete
9	Janatha Hospital & Research Centre	7947133113	7947133114	heart		View Details Update Details Delete
10	Hasanaba Multispeciality Hospital	9972371879	9972371880	Diabetes		View Details Update Details Delete
11	JSS Hosnital	7947135236	7947135236	liver		View Details Update Details Delete

Figure 7.14: Admin Hospital View Page

Medi Prognos Home Hospitals View Users View Feedbacks Logout							
View Users							
ID	Name	Phonenumber	Email Address	Username	Address	State	City
1	test	55443333433	test@gmail.com	test	ghghmgmmmgmg	ghtnnnn	thththht
2	varshita	8956214781	varshitha@gmail.com	varshitha	hassan	Karnataka	Hassan
3	ravi	8956214782	ravi@gmail.com	ravi	belagavi	Karnataka	belagavi
4	kiran	8956214783	kiran@gmail.com	kiran	hassan	Karnataka	Hassan

Figure 7.15: Admin User View Page

Medi Prognos Home Hospitals View Users View Feedbacks Logout		
ID	User ID	Query
1	1	comment
2	1	hihi
3	1	efeF
4	3	good website
5	1	hello

Figure 7.16: Admin Feedback View Page

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

8.1 CONCLUSION

The main objective of this project was to create a system that would predict more than one disease with high accuracy. The user doesn't need to traverse different websites which saves time as well. Diseases if predicted early can increase your life expectancy as well as save you from financial troubles. Multiple disease prediction model is used to predict multiple diseases at a time. Here based on the user input disease will be predicted. The choice will be given to the user. If the user wants to predict a particular disease or if the user doesn't enter any disease type then based on user entered inputs corresponding disease model will be invoked and predicted. The advantage of a multi disease prediction model in advance can predict the probability of occurrence of various diseases and also can reduce mortality ratio. In the future we can add more diseases in the existing API. We can try to improve the accuracy of prediction in order to decrease the mortality rate. Try to make the system user-friendly and provide a chatbot for normal queries.

8.2 FUTURE SCOPE

- 1. Early Detection and Diagnosis:** ML algorithms can analyze large datasets including medical records, imaging scans, genetic information, and lifestyle factors to identify patterns indicative of various diseases. Early detection can significantly improve treatment outcomes and prognosis.
- 2. Personalized Medicine:** ML models can analyze individual patient data to tailor treatment plans according to specific characteristics, such as genetic makeup, medical history, and lifestyle. This personalized approach can enhance the effectiveness of treatments and reduce adverse effects.
- 3. Predictive Analytics:** ML algorithms can be trained to predict disease progression and recurrence based on various factors. This can help healthcare providers anticipate and proactively manage patients' health, leading to better outcomes and reduced healthcare costs.

- 4. Drug Discovery and Development:** ML techniques, such as deep learning and reinforcement learning, can accelerate drug discovery by analyzing biological data, identifying potential drug candidates, and predicting their efficacy and safety profiles.
- 5. Remote Monitoring and Telemedicine:** ML-powered wearable devices and remote monitoring systems can continuously collect and analyze health data in real-time, enabling early detection of health issues and timely interventions, especially for chronic diseases.
- 6. Public Health Surveillance:** ML algorithms can analyze population-level health data from sources like electronic health records, social media, and wearable devices to identify disease outbreaks, monitor disease trends, and allocate resources efficiently.
- 7. Integration with Healthcare Systems:** ML-powered decision support systems can assist healthcare providers in clinical decision-making by providing evidence-based recommendations and alerts for potential health risks or interventions.
- 8. Challenges and Ethical Considerations:** Despite the potential benefits, there are challenges such as data privacy concerns, algorithm bias, interpretability of ML models, and regulatory hurdles that need to be addressed for the widespread adoption of ML in healthcare.

REFERENCES

- [1] Priyanka Sonar, Prof. K. JayaMalini,” DIABETES PREDICTION USING DIFFERENT MACHINE LEARNING APPROACHES”, 2019 IEEE ,3rd International Conference on Computing Methodologies and Communication (ICCMC)
- [2] Archana Singh ,Rakesh Kumar, “Heart Disease Prediction Using Machine Learning Algorithms”, 2020 IEEE, International Conference on Electrical and Electronics Engineering (ICE3)
- [3] A.Sivasangari, Baddigam Jaya Krishna Reddy,Annamareddy Kiran, P.Ajitha,” Diagnosis of Liver Disease using Machine Learning Models” 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)
- [4] Effective Heart Disease Prediction Using Hybrid Machine Learning Techniques Author - Senthilkumar Mohan, Chandrasegar Thirumalai, And Gautam Srivastava <https://ieeexplore.ieee.org/document/8740989>
- [5] An Empirical Evaluation of Machine Learning Techniques for Chronic Kidney Disease Prophecy Author - Bilal Khan, Rashid Naseem, Fazal Muhammad, Ghulam Abbas, And Sunghwan Kim, <https://ieeexplore.ieee.org/document/9040562>
- [6] Diabetes Prediction Using Ensembling of Different Machine Learning Classifiers Author - MD. Kamrul Hasan, MD. Ashraful Alam, Dola Das, Eklas Hossain, (Senior Member, IEEE), and Mahmudul Hasan <https://ieeexplore.ieee.org/document/9076634>
- [7] Kaggle Website for dataset collection (<https://www.kaggle.com/>)