# Title of project

**by**

**Student1 Name (Roll. No)**

**Student2 Name (Roll. No)**

**Student3 Name (Roll. No)**

**Student4 Name (Roll. No)**

**Department of Computer Science & Engineering- Data Science**

**ABES Engineering College**

**19th Km Stone, NH-09, Ghaziabad (U.P)**

**April, 2023**

## // Title of your project //

**by**

**Neha Gangwar (0903231070)**

**Neha Srivastava (0903231072)**

**Shweta Teotia (0903213054)**

**Smriti Singh (0903231112)**

**Submitted to the Department of CSE-Data Science**

**in partial fulfillment of the requirements**

**for the degree of**

**Bachelor of Technology**

**in**

**CSE-Data Science**

****

**ABES Engineering College, Ghaziabad**

**Dr. A.P.J. Abdul Kalam Technical University, Uttar Pradesh Lucknow**

**May, 2024**

## DECLARATION

*We hereby declare that this submission is our own work that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.*

*Signature:*

*Name:*

*Roll number:*

*Date:*

*Signature:*

*Name:*

*Roll number:*

*Date:*

*Signature:*

*Name:*

*Roll number:*

*Date:*

*Signature:*

*Name:*

*Roll number:*

*Date:*

## CERTIFICATE

This is to certify that the project report entitled “**XXXXXXXXXXXXXXXXXXX”** which is submitted by **Neha Gangwar, Neha Srivastava, Smriti Singh, and Shweta Teotia** in partial fulfillment of the requirement for the award of degree B.Tech. in the Department of Computer Science and Engineering-Data Science of Dr. A.P.J. Abdul Kalam, Technical University, is a record of the candidates’ work carried out by them under my supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree.

Date: (Supervisor Signature)

*Name Designation*

*CSE-Data Science*

ABES Engineering College, Ghaziabad.

## ACKNOWLEDGEMENT

*It gives us a great sense of pleasure to present the report of the B.Tech. Project undertaken during B.Tech Final Year. We owe special debt of gratitude to* ***Guide Designation & Name*** *Department of Computer Science & Engineering- Data Science, ABES Engineering College, Ghaziabad for his constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavours have seen light of the day.*

*We also take the opportunity to acknowledge the contribution of* ***Mr. Prabhat Singh,*** *Head of Department of Computer Science & Engineering- Data Science, ABES Engineering College, Ghaziabad for his full support and assistance during the development of the project.*

*We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.*

*Signature:*

*Name:*

*Roll No.*

*Date:*

*Signature:*

*Name:*

*Roll No. Date:*

*Signature:*

*Name:*

*Roll No.*

*Date:*

*Signature:*

*Name:*

*Roll No.*

*Date:*

## ABSTRACT

*The project resembles a device which can control various electrical devices simultaneously using a remote. The switching operation of the devices can be preset using remote according to the date time and year. This can also perform the operation repeatedly up to 100 years. The whole operation can be protected by a password which can be changed according to the privacy. This project based on the Atmel AT89C52 and Dallas real-time clock (RTC) chip DS12887 to control and remotely program the switching operation of the electrically operated devices. The devices can be switched on/off at precise times repeatedly every day, every month. The microcontroller can be programmed for device control using a normal Philips TV remote control.*

*Through this circuitry we can control the operation of devices by distant places. To build this system, sensing technology, long distance communication system, microcontroller has been used. This Project increases the operational security of devices in an Organization.*

[Title of project ii](#_Toc167660912)

[// Title of your project // iii](#_Toc167660913)

[DECLARATION iv](#_Toc167660914)

[CERTIFICATE v](#_Toc167660915)

[ACKNOWLEDGEMENT vi](#_Toc167660916)

[ABSTRACT 7](#_Toc167660917)

[Chapter –1 Introduction 18](#_Toc167660918)

[1.1 Health Care System 18](#_Toc167660919)

[1.1.1 Introduction 18](#_Toc167660920)

[1.1.2 Backgound and Context 18](#_Toc167660921)

[1.1.3 Problem Statement 18](#_Toc167660922)

[1.1.4 Objectives 18](#_Toc167660923)

[1.1.5 Scope 19](#_Toc167660924)

[1.2 History of Health and Machine Learning 19](#_Toc167660925)

[1.2.1 Early Beginnings (1950s-1970s) 19](#_Toc167660926)

[1.2.2 Development of Machine Learning Algorithms (1980s-1990s) 19](#_Toc167660927)

[1.2.3 The Rise of Big Data and Advanced Algorithms (2000s) 20](#_Toc167660928)

[1.2.4 Deep Learning and Real-World Applications (2010s) 20](#_Toc167660929)

[1.2.5 Current Trends and Future Directions (2020s and beyond) 20](#_Toc167660930)

[1.3 Rationale 21](#_Toc167660931)

[1.4 Project Timeline 22](#_Toc167660932)

[Chapter – 2 Literature Review 24](#_Toc167660933)

[Chapter-3 Methodology 25](#_Toc167660934)

[3.1 Introduction and Objective 25](#_Toc167660935)

[3.1.1 Introduction 25](#_Toc167660936)

[3.1.2 Objective 25](#_Toc167660937)

[3.2 Analysis of Problem Statement 25](#_Toc167660938)

[3.3 Project Scope 26](#_Toc167660939)

[3.3.1 Patient Management 26](#_Toc167660940)

[3.3.2 Predictive Analytics 26](#_Toc167660941)

[3.3.3 Medication Optimization 26](#_Toc167660942)

[3.3.4 Appointment Scheduling 26](#_Toc167660943)

[3.3.5 Technical Components 26](#_Toc167660944)

[3.4 Feasibility Study 27](#_Toc167660945)

[3.4.1 Technical Feasibility 27](#_Toc167660946)

[3.4.2 Economic Feasibility 28](#_Toc167660947)

[3.4.3 Operational Feasibility 28](#_Toc167660948)

[3.4.4 Schedule Feasibity 28](#_Toc167660949)

[3.4.5 Facilities required 28](#_Toc167660950)

[3.5 Project Planning and Timeline 28](#_Toc167660951)

[3.6 Team and Responsibilities 29](#_Toc167660952)

[3.7 Ethical Considerations 30](#_Toc167660953)

[3.8 Tools and Framework 31](#_Toc167660954)

[3.9 Data Collection 33](#_Toc167660955)

[3.10 Data Preprocessing 33](#_Toc167660956)

[3.11 Data Analysis 34](#_Toc167660957)

[3.12 Model Building 36](#_Toc167660958)

[3.13 Model Testing 37](#_Toc167660959)

[3.14 Model Deployment 37](#_Toc167660960)

[3.15 Django Framework 37](#_Toc167660961)

[3.16 Expected Outcomes 38](#_Toc167660962)

[Chapter-4 Health Care System 40](#_Toc167660963)

[4.1 Introduction to implementation 40](#_Toc167660964)

[4.1.1 Objective 40](#_Toc167660965)

[4.1.2 Scope 40](#_Toc167660966)

[4.2 Data Processing and Analysis 40](#_Toc167660967)

[4.2.1 Disease Prediction Data Set 40](#_Toc167660968)

[4.2.1.1-Data Attributes 40](#_Toc167660969)

[4.2.1.2-Data Head- 42](#_Toc167660970)

[4.2.1.3-Splitting the data into Symptoms and Disease 42](#_Toc167660971)

[4.2.1.4-Heatmap of df\_x 43](#_Toc167660972)

[4.2.2 Drug Prediction Data Set 43](#_Toc167660973)

[4.2.2.1-Data head and Categorical distribution 43](#_Toc167660974)

[4.2.2.2-Data Preprocssing 44](#_Toc167660975)

[4.2.2.3-Fitting the model 46](#_Toc167660976)

[4.2.2.4-Disease vs Age plot 46](#_Toc167660977)

[4.2.2.5-Disease vs Gender plot 47](#_Toc167660978)

[4.2.2.6-Heatmap 48](#_Toc167660979)

[4.3 Model Building 48](#_Toc167660980)

[4.3.1 Disease Prediction Model 48](#_Toc167660981)

[4.3.2 Drug Prediction Model 55](#_Toc167660982)

[4.4 Setting Up the Django Framework for Healthcare project 58](#_Toc167660983)

[4.4.1 Installing Python 58](#_Toc167660984)

[4.4.2 Installing Django 58](#_Toc167660985)

[4.4.3 Setting Up the project 59](#_Toc167660986)

[4.4.4 Setting Up the project app 60](#_Toc167660987)

[4.5 Django Admin 61](#_Toc167660988)

[4.6 Creating MySQL database 62](#_Toc167660989)

[4.7 Setting Up the Static file 65](#_Toc167660990)

[4.8 Setting Up the template folder inside the Core app 66](#_Toc167660991)

[4.8.1 Patient ajax.html 66](#_Toc167660992)

[4.8.2 Doctor ajax.html 66](#_Toc167660993)

[4.9 Setting Up the urls.py 66](#_Toc167660994)

[4.9.1 Importing files 67](#_Toc167660995)

[4.9.2 Setting the path 67](#_Toc167660996)

[4.10 Setting Up the views.py 67](#_Toc167660997)

[4.10.1 Importing files 68](#_Toc167660998)

[4.10.2 Home page 68](#_Toc167660999)

[4.10.3 User Registration 69](#_Toc167661000)

[4.10.4 User Login 69](#_Toc167661001)

[4.10.5 Doctor Authorization 70](#_Toc167661002)

[4.10.6 Home doctor dashboard 70](#_Toc167661003)

[4.10.7 Home patient dashboard 71](#_Toc167661004)

[4.10.8 User Profile 72](#_Toc167661005)

[4.10.9 Model Integration and Prediction 73](#_Toc167661006)

[4.10.10 Patient Result 74](#_Toc167661007)

[4.10.11 Appointment Requested by Patient 75](#_Toc167661008)

[4.10.13 Drug Recommendation 76](#_Toc167661009)

[4.10.14 Doctor approving Appointment 79](#_Toc167661010)

[4.10.15 Logout 80](#_Toc167661011)

[Chapter –5 Result and Discussion 81](#_Toc167661012)

[5.1 Result 81](#_Toc167661013)

[5.1.1 Patient Registration 81](#_Toc167661014)

[5.1.2 Patient Login 82](#_Toc167661015)

[5.1.3 Patient Profile 83](#_Toc167661016)

[5.1.4 Doctor Resgistartion 84](#_Toc167661017)

[5.1.5 Disease Prediction 87](#_Toc167661018)

[5.1.6 Appointment Request 88](#_Toc167661019)

[5.1.7 Drug Prediction 89](#_Toc167661020)

[5.1.8 Appointment Schedule 90](#_Toc167661021)

[5.2 Discussion 93](#_Toc167661022)

[5.2.1 User Authentication and Dashboard Functionality 93](#_Toc167661023)

[5.2.2 Modular Template Structure 93](#_Toc167661024)

[5.2.3 Database Management and User Profile 93](#_Toc167661025)

[5.2.4 Appointment Scheduling and Communication 93](#_Toc167661026)

[Chapter-6 Conclusion and Future Scope 94](#_Toc167661027)

[6.1 Conclusion 94](#_Toc167661028)

[6.2 Future Scope 94](#_Toc167661029)

[1. Enhanced Machine Learning Models: 94](#_Toc167661030)

[2. Telemedicine Integration: 95](#_Toc167661031)

[3. Mobile Application: 95](#_Toc167661032)

[References – 98](#_Toc167661033)

[Publication 99](#_Toc167661034)

[Plagiarisms Report 100](#_Toc167661035)

|  |  |  |
| --- | --- | --- |
| **No.** | **LIST OF TABLES**  **Title** | **Page No.** |
| 1 | Project Timeline | 17 |
| 2 | Data attributes | 36 |
| 3 | Classification report – 1 | 46 |
| 4 | Classification report – 2 | 50 |
| 5 | Classification report – 3 | 52 |
| 6 | Classification report – 4 | 54 |

|  |  |  |
| --- | --- | --- |
|  | **LIST OF FIGURES** |  |
| **No.** | **Title** | **Page No.** |
| 1 | Components of Feasibility Studies |  |
| 2 | Disease data head |  |
| 3 | df\_x data head |  |
| 4 | Heat map |  |
| 5 | Drug data head |  |
| 6 | Plotting data |  |
| 7 | Categorical distribution of drug dataset |  |
| 8 | Data preprocessing – 1 |  |
| 9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111 | Data preprocessing – 2  Data head after preprocessing  Model fitting  Plotting disease vs age  Disease vs Age plot  Plotting disease vs gender  Disease vs gender plot  Drug heatmap  Splitting of data for training and testing  Fitting gaussian naïve bayes  Confusion matrix – 1  Heatmap of confusion matrix – 1  Testing accuracy  Fitting decision tree classifier  Confusion matrix -2  Heatmap confusion matrix – 2  Calculating accuracy  Dumping decision tree model  Splitting of data for testing and training  Fitting the model and calculating accuracy  Fitting the data for decision tree  Model testing and calculating accuracy  Saving the model  Checking python version  Installing Django – 1  Installing Django – 2  Project setup  Setting up core app  Connecting the core app  Including the path  Starting the server  Admin panel  Connection MySql workbench  Importing files  Creating table – 1  Creating table – 2  Creating table – 3  Creating table – 4  Make migration  Migration for table members  Final migration  Applying all migrations  Database  Static file  Importing files  Setting up the paths  Importing files  Home page  Redirecting to home and register page  User registration function  Registration page  User login  Login page  Doctor home  Doctor dashboard  Patient home  Patient dashboard  User profile  User profile page  List of diseases  Taking symptoms from backend  Predicting disease  Disease prediction panel  Patient result  List of disease  Requesting appointment  Requesting appointment – 2  Taking the disease from the backend  Predicting drug – 1  Predicting drug – 2  Predicting drug – 3  Predicting drug – 4  Disease diagnosed  Appointment request  Schedule the appointment  Appointment schedule  Logout function  Logout  User registration  User registered  User login  User dashboard  Creating profile  Profile created  Doctor registration as patient  Registration succsedd fiulll  Activating doctors account  SQL query  Account activated  Doctor login  Disease prediction Pannel  Giving the symptoms and predicting the disease  Drug recommended  Appointment request  Appointment schedule  Drug prediction – 1  Drug prediction – 2  Medicine recommended  Scheduling appointment  Fixing date and time  Scheduled appointment  Patient appointment view  Logout |  |

# Chapter –1 Introduction

## 1.1 Health Care System

### 1.1.1 Introduction

In an era marked by enormous advancements in medical research and technology, the landscape of healthcare is ever evolving. Healthcare systems need to change to meet the needs of patients and providers as their needs change with the introduction of data-driven healthcare, the use of artificial intelligence, and the rising incidence of serious illnesses. The dynamic intersection of data analysis, illness prediction, and medical consultation within the healthcare system is the subject of this research. Predicting major diseases and identifying the age groups most susceptible to these conditions is our main goal through data analysis.

Medical personnel can now use large, complicated datasets to extract relevant insights thanks to the integration of machine learning (ML) and healthcare. This ushers in a new era of data-driven decision-making. From forecasting epidemics of disease.

Patient care, diagnosis, and treatment approaches are constantly being redefined in the dynamic field of healthcare by the convergence of technology and medicine. It is more important than ever to innovate and improve healthcare delivery as the globe struggles to deal with the COVID-19 epidemic and other health emergencies. Our project aims to use the Django framework to create an advanced healthcare system by utilizing machine learning algorithms and web-based apps.

### 1.1.2 Backgound and Context

A paradigm shift towards preventive and personalized medicine is occurring in the healthcare industry. While healthcare practitioners work to create individualized treatment regimens that maximize outcomes and minimize side effects, patients are increasingly looking for proactive methods for illness prevention, early identification, and management. These changing expectations are frequently not met by traditional healthcare systems because of their limitations like fragmented communication channels, segregated data, and inadequate predictive analytics capabilities.

### 1.1.3 Problem Statement

Our project aims to create an all-inclusive healthcare system with seamless integration of patient management, medication optimization, and predictive analytics features. Fundamentally, the system seeks to give people self-assessment and early disease detection capabilities while giving medical professionals useful information for individualized treatment planning and prescription management. Our approach aims to close the gap between symptom presentation and precise diagnosis by utilizing machine learning models and data-driven algorithms. This will improve patient outcomes and streamline healthcare operations.

### 1.1.4 Objectives

Our project's main goals include creating and deploying an easy-to-use patient dashboard that lets users enter personal information and symptoms to predict and assess risk for disease. Furthermore, our goal is to create a strong physician dashboard that has predictive analytics capabilities for managing medication prescriptions, treatment planning, and patient tracking. We will examine patient data and combine machine learning methods, such as Decision Trees and Random Forests, to produce precise forecasts for medication effectiveness and disease diagnosis. To promote smooth contact between patients and healthcare providers, our project will also set up secure communication channels and appointment scheduling features. To support future additions, upgrades, and interaction with other healthcare systems, it is critical to guarantee the healthcare system's scalability, dependability, and user-friendliness.

### 1.1.5 Scope

Our project's scope includes creating a prototype healthcare system with an emphasis on managing prescriptions and predicting diseases. The initial implementation focuses on common pharmaceuticals and conditions, but the extensible design and modular architecture open the possibility of future improvements and interaction with other specialist healthcare areas like neurology, cardiology, and cancer. Furthermore, interoperability with current telemedicine platforms and electronic health record (EHR) systems is made possible by the system architecture, which improves patient engagement and continuity of care.

## 1.2 History of Health and Machine Learning

The intersection of health and machine learning (ML) has a rich history, evolving significantly over the past few decades. Here is a detailed overview of the key milestones in this domain:

### 1.2.1 Early Beginnings (1950s-1970s)

1950s: The inception of artificial intelligence (AI) laid the groundwork for future ML applications in health. Early AI research aimed at mimicking human cognitive processes, which indirectly influenced healthcare.

1960s-1970s: Initial applications of computer-aided diagnosis began to appear. Systems like MYCIN, developed at Stanford University in the 1970s, used rule-based algorithms to identify bacterial infections and recommend antibiotics. Though primitive, MYCIN demonstrated the potential of AI in clinical decision support.

### 1.2.2 Development of Machine Learning Algorithms (1980s-1990s)

1980s: With the creation of increasingly complex algorithms like neural networks, decision trees, and Bayesian networks, the field of machine learning began to evolve. The first uses of these algorithms were in health, specifically in image analysis and predictive modeling.

1990s: T Processing of greater datasets became possible with the introduction of the internet and more potent computers. During this time, ML was used in radiology to identify abnormalities in X-rays and MRI scans, namely in medical imaging. Furthermore, the emergence of electronic health records, or EHRs, offered a wealth of data for building predictive models.

### 1.2.3 The Rise of Big Data and Advanced Algorithms (2000s)

2000s: T The broad use of EHRs and genetic data, which led to the explosion of big data in healthcare, opened new applications for machine learning. Methods like ensemble learning and supporting vector machines gained popularity.

2004: An important turning point was reached with the start of the Human Genome Project, which mapped the whole human genome. Advances in customized medicine resulted from the application of machine learning algorithms to analyze genetic data.

### 1.2.4 Deep Learning and Real-World Applications (2010s)

2010s: The area underwent a revolution with the advent of deep learning, specifically with the use of convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Deep learning models outperformed other models in challenges involving speech and image recognition.

2012: Google's DeepMind developed algorithms capable of learning and mastering complex games, showcasing the potential of deep learning.

2014: IBM Watson’s victory in the game show Jeopardy! highlighted AI’s potential in processing natural language and retrieving relevant information, which translated into healthcare applications for analyzing medical literature and clinical data.

2015: Google’s DeepMind applied deep learning to healthcare, resulting in systems capable of diagnosing eye diseases from retinal scans with high accuracy.

### 1.2.5 Current Trends and Future Directions (2020s and beyond)

2020s: With developments in reinforcement learning, federated learning, and natural language processing (NLP), the use of machine learning (ML) in healthcare is growing quickly. These days, machine learning (ML) models are utilized for medication development, tailored treatment planning, disease outbreak prediction, and hospital operations optimization.

COVID-19 Pandemic: The pandemic accelerated the adoption of ML in healthcare for predicting virus spread, developing diagnostic tools, and expediting vaccine research.

## 1.3 Rationale

The need to meet the rising demands of contemporary healthcare is driving research into data-driven healthcare, disease prediction, and physician consultation inside the healthcare system. This work's importance and worth are highlighted by its focus on related fields.

The project's primary need was prompted by the COVID-19 pandemic, in which individuals with various health concerns felt ignored as the healthcare system focused on treating COVID emergency. Due to the paucity of doctors, these patients frequently suffered from ignorance about the nature of their ailments or the proper drugs. To solve this, our study uses a dataset that includes information about a variety of health issues, possible illnesses, and related prescription medications to build a machine learning model.

Mental health problems like stress, despair, and loneliness increased in frequency during the pandemic. It can be difficult to diagnose these conditions since people are sometimes unwilling to talk about their troubles. Our machine learning algorithm will help with early intervention by predicting the likelihood that an individual may experience mental health concerns.

Additionally, our project will help consumers choose health insurance plans that best fit their needs and budget. By utilizing machine learning and data analytics, this will give people advantages like financial security, peace of mind, and access to top-notch healthcare.

Additional requirements for the project are as follows:

Transformation of Healthcare: Advances in technology and data analytics are transforming healthcare, necessitating an understanding of these changes.

Data as a Catalyst: Data-driven techniques offer the potential for more personalized, efficient, and preventive care.

Disease Prognosis for Early Intervention: Early illness prediction models can improve patient outcomes and reduce healthcare costs.

Age-Group Specific Approaches: Recognizing age-related illness patterns allows for targeted therapies, enhancing healthcare delivery.

The Expanding Role of Telemedicine: Telemedicine increases healthcare accessibility and convenience by overcoming geographical barriers and inequities.

Ethical and Privacy Considerations: Ensuring ethical practices regarding patient privacy, data security, and bias is crucial for fair healthcare benefits.

Ultimately, the need to adjust to and prosper in a quickly evolving healthcare ecosystem is what drives research into data-driven healthcare, disease prediction, and physician consultation. Healthcare systems can secure the sustainability of healthcare delivery in an era defined by data and technology, improve patient-centric care, address current difficulties and embrace innovative applications by utilizing data analytics.

## 1.4 Project Timeline

Our project timeline encompasses a series of structured phases, each focusing on specific tasks and deliverables:

Requirement Analysis and Stakeholder Engagement: Collaborate with healthcare professionals, patients, and domain experts to identify user requirements, system functionalities, and regulatory considerations.

System Design and Architecture: Design the system architecture, user interface, and database schema, considering scalability, interoperability, and security requirements.

Data Collection and Preprocessing: Gather and preprocess diverse datasets, including patient demographics, symptoms, medical history, and drug efficacy data, ensuring data quality, privacy, and compliance with regulatory standards.

Implementation of Patient and Doctor Dashboards: Develop intuitive and responsive web interfaces for patients and healthcare providers, incorporating features such as symptom input, risk assessment, drug recommendation, and appointment scheduling.

Integration of Machine Learning Models: Implement machine learning algorithms, including Decision Trees and Random Forests, for disease prediction and drug efficacy analysis, fine-tuning model parameters and evaluating performance metrics.

Testing and Validation: Conduct comprehensive testing, including unit tests, integration tests, and user acceptance tests, to validate system functionality, performance, and usability across diverse use cases and user scenarios.

Deployment and User Training: Deploy the healthcare system on a secure and scalable infrastructure, providing user training, technical support, and documentation to facilitate seamless adoption and utilization by healthcare stakeholders.

Following is the execution plan of the project -

|  |  |
| --- | --- |
| Task | Deadline |
| Problem analysis | 25th August 2023 |
| Feasibility study | 7th September 2023 |
| Data preprocessing module and Exploratory data analysis | 14th September 2023 |
| Data analysis | 25th September 2023 |
| Model building | 30th September 2023 |
| Model testing | 10th October 2023 |
| Backend development | 15th October 2023 |
| Frontend development | 25th October 2023 |
| Web app deployment | 5th November 2023 |

Table – 1 Deadline

# Chapter – 2 Literature Review

One such method, proposed by Jianfang et al. [1], employs the Support Vector Machine (SVM) for disease classification based on symptoms. SVM is known for its efficiency in prediction; however, it can be time-consuming, and its accuracy may be limited by the challenge of using a hyperplane for classification. This method is particularly effective for binary classifications but less so for scenarios involving more than two disease classes [2, 3, 4].

Another notable approach is the K-Nearest Neighbors (KNN) algorithm, utilized by Keniya et al. [5] and Kashvi et al. [6]. KNN assigns a data point to the class most represented by its nearest neighbors. While KNN is straightforward, it can be sensitive to noisy or missing data and may exhibit lower accuracy when dealing with complex datasets involving multiple parameters such as age group, symptoms, and gender [5, 6]. Despite its potential, the challenge of working with smaller datasets for classification remains an obstacle .

In contrast, the Naïve Bayes method has been employed by Pingale et al. , Gomathy and Rohith Naidu [18], and Chhogyal and Nayak [19] for disease prediction. These methods are characterized by their simplicity and accessibility, making them suitable for applications like web-based disease prediction tools. However, their accuracy can be compromised by the limited scope of diseases considered and the quality of training data utilized .

Furthermore, Kumar et al. introduced the Rustboost Algorithm to address class imbalance issues in disease prediction. While this approach shows promise in handling imbalanced datasets, the use of random under-sampling as a resampling method may lead to the loss of critical information during model training .

Collectively, these machine learning techniques represent a diverse array of approaches for disease prediction, each with its own strengths and limitations. Despite the advancements in machine learning models for disease prediction, there are common challenges that need to be addressed. Issues such as efficiency, accuracy, the limited size of training datasets, and the scope of symptoms considered remain significant hurdles.

To overcome these challenges, there is a clear call for the development of modified and more accurate models for predicting human diseases. The subsequent sections of the research paper likely delve into proposing and detailing such a model, aiming to enhance the effectiveness and reliability of disease prediction systems in healthcare settings.

# Chapter-3 Methodology

The methodical strategy and organized approach that will be employed to accomplish the project's objectives are described in the section on project methodology. It ensures clarity, reproducibility, and rigor by covering all methods, approaches, and instruments employed during the project. Here is a thorough explanation of the project process for, Health Care System together with its complete structure:

## 3.1 Introduction and Objective

### 3.1.1 Introduction

The medical field is constantly changing in this period of tremendous technological and scientific developments. With the advent of data-driven healthcare, the use of artificial intelligence, and the rise in the prevalence of serious illnesses, healthcare systems must adapt to suit the demands of both patients and providers. This study focuses on the dynamic interaction of medical consultation, disease prediction, and data analysis within the healthcare system. Our primary objective through data analysis is to predict major diseases and identify the age groups most susceptible to these conditions.

The dynamic intersection of technology and medicine is continuously redefining patient care, diagnosis, and treatment techniques in the healthcare industry. As the world tries to contain the COVID-19 pandemic and other health problems, it is more crucial than ever to innovate and improve healthcare services. Our project uses web-based apps and machine learning algorithms to build an advanced healthcare system using the Django framework.

### 3.1.2 Objective

One of the primary objectives of our project is to develop and implement a user-friendly patient dashboard that allows users to input personal data and symptoms in order to estimate and evaluate risk for illness. In addition, we want to develop a powerful physician dashboard with predictive analytics features for tracking patients, treatment planning, and prescription management. To provide accurate predictions for pharmaceutical effectiveness and disease diagnosis, we will integrate machine learning techniques, such as Random Forests and Decision Trees, with patient data analysis. Our project will also set up secure communication channels and appointment scheduling capabilities to facilitate easy communication between patients and healthcare providers. Ensuring the scalability, dependability, and user-friendliness of the healthcare system is essential to accommodate future updates, additions, and integration with other systems.

## 3.2 Analysis of Problem Statement

Develop a patient dashboard for managing health records, appointments, and doctor communication in addition to a doctor dashboard for accessing patient records, scheduling appointments, and writing prescriptions in order to establish a comprehensive healthcare system with integrated medication optimization, patient management, and predictive analytics. Use a machine learning model to forecast diseases based on symptoms reported by patients, then present the findings for a doctor's review. Keep an extensive medication database and offer prescription management solutions that take the patient's medical history and possible drug interactions into account. With calendar integration and notification systems, you may facilitate appointment scheduling that is started by the patient or by the doctor. Make sure your relational database is used for solid backend development.

## 3.3 Project Scope

Develop an all-inclusive healthcare system that integrates patient management, medication optimization, and predictive analytics to enhance patient care and streamline healthcare processes.

### 3.3.1 Patient Management

* Patient Dashboard: Interface for patients to manage personal health records, view upcoming appointments, and communicate with healthcare providers.
* Doctor Dashboard: Interface for doctors to access patient records, manage appointments, prescribe medications, and review predictive analytics.

### 3.3.2 Predictive Analytics

* Symptom Input: Feature for patients to input their symptoms.
* Disease Prediction Model: Machine learning model that predicts potential diseases based on patient symptoms.
* Prediction Display: Display of predicted diseases with confidence scores for doctors to review and validate.

### 3.3.3 Medication Optimization

* Prescription Management: Tools for doctors to prescribe medications based on predicted diseases and patient medical history.
* Medication Database: Comprehensive database containing information on medications, their uses, dosages, and interactions.

### 3.3.4 Appointment Scheduling

* Doctor-initiated Appointments: Doctors can schedule appointments for patients, which are then displayed on the patient dashboard.
* Patient-initiated Requests: Patients can request appointments, and doctors can approve and schedule them.

### 3.3.5 Technical Components

#### **3.3.5.1** Backend Development

* Database: Design and implement a relational database for storing patient records, symptoms, disease predictions, medication information, and appointments.
* Security: Implement encryption, authentication, and authorization protocols to protect patient data.

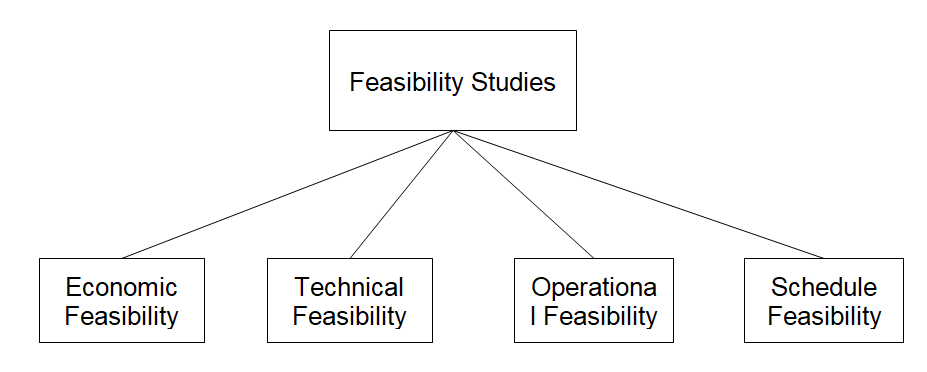
#### 3.3.5.2 Frontend Development

* User Interface: Design intuitive and responsive interfaces for both patient and doctor dashboards.
* User Experience: Ensure ease of use and accessibility for all users.

#### 3.3.5.3 Machine Learning

* Data Collection and Preprocessing: Gather and preprocess data for training the disease prediction model.
* Model Training and Deployment: Train the model using appropriate algorithms and deploy it in a scalable environment.
* Continuous Learning: Update the model with new data to improve accuracy over time.

## 3.4 Feasibility Study

Figure – 1 Components of Feasibility Studies

The feasibility study is the first part of the project, and it evaluates the project's viability, necessity, and relevance. This study will look into:

### 3.4.1 Technical Feasibility

An essential part of determining if a large-scale project, such as a hospital system, is viable is conducting a technical feasibility study. Its main objective is to assess the technical viability of implementing the suggested healthcare system. To eliminate redundancies from the data set, the project requires a complete data set, which will then undergo data preparation. To determine the relationships and interdependencies between the data, data analysis will be performed. In the third section, a web application built on the Django framework will be constructed. To hold user data, the project will feature an extensive database. Because the technology and infrastructure are readily available, the project can be implemented technically.

### 3.4.2 Economic Feasibility

An economic feasibility analysis is an essential part of determining if a healthcare system is viable. Its main objective is to assess the financial implications of setting up and running the healthcare system. The only expense associated with creating the project is time, as it is built using pre-existing backend and frontend technologies and an existing data source.Other than $4–$5 to install the project on cloud platforms, no investment is needed for it.

### 3.4.3 Operational Feasibility

A vital component of a feasibility assessment for a healthcare system is operational viability. It evaluates the feasibility of operating the suggested healthcare system effectively and efficiently in order to achieve the desired aims and objectives.People with data science, front-end, and back-end development skills are essential to the health care system's operations.For our project, we need four people: a front-end developer, a back-end developer, and two members of the data science stream.The health care system will be adaptable to future changes and efficient.It will be able to provide an easy-to-use, interactive GUI together with an effective backend system.The correctness of the data set and the model's forecast is the only risk associated with this project, thus we need to achieve high accuracy

### 3.4.4 Schedule Feasibity

A healthcare system feasibility assessment must consider schedule feasibility. It evaluates if the project can be finished in the allotted time and whether the schedule is reasonable and doable.The project will take about four to five months to complete, during which time the work will be divided among the group members who are each responsible for a certain area of expertise. This will save time and lower risk, both of which will ultimately boost efficiency.

### 3.4.5 Facilities required

* A high-quality dataset on healthcare in the areas of insurance, mental health, and medical care.
* GPU with high computational power to construct intricate neural networks.
* A database with low latency for quicker communication.
* Strong encryption software to safeguard user information.
* Front-end framework for GUI development.
* A communication API structure.

## 3.5 Project Planning and Timeline

Our project schedule is divided into several organized stages, with each stage concentrating on particular duties and outputs:

1. Requirement analysis and Stakeholder Engagement: Work together to determine user requirements, system functions, and regulatory considerations with patients, healthcare professionals, and domain experts.
2. System Design and Architecture: Create the database schema, user interface, and system architecture while taking security, scalability, and interoperability needs into account.
3. Data Collection and Preprocessing: To ensure data quality, privacy, and compliance with regulatory standards, collect and preprocess a variety of datasets, including patient demographics, symptoms, medical histories, and drug efficacy data.
4. Putting in place dashboards for doctors and patients: Provide user-friendly, responsive web interfaces with functions including medication suggestion, risk assessment, symptom entry, and appointment scheduling for patients and healthcare professionals.
5. Integrating Machine Learning Models: Apply machine learning methods for drug efficacy analysis and disease prediction, such as Random Forests and Decision Trees. Adjust model parameters and assess performance metrics.
6. Testing and Validation: To verify system functionality, performance, and usability across a range of use cases and user scenarios, conduct thorough testing, including unit tests, integration tests, and user acceptance tests.
7. Deployment and User Training: To enable smooth acceptance and use by healthcare stakeholders, deploy the healthcare system on a safe and scalable infrastructure and provide user training, technical support, and documentation.

## 3.6 Team and Responsibilities

Anand Kumar Rohal

* Research Papers: Conduct and document research to support the project.
* Database Development – Developed and wrote migrations for the database.
* Patent Ideas: Develop and document innovative ideas for potential patents.
* Data Analysis: Analyze the data to extract meaningful insights.

Abhinav Akash Tripathi

* Machine Learning Models: Develop and train the machine learning models for disease prediction.
* Backend Development: Implement the backend using Django, including database management and API development.
* Dashboard Integration: Integrate the machine learning models and data management into the doctor and patient dashboards.
* Documentation: Create comprehensive documentation for the project.

Kamakshi Agrawall

* Data Preprocessing: Prepare and clean the data for analysis and model training.

Akshat Sharma

* UI/UX Design: Design the user interfaces for the patient and doctor dashboards.
* Frontend Development: Implement the frontend using HTML, CSS, and JavaScript.

## 3.7 Ethical Considerations

* Data Encryption: Implement robust encryption methods for data storage and transmission to prevent unauthorized access.
* Access Controls: Ensure strict access control mechanisms so that only authorized personnel can access sensitive patient data.
* Informed Consent
* Data Usage Transparency: Clearly inform patients how their data will be used, particularly regarding data collected for predictive analytics and machine learning.
* Bias and Fairness in Predictive Analytics
* Bias Mitigation: Ensure the machine learning models are trained on diverse datasets to avoid biases that could lead to unfair treatment of certain patient groups.
* Regular Audits: Conduct regular audits of the predictive analytics models to identify and rectify any biases or inaccuracies.
* Explainability: Ensure that the predictions made by the machine learning models are explainable and understandable to healthcare providers, enabling them to make informed decisions.
* Accuracy and Reliability of Predictions
* Model Validation: Regularly validate and test the predictive models to ensure their accuracy and reliability.
* Continuous Learning: Implement a system for continuous learning and updating of the models to improve their predictive accuracy over time.
* Human Oversight: Ensure that all predictions are reviewed by qualified healthcare professionals before any clinical decisions are made.
* Patient Autonomy and Empowerment
* Patient Control: Allow patients to have control over their health information and how it is used within the system.
* Educational Resources: Provide patients with educational resources to help them understand their health data and the implications of predictive analytics.
* Transparency and Accountability
* Clear Communication: Maintain clear and open communication with patients about the capabilities and limitations of the system.
* Ethical Use of AI
* Beneficence and Non-Maleficence: Ensure that the AI-driven components of the system are designed to benefit patients and do no harm.
* Ethical AI Guidelines: Follow established ethical guidelines for AI development and deployment, ensuring the system’s integrity and ethical soundness.
* Equity in Access
* Accessibility: Ensure the system is accessible to all patients, regardless of socioeconomic status, geographic location, or technological proficiency.

## 3.8 Tools and Framework

Jupyter Notebook

* Definition: Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. It supports various programming languages, including Python, and is widely used for data cleaning, transformation, numerical simulation, statistical modeling, data visualization, and machine learning.
* Usage in Project: Used for data preprocessing, analysis, and developing machine learning models.

VS Code (Visual Studio Code)

* Definition: Visual Studio Code is a free, open-source code editor developed by Microsoft. It includes support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. It is highly customizable with a variety of extensions available.
* Usage in Project: Used for writing and editing code, developing backend and frontend components, and overall project management.

Python

* Definition: Python is a high-level, interpreted, and general-purpose programming language known for its readability and versatility. It has extensive libraries and frameworks that support data analysis, machine learning, web development, automation, and more.
* Usage in Project: Used for developing machine learning models, backend logic, and data analysis tasks.

MySQL Workbench

* Definition: MySQL Workbench is a unified visual tool for database architects, developers, and DBAs. It provides data modeling, SQL development, and comprehensive administration tools for server configuration, user administration, backup, and more.
* Usage in Project: Used for designing, managing, and interacting with the relational database that stores patient records, symptoms, disease predictions, medication information, and appointments.

HTML (HyperText Markup Language)

* Definition: HTML is the standard markup language used for creating web pages and web applications. It structures the content on the web and is used to define elements such as headings, paragraphs, links, images, and other types of multimedia.
* Usage in Project: Used for structuring the content of the frontend user interfaces.

CSS (Cascading Style Sheets)

* Definition: CSS is a style sheet language used for describing the presentation of a document written in HTML or XML. It defines how HTML elements are to be displayed, including layout, colors, fonts, and overall visual appearance.
* Usage in Project: Used for styling the frontend user interfaces to ensure a visually appealing and consistent design.

JavaScript

* Definition: JavaScript is a high-level, interpreted programming language that enables interactive web pages. It is a core technology of the World Wide Web, alongside HTML and CSS. JavaScript allows the implementation of dynamic features such as interactive forms, animations, and other user interactions.
* Usage in Project: Used for adding interactive elements to the frontend user interfaces and for client-side scripting.

Django Framework

* Definition: Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. It follows the model-template-views (MTV) architectural pattern and provides tools for building robust web applications with reusable components.
* Usage in Project: Used for developing the backend of the healthcare system, including the patient and doctor dashboards, integrating the machine learning models, and managing the database interactions.

## 3.9 Data Collection

## 3.10 Data Preprocessing

Data Preprocessing:

* Definition: Data preprocessing is a crucial step in the data analysis and machine learning pipeline. It involves transforming raw data into a clean and usable format, preparing it for analysis and modeling. This process includes tasks such as data cleaning, normalization, transformation, and reduction to enhance the quality and performance of the machine learning model.
* Steps Involved:
  + Data Cleaning: Removing or correcting errors, inconsistencies, and missing values in the dataset.
  + Data Integration: Combining data from different sources into a coherent dataset.
  + Data Transformation: Converting data into suitable formats or structures, such as normalizing numerical values or encoding categorical variables.
  + Data Reduction: Reducing the volume of data while maintaining its integrity, which might include feature selection and dimensionality reduction.
  + Data Normalization: Scaling data to fit within a specific range, improving the model's convergence during training.

Data Preprocessing for Disease and Drug CSV Files

Process Overview:

* Loading the Data:
  + Read the Disease and Drug CSV files into data frames using a tool like Pandas in Python.
* Data Cleaning:
  + Remove Redundancies: Identify and eliminate duplicate rows or records to ensure the dataset's integrity.
  + Handle Missing Values: Fill or remove missing data points to prevent inaccuracies in the model.
  + Correct Inconsistencies: Standardize formats, correct typos, and ensure consistency across the dataset.
* Data Integration:
  + Combine disease and drug data if needed, ensuring a unified structure for analysis.
* Data Transformation:
  + Normalization: Scale numerical features to a standard range (e.g., 0 to 1) to improve model performance.
  + Encoding: Convert categorical variables into numerical format using techniques like one-hot encoding or label encoding.
* Data Reduction:
  + Feature Selection: Select relevant features that contribute most to the prediction, reducing the dimensionality of the data.
  + Dimensionality Reduction: Use techniques like Principal Component Analysis (PCA) to further reduce the number of features while retaining essential information.

Outcome:

The preprocessing of Disease and Drug CSV files ensures that the dataset is clean, consistent, and ready for modeling. By removing redundancies and handling inconsistencies, the accuracy and efficiency of the machine learning model are significantly improved.

By carefully preprocessing the data, the project ensures that the resulting machine learning models are trained on accurate and high-quality datasets, ultimately leading to better predictions and more reliable outcomes.

### 3.11 Data Analysis

Data Analysis:

* Definition: Data analysis is the process of inspecting, cleaning, transforming, and modeling data to uncover meaningful insights, patterns, and trends. It involves applying statistical and computational techniques to understand the structure and content of datasets, extract valuable information, and support decision-making processes.
* Purpose: Data analysis helps organizations and researchers make informed decisions, solve problems, and improve efficiency by leveraging the knowledge hidden within large volumes of data.

Data Analysis Process for Heatmap and Correlation Matrix

Process Overview:

* Loading the Data:
  + Load the preprocessed dataset into a data analysis tool or library like Pandas in Python.
* Exploratory Data Analysis (EDA):
  + Summary Statistics: Calculate descriptive statistics such as mean, median, standard deviation, etc., to understand the central tendency and variability of the data.
  + Visualization: Create visual representations of the data using plots and charts to identify patterns and relationships.
* Heatmap and Correlation Matrix:
  + Correlation Analysis: Compute pairwise correlations between numerical features in the dataset to measure the strength and direction of their linear relationships.
  + Heatmap Visualization: Generate a heatmap visualization of the correlation matrix to visually represent the correlation coefficients between features. This helps identify highly correlated features and potential multicollinearity issues.
* Interpretation and Insights:
  + Analyze the heatmap and correlation matrix to identify hidden relationships and dependencies between variables.
  + Use the insights gained to guide feature selection, model building, and further data exploration.

Outcome:

The heatmap and correlation matrix provide valuable insights into the relationships between variables in the dataset. By visualizing the correlations, researchers can identify important features, detect multicollinearity, and gain a deeper understanding of the data's structure. This information is crucial for optimizing model performance and making informed decisions during the modeling process.

### 3.12 Model Building

Machine Learning Model Definition

Machine Learning Model:

* Definition: A machine learning model is a mathematical representation of a real-world process that is learned from data. It is trained on historical data to make predictions or decisions without being explicitly programmed for specific tasks. Machine learning models use algorithms and statistical techniques to identify patterns, relationships, and trends in data, enabling them to make accurate predictions or classifications on new, unseen data.
* Purpose: Machine learning models are used across various domains, including healthcare, finance, retail, and more, to automate tasks, make predictions, and gain insights from data.

Disease Prediction Model and Drug Prediction Model

Disease Prediction Model:

* Definition: The disease prediction model is a machine learning model trained on historical patient data to predict the likelihood of a patient having a particular disease based on their symptoms, medical history, and other relevant factors. It uses algorithms such as logistic regression, decision trees, or neural networks to analyze the input features and generate predictions.
* Integration: The disease prediction model is integrated into the patient dashboard, allowing patients to input their symptoms and receive predictions about potential diseases. This empowers patients to proactively manage their health and seek appropriate medical care.

Drug Prediction Model:

* Definition: The drug prediction model is a machine learning model trained on historical patient and medication data to recommend appropriate drugs or treatments for specific diseases or conditions. It analyzes factors such as patient demographics, disease severity, medication efficacy, and potential side effects to generate personalized recommendations.
* Integration: The drug prediction model is integrated into the doctor dashboard, enabling healthcare providers to prescribe medications based on predicted diseases and patient characteristics. This assists doctors in making informed decisions about treatment plans and medication choices for their patients.

Key Characteristics:

* Accuracy: The models are evaluated based on their ability to accurately predict diseases and recommend appropriate drugs, ensuring high-quality patient care.
* Scalability: The models should be scalable to handle large volumes of patient data and accommodate future growth in the healthcare system.
* Interpretability: The models should provide transparent and interpretable results, allowing healthcare providers and patients to understand the reasoning behind the predictions.
* Continual Improvement: The models should be continuously monitored and updated with new data to maintain their accuracy and effectiveness over time.

By integrating machine learning models into the healthcare system, healthcare providers and patients can benefit from more efficient and personalized medical care, leading to improved health outcomes and patient satisfaction.

### 3.13 Model Testing

After training the machine learning models, they were tested with the remaining 80% of the data, and achieved good accuracy. This indicates that the models are performing well in predicting diseases and recommending drugs based on the input data. The high accuracy achieved during testing demonstrates the effectiveness of the models in making accurate predictions on unseen data, which is essential for their practical application in real-world scenarios. This successful testing phase reinforces confidence in the reliability and usability of the disease prediction model integrated into the patient dashboard and the drug prediction model integrated into the doctor dashboard.

### 3.14 Model Deployment

After successful testing, the machine learning models were deployed to the healthcare system. Deployment involves integrating the trained models into the healthcare system's infrastructure, making them accessible and operational for real-time use. The deployment process ensures that the models can efficiently process incoming data, provide predictions or recommendations, and seamlessly integrate with the patient and doctor dashboards. By deploying the models to the healthcare system, healthcare providers and patients can leverage their predictive capabilities to enhance decision-making, improve patient care, and optimize treatment outcomes.

### 3.15 Django Framework

The healthcare system was developed using the Django framework, integrating DS09 machine learning models. HTML, CSS, and JavaScript were employed to create the frontend interfaces, ensuring a user-friendly experience. AJAX was utilized to facilitate asynchronous data transfer between the frontend and backend, enhancing system responsiveness. MySQL served as the backend database, storing patient records and other relevant data securely.

To uphold security measures, patients are required to register and generate passwords for login access. Only authorized doctors, vetted by the backend team, are granted access to the system. This approach ensures adherence to best practices and safeguards patient confidentiality and data integrity.

### 3.16 Expected Outcomes

t appears you have a clear plan for developing a Django-based web application with two key components: Health Recommendation and Drug Prediction. Here's a structured approach to achieve your goals:

* Setting Up Django Project:
  + Create a new Django project to serve as the foundation of the web application.
  + Configure project settings, including database connection, static files, and security settings.
* Database Design:
  + Design a database structure to efficiently store user data and information required for each component.
  + Define database models for users, health recommendations, and drug predictions.
* User Authentication and Security:
  + Implement user registration and login functionalities to ensure user data security.
  + Utilize Django's built-in authentication system or third-party packages for enhanced security features.
* Component Development:
  + Health Recommendation: Develop a feature to provide tailored health advice based on user input. Utilize machine learning models for personalized recommendations if necessary.
  + Drug Prediction: Implement functionality to predict required medications based on user-supplied symptoms or conditions. Integrate machine learning models and algorithms for accurate predictions.
* Integration of Machine Learning Models:
  + Integrate machine learning models and algorithms into the Django application for accurate predictions.
  + Use frameworks like scikit-learn or TensorFlow to develop and deploy machine learning models within Django views.
* Testing and Quality Assurance:
  + Conduct rigorous testing of all components to ensure they function as intended.
  + Perform unit tests, integration tests, and end-to-end tests to validate the functionality and reliability of the application.
* Deployment and Security:
  + Deploy the web application to a web server following security best practices.
  + Implement HTTPS, secure authentication mechanisms, and other security measures to protect user data.
* Documentation:
  + Create comprehensive documentation for users, including user guides, API documentation (if applicable), and troubleshooting instructions.
  + Provide clear instructions on how to use each component of the application and how to troubleshoot common issues.

# Chapter-4 Health Care System

## 4.1 Introduction to implementation

### 4.1.1 Objective

The goal of our project is to create a Django-based web application for healthcare that uses machine learning to identify diseases and suggest medications. Physicians and patients will have different login and registration sites in the program, which will support user authentication. Patients will have a dashboard where they may enter their symptoms to get predictions about their disease and, if necessary, schedule visits. medicine predictions based on patient data will be possible with the doctor dashboard. If the medicine cannot be anticipated or if the doctor feels that a patient examination is required, appointments can be scheduled. The system will take care of appointment setting and management to guarantee smooth communication between patients and medical professionals, improving healthcare.

### 4.1.2 Scope

The implementation's scope includes creating a comprehensive online application for healthcare using Django that incorporates machine learning for medication and disease prediction. Developing strong user authentication systems with separate login and registration pages for physicians and patients is part of this. Patients will be able to schedule appointments and enter symptoms to receive disease forecasts on a dashboard. Doctors will have access to a dashboard where they may anticipate drugs based on patient data. If the medicine cannot be predicted or additional testing is needed, the dashboard will allow doctors to arrange visits. To provide effective and smooth interactions between patients and healthcare providers and to improve the overall delivery of healthcare services, the program will also contain tools for managing and scheduling appointments.

### 4.2 Data Processing and Analysis

In this section, we will focus on the preprocessing and analysis of the provided healthcare dataset. The preparation processes will involve cleaning the data by addressing missing values, removing duplicates, and resolving conflicts to ensure the reliability and quality of the information. We will also scale and normalize the data in order to standardize the features and get them ready for machine learning techniques..

### 4.2.1 Disease Prediction Data Set

This dataset contains various symptoms and the diseases that may be associated with those symptoms.Working on the data set is as follows-

### 4.2.1.1-Data Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Attribute | Meaning | Values |
| 1 | itching | Is patient having itching | {0, 1} |
| 2 | Skin rash | Is patient having skin rashes | {0, 1} |
| 3 | Nodal skin | Is patient having skin | {0, 1} |
| 4 | Sneezing | Does patient have continuous sneezing | {0, 1} |
| 5 | chills | Is patient facing chills | {0, 1}} |
| 6 | vomiting | Is patient facing vomiting | {0, 1} |
| 7 | fatigue | Is patient facing fatigue | {0, 1} |
| 8 | High fever | Was patient facing high fever | {0, 1} |
| 9 | dehydration | Was patient having dehydration | {0, 1} |
| 10 | Neck pain | Was patient facing neck pain | {0, 1} |
| 11 | excessive\_hunger | Was patient hungry | {0, 1} |
| 12 | Weakness of one body side | Was facing weakness on one side of body | {0, 1} |

Table – 2 Data Attributes

### 4.2.1.2-Data Head-

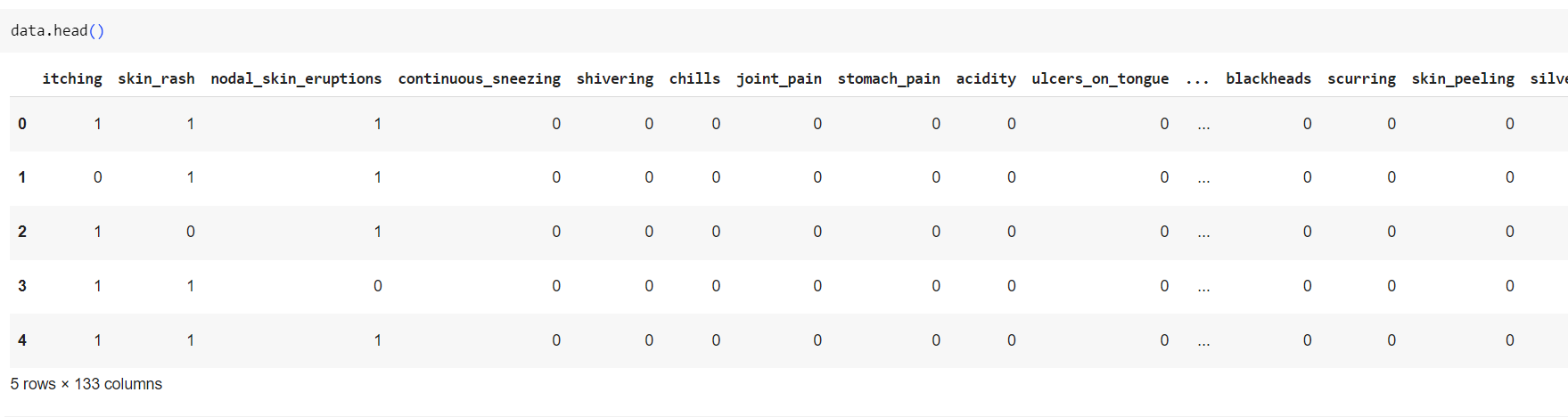


Figure – 2 Disease data head

### 4.2.1.3-Splitting the data into Symptoms and Disease

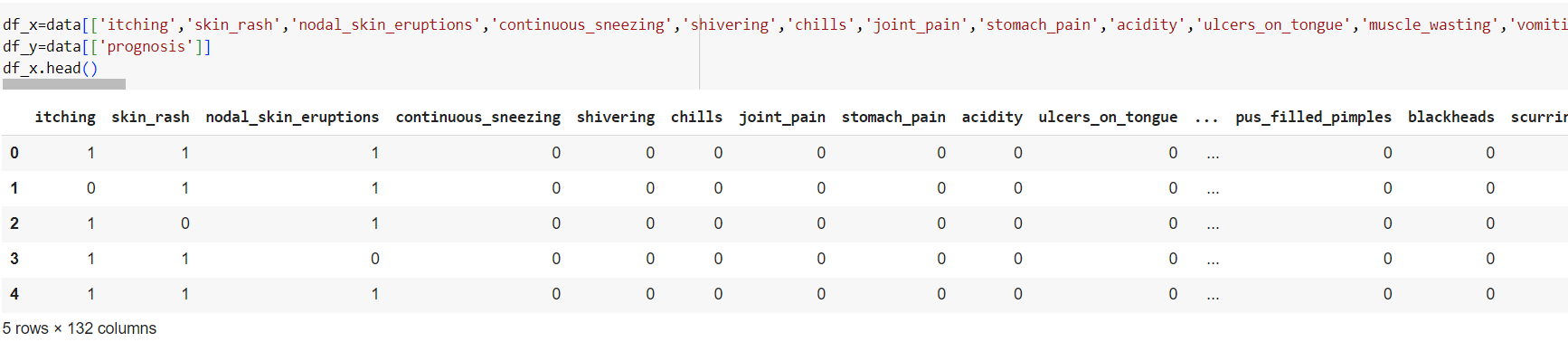


Figure – 3 df\_x data head

### 4.2.1.4-Heatmap of df\_x

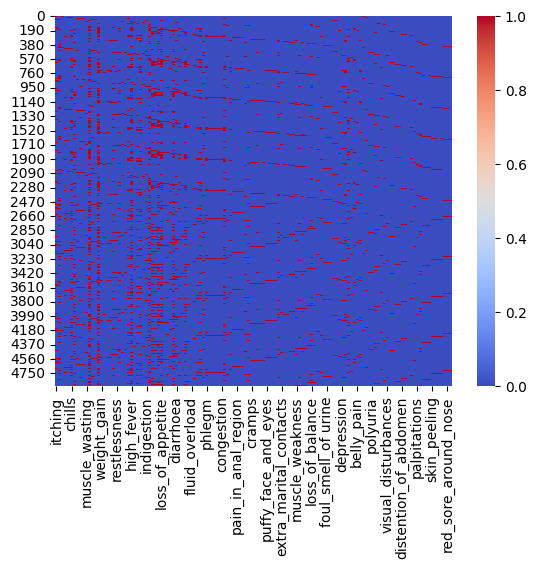


Figure – 4 Heat map df\_x

### 4.2.2 Drug Prediction Data Set

The drug prediction dataset includes attributes such as drug, disease, gender, and age. This dataset will be used to develop a drug prediction model.Working on the dataset is as follows-

### 4.2.2.1-Data head and Categorical distribution

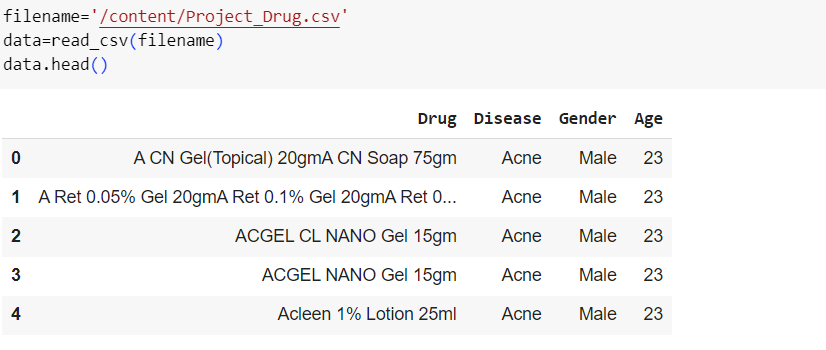


Figure – 5 Drug data head

We created a categorical distribution of the dataset using Matplotlib, which is as follows:

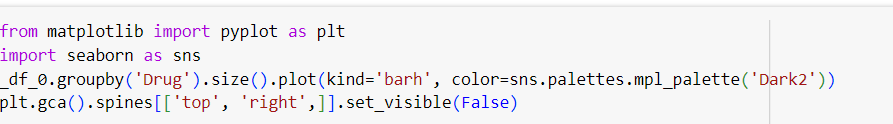


Figure – 6 Plotting data

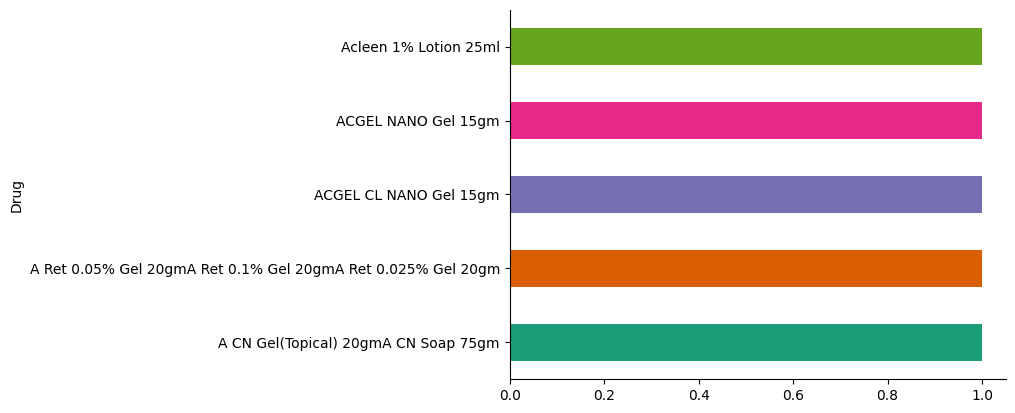


Figure – 7 Categorical distribution of drug dataset

### 4.2.2.2-Data Preprocssing

The .isnull().sum() function is used in data analysis to identify and count the number of missing values (NaNs) in a dataset.This we can see there are no missing values.

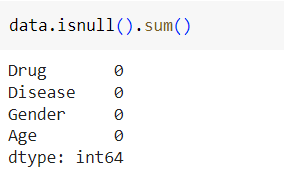


Figure – 8 Data preprocssing – 1

Here we can see that we have replaced Male with 1 and Female with 0 in the data set also we have replaced the disease with different categorical values.

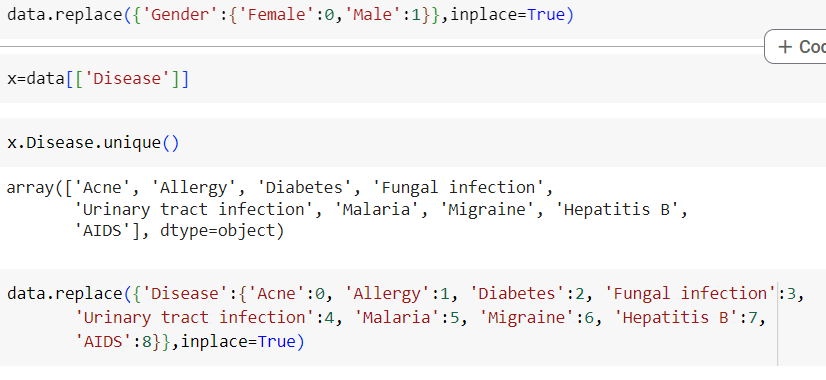


Figure – 9 Data processing – 2

Thus new data is -

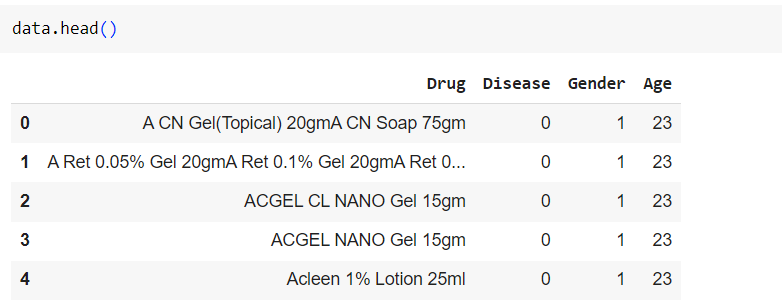


Figure – 10 Data head after preprocessing

### 4.2.2.3-Fitting the model



Figure – 11 Model fitting

Here we can see that the df\_x list contains Disease,Gender,Age and df\_y contains Drug

### 4.2.2.4-Disease vs Age plot



Figure – 12 Plotting disease vs age

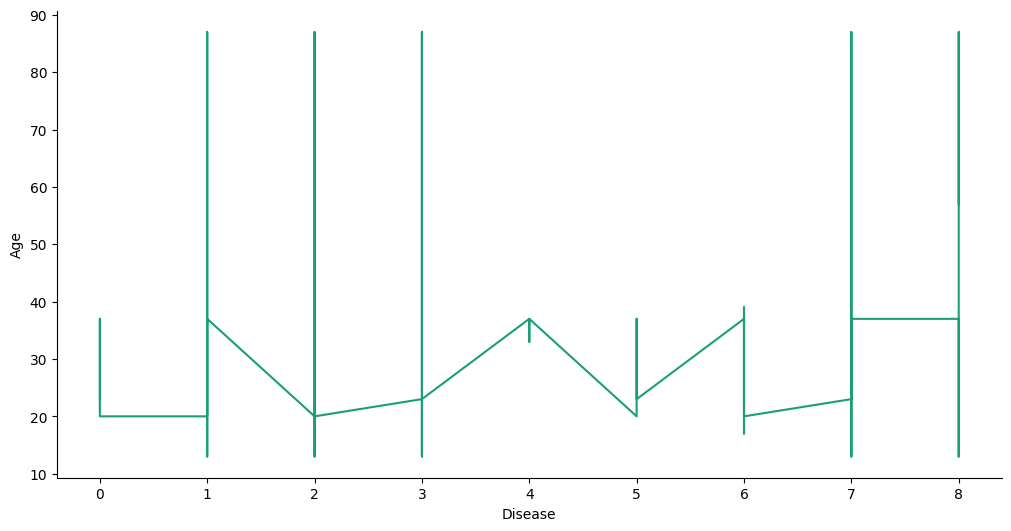


Figure – 13 Disease vs Age plot

### 4.2.2.5-Disease vs Gender plot



Figure – 14 Plotting disease vs gender

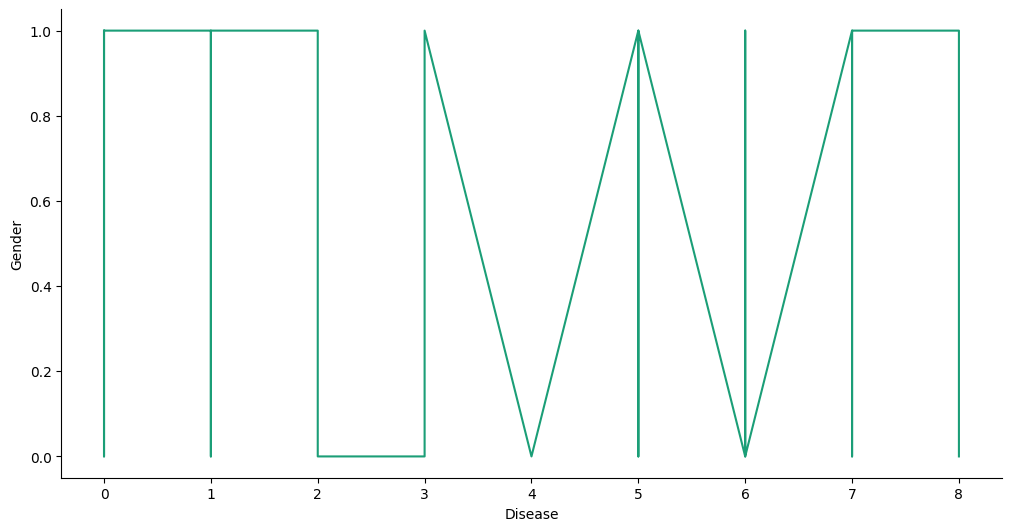


Figure – 15 Disease vs gender plot

### 4.2.2.6-Heatmap

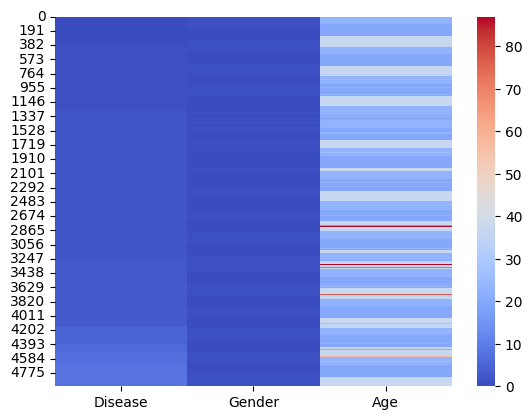


Figure – 16 Drug heatmap

## 4.3 Model Building

### 4.3.1 Disease Prediction Model

1-Splitting the data for testing and training

The dataset is split into 80% for training and 20% for testing

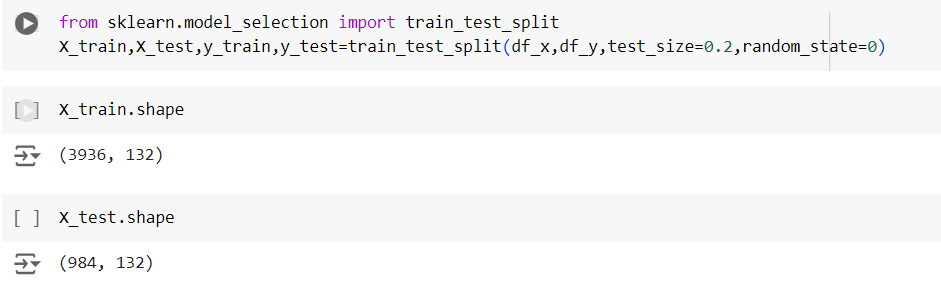


Figure – 17 Splitting of data for training and testing

2-Gussian Naive Bayes

Fitting the model

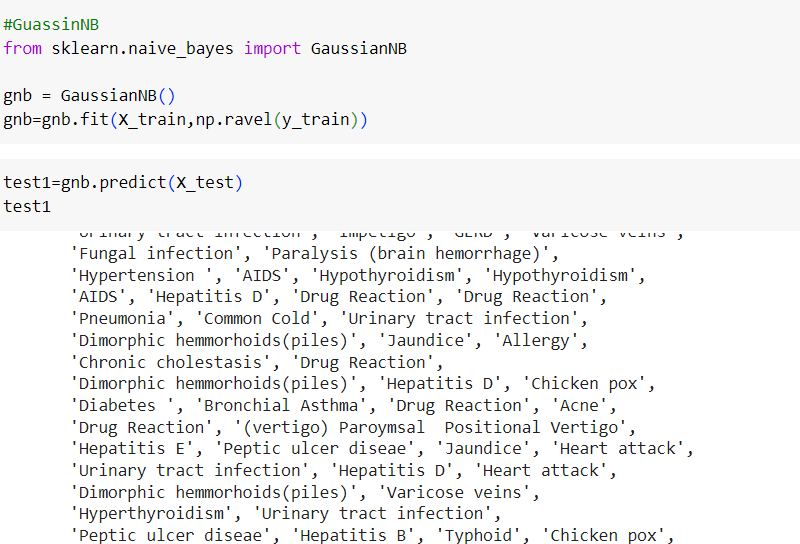


Figure – 18 Fitting gaussian naïve bayes

Confusion matrix and heatmap-

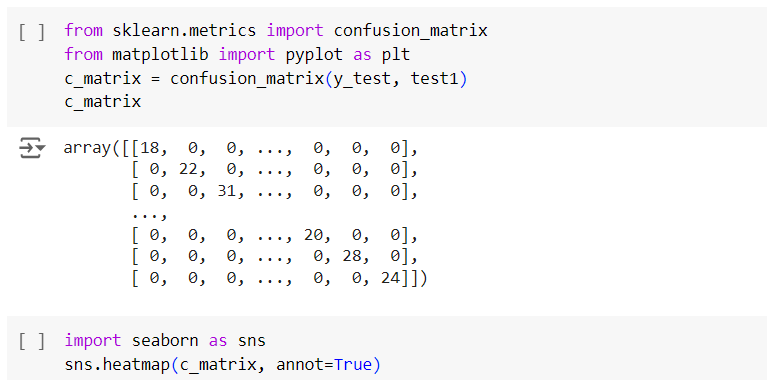


Figure – 19 Confusion matrix – 1



Figure – 20 Heatmap of confusion matrix – 1

Accuracy and Classification Report-

It tells us about the accuracy,f1 score of the model



Figure – 21 Testing accuracy

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | precision | recall | f1-score | recall |
| Drug Reaction | 1.00 | 1.00 | 1.00 | 27 |
| Allergy | 1.00 | 1.00 | 1.00 | 23 |
| Chicken pox | 1.00 | 1.00 | 1.00 | 21 |
| Common Cold | 1.00 | 1.00 | 1.00 | 34 |
| Dengue | 1.00 | 1.00 | 1.00 | 18 |
| Diabetes | 1.00 | 1.00 | 1.00 | 22 |
| Drug Reaction | 1.00 | 1.00 | 1.00 | 27 |
| Heart attack | 1.00 | 1.00 | 1.00 | 32 |
| Hepatitis B | 1.00 | 1.00 | 1.00 | 19 |
| Hepatitis C | 1.00 | 1.00 | 1.00 | 17 |
| Hepatitis D | 1.00 | 1.00 | 1.00 | 28 |
| Hepatitis E | 1.00 | 1.00 | 1.00 | 22 |
| Jaundice | 1.00 | 1.00 | 1.00 | 26 |
| Malaria | 1.00 | 1.00 | 1.00 | 25 |
| Migraine | 1.00 | 1.00 | 1.00 | 29 |
| Typhoid | 1.00 | 1.00 | 1.00 | 25 |
| hepatitis A | 1.00 | 1.00 | 1.00 | 24 |
| Pneumonia | 1.00 | 1.00 | 1.00 | 23 |
| accuracy |  |  | 1.00 | 984 |
| macro avg | 1.00 | 1.00 | 1.00 | 984 |
| weighted avg | 1.00 | 1.00 | 1.00 | 984 |

Table – 3 Classification report – 1

3-Decsion tree classifier

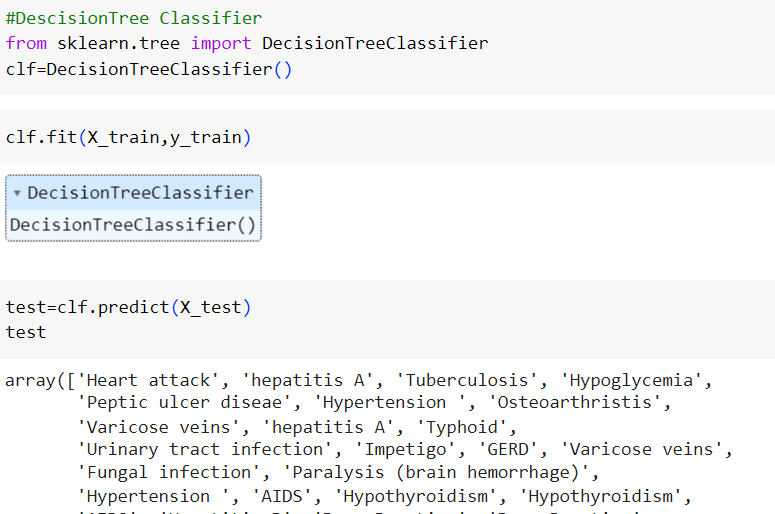


Figure – 22 Fitting decision tree classifier

Confusion matrix

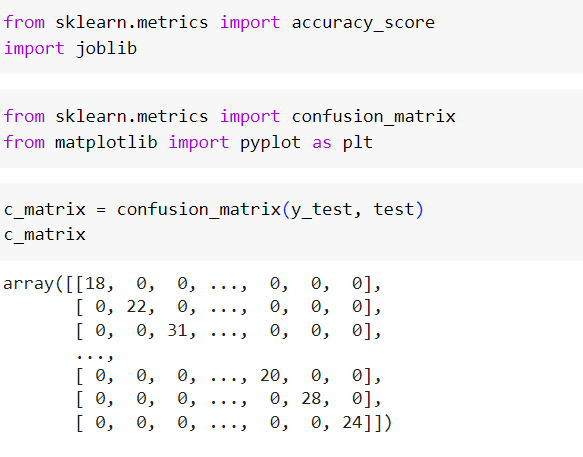


Figure – 23 Confusion matrix -2

Heat Map



Figure – 24 Heatmap confusion matrix – 2

Accuracy and Classfication report



Figure – 25 Calculating accuracy

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | precision | recall | f1-score | recall |
| Drug Reaction | 1.00 | 1.00 | 1.00 | 27 |
| Allergy | 1.00 | 1.00 | 1.00 | 23 |
| Chicken pox | 1.00 | 1.00 | 1.00 | 21 |
| Common Cold | 1.00 | 1.00 | 1.00 | 34 |
| Dengue | 1.00 | 1.00 | 1.00 | 18 |
| Diabetes | 1.00 | 1.00 | 1.00 | 22 |
| Drug Reaction | 1.00 | 1.00 | 1.00 | 27 |
| Heart attack | 1.00 | 1.00 | 1.00 | 32 |
| Hepatitis B | 1.00 | 1.00 | 1.00 | 19 |
| Hepatitis C | 1.00 | 1.00 | 1.00 | 17 |
| Hepatitis D | 1.00 | 1.00 | 1.00 | 28 |
| Hepatitis E | 1.00 | 1.00 | 1.00 | 22 |
| Jaundice | 1.00 | 1.00 | 1.00 | 26 |
| Malaria | 1.00 | 1.00 | 1.00 | 25 |
| Migraine | 1.00 | 1.00 | 1.00 | 29 |
| Typhoid | 1.00 | 1.00 | 1.00 | 25 |
| hepatitis A | 1.00 | 1.00 | 1.00 | 24 |
| Pneumonia | 1.00 | 1.00 | 1.00 | 23 |
| accuracy |  |  | 1.00 | 984 |
| macro avg | 1.00 | 1.00 | 1.00 | 984 |
| weighted avg | 1.00 | 1.00 | 1.00 | 984 |

Table – 4 Classification report – 2

4-Saving the model

A Python package called Joblib offers tools for effectively serializing and deserializing Python objects, especially ones with lots of data and computational models. It is frequently used to store and load models in machine learning workflows, which facilitates the sharing and persistence of trained models. Here we have used the Decision tree classifier model.

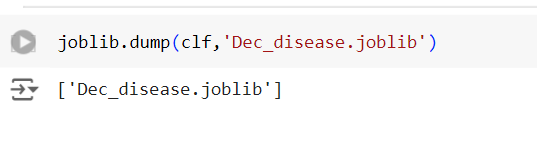


Figure-27 Dumping decision tree model

### 4.3.2 Drug Prediction Model

1-Splitting the data for Testing and Training

In this 30% data is used for training and 70% data is used for testing



Figure-28 Splitting of data for testing and training

2-Random Forest Model

Fitting the model and checking the Accuracy

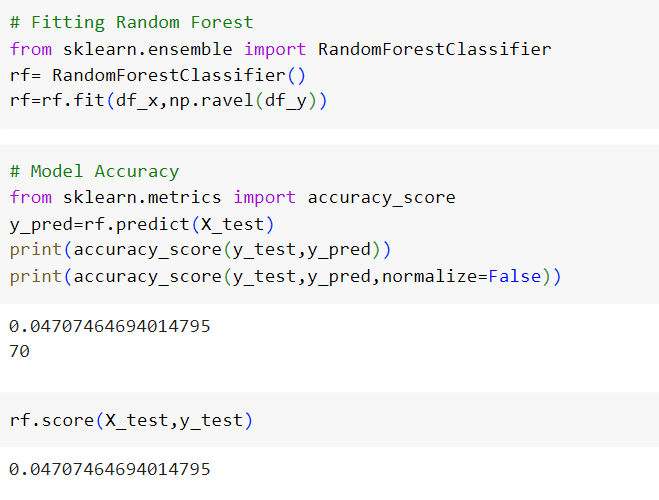


Figure-29 Fitting the model and calculating accuracy

Testing the model

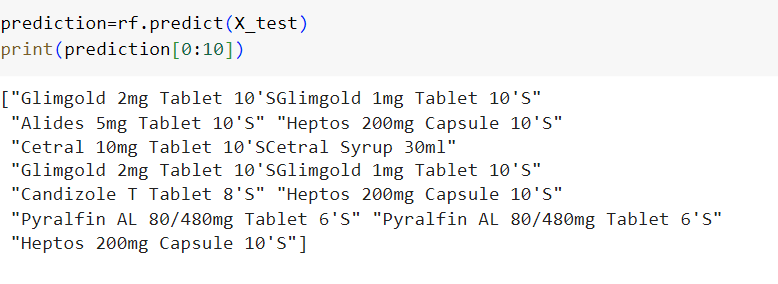


Figure-30 Model testing

Classification Report

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | precision | recall | recall | recall |
| accuracy |  |  | 0.05 | 1487 |
| Macro avg | 0.03 | 0.05 | 0.03 | 1487 |
| Weighted avg | 0.03 | 0.05 | 0.04 | 1487 |

Table – 5 Classification report – 3

3-Decision tree classifier

Fitting the data

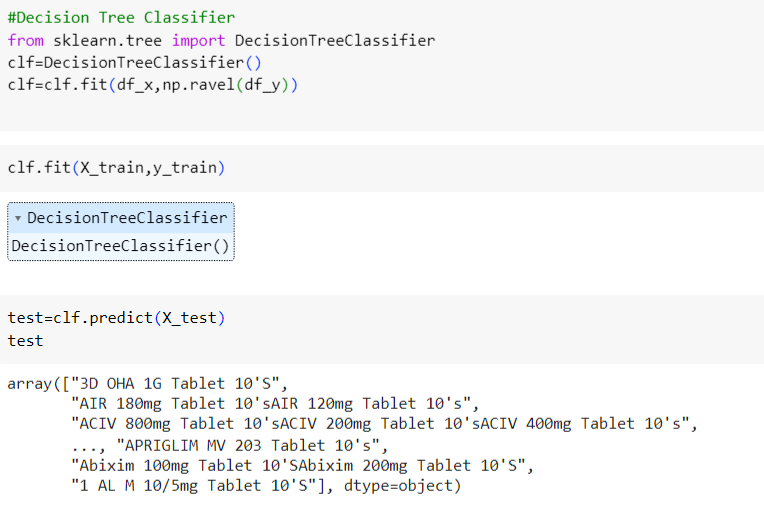


Figure-31 Fitting the data for decision tree

Model Accuracy and testing the model

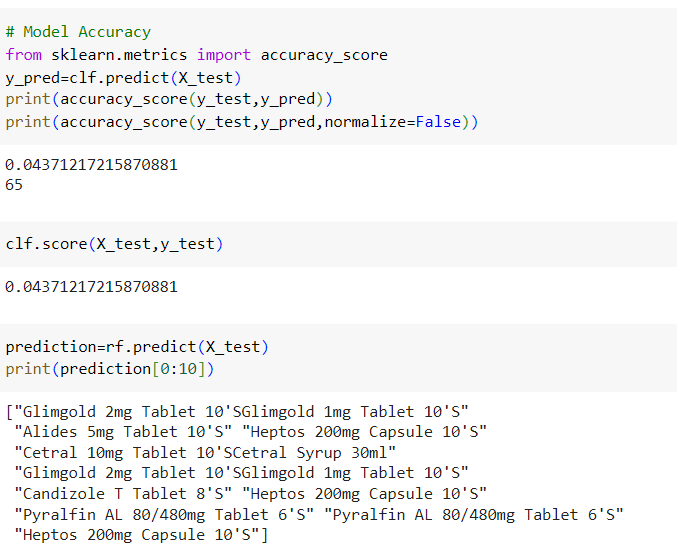


Figure-32 Model testing and calculating accuracy

Confusion Matrix and Classification Report

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | precision | recall | recall | recall |
| accuracy |  |  | 0.04 | 1487 |
| Macro avg | 0.03 | 0.04 | 0.03 | 1487 |
| Weighted avg | 0.03 | 0.04 | 0.03 | 1487 |

Table – 6 Classification report – 4

Saving the model

A Python package called Joblib offers tools for effectively serializing and deserializing Python objects, especially ones with lots of data and computational models. It is frequently used to store and load models in machine learning workflows, which facilitates the sharing and persistence of trained models.Here we have used the Random Forest Model as it has more accuracy.

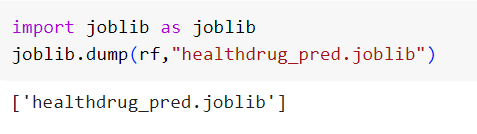


Figure-32 Saving the model

## 4.4 Setting Up the Django Framework for Healthcare project

### 4.4.1 Installing Python

Go to the downloads area of the official Python website in order to begin the installation process. Based on your operating system (Windows, macOS, or Linux) and architecture (32-bit or 64-bit), select the relevant installation. Launch the installer after downloading it. In order to facilitate command-line access to Python, be sure to tick the "Add Python to PATH" box during installation. To finish the installation procedure, adhere to the installer's instructions. After installation, you can confirm that Python has been installed by typing "python --version" into a terminal or command prompt to see the version of Python that has been installed. Moreover, you can use the pip package manager to install Python packages by entering "pip install " in the terminal or command line.

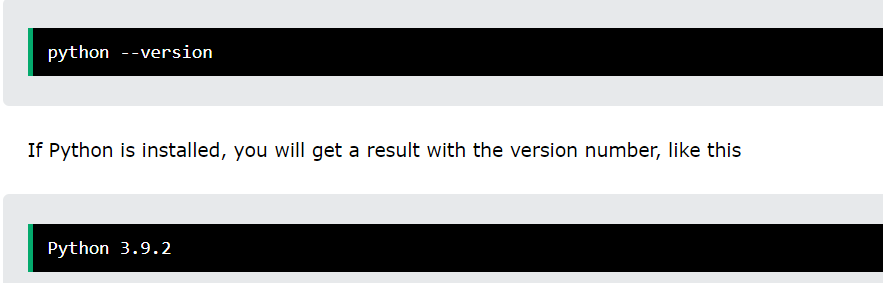


Figure-34 Checking python version

### 4.4.2 Installing Django

Prior to installing Django, make sure Python is set up on your computer. Next, use Python's package manager pip to install Django by opening a command prompt or terminal and typing "pip install django". The most recent version of Django and its dependencies will be immediately downloaded and installed by using this command. When the installation is finished, you may check it by typing "django-admin --version" into the terminal or command line, which should show the Django version that is installed. You may now use Django to create web applications by following its official tutorials and documentation.

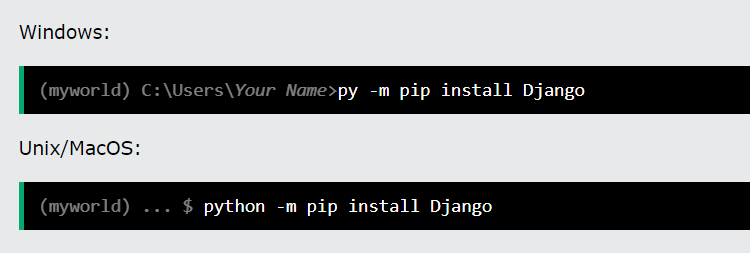


Figure-35 Installing Django – 1



Figure-36 Installing Django – 2

### 4.4.3 Setting Up the project

To set up the project type the command



Figure – 37 Project setup

To initiate a new Django project with the name "healthcare" and the necessary file structure, open a command prompt or terminal and navigate to the desired directory. Next, create the Django project "healthcare" by using the command "django-admin startproject healthcare". This will produce the "healthcare" directory, which will contain "manage.py" and all of the project's primary files. You will discover a app inside the "healthcare" directory that shares the same name as the project ("healthcare") and holds the wsgi files, urls, and settings ("settings.py", "urls.py", "wsgi.py"). Furthermore, when you build new applications using the "django-admin startapp " command, Django will automatically produce a "models.py" file inside the app directory.

### 4.4.4 Setting Up the project app

Open the command prompt or terminal and navigate to the root directory of your Django project to construct the Django app named "core" with the desired file structure. The "core" Django app will then be created by running the command "python manage.py startapp core." After the app is formed, go to the "core" directory and make a "templates" folder, which is where you will store the HTML and AJAX files for the healthcare web application's homepage, patient and doctor dashboards, and other files. This modular design makes it easier to manage and maintain the project by breaking it up into discrete components.

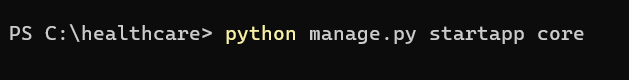


Figure – 38 Setting up core app

Then we need to connect the core app to the healthcare app this can be done by adding core in INSTALLED APP section in Settings.py and set the path in urls.py

Settings.py -

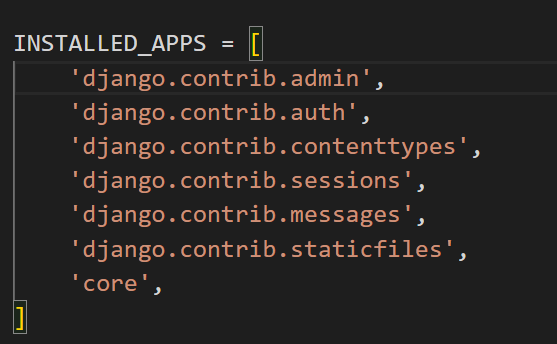


Figure – 39 Connecting the core app

Urls.py



Figure – 40 Including the path

To start the server write the command-

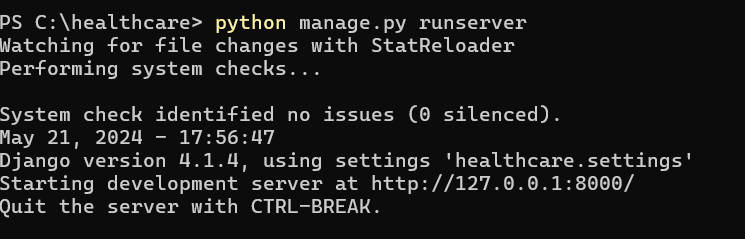


Figure – 41 Starting the server

## 4.5 Django Admin

A web-based platform for managing and interacting with the data models of a Django application is provided by an integrated administration interface known as Django Admin. It makes it easy to create, read, update, and delete records (CRUD) because it doesn't require any particular forms or views. Due to the high degree of configurable admin interface, developers can register models for management, change their appearance, and assign different levels of access to users using the permissions system. Using Django Admin, developers may quickly create a complex management interface for the data in their application.

A superuser is a special user account that has full access to the Django admin panel. Any object on the admin site can be added, edited, or removed by superusers. They have all permissions enabled, thus they can also oversee other users and perform any administrative tasks. Typically, to set up a Django project, you run "python manage.py createsuperuser," which prompts you for the password, email address, and username of the account. The initial setup and ongoing management of the Django application require this superuser account.

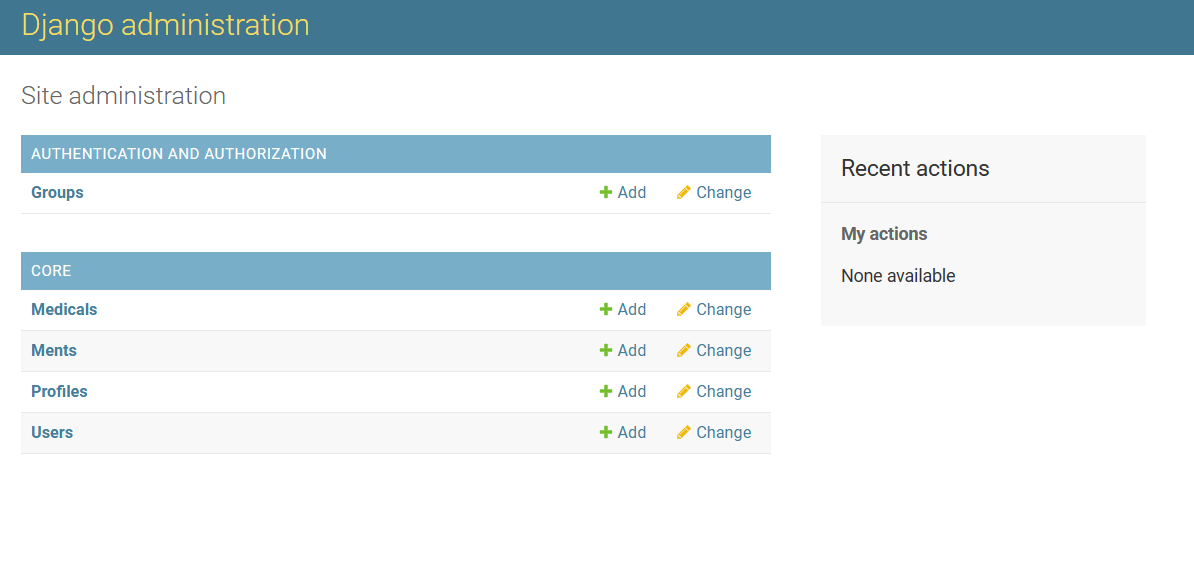


Figure – 42 Admin pannel

## 4.6 Creating MySQL database

The tables in the models.py portion of the main application must be created before we can construct the MySQL database in Django.Since classes are used to generate tables, creating a class user will result in a user table.We must first connect our Django application to the MySQL database, which is done inside Settings.py under the name healthcare\_db, before we can create the tables.

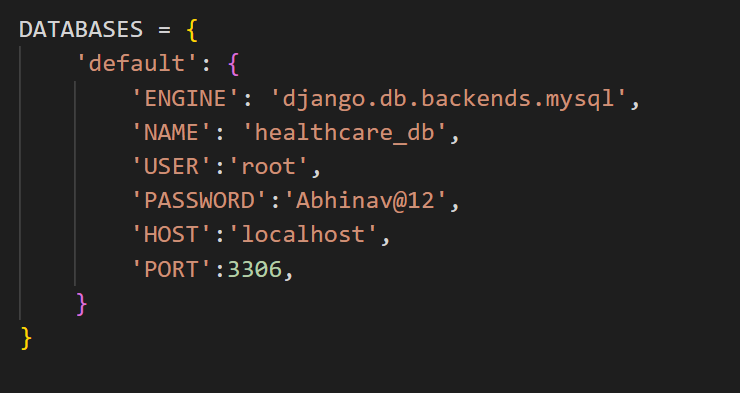


Figure – 43 Connection MySql workbench

Once the database is connected, we must create tables, which is done in the core application's models.py file.Like we can see we have different tables like User,Medical etc and each of the tables are containing differnt attributes with their datatypes like is\_patient,is\_doctor,disease,medicine etc.

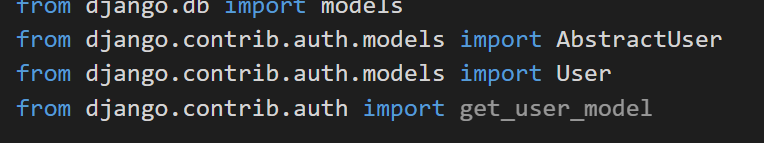


Figure – 44 Importing files

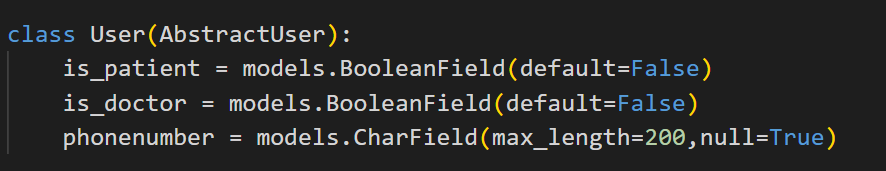


Figure – 45 Creating table – 1

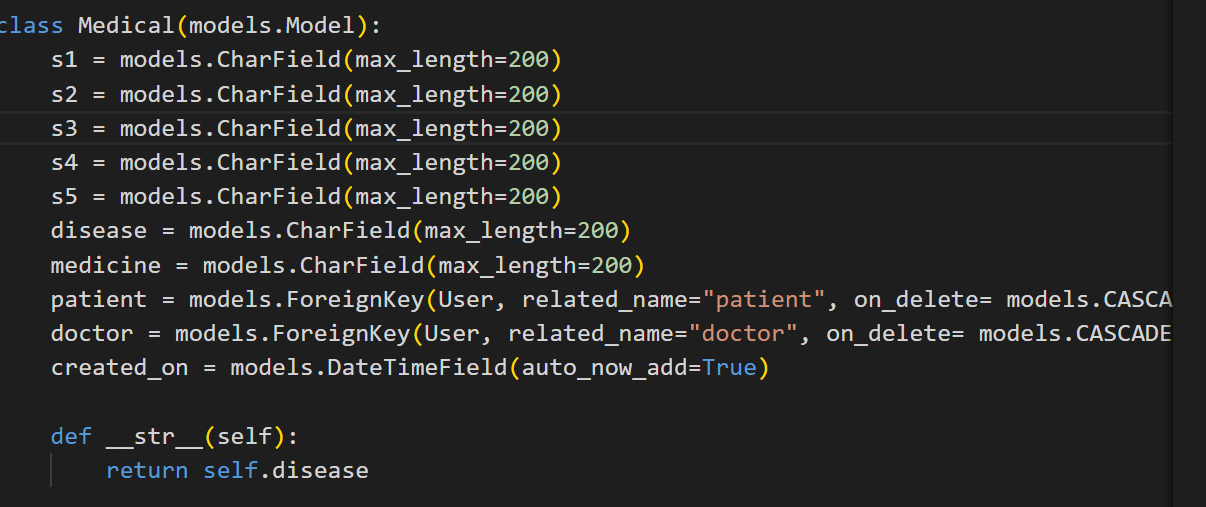


Figure – 46 Creating table – 2

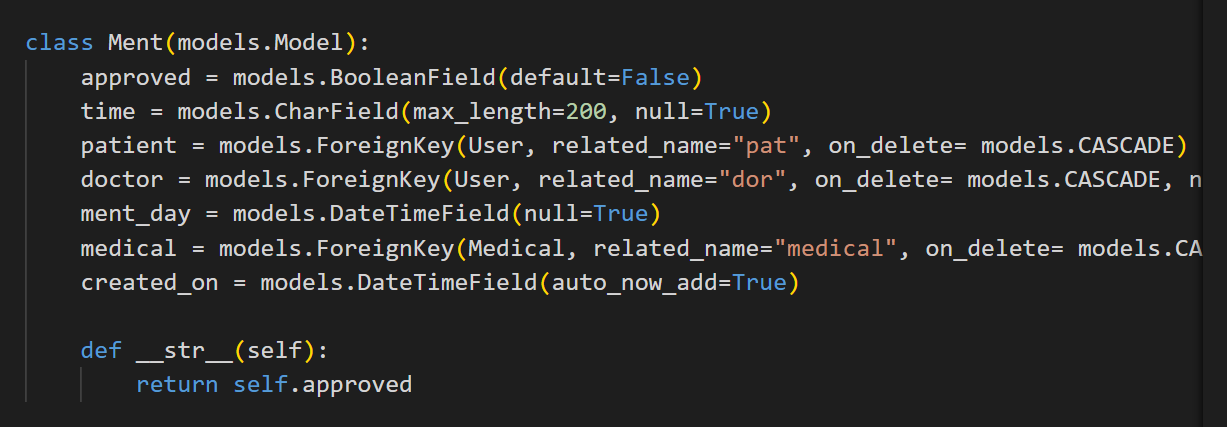


Figure – 47 Creating table – 3

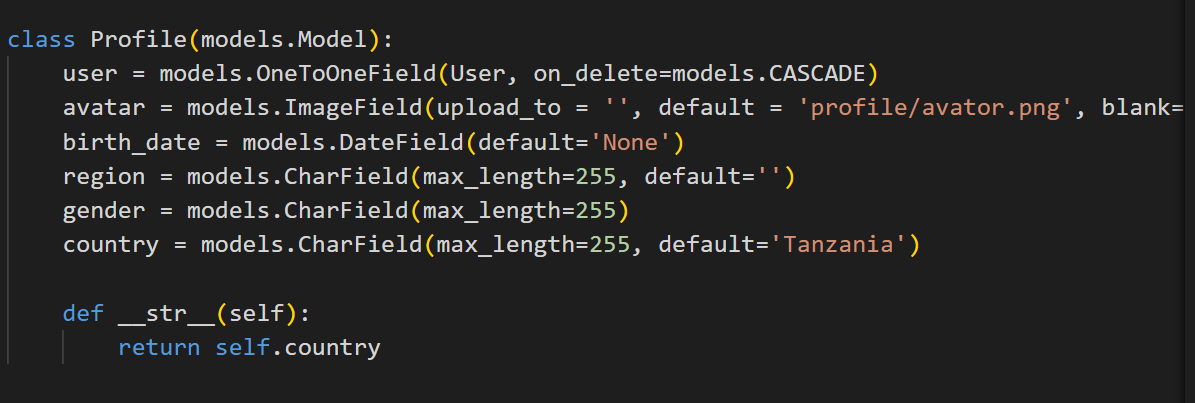


Figure – 48 Creating table – 4

We must execute the command py manage.py makemigrations table\_name after creating each table. Should a table member be terminated, the command will be executed.



Figure – 49 Make migration

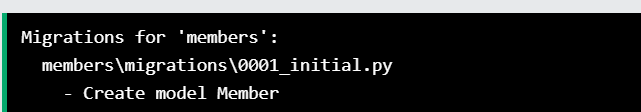


Figure – 50 Migration for table members

After completing all makemigrations we need to run the following command to create the table.

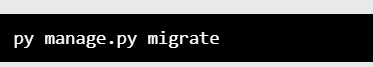


Figure – 51 Final migration

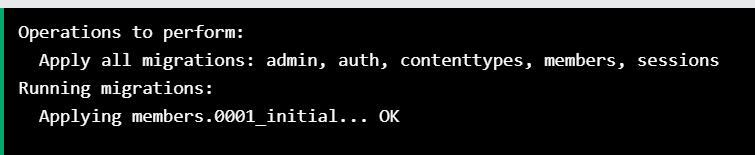


Figure – 52 Applying all migrations

The database will look like this-

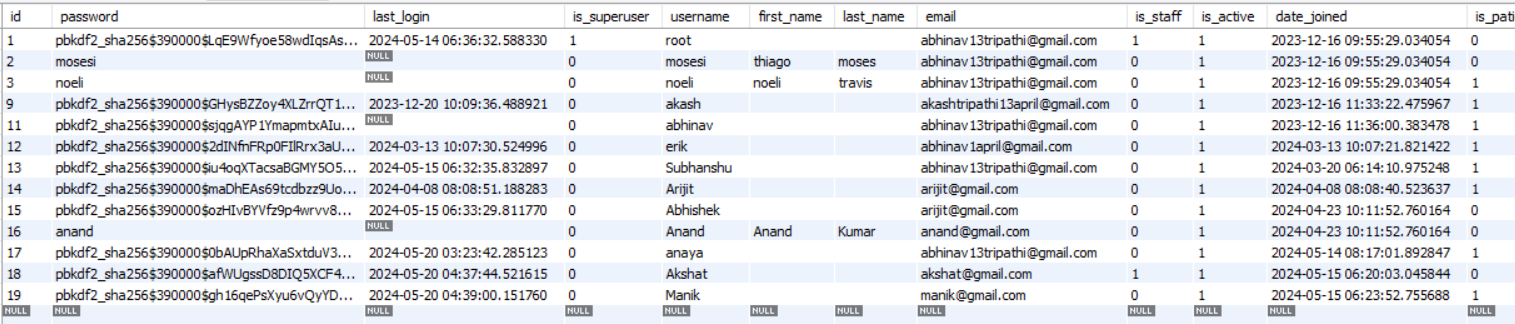


Figure – 53 Database

## 4.7 Setting Up the Static file

In Django, static files pertain to CSS, JavaScript, images, and other content that is given directly to the client rather than being created dynamically by the server. The front end of the web application requires these files. Django can handle these files by using the `STATICFILES\_DIRS` and `STATIC\_URL` options, which specify the URL path to access the static files and the directories Django searches for, respectively. While Django serves static files automatically during development, a web server like Nginx or Apache is typically needed for production to achieve best performance. For production purposes, compile all static files into a single directory using the `python manage.py collectstatic} command.

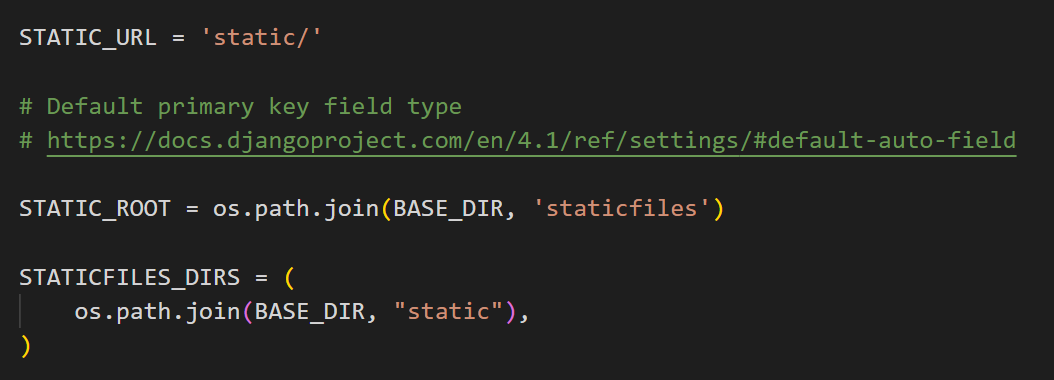


Figure – 54 Static file

## 4.8 Setting Up the template folder inside the Core app

The template folder contains all of the HTML files that make up a Django project's layout and structure. This category typically contains necessary templates such as home.html, base.html, login.html, and register.html. These templates are utilized for the user login, registration, base layout, and homepage, in that order. Additionally, it includes personalized templates for doctor-patient interfaces, such as diagnose.html and result.html. One important file in this folder called Ajax.html enables AJAX data transfers between the frontend and backend. For instance, the backend illness prediction algorithm receives the symptoms entered by the user through ajax.html, processes them, and delivers the expected disease. Similarly, doctors use AJAX to send disease data to the backend, and the drug prediction algorithm determines which prescription is best. The smooth and dynamic interactions between the backend processes and the user interface are guaranteed by this configuration.

### 4.8.1 Patient ajax.html

The patient AJAX system takes symptoms provided by the patient and sends them to the backend for the disease prediction model to predict the disease. Additionally, it handles appointment requests from the user and forwards them to the backend.

### 4.8.2 Doctor ajax.html

The doctor AJAX system receives the disease information from the backend and uses the drug prediction model to predict the appropriate drug. It also sends data about scheduled appointments to the backend.

### 4.9 Setting Up the urls.py

In a Django project, the urls.py file is necessary to pass URLs to the relevant views. It works as a mapping table by assigning URL patterns to the relevant view functions or classes that handle the requests. Usually located in the project's root directory adjacent to settings.py, the main urls.py contains the admin route and URLs for additional apps. Developers can define these mappings more efficiently and logically by using tools like path() and re\_path(). Additionally, by accessing URL configurations from other applications, include() promotes modular and maintainable code architectures.

### 4.9.1 Importing files

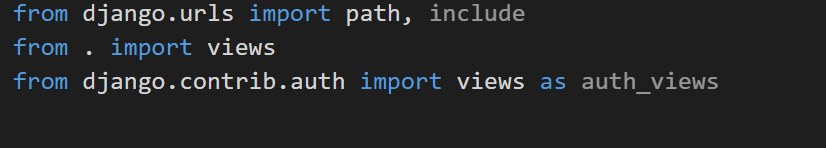


Figure – 55 Importing files

### 4.9.2 Setting the path

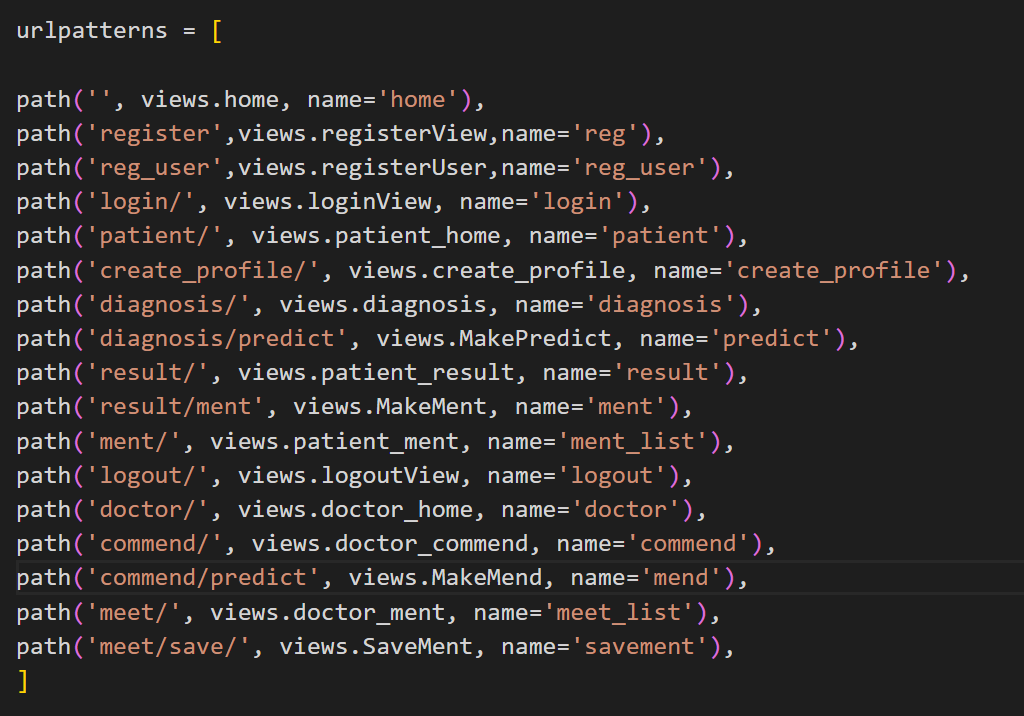


Figure – 56 Setting up the paths

## 4.10 Setting Up the views.py

A crucial component of any Django application is views.py. It contains the view functions or classes that handle requests and respond to them. Every view function receives an HTTP request, which it processes to produce an HTTP response. This response can be an HTML page, a redirect, a 404 error, or, in the case of an API, a JSON object. If you logically organize views within an application's views.py file, you can simply manage the logic that controls what data is delivered to the templates and how user interactions are handled.

### 4.10.1 Importing files

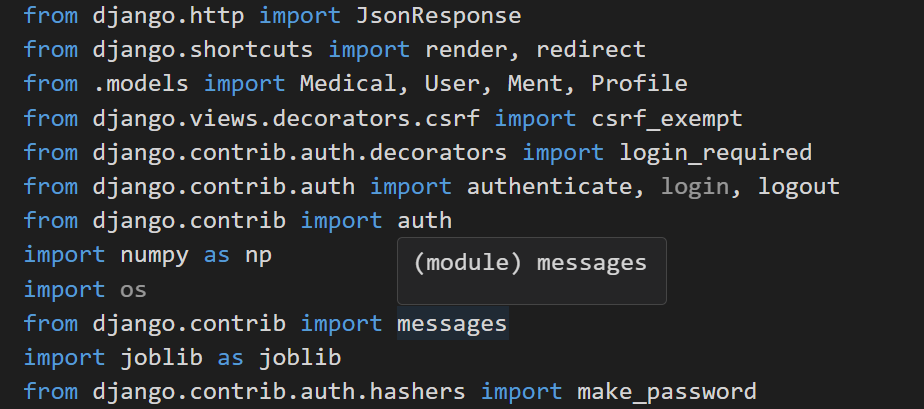


Figure – 57 Importing files

### 4.10.2 Home page

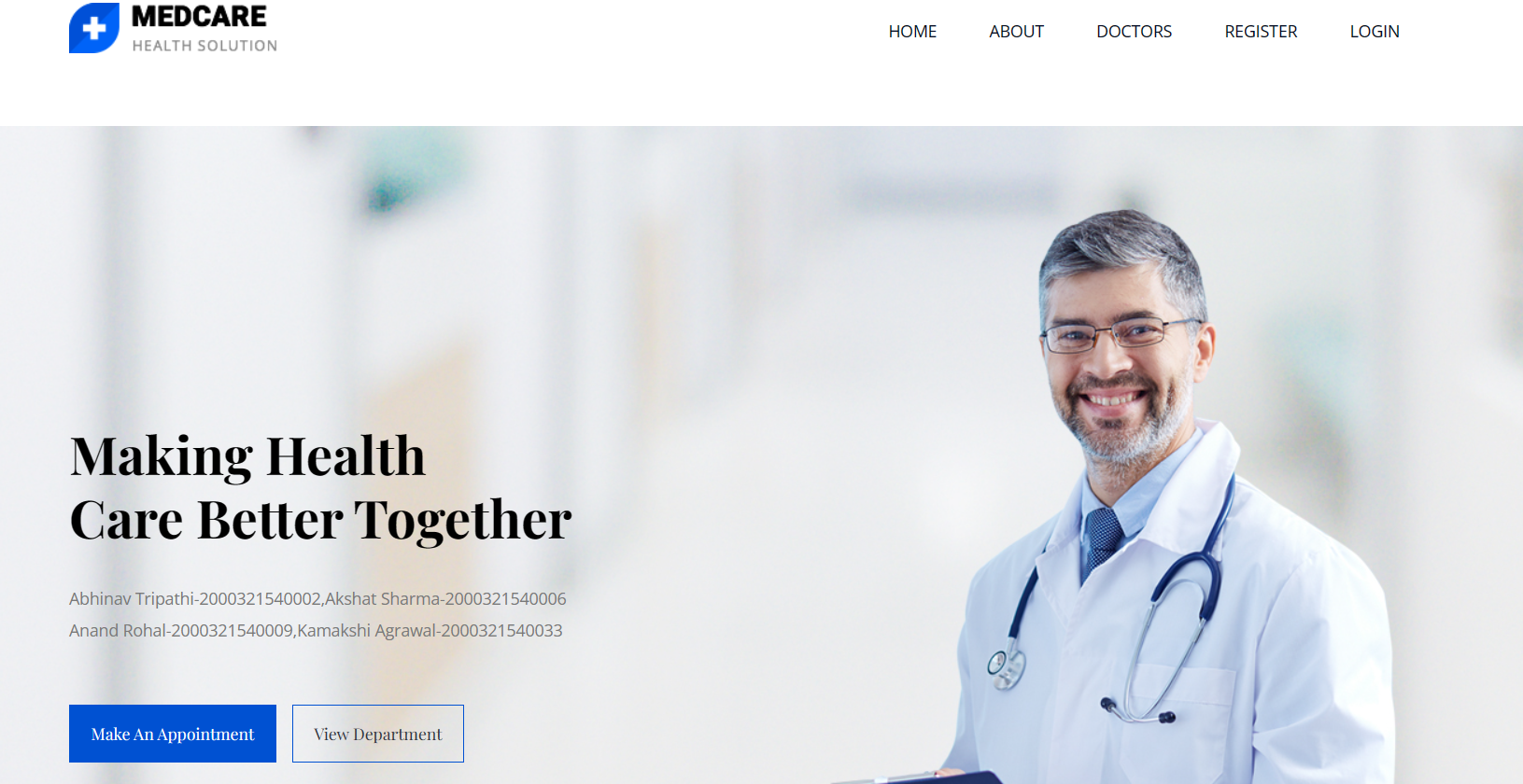


Figure – 58 Home page

To redirect to the home page or register page we use the below shown code

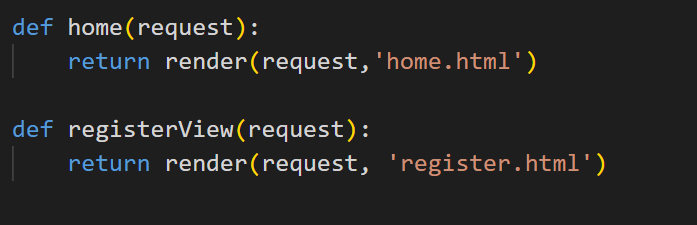


Figure – 59 Redirecting to home and register page

### 4.10.3 User Registration

This function shows a message indicating whether or not the account was successfully created after receiving the username, email address, and password from the front end. Following a successful registration, a "Account was created successfully" message is provided and the registration details are recorded in the backend. If not, the user will be taken to the registration page and see the message "Failed to register, try again later."

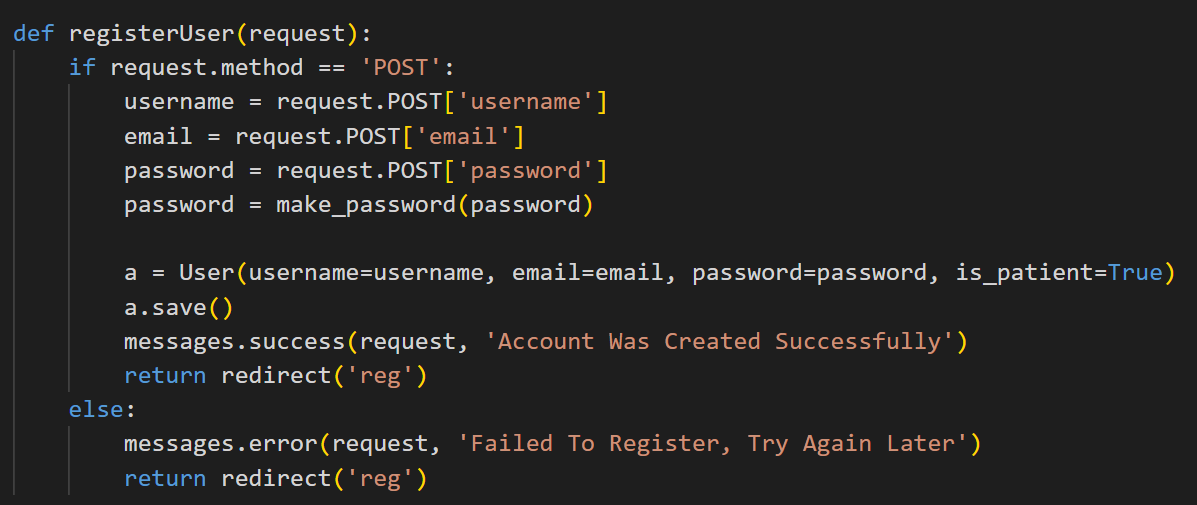


Figure – 60 User registration function

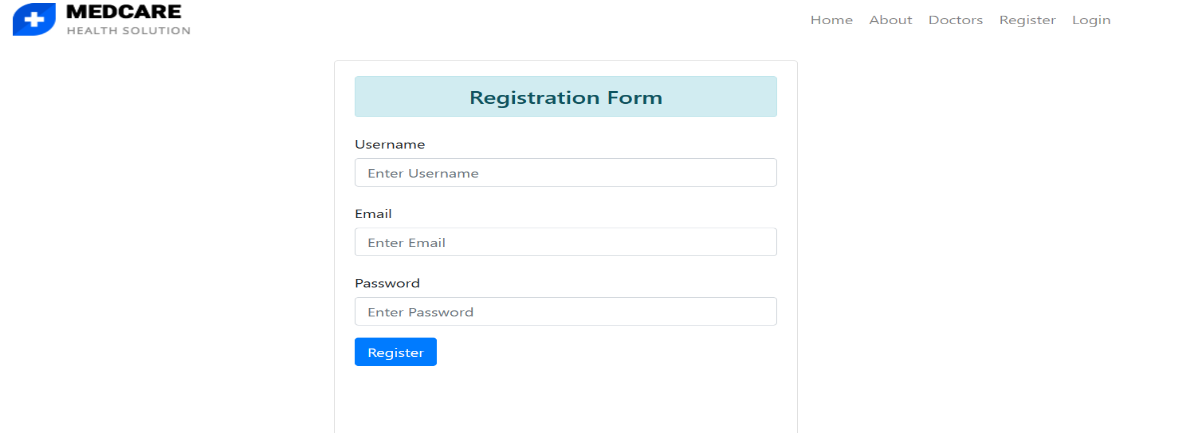


Figure – 61 Registration page

### 4.10.4 User Login

After receiving the username and password, the `loginView` method enables the user to access the dashboard. The user is forwarded to the Patient dashboard if he is a patient, and to the Doctor dashboard if he is a doctor. It will display the warning "Invalid Username or Password" if the login attempt fails.

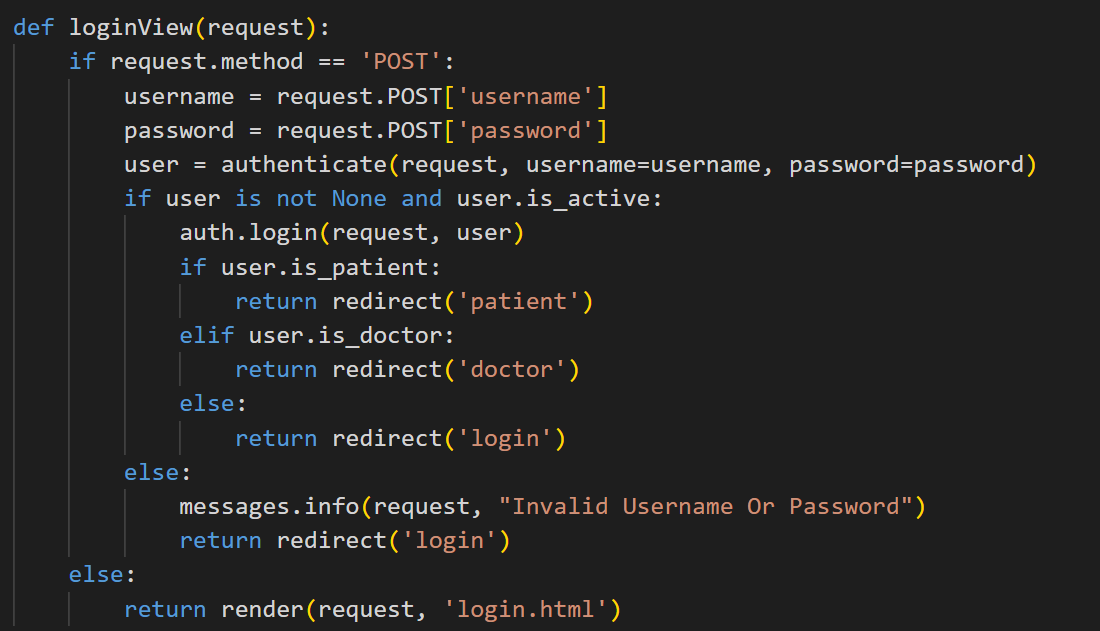


Figure – 62 User login

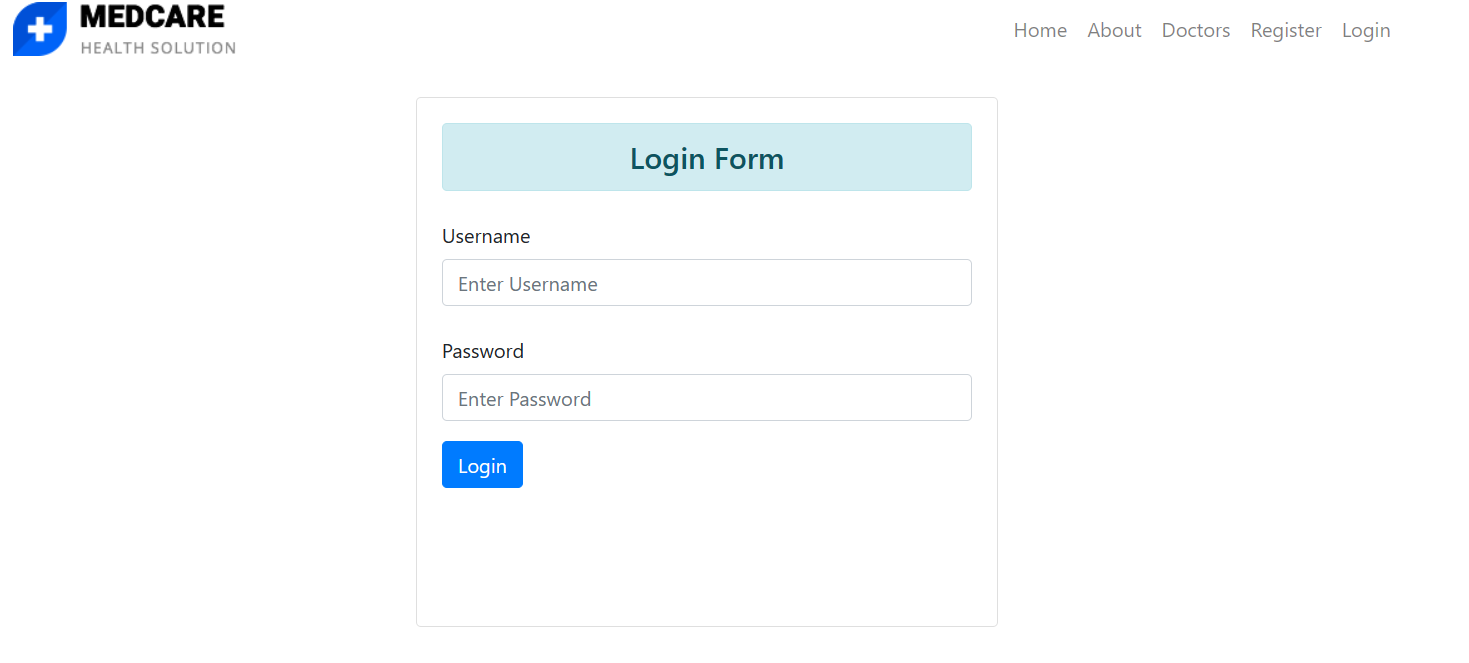


Figure – 63 Login page

### 4.10.5 Doctor Authorization

The backend staff will update the doctor's status in the database from patient to doctor after his background has been checked, and he must register in the same manner as other users.

### 4.10.6 Home doctor dashboard

The doctor\_home function aids in showing the quantity of physicians, quantity of patients, quantity of appointments, and quantity of drugs prescribed.

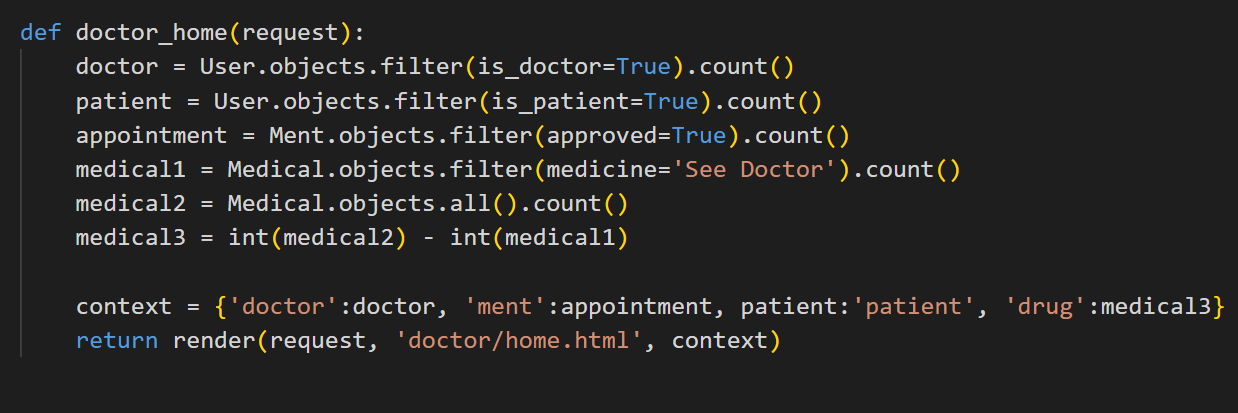


Figure – 65 Doctor home

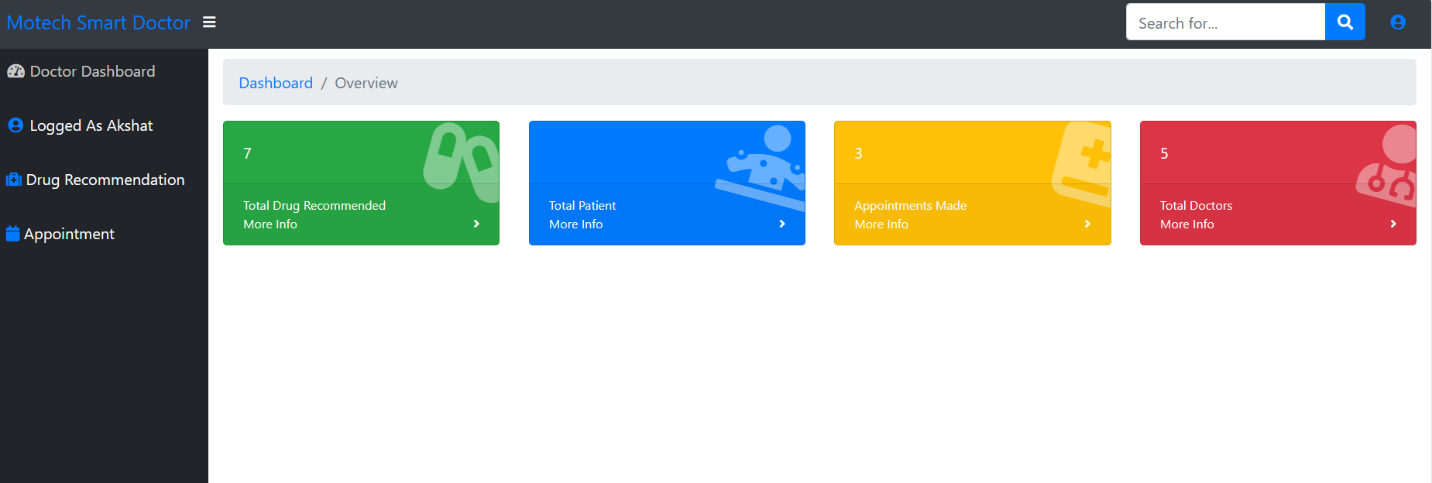


Figure – 66 Doctor dashboard

### 4.10.7 Home patient dashboard

It is possible to view the entire number of doctors, patients, appointments, and suggested drugs with the use of the `patient\_home` function. Additionally, it verifies that the patient has made his profile. If not, it displays the message "Please Create Profile To Continue." The patient cannot diagnose themselves if the user profile is not created.

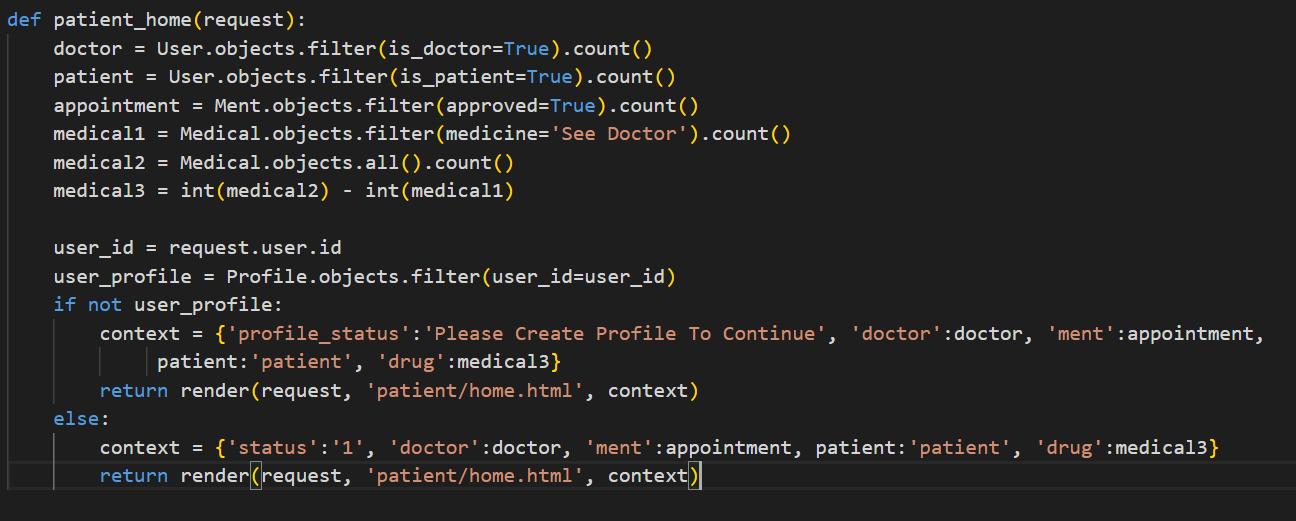


Figure – 68 Patient home

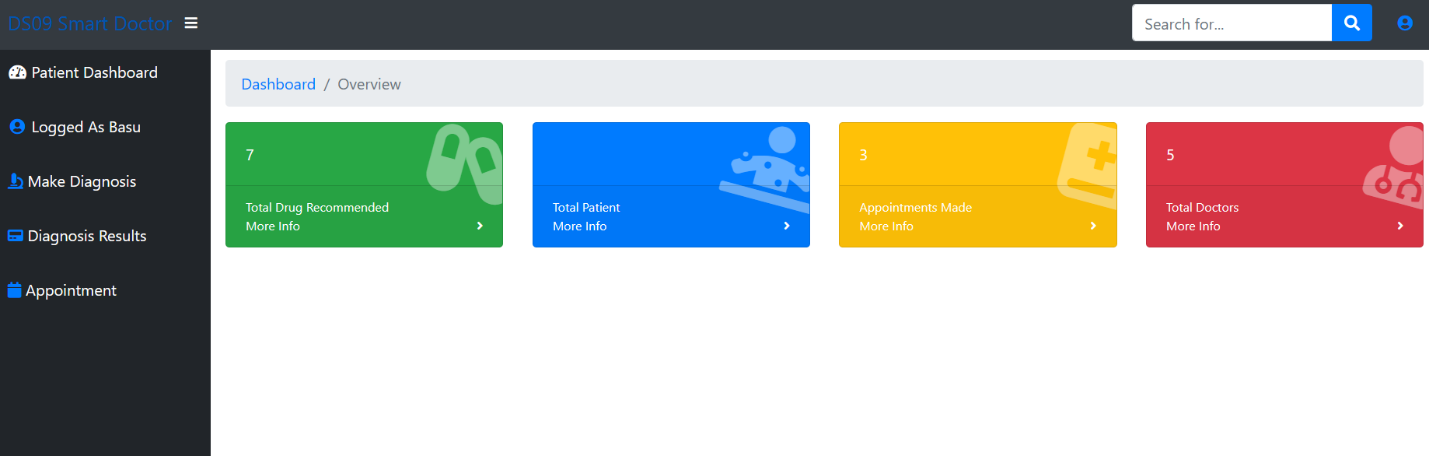


Figure – 69 Patient dashboard

### 4.10.8 User Profile

After obtaining the user's birthdate, gender, nationality, and area, the `create\_profile` function assists in creating their profile and displays the message "Your Profile was Created Successfully."

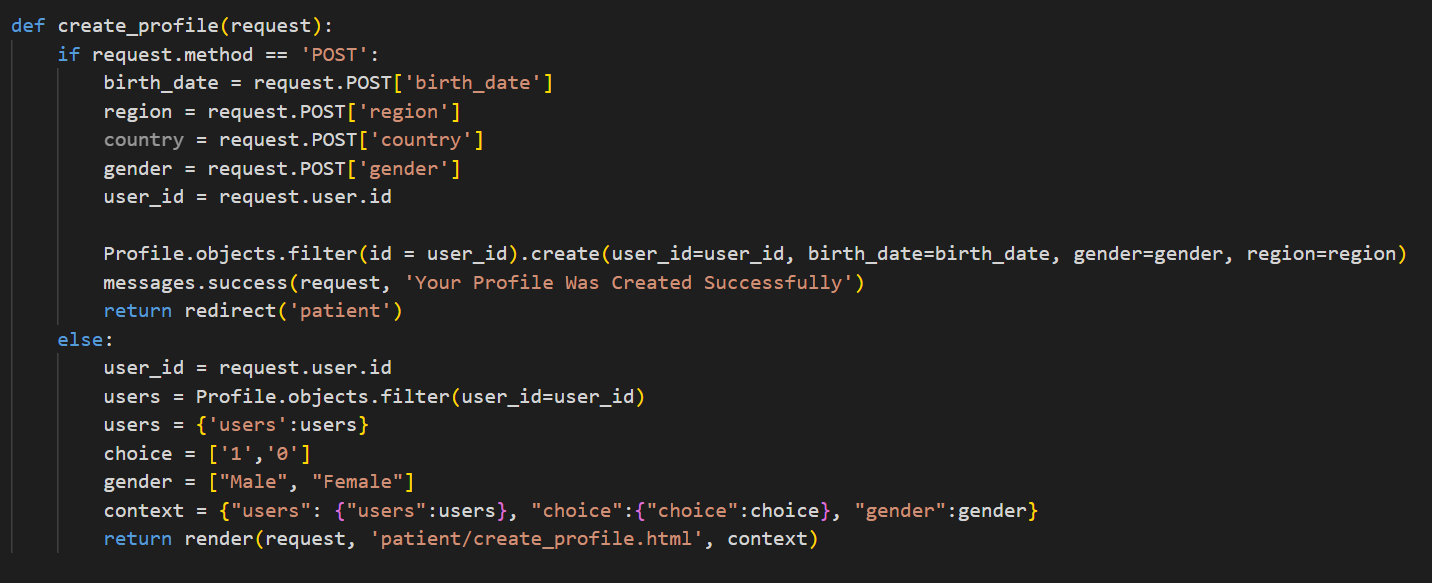


Figure – 70 User profile



Figure – 71 User profile page

### 4.10.9 Model Integration and Prediction

The list of symptoms in the `diagnosis` function is composed of several symptoms. After sorting, the list is fed into the `diagnosis.html` template located in the {patient} folder.

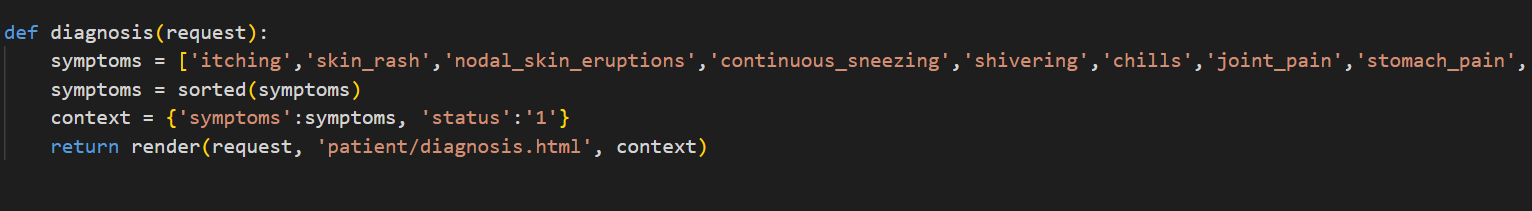


Figure – 72 List of desies

The user can choose five symptoms (s1, s2, s3, s4, s5), which are then saved in list\_b, with the aid of the list symptoms given from the diagnosis function. A for loop is used to append 0 from 0 to the length of list\_a after a list\_a is completed with all of the symptoms. This creates a new list\_c in which all of the symptoms are initially zero.After that, we check which entries from list\_b are in list\_a using a nested for loop, and we assign a 1 to the index in list\_c.Following this list\_c's conversion to an array using numpy, the model is used to forecast the outcome and the diease.

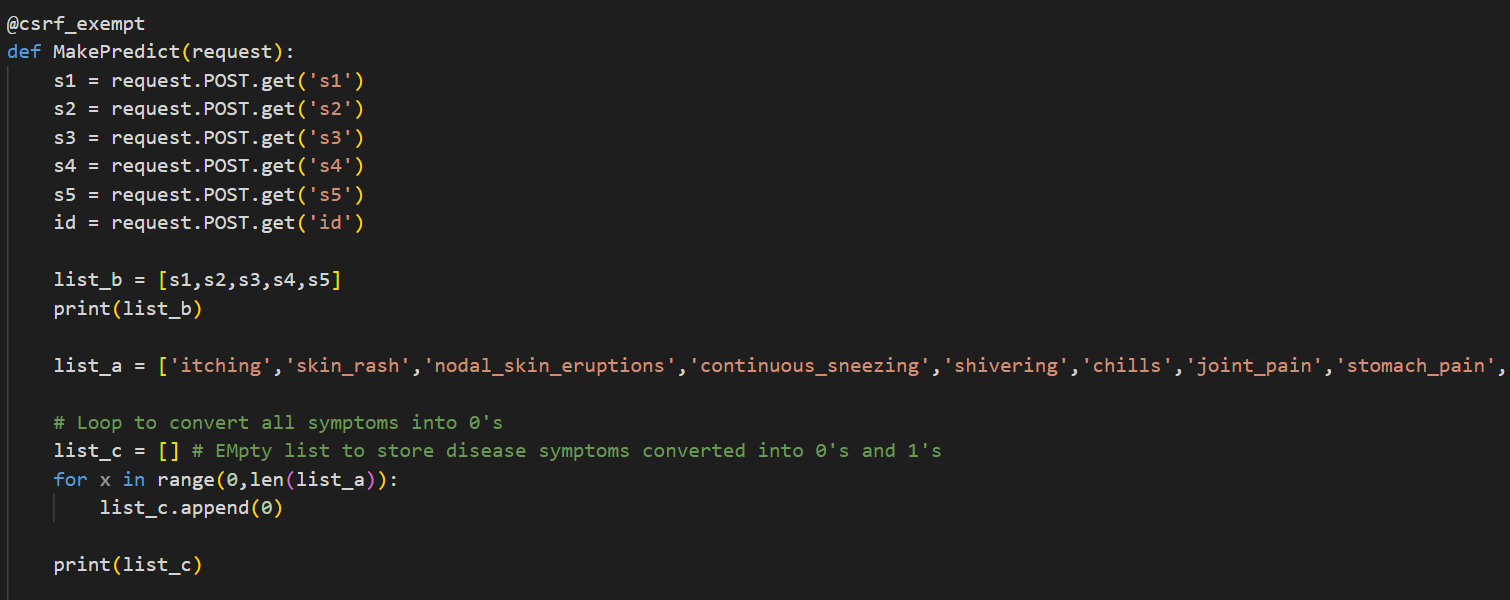


Figure – 73 Taking symptoms from backend

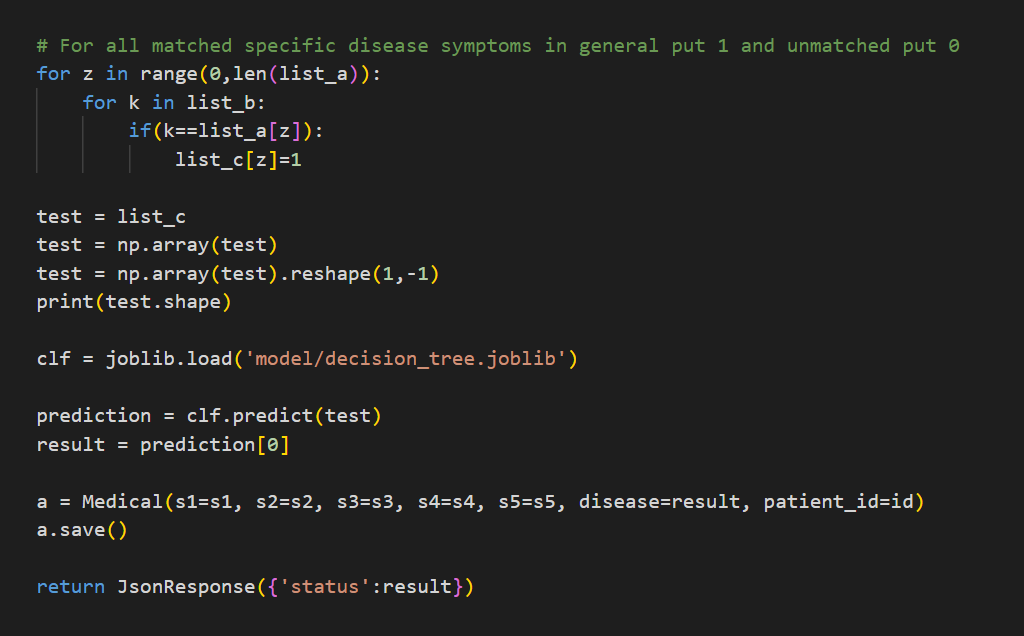


Figure – 74 Predicting disease

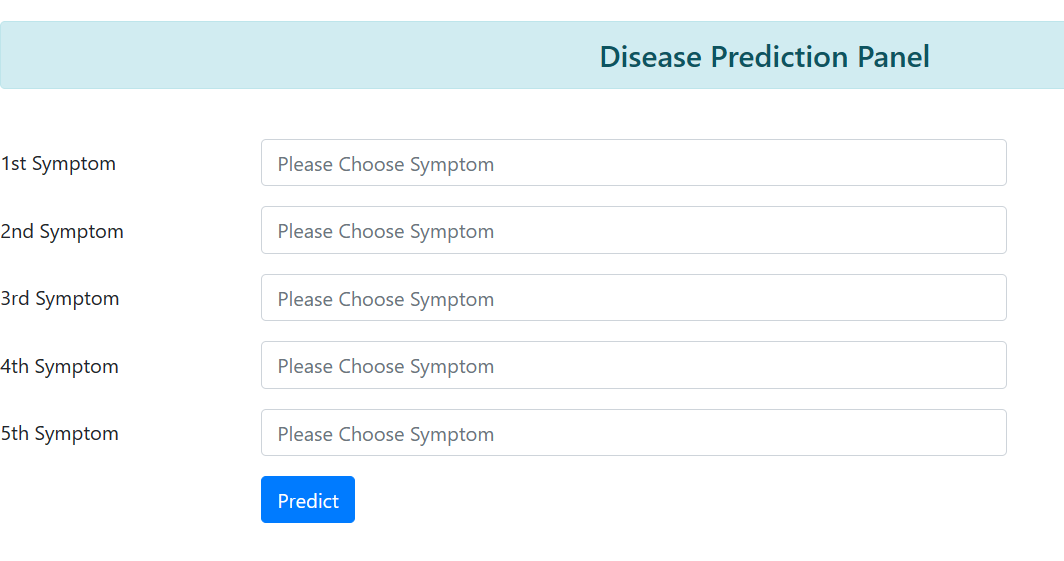


Figure – 75 Disease prediction pannel

### 4.10.10 Patient Result

The user can view Id, Name, Disease, Medicine, and Appointment in the diagnosis result section on the left sidebar after the disease has been predicted.

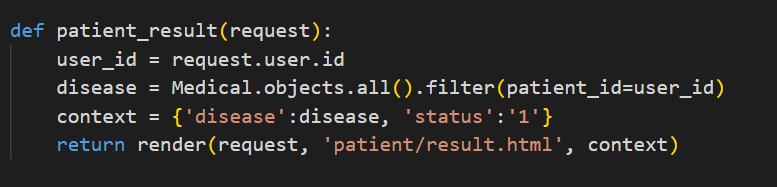


Figure – 76 Patient result

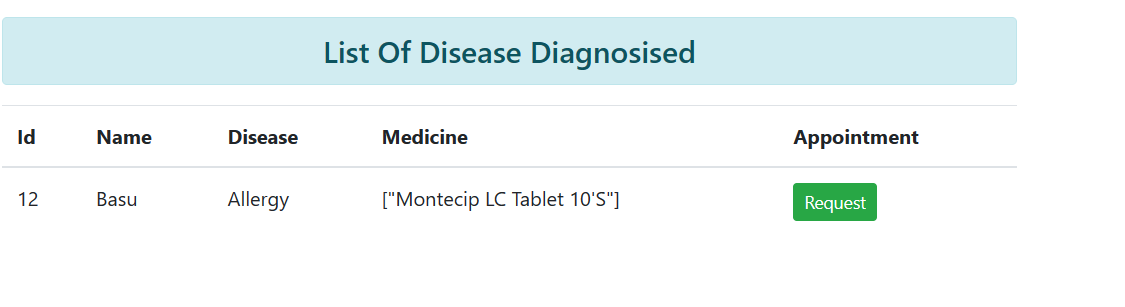


Figure – 77 List of disease

### 4.10.11 Appointment Requested by Patient

The user can make an appointment after forecasting the outcome. The message "Appointment Request Exist" will appear if an appointment is available; if not, the doctor will receive the request.

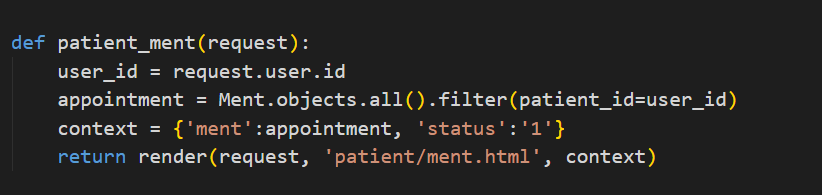


Figure – 78 Requesting appointment

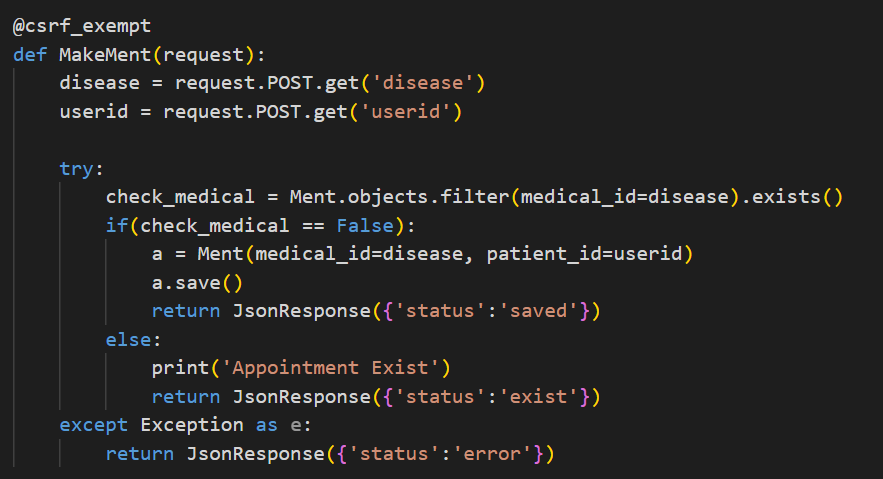


Figure – 79 Requesting appointment – 2

### 4.10.13 Drug Recommendation

Following the prediction of the disease, the user\_id and the disease are sent to the backend to the result.html file, where the doctor may view the following information: Id, Name, Disease, Medicine recommendation (if any), and Action column (where the doctor can click to propose a medication).After clicking the "recommend" button, the patient's user ID and the disease are taken by the Make\_mend function.The illness is then looked up in the disease\_list, which is a list of all the illnesses. If the disease the user has experienced is on the disease\_list, a new list is created with the predicted disease, gender, and age. It is then transformed into an array using an array and sent into the model to forecast the medication.In the event that the model is unable to propose a medication, it will indicate in the Medicine portion of the Drug Recommendation section on the left sidebar to "See Doctor." If the medication still needs to be recommended, it will indicate "Yet Recommended," otherwise it will display the suggested medication.

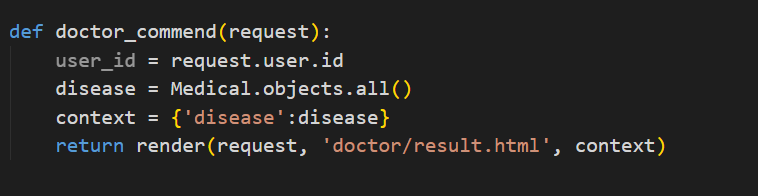


Figure – 80 Taking the disease from the backend

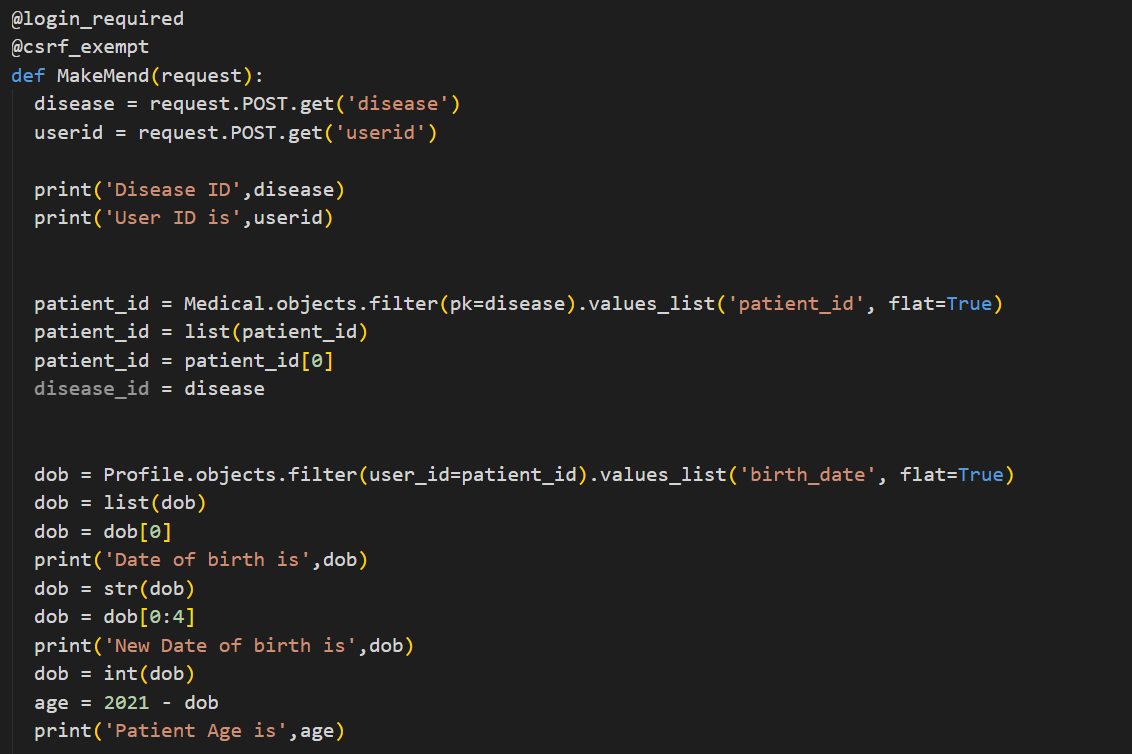


Figure – 81 Predicting drug – 1

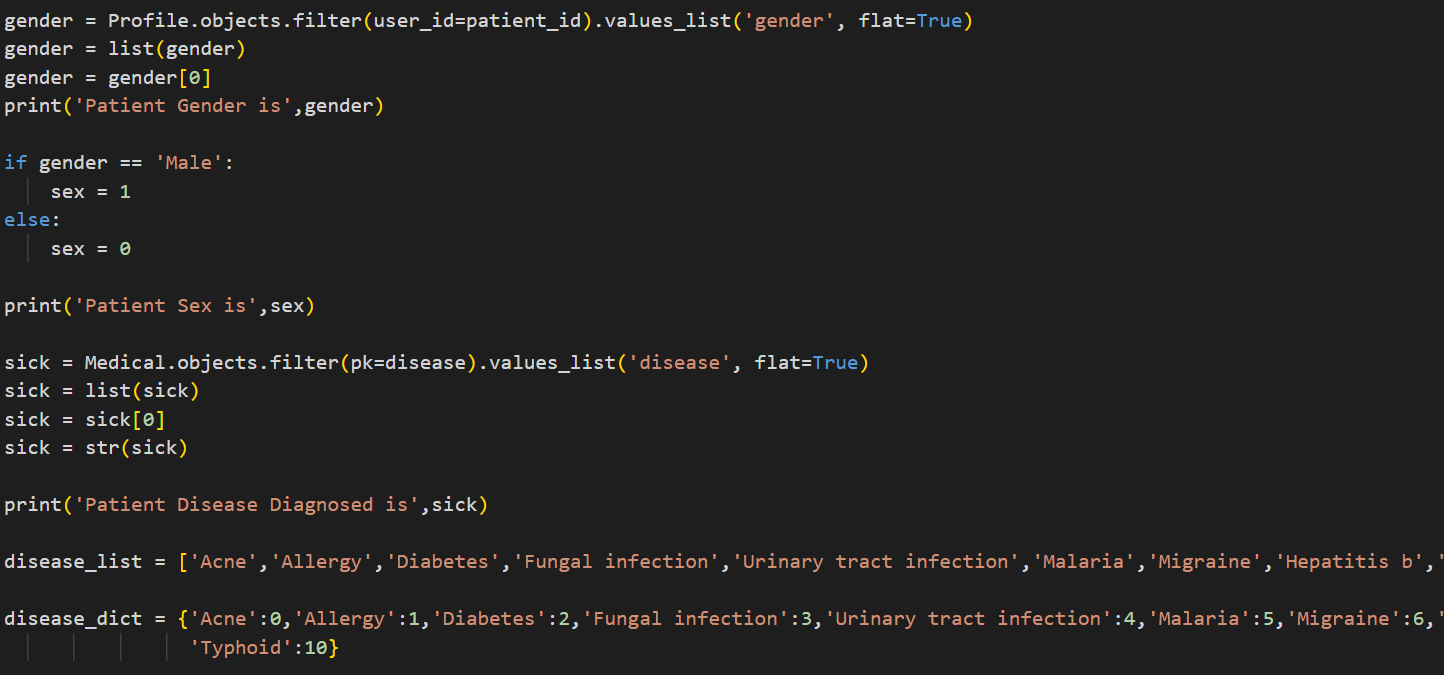


Figure – 82 Predicting drug – 2

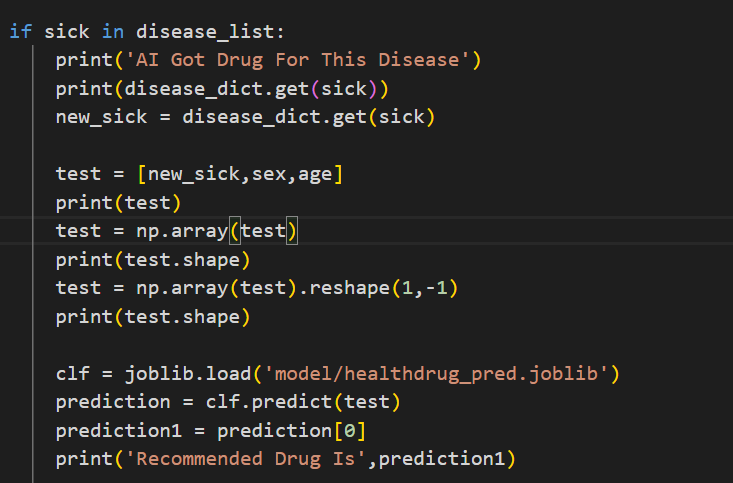


Figure – 83 Predicting drug – 3

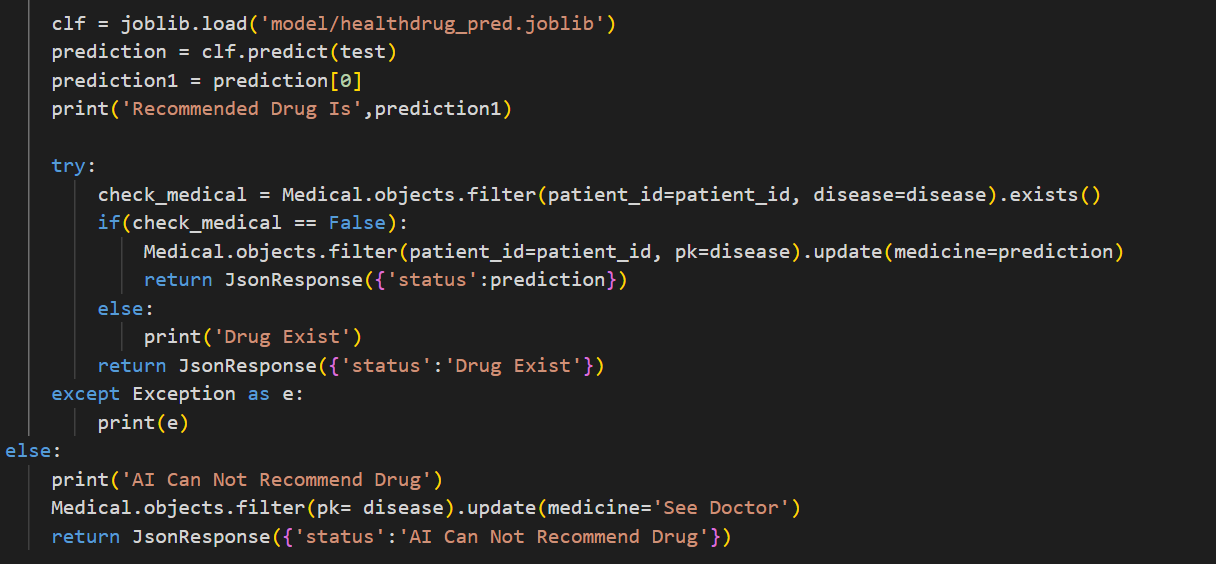


Figure – 84 Predicting drug – 4



Figure – 85 Disease diagnosed

### 4.10.14 Doctor approving Appointment

The doctor can reschedule the visit after recommending the medication.The patient's appointment status is passed, and the doctor then uses the Save\_ment function to enter the date and time, which the patient can view in the Appointment area of his dashboard.

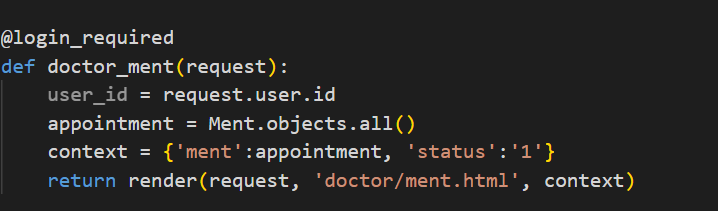


Figure – 86 Appointment request



Figure – 87 Schedule the appointment

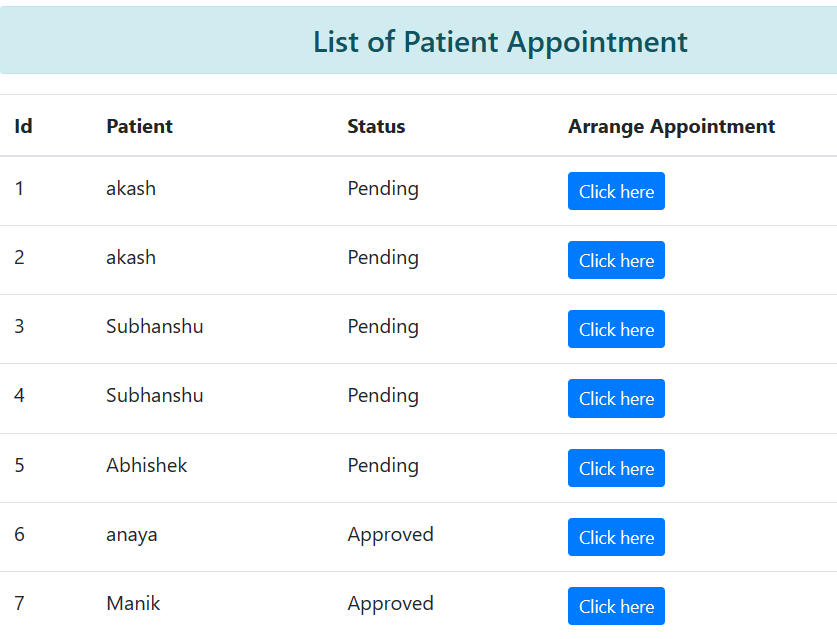


Figure – 88 Appointment schedule

### 4.10.15 Logout

This function assists the user in logging out of the dashboard and returning to the login screen.

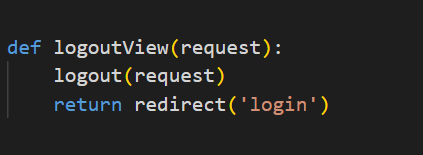


Figure – 89 Logout function

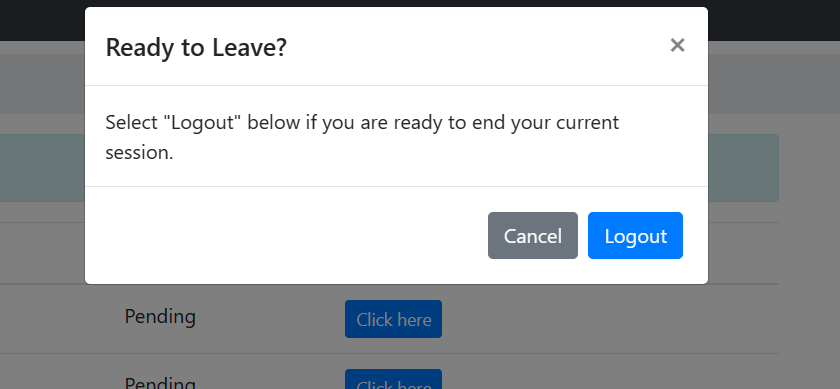


Figure – 90 Logout

# Chapter –5 Result and Discussion

## 5.1 Result

### 5.1.1 Patient Registration

Click the “Register” link in the top right corner of the homepage to register with the healthcare system. You’ll reach the signup page by doing this. In the corresponding sections, type your preferred username, password, and email address. To finish the registration procedure, click the blue “Register” button after completing the form. You will get a message that account was created Successfully.

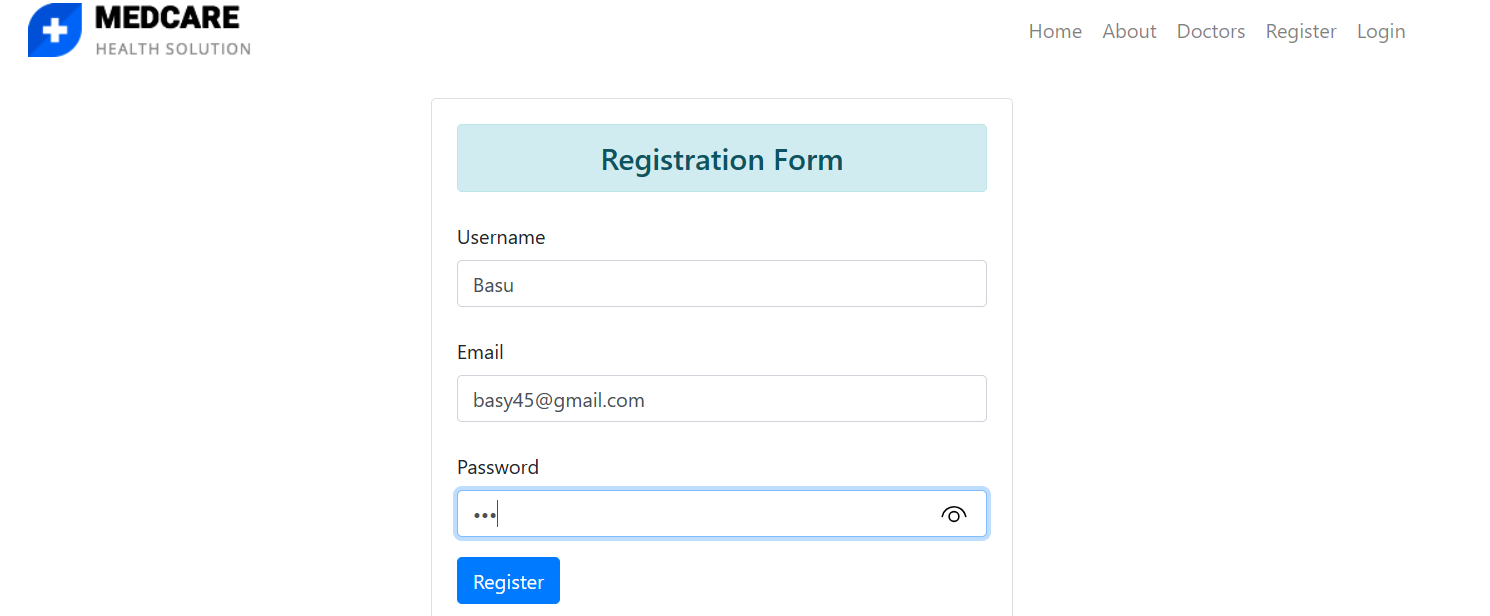


Figure – 91 User registration

After entering all the details click on Register button and you will be registered successfully

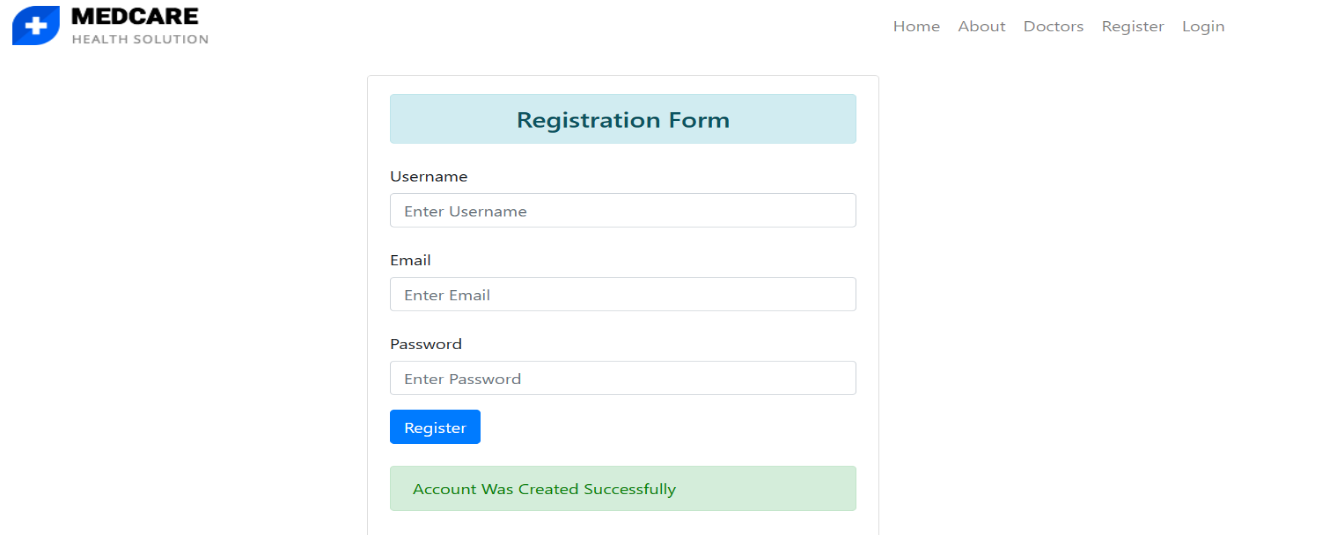


Figure – 91 User registered

### 5.1.2 Patient Login

Once you've successfully registered, click the "Login" icon in the top right corner of the homepage. On the login page, fill in the corresponding fields with your username and password.



Figure – 92 User login

To access the patient dashboard, click the blue “Login” button

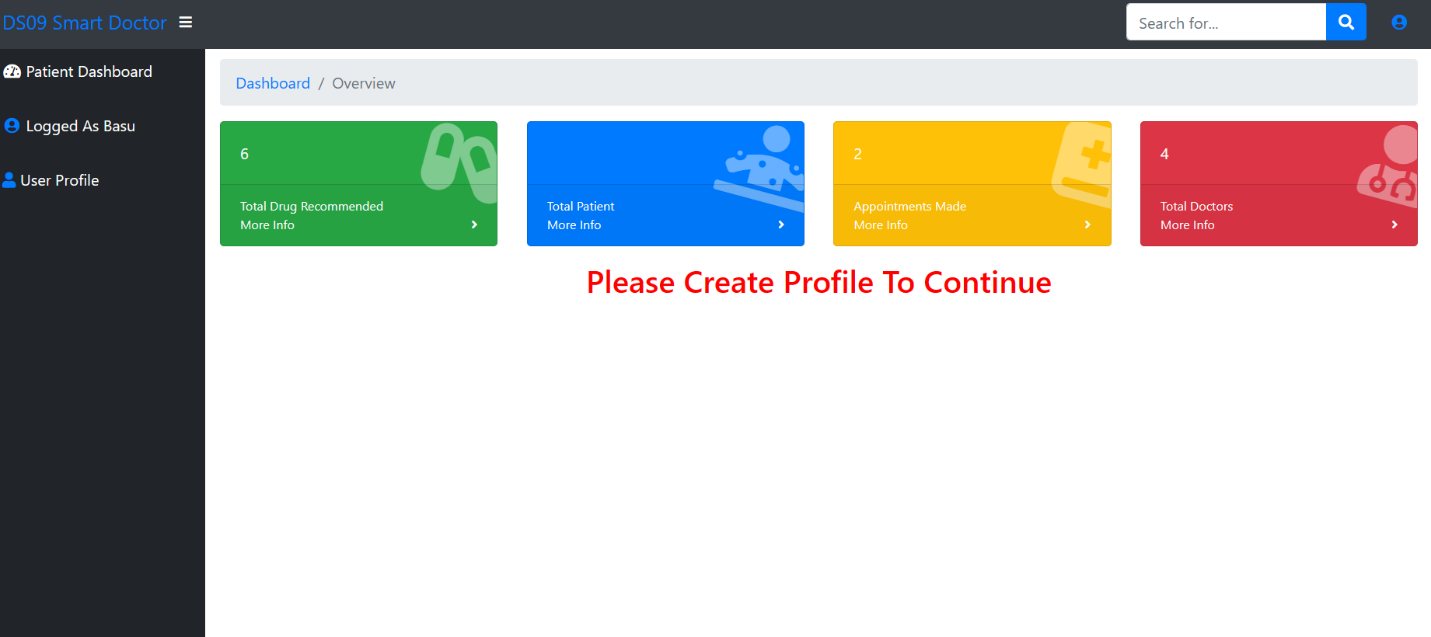


Figure – 93 User dashboard

### 5.1.3 Patient Profile

Before you can forecast a disease, the User Profile field needs to be filled out. This entails giving your gender, nationality, region, and date of birth. To accomplish this, select "User Profile" from the patient dashboard's left sidebar and enter the necessary information.



Figure – 94 Creating profile

Press the "Save" button to start building your profile. After your profile has been saved, the Patient Dashboard will appear.

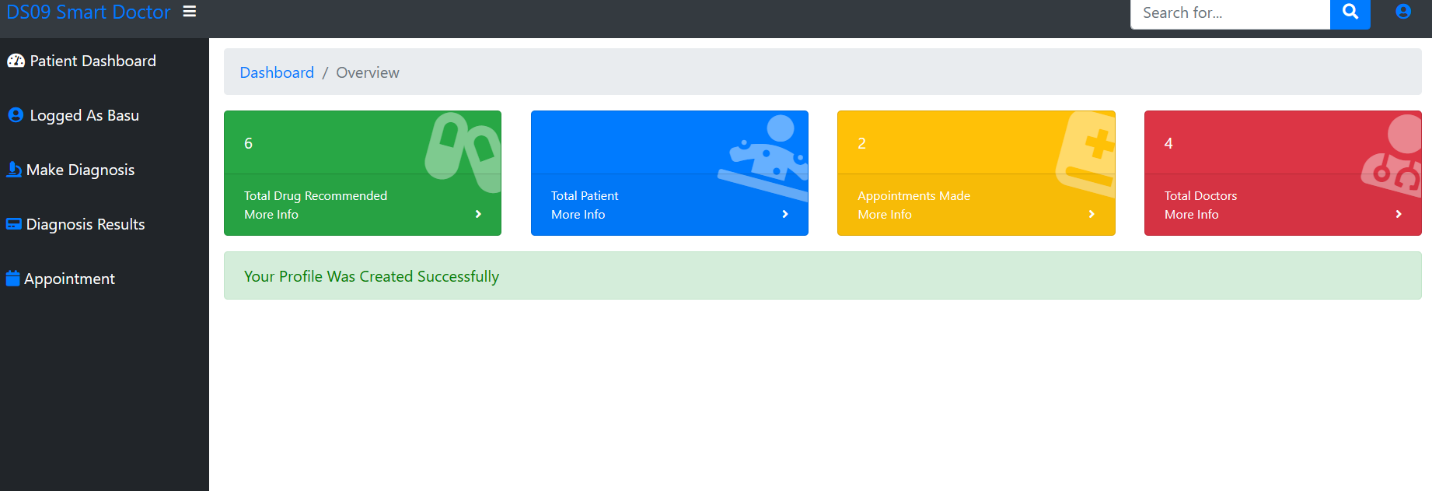


Figure – 95 Profile created

### 5.1.4 Doctor Resgistartion

The physician must first register as a patient in order to register as a doctor. The backend staff will authorize and upgrade the account to doctor status after the registration process is finished. Before giving doctors privileges, this procedure makes sure that the right authorization and verification are done.

Register as Patient-

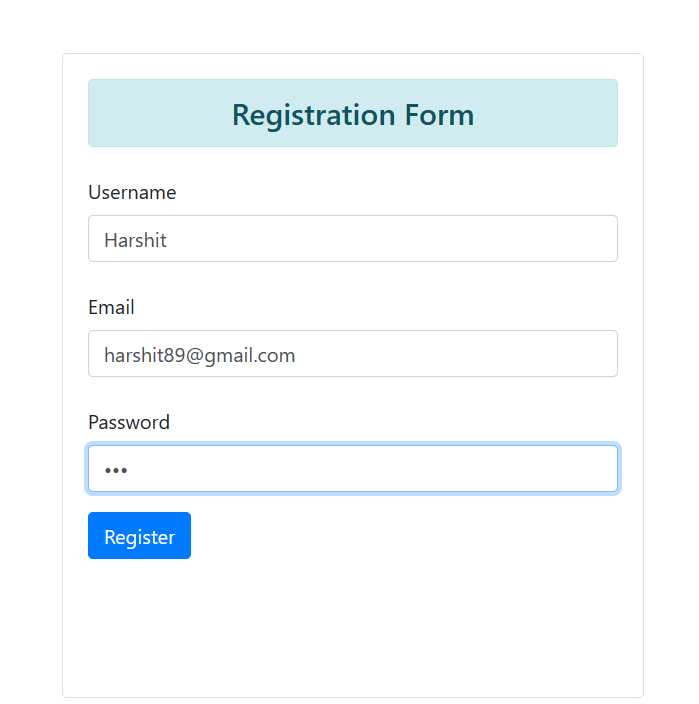


Figure – 96 Doctor registration as patient

Click on the Register to create the account-

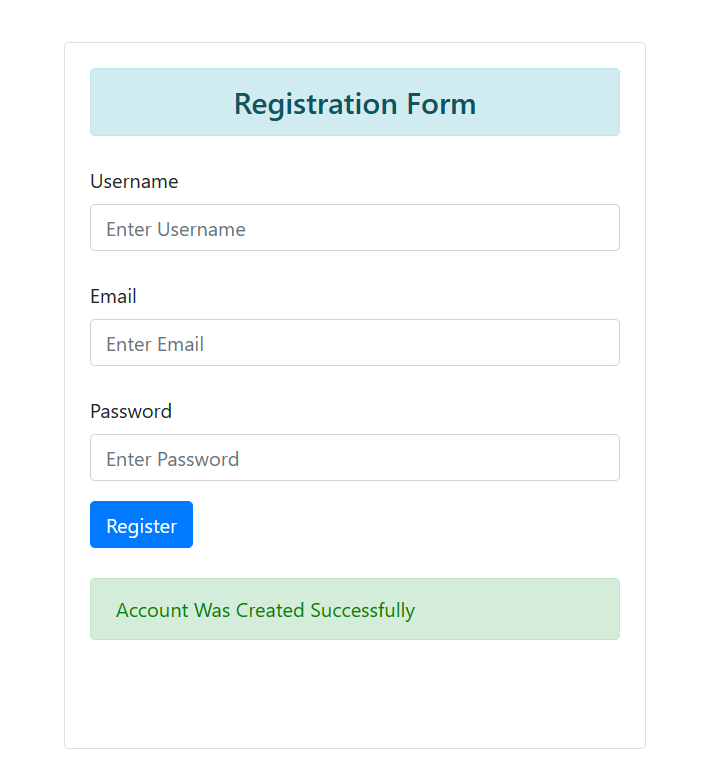


Figure – 97 Registration succseddfiulll

After the Doctor has registered as the patient his status will be changed by the backend team. Here we can see that the doctor Harshit is registered as a Patient -

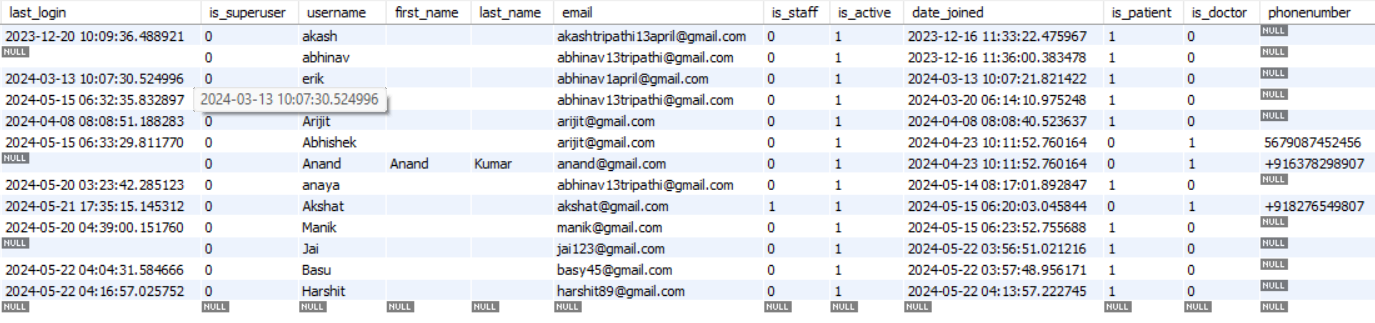


Figure – 98 Activating doctors account

Harshit can be designated as a doctor in the database by setting the is\_patient as 0, is\_doctor, and is\_staff values to 1, respectively, and by adding a phone number.

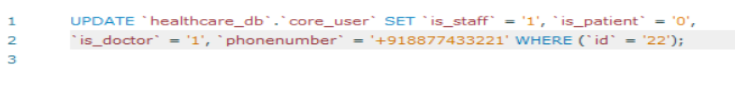


Figure – 99 SQL query

Following the execution of this query, the database will update as indicated below-

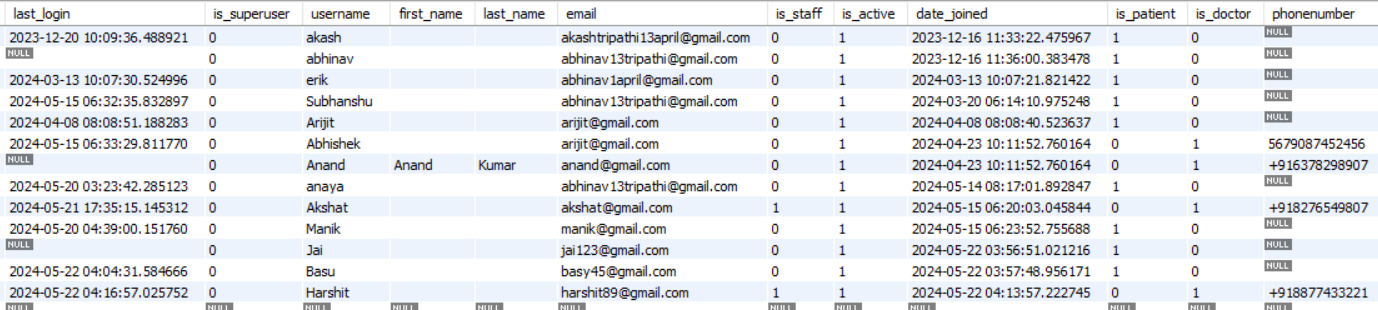


Figure – 100 Account activated

Now Login again and Harshit will be directed to the Doctor Dashboard-

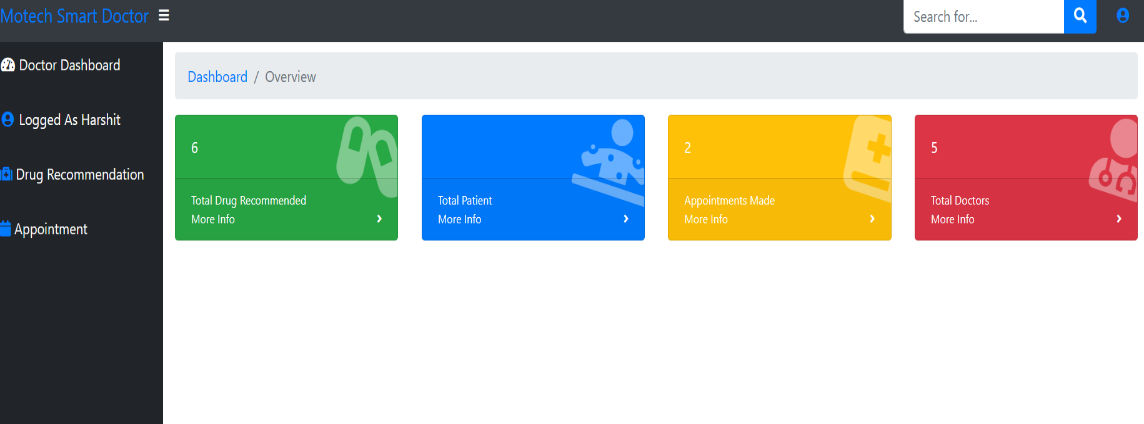


Figure – 101 Doctor login

### 5.1.5 Disease Prediction

For predicting a disease, the patient should log in and click on “Make Diagnosis” on the left sidebar-

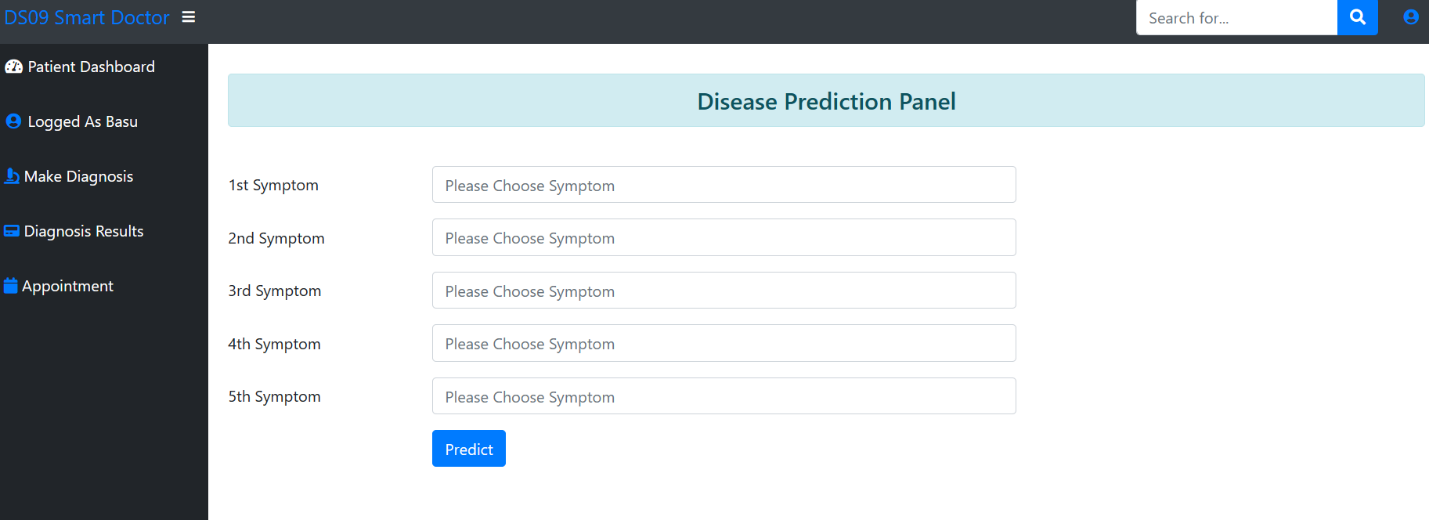


Figure – 102 Disease prediction Pannel

Fill the symptoms and click on the Predict button-

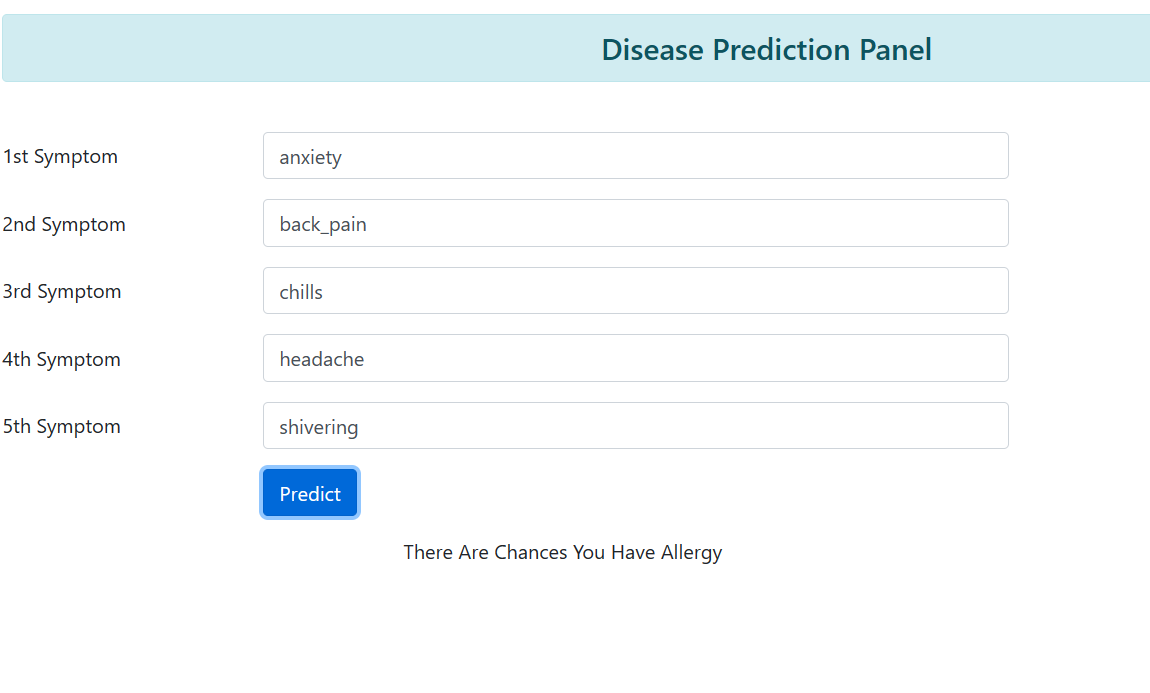


Figure – 103 Giving the symptoms and predicting the disease

You can see it is showing that You have chances of having Allergy.

Click on "Diagnosis Result" on the left sidebar to see if the doctor has prescribed any medicine or not.

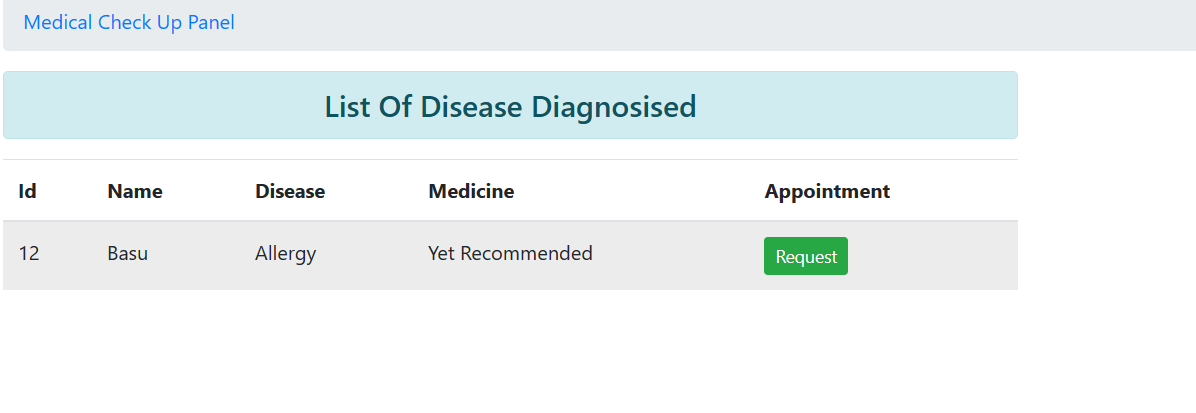


Figure – 104 Drug recommended

### 5.1.6 Appointment Request

Click on the green "Request" button in the Diagnosis Result section to request an appointment with the doctor-

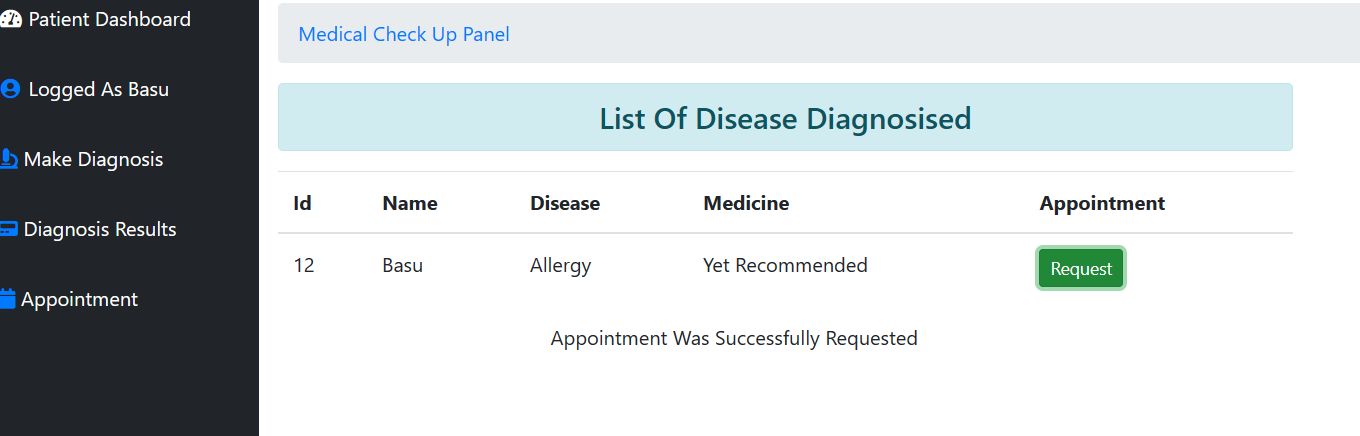


Figure – 105 Appointment request

Click on the “Appointment” section on the left sidebar to check if the doctor has provided the appointment schedule.

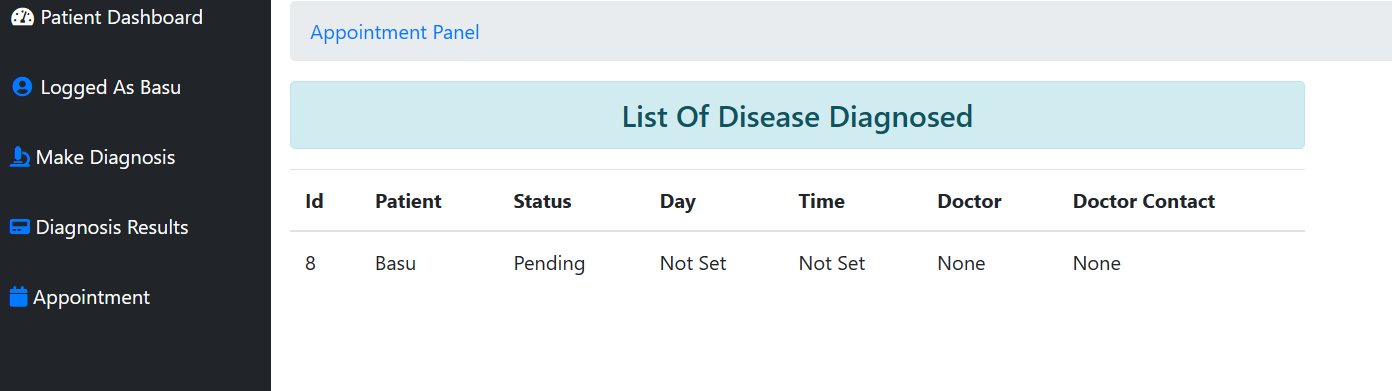


Figure – 106 Appointment schedule

### 5.1.7 Drug Prediction

To predict the drug, the doctor needs to click on the "Drug Recommendation" section in the left sidebar and then click on the blue "Recommend" button to suggest the appropriate medication.

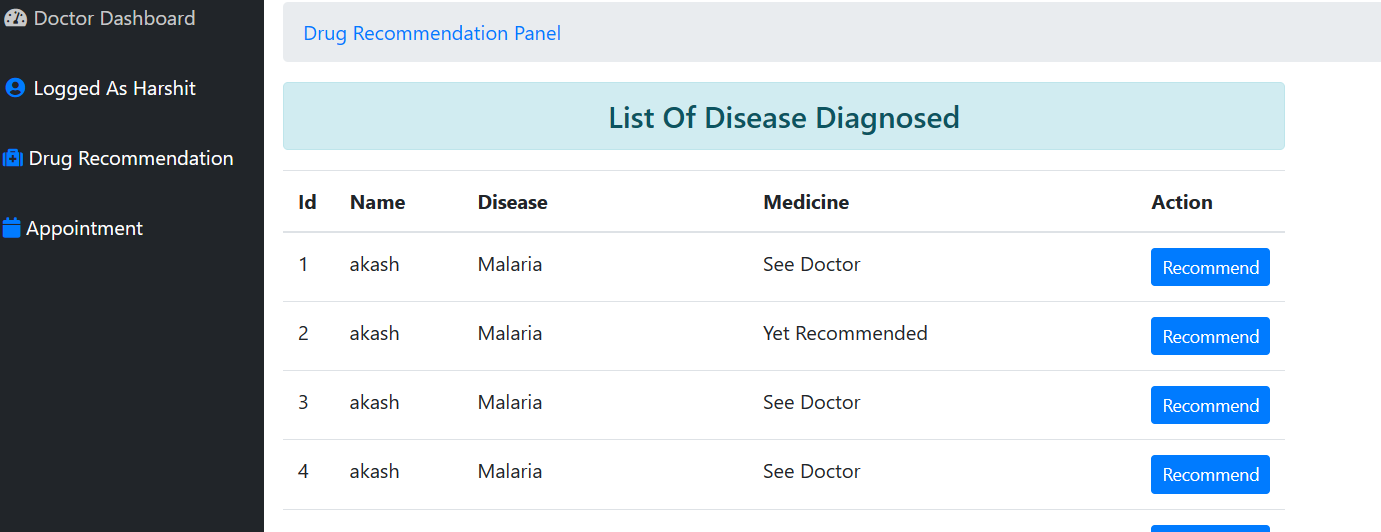


Figure – 107 Drug prediction – 1



Figure – 108 Drug prediction – 2

Patients can view if the doctor has predicted the drug in the "Diagnosis Result" section.

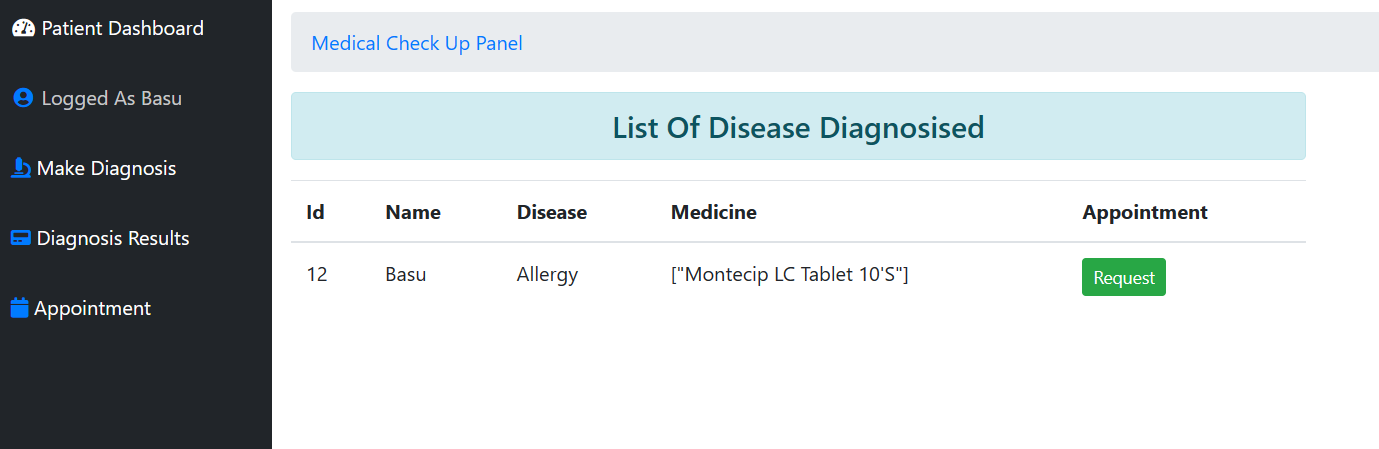


Figure – 109 Medicine recommended

### 5.1.8 Appointment Schedule

The doctor can schedule the appointment for the patient by navigating to the “Appointment” section, where there are three columns: ID, Patient, and Status. The Status column indicates whether the appointment has been approved or not. The doctor can arrange the appointment by clicking on the blue “Click Here” button.

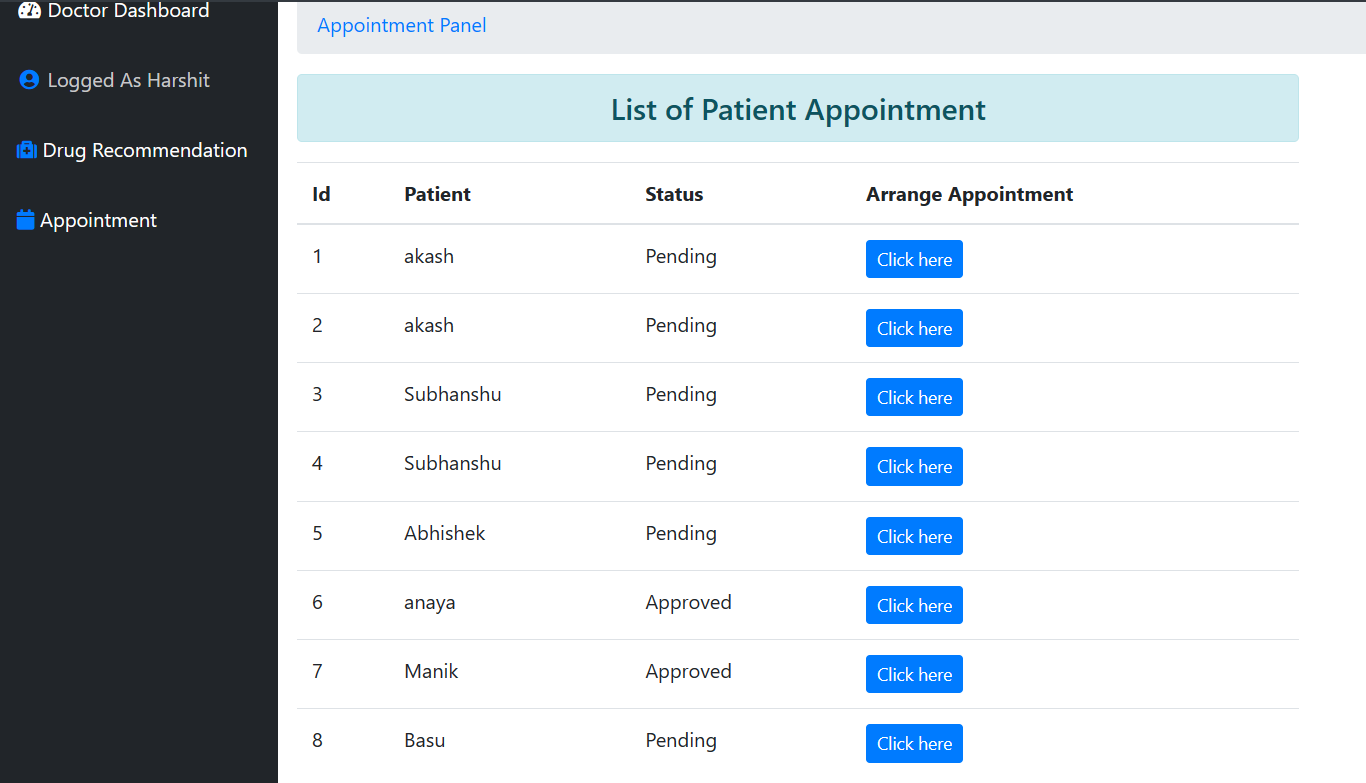


Figure – 110 Scheduling appointment

After clicking the "Arrange Appointment" button, the "Arrange Appointment For Patient" section will appear, allowing the doctor to choose the day and time for the appointment. Once the doctor has selected the desired day and time, they can click on the "Submit" button to finalize the appointment scheduling process.

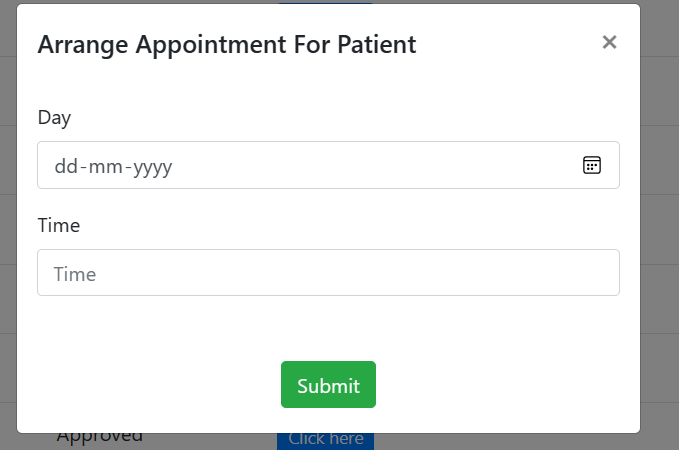
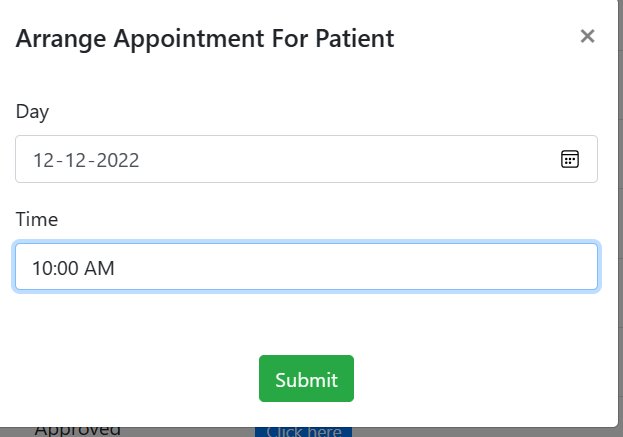


Figure – 111 Fixing date and time



Fixing – 112 Scheduled appointment

The patient can check if the appointment has been scheduled, including the date and time, doctor’s name, and doctor’s contact number, by clicking on the “Appointment” section in the left sidebar.

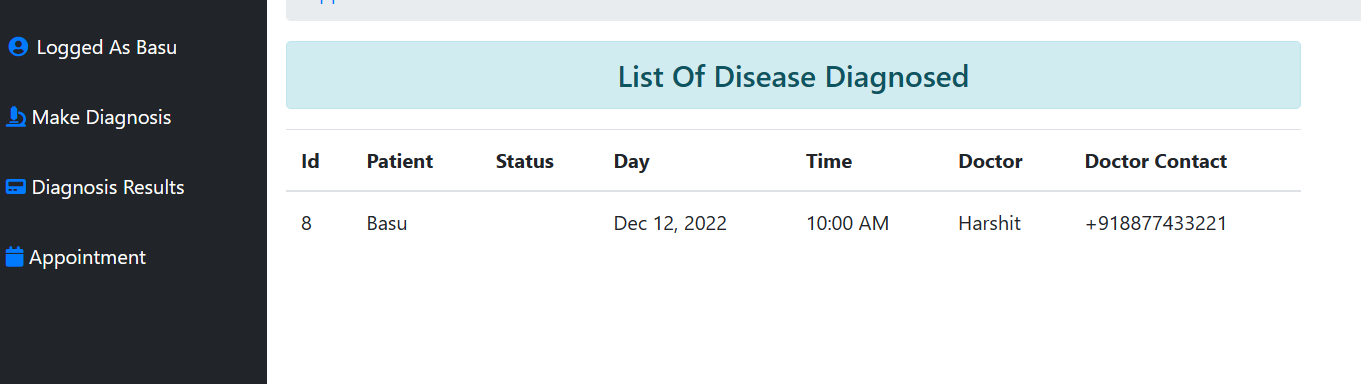


Figure – 113 Patient appointment view

5.1.9 Logout

To log out, click on the icon located at the upper right side of the dashboard, then select the “Logout” option.

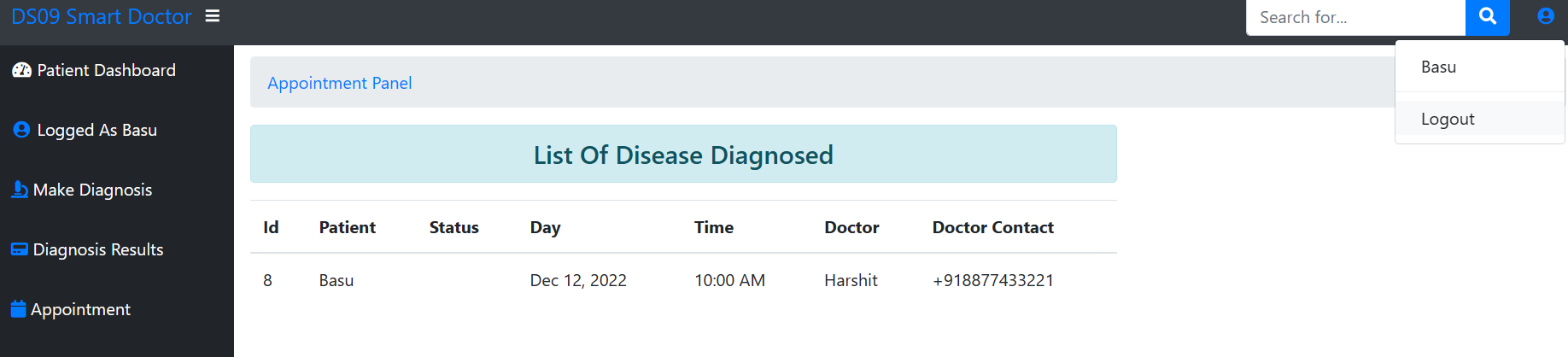


Figure – 114 Logout

## 5.2 Discussion

The creation of a healthcare web application using Django offers a viable path for enhancing healthcare services via technological integration. By utilizing machine learning for medication recommendations and illness prediction in conjunction with patient and physician-friendly interfaces, this initiative seeks to optimize the healthcare system.

### 5.2.1 User Authentication and Dashboard Functionality

With different login and registration sites for patients and doctors, the platform's user authentication feature guarantees secure access. After logging in, users are taken to their dashboards, which include customized features. While doctors may forecast medications, schedule appointments, and check patient data, patients can enter symptoms, request appointments, and get diagnosis results.

### 5.2.2 Modular Template Structure

Simple navigation and customization are made possible by the usage of a modular template structure, which uses separate HTML files for various pages and capabilities. Home.html, base.html, login.html, register.html, diagnose.html, result.html, and ajax.html are important templates. Throughout the application, this methodical approach guarantees readability and maintainability.

### 5.2.3 Database Management and User Profile

Managing profiles and keeping user data need database administration. Users can assign different roles and permissions within the system by specifying particular fields like is\_staff, is\_doctor, and is\_patient. Moreover, the incorporation of user profiles facilitates tailored experiences and effective data administration.

### 5.2.4 Appointment Scheduling and Communication

Efficient contact between physicians and patients is made possible by the appointment scheduling tool. Doctors may accept, decline, or reschedule appointments at the request of patients, depending on availability. Direct connection is ensured and the patient-doctor relationship is strengthened by including the doctor's contact information.

# Chapter-6 Conclusion and Future Scope

## 6.1 Conclusion

In conclusion, the Django framework was successfully used to construct the healthcare project and accomplish its goals. We have created a strong user registration and authentication system that guarantees safe access for physicians and patients alike. The backend team can verify doctors and maintain their status through a smooth registration process. The right procedures are followed to guarantee security and verification, demonstrating the legitimacy and dependability of doctors. Notably, the backend team must get authorization before adding doctors, thus only licensed medical practitioners are permitted access to the system. By effectively rerouting users to their corresponding dashboards according to their roles, the login functionality improves user experience.

The system shows important data like the overall number of physicians, patients, appointments, and prescribed medications, together with extensive information on the healthcare environment. Furthermore, it guarantees that patients establish their profiles prior to using diagnostic features, protecting user data integrity and completeness.

Users can quickly and precisely enter their information into a simplified profile setup, and their information is confirmed and stored with a success message. Using user-selected symptoms, the diagnostic component makes predictions about possible diseases. The system facilitates proactive healthcare management by providing dependable diagnostic recommendations by transforming user input into a predictive model-compatible format.

In addition, physicians have the ability to forecast drug outcomes using the disease forecasts produced by the patient dashboard. This feature improves the therapeutic process by allowing physicians to prescribe the right drugs to patients. Doctors can also schedule appointments by giving the time and date, which guarantees a well-organized and effective scheduling process.

All things considered, the project successfully integrates safe data management, easy-to-use interfaces, and efficient health diagnostics. Appropriate steps to guarantee security and verification give the system an additional degree of dependability and trust. This project offers a strong basis for scalability and future improvements in the healthcare industry. Because of the thorough implementation, all procedures are guaranteed to be well-documented, providing a clear route for future growth and development.

## 6.2 Future Scope

### Enhanced Machine Learning Models:

* Advanced Diagnostics: Increase the range and accuracy of disease forecasts by incorporating artificial intelligence and more advanced machine learning algorithms.
* Personalized Medicine: Create algorithms that will provide individualized therapy suggestions based on genetic and medical histories.

### Telemedicine Integration:

* Virtual Consultations: By using telemedicine capabilities, physicians can confer virtually with patients, improving access to medical treatment.
* Remote Monitoring: Provide patients with real-time health metrics tracking and prompt actions by introducing remote monitoring solutions.

### Mobile Application:

* Mobile Access: Provide mobile applications to improve convenience and usefulness by enabling doctors and patients to access the system while on the go.
* Push Notifications: Set up push notifications for health advice, medication alerts, and appointment reminders.

1. Comprehensive Health Records:

* EHR Integration: Make smarter decisions by integrating Electronic Health Records (EHR) to give a complete picture of a patient's medical history.
* Data Interoperability: Assure compatibility with databases and other healthcare systems to facilitate provider collaboration and data exchange.

1. Enhanced Security Measures:

* Two-Factor Authentication: Adding two-factor authentication (2FA) to the login process will provide an extra degree of protection.
* Data Encryption: Improve data encryption techniques to safeguard private patient and physician data.

1. Expanded User Roles:

* Role-Based Access Control: Provide more precise role-based access control to regulate the system's permissions and access levels for various user types.
* Support Staff Integration: Include features that will allow nurses and other administrative staff members, as well as other healthcare support staff, to contribute to and access pertinent areas of the system.

1. Patient Engagement Tools:

* Health Education: Assist patients in understanding their diseases and available treatments by providing them with educational information and tools.
* Feedback Mechanism Establish a feedback system so that clients may evaluate and assess their experiences, which will enhance the quality of the services provided.

1. Calability and Performance:

* Load Balancing: Use performance optimization strategies like load balancing to make sure the system can manage growing traffic as the user base expands.
* Cloud Integration: Investigate cloud-based options to ensure that the system can expand to meet the demands of patients and healthcare providers by providing scalable processing and storage capacity.

1. Regulatory Compliance:

* HIPAA Compliance: Ascertain that the system satisfies all applicable legal standards for managing patient data, including HIPAA in the US.
* Global Standards: Modify the system for worldwide deployment so that it complies with international standards and laws.

1. Mental Health Prediction:

* Research Integration: By utilizing current findings, expand the system to include mental health prediction skills. A study article in this area, which our team has previously published, offers a strong basis for incorporating mental health diagnoses and support into the system.
* Mental Health Tools: Provide features and tools for mental health evaluation and assistance so that complete healthcare can be provided, addressing both mental and physical health issues.

1. Sentinel Health Assist Integration:

* Advanced Monitoring: Incorporate Sentinel Health Assist, a cutting-edge health monitoring system with an intricate sensor array intended to track behavioral and physiological signs. This system collects detailed data about movement, heart rate, stress levels, and anxiety using sensors such as accelerometers, temperature sensors, electrodermal activity sensors, and heart rate monitors.
* Real-Time Analysis: Employ cutting-edge deep learning and machine learning techniques, like Generative Adversarial Networks, Random Forest, Naïve Bayes, K-Nearest Neighbor, Convolutional Neural Networks, and Naïve Bayes, to evaluate data in real-time and spot patterns that could point to mental health problems and health hazards.
* Personalized Alerts: Encourage users to take proactive steps like staying hydrated, interacting with others, getting medical attention, or exercising by sending them individualized notifications and recommendations.
* Family and Friends Alerts: To enable prompt support and help, gather family members' or friends' contact information in advance and advise them of the user's condition.

Additionally, we want to patent this idea.

1. Community and Support:

* Support Forums: Establish community areas and support forums where users may ask questions, share experiences, and receive guidance from professionals and other users..
* Continuous Training: Provide doctors and staff with ongoing training and development opportunities to stay current on the newest features and industry best practices.

By concentrating on these upcoming improvements, the healthcare project can proceed to develop, offering users more complete, easily accessible, and safe healthcare options. The enhancements will not only augment the system's functionality and user experience, but also guarantee its continued adaptability to the dynamic field of healthcare technology.

# References –

1. Corley C. D., Pullum L. L., Hartley D. M., et al. Disease prediction models and operational readiness. PLoS One . 2014;9(3):p. 91989. doi: 10.1371/journal.pone.0091989. [[PMC free article](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3960139/)] [[PubMed](https://pubmed.ncbi.nlm.nih.gov/24647562)] [[CrossRef](https://doi.org/10.1371%2Fjournal.pone.0091989)] [[Google Scholar](https://scholar.google.com/scholar_lookup?journal=PLoS+One&title=Disease+prediction+models+and+operational+readiness&author=C.+D.+Corley&author=L.+L.+Pullum&author=D.+M.+Hartley&volume=9&issue=3&publication_year=2014&pages=p.+91989&doi=10.1371/journal.pone.0091989&)]

2. Bashir S., Qamar U., Khan F. H., Javed M. Y. MV5: a clinical decision support framework for heart disease prediction using majority vote based classifier ensemble. Arabian Journal for Science & Engineering . 2014;39(11):7771–7783. doi: 10.1007/s13369-014-1315-0. [[CrossRef](https://doi.org/10.1007%2Fs13369-014-1315-0)] [[Google Scholar](https://scholar.google.com/scholar_lookup?journal=Arabian+Journal+for+Science+&+Engineering&title=MV5:+a+clinical+decision+support+framework+for+heart+disease+prediction+using+majority+vote+based+classifier+ensemble&author=S.+Bashir&author=U.+Qamar&author=F.+H.+Khan&author=M.+Y.+Javed&volume=39&issue=11&publication_year=2014&pages=7771-7783&doi=10.1007/s13369-014-1315-0&)]

3. Manemann S. M., Sauver J., Liu H. Longitudinal cohorts for harnessing the electronic health record for disease prediction in a US population. BMJ Open . 2021;11(6, article 044353) doi: 10.1136/bmjopen-2020-044353. [[PMC free article](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8190051/)] [[PubMed](https://pubmed.ncbi.nlm.nih.gov/34103314)] [[CrossRef](https://doi.org/10.1136%2Fbmjopen-2020-044353)] [[Google Scholar](https://scholar.google.com/scholar_lookup?journal=BMJ+Open&title=Longitudinal+cohorts+for+harnessing+the+electronic+health+record+for+disease+prediction+in+a+US+population&author=S.+M.+Manemann&author=J.+Sauver&author=H.+Liu&volume=11&issue=6,+article+044353&publication_year=2021&doi=10.1136/bmjopen-2020-044353&)]

4. Collier R. H. Pest and Disease Prediction Models. Encyclopedia of Applied Plant Sciences . 2017;3:120–123. doi: 10.1371/journal.pone.0091989. [[CrossRef](https://doi.org/10.1371%2Fjournal.pone.0091989)] [[Google Scholar](https://scholar.google.com/scholar_lookup?journal=Encyclopedia+of+Applied+Plant+Sciences&title=Pest+and+Disease+Prediction+Models&author=R.+H.+Collier&volume=3&publication_year=2017&pages=120-123&doi=10.1371/journal.pone.0091989&)]

5. El-Shafiey M. G., Hagag A., El-Dahshan E. A hybrid GA and PSO optimized approach for heart-disease prediction based on random forest. Multimedia Tools and Applications . 2022;81(13):18155–18179. doi: 10.1007/s11042-022-12425-x. [[CrossRef](https://doi.org/10.1007%2Fs11042-022-12425-x)] [[Google Scholar](https://scholar.google.com/scholar_lookup?journal=Multimedia+Tools+and+Applications&title=A+hybrid+GA+and+PSO+optimized+approach+for+heart-disease+prediction+based+on+random+forest&author=M.+G.+El-Shafiey&author=A.+Hagag&author=E.+El-Dahshan&volume=81&issue=13&publication_year=2022&pages=18155-18179&doi=10.1007/s11042-022-12425-x&)]

# Publication

Title – Diagnosis of Brain’s Health Condition through Smart ML Algorithm through Brain Waves

Publisher – IEEE

Authors - Dimple Tiwari, Anand Kumar, Abhinav Akash, Kamakshi Agarwal, Akshat Sharma, Nitesh Singh

Abstract:

Mental health problems, when diagnosed early on, help professionals to provide better care to patients in the starting phase itself and improve the quality of life for the patients. Mental health issues, if not treated at an early age, lead to degradation in the conditions of patients. Machine learning techniques in recent years have given promising results in predicting patients’ conditions. This study assessed the efficacy of five machine learning algorithms in detecting common mental health issues using various accuracy metrics, ROC, and classification reports. More than 1200 examples comprise the data collection gathered for approach testing and training. Fourteen – five crucial characteristics have been determined for the machine learning algorithms. The accuracy of the machine learning algorithms was compared. The decision tree classifier gave more accurate results, and the gap between their performances over the complete attribute set and the selected attribute set is insignificant and can be overcome by hyperparameter tunning. This research will be useful in developing a future health case system.

# Plagiarisms Report