A PROJECT REPORT

ON

"SMART HEALTH DISEASE PREDICTION SYSTEM"

Submitted to

UNIVERSITY OF MUMBAI

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER ENGINEERING

BY

SHAIKH FAIYYAZ TANVIR UZMA	17CO45
SHAIKH SOAIB RIYAZ RABBANA	17CO48
SHAIKH UBED IKRAM SHABANA	17CO49
NIVEKAR BILAL HAMZA SHAHEEN	16DCO66

UNDER THE GUIDANCE OF
PROF. ABDUL SALAM SHAIKH

DEPARTMENT OF COMPUTER ENGINEERING
Anjuman-I-Islam's Kalsekar Technical Campus
SCHOOL OF ENGINEERING & TECHNOLOGY
Plot No. 2 3, Sector - 16, Near Thana Naka,
Khandagaon, New Panvel - 410206

2020-2021

AFFILIATED TO

UNIVERSITY OF MUMBAI

A PROJECT II REPORT
ON

"SMART HEALTH DISEASE PREDICTION SYSTEM"

Submitted to
UNIVERSITY OF MUMBAI

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER ENGINEERING

BY

SHAIKH FAIYYAZ TANVIR UZMA	17CO45
SHAIKH SOAIB RIYAZ RABBANA	17CO48
SHAIKH UBED IKRAM SHABANA	17CO49
NIVEKAR BILAL HAMZA SHAHEEN	16DCO66

UNDER THE GUIDANCE OF
PROF. ABDUL SALAM SHAIKH

DEPARTMENT OF COMPUTER ENGINEERING
Anjuman-I-Islam's Kalsekar Technical Campus
SCHOOL OF ENGINEERING & TECHNOLOGY
Plot No. 2 3, Sector - 16, Near Thana Naka,
Khandagaon, New Panvel - 410206

2020-2021
AFFILIATED TO

UNIVERSITY OF MUMBAI

Anjuman-i-Islam's Kalsekar Technical Campus

Department of Computer Engineering

SCHOOL OF ENGINEERING & TECHNOLOGY

Plot No. 2 3, Sector - 16, Near Thana Naka,

Khandagaon, New Panvel - 410206

CERTIFICATE

This is certify that the project entitled

"SMART HEALTH DISEASE PREDICTION SYSTEM"

submitted by

SHAIKH FAIYYAZ TANVIR UZMA	**17CO45**
SHAIKH SOAIB RIYAZ RABBANA	**17CO48**
SHAIKH UBED IKRAM SHABANA	**17CO49**
NIVEKAR BILAL HAMZA SHAHEEN	**16DCO66**

is a record of bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Computer Engineering) at ***Anjuman-I-Islam's Kalsekar Technical Campus, Navi Mumbai*** under the University of MUMBAI. This work is done during year 2020-2021, under our guidance.

Date: / /

(Prof. ABDUL SALAM SHAIKH)
Project Supervisor

(Prof. KALPANA BODKE)
Project Coordinator

(Prof. TABREZ KHAN)
HOD, Computer Department

DR. ABDUL RAZAK HONNUTAGI
Director

External Examiner

Acknowledgements

We would like to take the opportunity to express our sincere thanks to our guide **Abdul Salam Shaikh**, Assistant Professor, Department of Computer Engineering, AIKTC, School of Engineering, Panvel for his invaluable support and guidance throughout our project research work. Without his kind guidance & support this was not possible.

We are grateful to him for his timely feedback which helped us track and schedule the process effectively. His time, ideas and encouragement that he gave helped us to complete our project efficiently.

We would like to express deepest appreciation towards **DR. ABDUL RAZAK HONNUTAGI**, Director, AIKTC, Navi Mumbai, **Prof. TABREZ KHAN**, Head of Department of Computer Engineering and **Prof. KALPANA BODKE**, Project Coordinator whose invaluable guidance supported us in completing this project.

At last we must express our sincere heartfelt gratitude to all the staff members of Computer Engineering Department who helped me directly or indirectly during this course of work.

SHAIKH FAIYYAZ
SHAIKH SOAIB
SHAIKH UBED
NIVEKAR BILAL

Project I Approval for Bachelor of Engineering

This project entitled ***Smart Health Disease Prediction System*** by ***Shaikh Faiyyaz, Shaikh Soaib, Shaikh Ubed and Nivekar Bilal*** is approved for the degree of ***Bachelor of Engineering in Department of Computer Engineering.***

Examiners

1. .....
2. .....

Supervisors

1. .....
2. .....

Chairman

.....

Declaration

We declare that this written submission represents our ideas in our own words and where others ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Shaikh Faiyyaz
Roll Number: 17CO45

Shaikh Soaib
Roll Number: 17CO48

Shaikh Ubed
Roll Number: 17CO49

Nivekar Bilal
Roll Number: 16DCO66

ABSTRACT

Title: Smart Health Disease Prediction System

We are living in an era where technology is advancing at unprecedented rate. Nowadays, Artificial Intelligence and Machine Learning are used in every domain wherein machines or computers are being trained to work automatically with very less human efforts. Machine Learning Algorithms are very useful in prediction, analysis and training. We will use ML Algorithms in predicting and analysis of various diseases in human beings. Health is one of the precious asset for a human being but due to the ongoing pandemic people can't recognize and treat their diseases from their home so we are aiming to develop a disease prediction system using ML Algorithms for better prediction of diseases by providing their symptoms to recognize the diseases more precisely with its consequences and treatment with the ease of using it at their own comfort zone. We are developing this system also because people can consult with their respective doctors via live consultation without physical contact.

Our interface would help people to some extent in-order to reduce the risk associated with predicted diseases to reduce the impact on other body parts. With the help of extensive powerful ML Algorithms accuracy is highest. People can also maintain safety protocols in this pandemic by predicting the diseases from their home.

Keywords: Scikit-learn, NumPy, Data Preprocessing, Dataset, Algorithms, Training Data, Training Set, Machine Learning, Naive Bayes, KNN, Decision Tree, Kernel SVM, Logistic Regression, Random Forest, Django, Training Model, Web Module, Data Collection Module, APIs, Artificial Intelligence, Disease Detection Module, Authentication, User Interface, SHDPS.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Smart Health Disease Prediction System is a Web Application used for predicting Human diseases by providing respective symptoms. Our system uses powerful Machine Learning Algorithms to predict the disease based on the symptoms provided by the users. This interface also helps users to track their consultation history, medical records and directly consult with specialist doctors to get cured as soon as possible.

1.1 Purpose

As we know that there is quite confusion amongst patients in recognizing and predicting their diseases based on their symptoms. This problem of patients should be minimized by developing a Disease Prediction System with providing their own symptoms with accurate prediction and analysis of that disease using powerful Machine Learning Algorithms. Since we're living in a pandemic where people can't consult doctors with physical communication. So this interface would surely facilitate them.

1.2 Project Scope

SHDPS will provide powerful analysis features using various libraries, APIs, and Machine Learning Algorithms with appropriate and precise disease prediction by providing symptoms. This will not only reduce anxiety of people but will also give full satisfactory results. SHDPS will have specialist recommendation for the respective diseases and patients can communicate with doctors with our unique feature Live Consultation as per their own convenience and comfort without physical contact due this pandemic.

1.3 Project Goals and Objectives

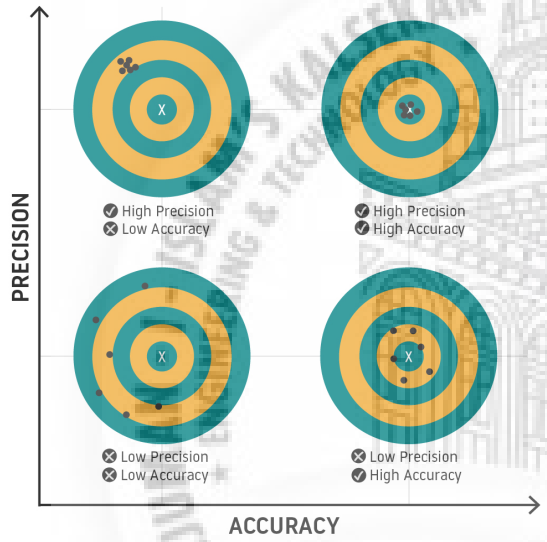
1.3.1 Goals

Early detection of Diseases

Prediction and detecting the diseases with the help of some symptoms at early stage would be much more beneficial to users in improving their health.

Effective and Accurate Prediction

Use of Extensive Machine learning algorithms to develop an interface with much better accuracy to avoid Emergency situations. To provide quick results in critical situations.



To Reduce Anxiety

Patients can have a Virtual meet with the Doctor so that they can convey their symptoms and it will diagnose and assist them in more effective manner rather than creating stress due to traditional disease predictions.

1.3.2 Objectives

Powerful Analysis Features

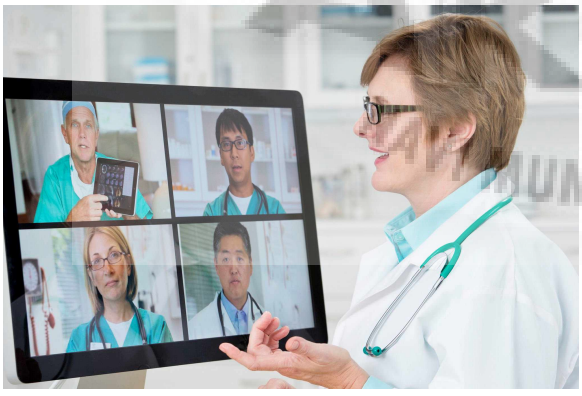
This system will diagnose and analyze patients symptoms more effectively using powerful Machine learning algorithms and classifiers and can recommend best available doctors to them.

Take care of Patient's Health

Patients can set reminders for their medicinal routine checkup and also for their regular appointments with their consulting Doctors.

Live Consultation

Patients can have a live chat or consultation as per their convenience with doctors for appropriate diagnosing the health problems which they are facing in order to treat them with proper care and prescription.



1.4 Organization of Report

The report is divided into nine chapters. Each chapter deals with the different aspects of SHDPS. Chapter 1 is about Introduction, it gives more emphasis on the purpose of creating. It also provides different project goals and objectives. Chapter 2 focuses on the literature survey of existing systems especially pros and cons. Also provides optimal resolutions to drawbacks of traditional systems.

Chapter 3 is about the overview of complete Project planning starting from scratch to its completion. It describes about the job roles assigned to different members of group and their project methodology approach. It also gives idea about the budget, constraints and chronology. Chapter 4 is Software Requirement Specification. It describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill all stakeholders needs. It explains about the Functional as well as Nonfunctional requirements in depth. Chapter 5 is about System Design, it provides more detail and classification of System design. It explains the Overall System Architecture of interface and more about it.

Chapter 6 provides the implementation of project in various chunks or modules, categorizing differently. It explains about the implementation of various ML Algorithms in our system to acquire highest accuracy. It describes the development of system using various languages, libraries, frameworks, databases and ML Algorithms. Chapter 7 is about Testing the system using various test cases for fetching the accurate predicted results. It also provides various set of actions executed to verify a particular feature or functionality of our system. Chapter 8 consists of various Screenshots of system including its various functionalities, basic working, operations, and many more features. Chapter 9 concludes about the project and gives more essence on the future scope of our system in various ways. This chapter summarizes the whole project with better prospects.

Chapter 2

Literature Survey

2.1 Designing Disease Prediction Model Using Machine Learning Approach

Now-a-days, people face various diseases due to the environmental condition and their living habits. So, the prediction of disease at earlier stage becomes important task. But the accurate prediction on the basis of symptoms becomes too difficult for doctor.

The correct prediction of disease is the most challenging task. To overcome this problem data mining plays an important role to predict the disease. Medical science has large amount of data growth per year. Due to increase amount of data growth in medical and healthcare field the accurate analysis on medical data which has been benefits from early patient care. With the help of disease data, data mining finds hidden pattern information in the huge amount of medical data.

In this paper, they have proposed a general disease prediction based on symptoms of the patient. For the disease prediction, they used K-Nearest Neighbour (KNN) and Convolutional neural network (CNN) machine learning algorithm for accurate prediction of disease. For disease prediction required disease symptoms dataset. Ensemble classification technique is used in this model before prediction. In this general disease prediction, the living habits of person and check-up information consider for the accurate prediction.

Machine Learning Algorithms Used: KNN (K-Nearest Neighbour), CNN (Convolutional neural network)

Accuracy: CNN – 84.50%, KNN – 81.12%

2.1.1 Advantages of Paper

a. Precise and accurate.

b. Algorithms are working efficiently with larger datasets.

2.1.2 Disadvantages of Paper

a. Risks associated with disease are not displayed.

b. Higher Memory Requirement.

c. Less Efficiency and accuracy.

d. Time required for classification is higher about 12900ms.

2.1.3 How to overcome the problems mentioned in Paper

a. Prediction will be done at faster rate.

b. Memory Requirement will be reduced.

c. Risks associated with the diseases will be displayed.

2.2 Symptom Based Health Prediction using Data Mining and ML

The general day to day health of a person is vital for the efficient functioning of the human body. In this Paper they are taking certain prominent symptoms and their diseases to build a Machine learning model to predict common diseases based on real symptoms is the objective of their research with the dataset of the most commonly exhibited diseases, they built a relation for predicting the possible disease based on the input of symptoms.

The proposed model utilizes the capability of different Machine learning algorithms combined with text processing to achieve accurate prediction. Text processing has been implemented using Tokenization and, is combined with various algorithms to test the similarities and the outputs. In health industry, it provides several benefits such as pre-emptive detection of diseases, faster diagnosis, medical history for review of patients etc.

Training Models, ML Algorithms Used: Scikit Learn Library and Pandas data frames of Python to process data. Decision Trees, Random Forest and Naïve Bayes Algorithms.

Accuracy: Decision tree Algorithm - 98.18%, Random Forest Algorithm - 98.05%, Naïve Bayes Algorithm – 98.55%.

2.2.1 Advantages of Paper

a. Less Pre processing.

b. Strong independent Assumptions.

c. Faster due to usage of conditional probability.

2.2.2 Disadvantages of Paper

a. Less data set and number of parameters for classification.

b. Decision trees are built on the entire dataset.

c. Time required for training dataset is higher.

d. It does not show all components of the Prediction system, but it focuses on the feature of classification.

2.2.3 How to overcome the problems mentioned in Paper

a. More number of parameters will taken for the accurate prediction.

b. Time required for training dataset will be reduced by doing preprocessing of the dataset for faster results.

c. Selection of best ML Algorithm will taken so as to improve the efficiency and accuracy of our project in terms of all requirements.

d. Features of the predictions system will be showcased in a detailed manner for better analysis.

2.3 A Novel Approach to Predict Diabetes by Using Naive Bayes Classifier

Diabetes can be mentioned as one of the most fatal and constant sicknesses that may cause a rise in the glucose levels. In this IEEE Paper, the main target of this model is to analyze the database of diabetic patients and to predict the diabetic disease in the early stage.

In this proposed system, Naïve Bayes Classification is used for predicting the diabetes. Information mining is a procedure of extricating data from a dataset and change it into justifiable structure for additional utilization. The information arrangement is diabetic patients informational collection is created by gathering information from clinic storehouse comprises of 1865 occurrences with various qualities. The outcomes show that the proposed novel strategy can foresee the diabetes with higher exactness levels than the customary/existing techniques.

In this proposed system using Naïve Bayes Classifier, Output will be theWeb Interface showing the Outcome of having diabetes or not by taking the input values like Insulin level, age and so on. This increases the accuracy of the system.

Machine Learning Algorithms Used: Naïve Bayes

Accuracy: 96%

2.3.1 Advantages of Paper

a. Simple and easy to predict.

b. Performs well due to multiple class expectation.

c. Accuracy is highest.

2.3.2 Disadvantages of Paper

a. Implicitly assumes that all the attributes are mutually independent.

b. The assumption of independent predictor features.

2.3.3 How to overcome the problems mentioned in Paper

a. Prediction will be done faster and with less memory requirement and classification will be done more precisely with appropriate symptoms.

b. Risk associated with the disease will be showcased after prediction.

c. Selection of best ML Algorithms will be taken for development.

d. Implementation of Live Consultation with Doctors for better treatment and diagnosing of diseases.

e. Reduce the limitations of less accurate Algorithms by implementing them in a sequenced manner.

f. Specialized Doctor recommendation and nearest health centers from your location available for quick treatment.

2.4 Technical Review

We have used Django Framework to build the functioning of our webapp and Machine Learning algorithms like Naive Bayes, Linear Regression, K-Nearest Neighbour, Kernel SVM, Decision Trees and Random Forest Algorithms for accurate prediction and analysis of diseases.

2.4.1 Advantages of Technology

a. Django is implemented in Python

b. Better CDN connectivity and Content Management

c. Batteries Included Framework

d. Fast Processing

e. Naive Bayes Algorithm is simple and easy to implement. It doesn't require as much training data. It handles both continuous and discrete data

2.4.2 Reasons to use this Technology

a. Django Offers Rapid-development and provides Security

b. Django has a Model-View-Template (MVT) architecture. The MVT (Model View Template) is a software design pattern.

c. It is a collection of three essential components Model, View, and Template. These three layers are responsible for different things and can be used independently.

d. Naive Bayes Algorithm is highly scalable with the number of predictors and data points.

e. It is fast and can be used to make real-time predictions

f. It is not sensitive to irrelevant features

g. Naive Bayes Algorithm performs better than other models with less training data if the assumption of independence of features holds.

h. If you have categorical input variables, the Naive Bayes algorithm performs exceptionally well in comparison to numerical variables.

Chapter 3

Project Planning

3.1 Members and Capabilities

Table 3.1: Table of Capabilities

SR. No	Name of Member	Capabilities
1	Shaikh Faiyyaz	Frontend, UI Design, Database
2	Shaikh Soaib	UI/UX, Integration, Database
3	Shaikh Ubed	Machine Learning, Backend, Django
4	Nivekar Bilal	Documentation, Backend

3.2 Roles and Responsibilities

Table 3.2: Table of Responsibilities

SR. No	Name of Member	Role	Responsibilities
1	Shaikh Faiyyaz	Team Leader, Frontend	UI Design, Frontend, Project Workflow
2	Shaikh Soaib	UI/UX	UI Django Integration, Database
3	Shaikh Ubed	Backend	Machine Learning, Backend, Django
4	Nivekar Bilal	Documentation	Documentation, Backend

3.3 Assumptions and Constraints

- Users of this Webapp can use this system.
- Users of this Webapp should have active internet.
- Users should provide appropriate symptoms for prediction.
- There will be a backup of the data after every cycle is completed.
- There will be a admin present 24/7 at the server.
- Highest accuracy is fetched with more than 2 or 3 symptoms.
- We have to work with the available resources.
- We need to manage the entire project within the team of developers.

3.4 Project Management Approach

We will be working in an Agile project management approach. As Agile is a project management methodology that uses short development cycles called "sprints" to focus on continuous improvement in the development of a product or service. We will be using the following principles while working:

- Customer satisfaction is always the highest priority and will be achieved through rapid and continuous delivery.
- Changing environments are embraced at any stage of the process to provide the customer with a competitive advantage.
- We will deliver a product or service with higher frequency.
- Our Stakeholders and developers collaborate closely on a daily basis.
- All stakeholders and team members remain motivated for optimal project outcomes, while teams are provided with all the necessary tools and support, and are trusted to accomplish project goals.
- Face-to-face meetings are deemed the most efficient and effective format for project success.
- A final working product is the ultimate measure of success.
- Sustainable development is accomplished through agile processes whereby development teams and stakeholders are able to maintain a constant and ongoing pace.
- Agility is enhanced through a continuous focus on technical excellence and proper design.
- Simplicity is an essential element.
- Self-organizing teams are most likely to develop the best architectures and designs and to meet requirements. Regular intervals are used by teams to improve efficiency through fine-tuning behaviors.

3.5 Ground Rules for the Project

- Be on time for all team meetings.
- Team leader must create and disseminate agendas for each team meeting.
- Team leader must create and disseminate minutes after each team meeting.
- Attend full duration of all team meetings unless a case of emergency.

- Avoid informal/social talk during team meetings.
- Avoid apathetic/passive decision making (e.g., “whatever you all think is right”).
- Inform team leader if unable to complete work on time.
- Set deadlines for each deliverable in advance of due date to allow for collaborative revisions.
- Rotate responsibilities so each person gets experience with several aspects regardless of quality or qualifications.
- Make criticisms constructive with suggestions for improvement and non-judgmental language.
- Confront issues directly and promptly.
- Promptly relay all interpersonal concerns/conflicts to team leader.
- Keep a positive attitude toward the team, individual members, projects and course.
- Take initiative by offering ideas and volunteering for tasks.
- Play an equal role in the team by contributing equally to every task.
- Be honest with any team member who is not pulling her/his weight.
- Help one another with difficult or time consuming deliverables.
- Ask for help from the team or other resources if “stuck” or falling behind.
- Treat each other with respect.
- Accept responsibility and accountability along with the authority given.

3.6 Project Budget

- It is cost efficient Project.
- Easily deployable accross compatible devices.

3.7 Project Timeline

	Name	Duration	Start
1	Project initialization	0 days	26/01/21, 8:00 AM
2	**Requirement Analysis**	21 days	**26/01/21, 8:00 AM**
3	Market reaseach	7 days	26/01/21, 8:00 AM
4	Brainstroming session	4 days	04/02/21, 8:00 AM
5	Identify Problem	5 days	10/02/21, 8:00 AM
6	Feature list	2 days	17/02/21, 8:00 AM
7	Cost Analysis	3 days	19/02/21, 8:00 AM
8	**Design**	22 days	**24/02/21, 8:00 AM**
9	Architecture Defination	5 days	24/02/21, 8:00 AM
10	Prepare SRS	4 days	03/03/21, 8:00 AM
11	Prepare UML diagram	3 days	09/03/21, 8:00 AM
12	Task allotment	2 days	12/03/21, 8:00 AM
13	Devlope demo model	4 days	16/03/21, 8:00 AM
14	unit testing and fixing	4 days	22/03/21, 8:00 AM
15	**Integration of unit Model**	20 days	**26/03/21, 8:00 AM**
16	Integration of model	5 days	26/03/21, 8:00 AM
17	system testing	5 days	02/04/21, 8:00 AM
18	acceptance testing	5 days	09/04/21, 8:00 AM
19	Documentation	5 days	16/04/21, 8:00 AM
20	**Deployment**	5 days	**23/04/21, 8:00 AM**
21	Deploye online	3 days	23/04/21, 8:00 AM
22	testing performance	2 days	28/04/21, 8:00 AM
23	Finish	0 days	29/04/21, 5:00 PM

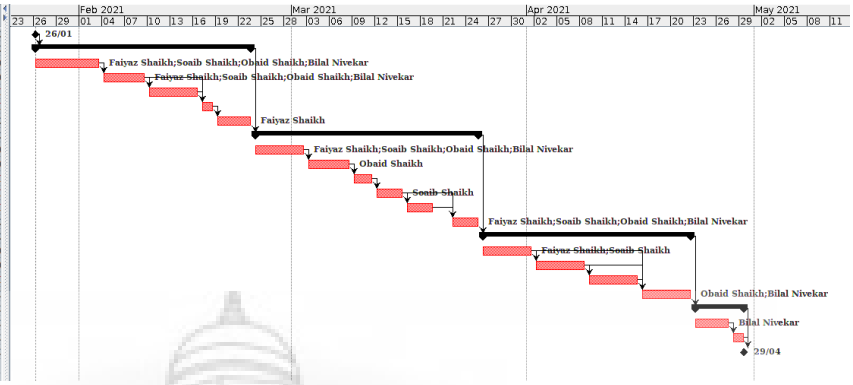


Figure 3.1: Project Timeline

Chapter 4

Software Requirements Specification

4.1 Overall Description

4.1.1 Product Perspective

Our disease prediction system web application is based that tries to ease the development process by simplifying repetitive tasks used in most of today's web applications, including routing, authentication, caching and sessions. The vertically integrated web development environment is meant to offer an improved and smooth workflow of data.

4.1.2 Product Features

The following are the main features that are included in our Smart Health Prediction System:

1. Cross platform support: Offers operating support for most of the known and commercial operating systems.

2. User account: The system allows the user to create their personal accounts for patient and doctors in the system and provide features of updating and viewing profiles.

3. Number of users being supported by the system: Though the number is precisely not mentioned, the system is able to support a large number of online users at a time.

4. Analysis summary: Provides patient users with a dashboard to review the results of disease prediction based on their entered input symptoms.

5. Communication network: There is a platform between patients and doctors to contact with each other for discussing about the health related issues.

4.1.3 User Classes and Characteristics

The two class of users called Patient and Doctor have them their different functionalities of the application, including:

1. Patient Disease Prediction : Patient will be able to predict disease prediction in probability based entered symptoms without contacting doctors.
2. Consult Page : Patient can consults with provided list of doctors after predicting disease result for further procedure.
3. Doctors access : Doctors can view the details of patient about the disease and also the symptoms that mentioned while predicting through our portals so that doctors can study more while contacting with the patient.

4.1.4 Operating Environment

a. The application is developed in the Django framework to enable the creation of a web-based application, which can be accessed from any web browser.
b. The application will connect to the backend to store and retrieve data from a PostgreSQL database.

4.1.5 Design and Implementation Constraints

a. The application should run in a latest JavaScript enabled web browser.
b. The application might take a few seconds to load the analysis data generated by the api.
c. This system is provisioned to be built on the Django framework which is highly flexible. Decision regarding which database to use is taken considering the fact that data being exchanged or stored is large, and the appropriate data management system will yield efficient performance.

4.2 System Features

4.2.1 Accurate Prediction with faster results

Smart Health Disease Prediction System will predict accurate diseases based on the user's symptoms with full fledged efficiency and swiftness. SHDPS will also showcase the consequences of that disease and the impact of that disease on other organs of the body. SHDPS will work on the fastest Machine Learning Algorithm with the data set and it will predict the diseases with its powerful prediction capabilities.

4.2.2 Live Consultation

SHDPS will not only predict diseases but will also provide Live Consultation with the specialist for that disease in this amid pandemic. People are facing difficulties in consulting with the doctors as they are unable to have physical communication or contact with their doctors so this would be a beneficial interface for patients to consult with their doctors with these unique feature.

4.2.3 Patient's Health Care

SHDPS will provide regular followup appointments, reminders for routine checkups and also maintain the data records of patients previous health history. Some people have the habit of forgetting regular followups with their doctors due to which the health is affected. So to minimize this concern SHDPS will continuously notify its users for consultations.

4.3 External Interface Requirements

4.3.1 User Interfaces

a. Front-end software: HTML, CSS, JavaScript, Bootstrap, jQuery

b. Back-end software: Django Framework, Python

c. Database software: PostgreSQL

4.3.2 Hardware Interfaces

a. Windows or Mac operating systems

b. Android or IOS mobile phones.

c. Devices should be enabled with the Internet.

4.3.3 Software Interfaces

a. The user's browser should be HTML5 and JavaScript compatible for all the functionalities to work.

4.3.4 Communications Interfaces

a. The application will generate feedback form to receive the responses from the standard users.

4.4 Nonfunctional Requirements

4.4.1 Performance Requirements

a. The dashboard page is displayed to the user immediately after the login. It takes 1-2 seconds to display the predicted data based on different symptoms and training of date to predict disease.

b. The disease prediction part of the application is flexible and smooth so that it does not consume any time of the user.

c. The database should be normalized to prevent redundant data and improve performance.

4.4.2 Safety Requirements

a. Databases should use sharding to be redundant to prevent loss of data.

b. Information transmission should be securely transmitted to server without any changes in information

c. Backups of the databases should be done monthly and be kept for a long time.

4.4.3 Security Requirements

a. Any keys used for the REST api should be stored securely. Only the REST api should be able to connect to the databases.

b. Databases should be behind a firewall.

c. For users' accounts, a proper login mechanism should be used to avoid hacking.

Chapter 5

System Design

5.1 System Requirements Definition

System requirement definitions specify what the system should do, its functionality and its essential and desirable system properties. The techniques applied to elicit and collect information in order to create system specifications and requirement definitions involve consultations, interviews, requirements workshop with customers and end users. The objective of the requirements definition phase is to derive the two types of requirement:

5.1.1 Functional requirements

They define the basic functions that the system must provide and focus on the needs and goals of the end users.

- Authentication of user whenever he/she logs into the system.
- System shutdown in case of a cyber attack.
- Automatic Backup of system.

Use-case Diagram

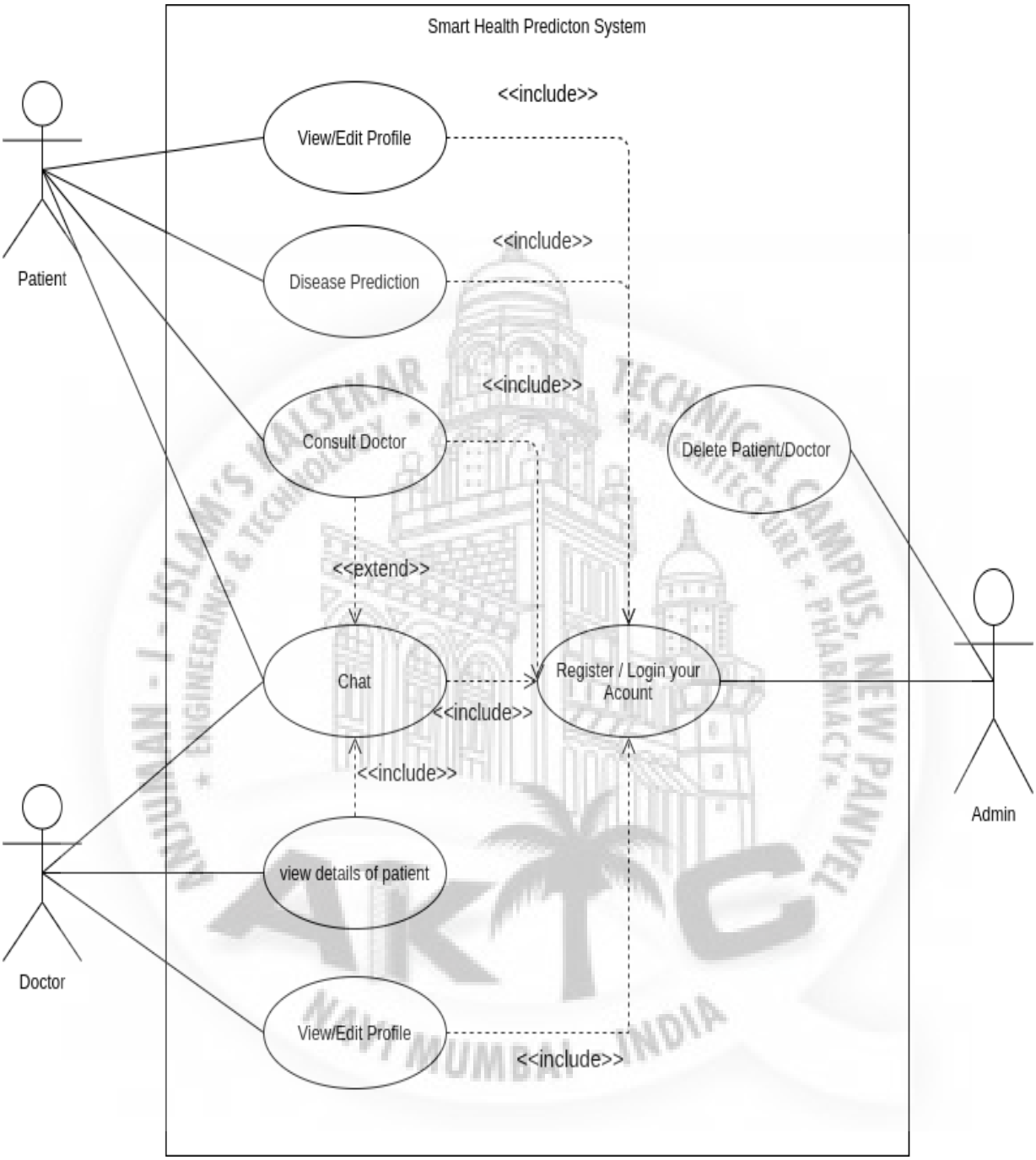


Figure 5.1: Use case Diagram For SHDPS

Data-flow Diagram

Level 0

DFD Level 0 for our systems focuses on the main output where patient gets the result of his/her disease analysis.

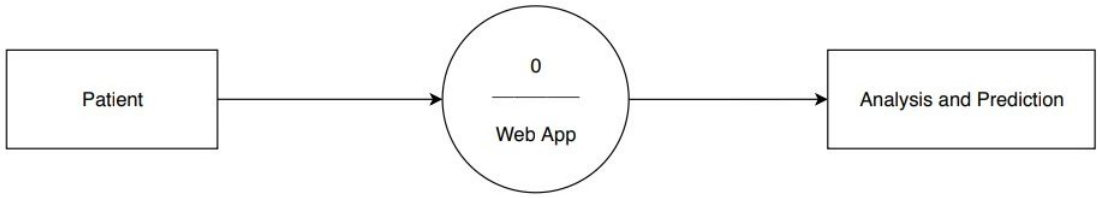


Figure 5.2: DFD Level 0 for SHPS

Level 1

DFD level 1 depicts that Patients symptoms will be saved in database as well as sent for classification and prediction accompanied by trained Machine Learning Model.

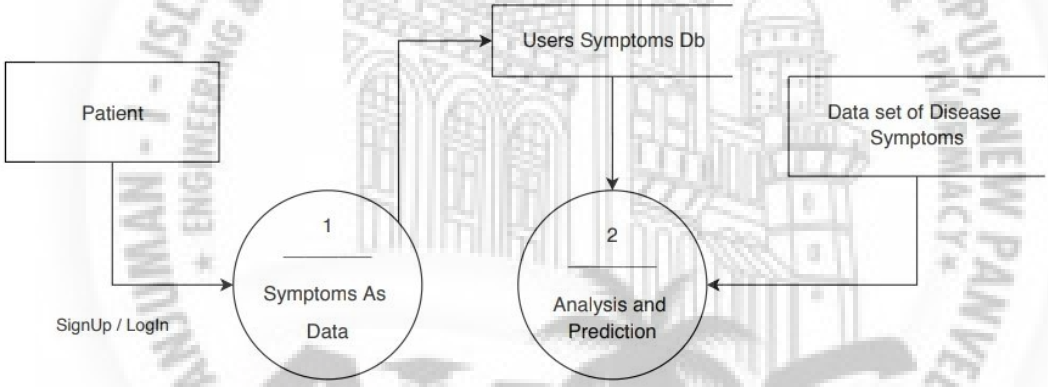


Figure 5.3: DFD Level 1 for SHPS

Level 2

DFD Level 2 the data will be analyzed and a complete prediction will be made. The predicted disease will be accompanied by some of it's information as well as a doctor will be referred according to the classified disease. Patient can see the doctor's ratings and can decide whether to go for live consultation or not.

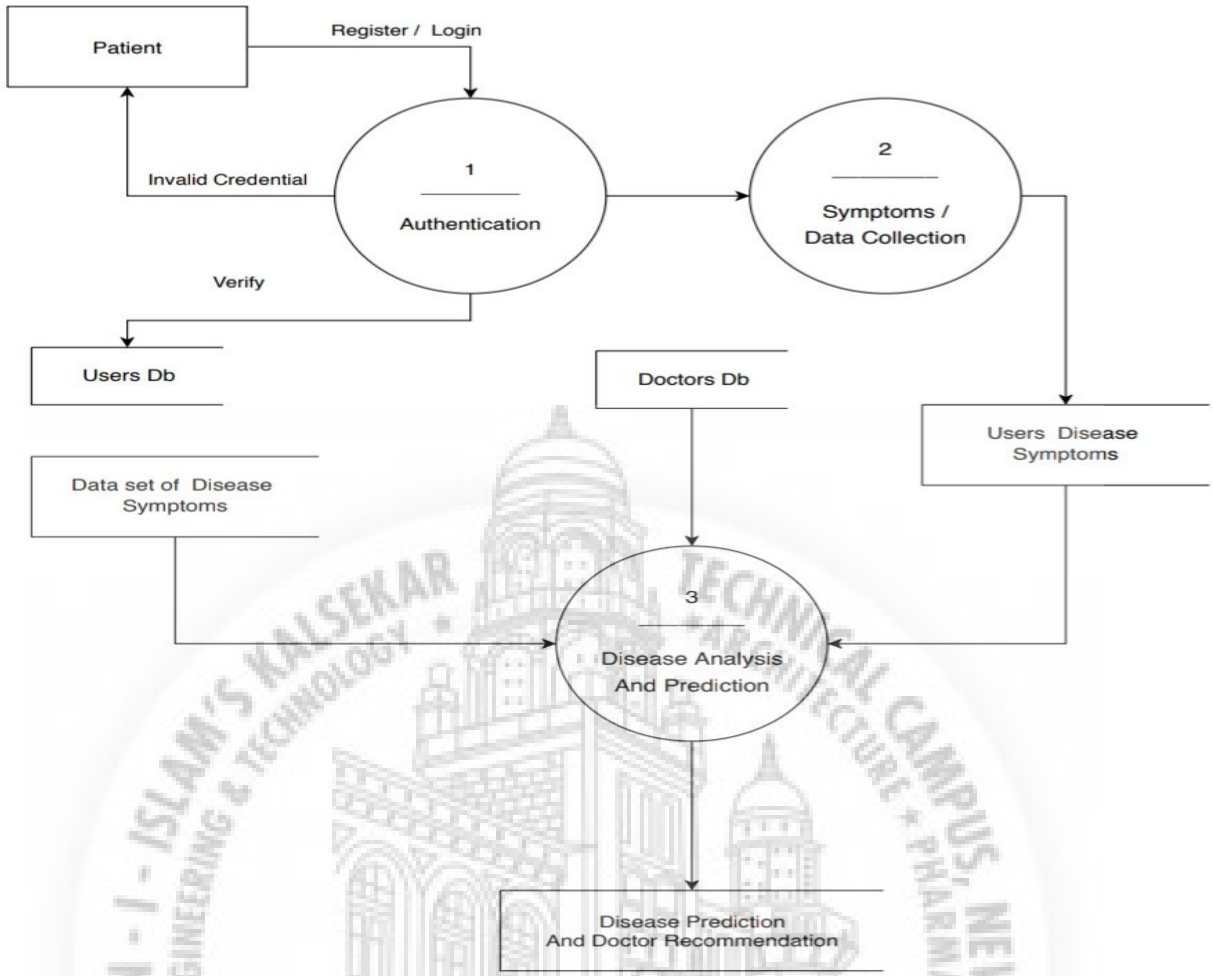


Figure 5.4: DFD Level 2 for SHPS

Activity Diagram

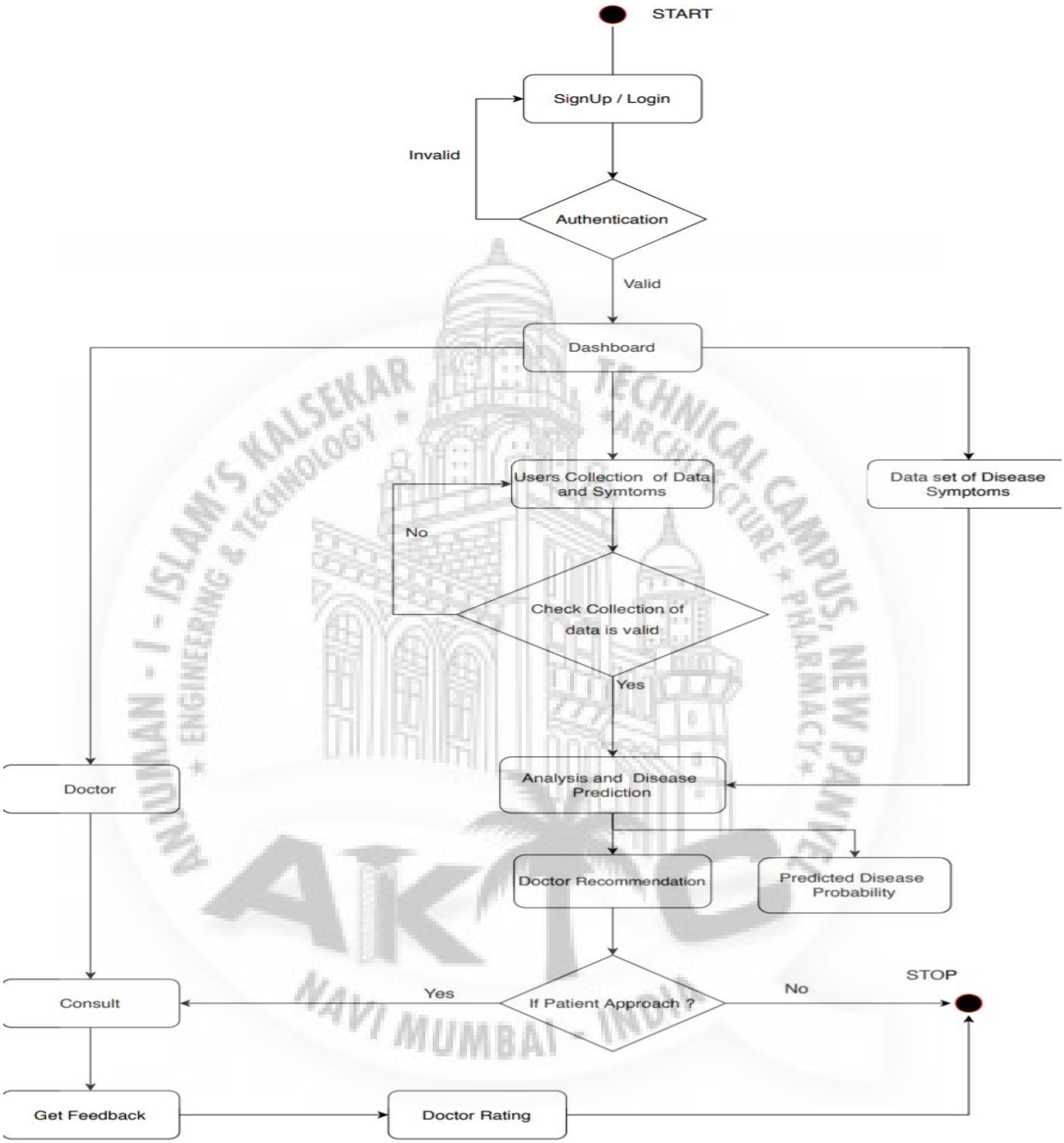


Figure 5.5: Activity Diagram for SHPS

5.1.2 System requirements (non-functional requirements)

a. Emails should be sent with a latency of no greater than 12 hours from such an activity.

b. The processing of each request should be done within 10 seconds.

c. The site should load in 3 seconds when the number of simultaneous users are greater than 10000.

5.2 System Architecture Design

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

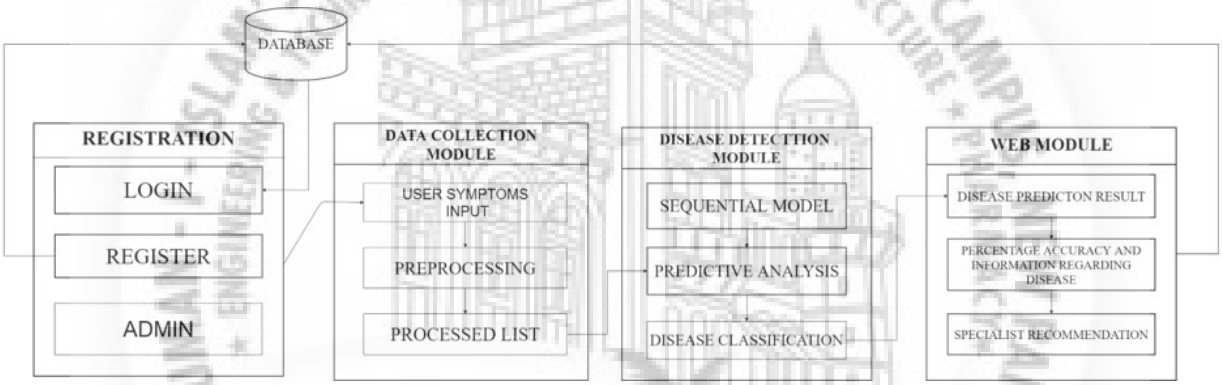


Figure 5.6: System Architecture Design Diagram for SHPS

5.3 Sub-system Development

This system consist of four modules Registration , Data Collection Module , Disease Prediction Module and Web Module. In Registration module authentication must be done and in second module Data collection Module user symptoms input will be taken and based on that input symptoms Disease Prediction Module will analyse the disease probability and that will display through Web Module.

5.3.1 Registration Module

Registration of patients and Doctors will be there through their personal portals and after that there will be a personal page for their purpose of patient and doctor.

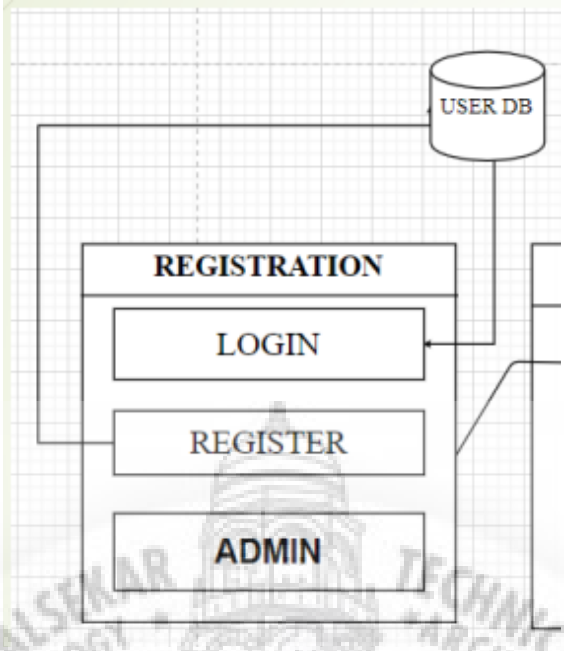


Figure 5.7: Registration Module

5.3.2 Data Collection Module

In Data Collection Module there will be collection of symptoms of patient selected by the patient from our given symptoms list. This module is mainly concern with patient side.

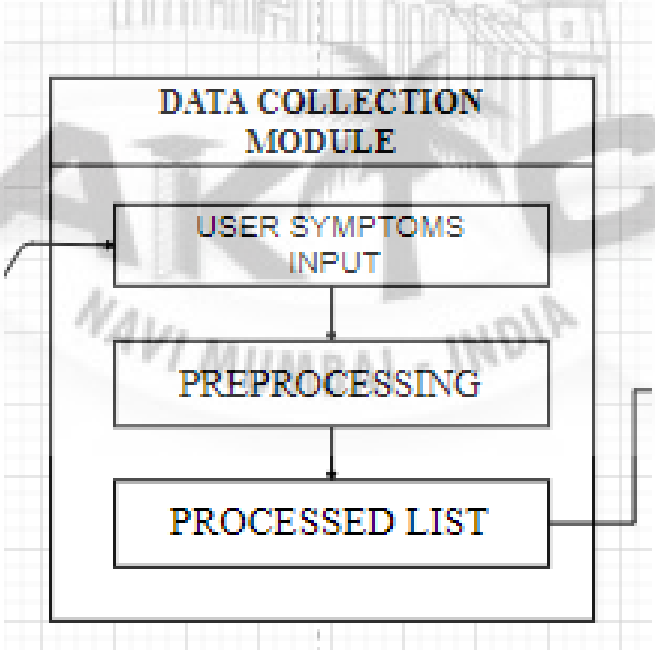


Figure 5.8: Data Collection Module

5.3.3 Disease Prediction Module

In Data prediction Module there a prediction of disease will be done with the help of algorithms provided in back-end. The better probability of disease prediction result will be selected for further procedure.

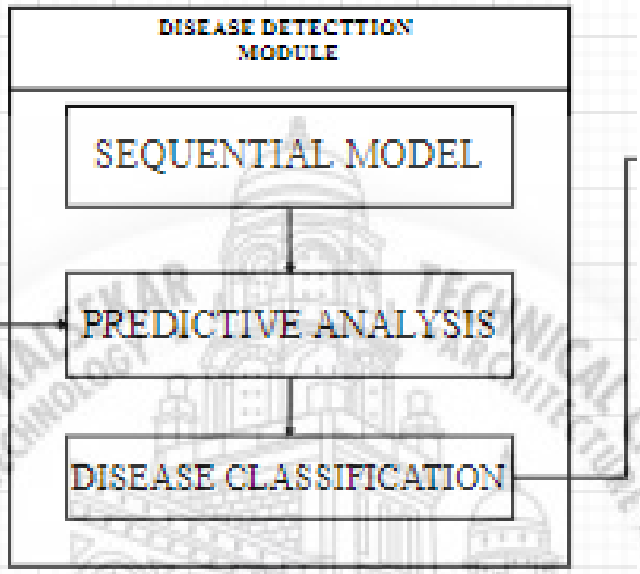


Figure 5.9: Disease Prediction Module

5.3.4 Web Module

In web module there is contact between patient and doctor will be perform based on their disease predicted through previous module and they will contact with each other due this module in which we design a chat body system for them.

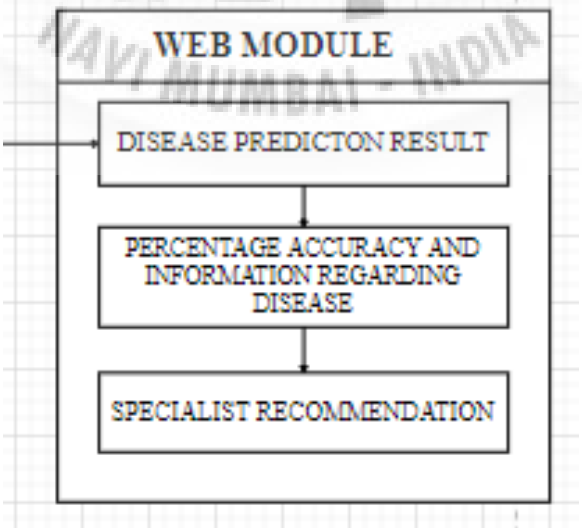


Figure 5.10: Web Module

5.4 Systems Integration

System integration (SI) is an engineering process or phase concerned with joining different subsystems or components as one large system. It ensures that each integrated subsystem functions as required. SI is also used to add value to a system through new functionalities provided by connecting functions of different systems.

5.4.1 Component Diagram

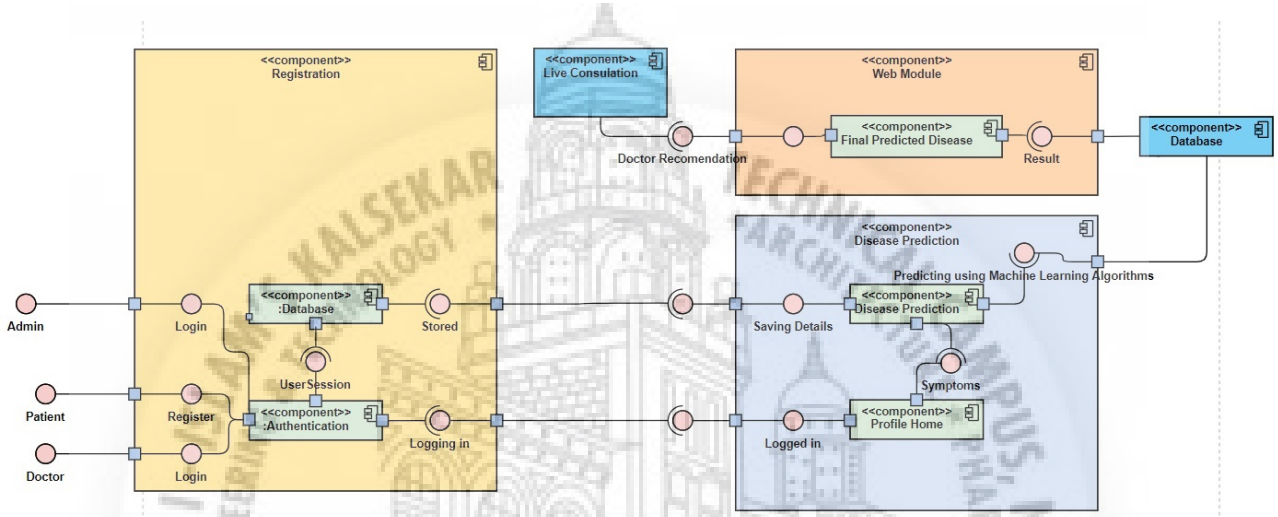


Figure 5.11: Component Diagram for SHDPS

5.4.2 Class Diagram

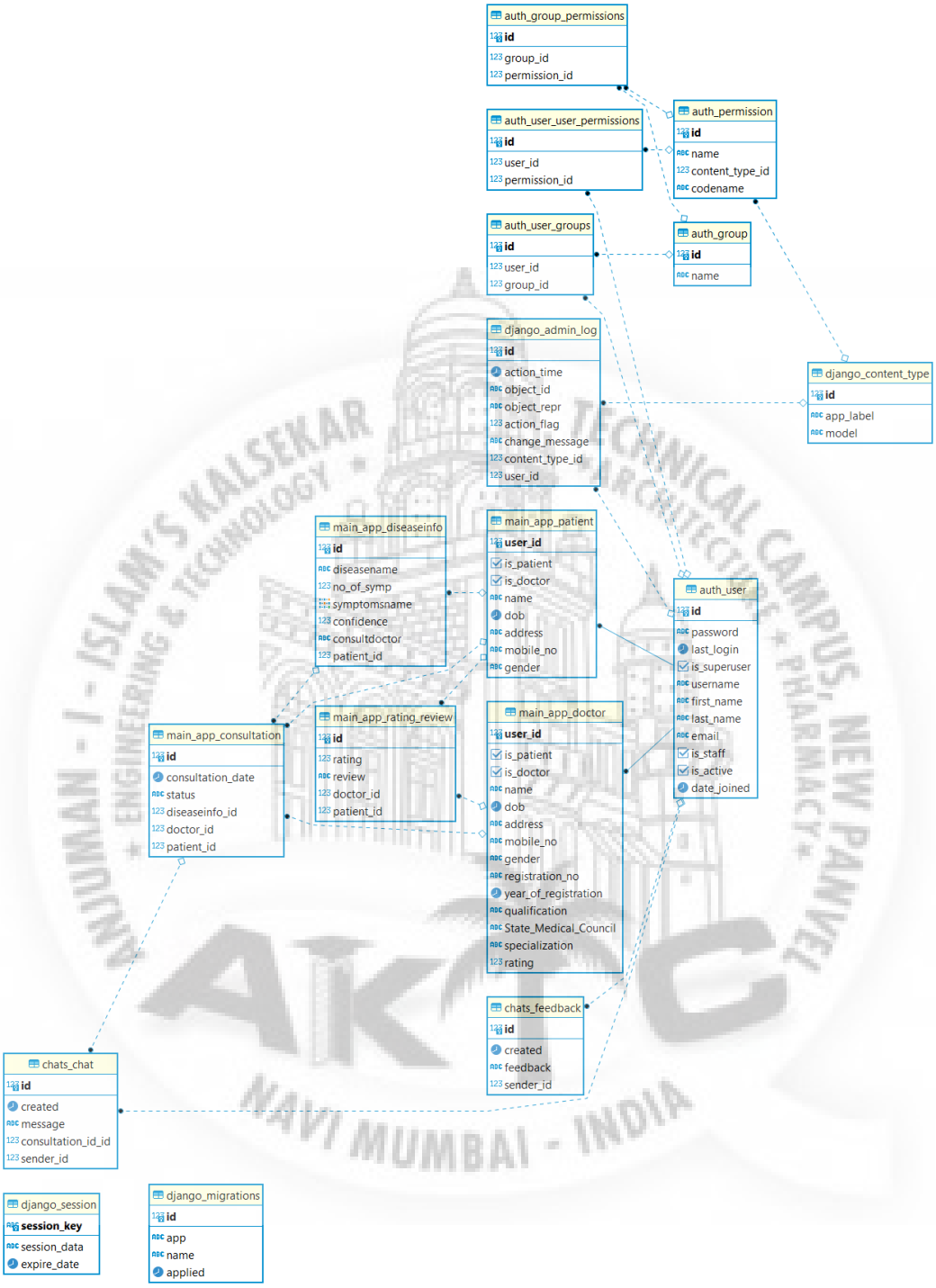


Figure 5.12: Class Diagram for Smart Health Disease Prediction System

5.4.3 Sequence Diagram

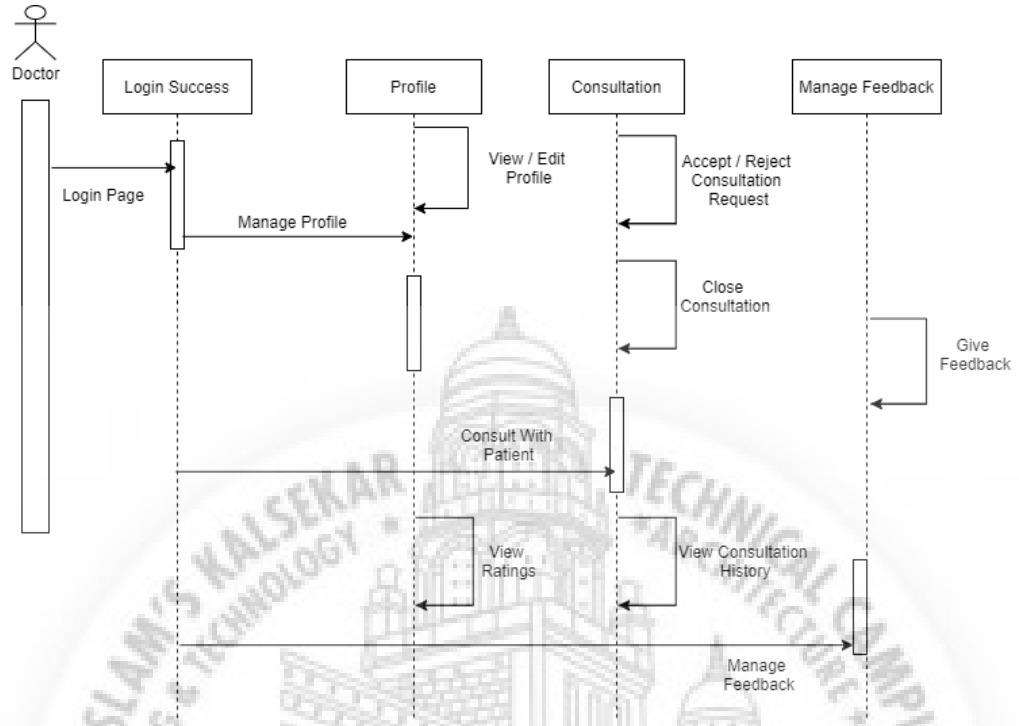


Figure 5.13: Sequence Diagram for Doctor User

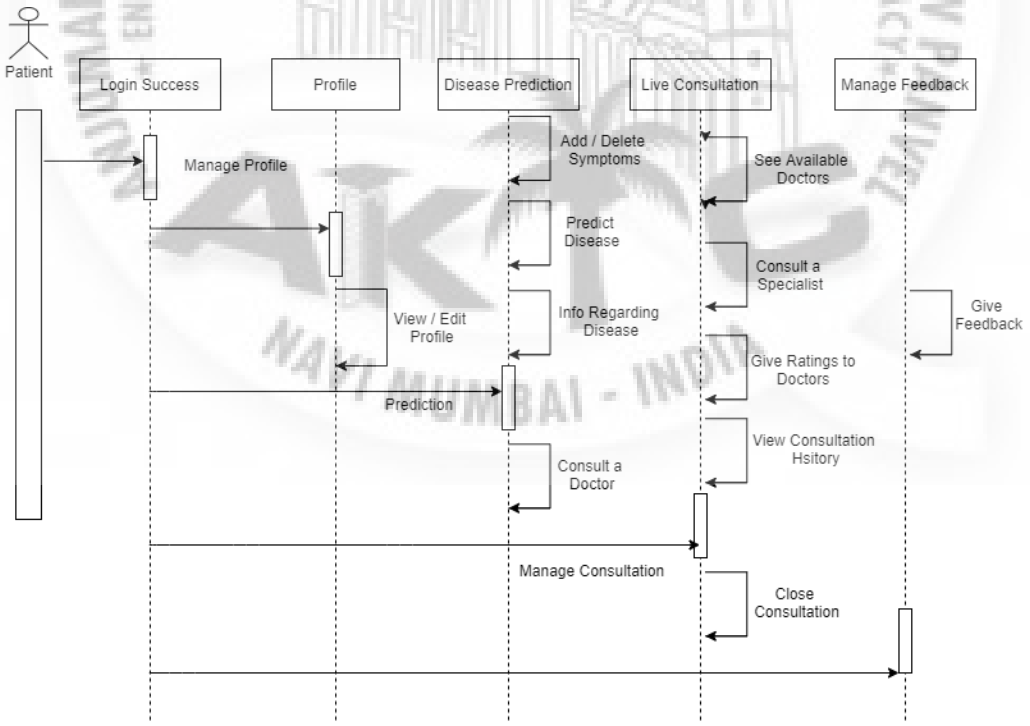


Figure 5.14: Sequence Diagram for Patient User

5.4.4 Deployment Diagram

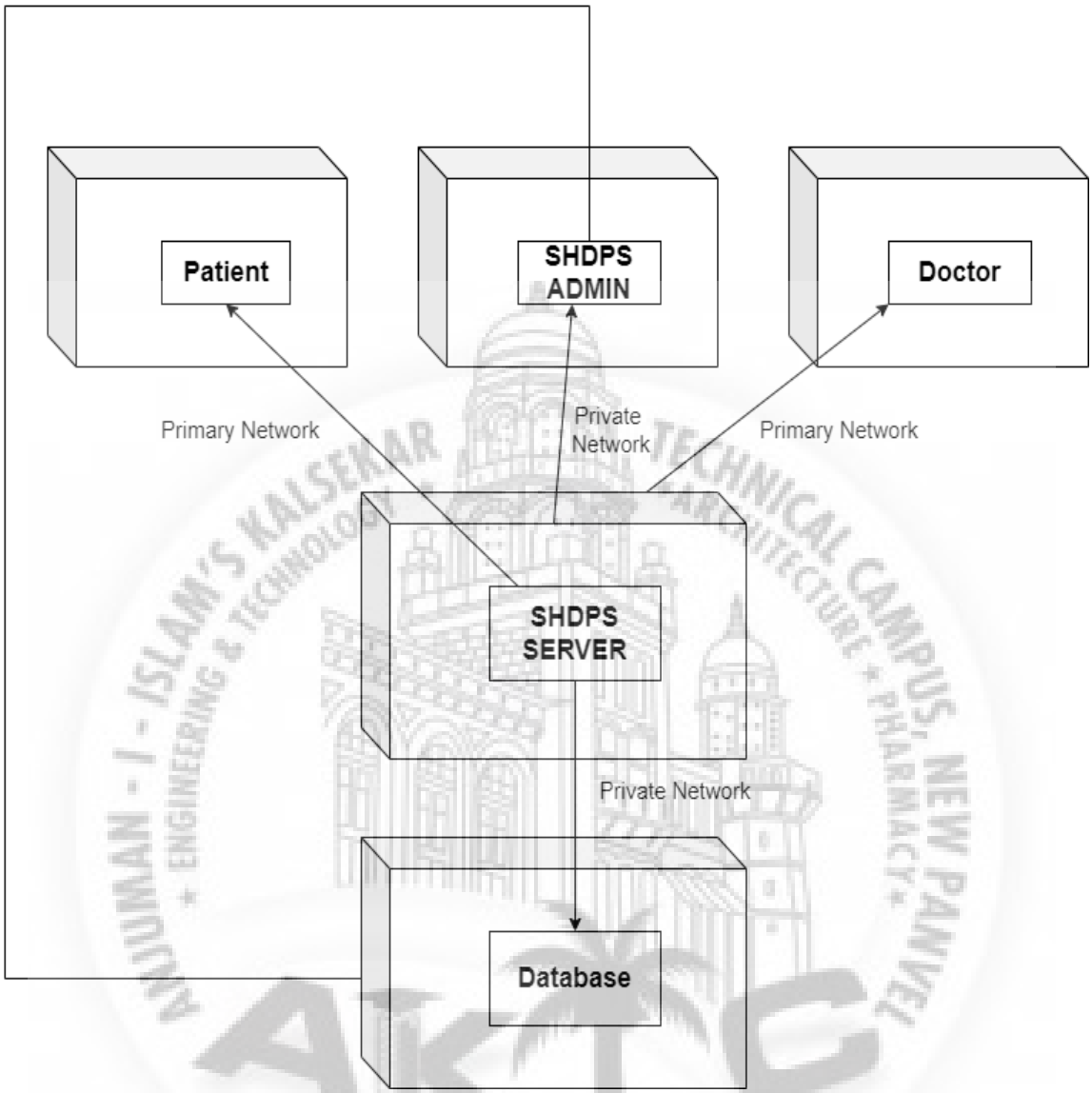


Figure 5.15: Deployment Diagram for SHDPS

Chapter 6

Implementation

6.1 Disease Prediction

Disease Prediction the core and sole feature of our project named SMART HEALTH DISEASE PREDICTION SYSTEM. The main goal is to provide powerful analysis features with utmost accuracy in predicting diseases because health is an essential factor with which no one is willing to take any risk. As we know that Health is the most important asset for a human being. So to maintain a healthy lifestyle by anticipating or predicting diseases and impact of them on other organs at early stage would be more beneficial and constructive in analysis for the treatment of that disease. To overcome this Real life problem, we developed a Smart Health Disease Predictor with the help of Machine Learning.

Smart Health Disease Prediction System will predict accurate diseases based on the user's symptoms with full fledged efficiency and swiftness. SHDPS will also showcase the consequences of that disease and the impact of that disease on other organs of the body. SHDPS will work on the fastest Machine Learning Algorithm with the data set and it will predict the diseases with its powerful prediction capabilities.

The precise goal of our project is accuracy. Since our system deals with problems related to health and it could be sometimes serious if it lacks accuracy hence we will try to achieve as much accuracy as we can in comparison with the systems previously proposed. Our system will not only bound till prediction we will try to make it more User specific so that both patients as well medical professionals can seek profit through our system.

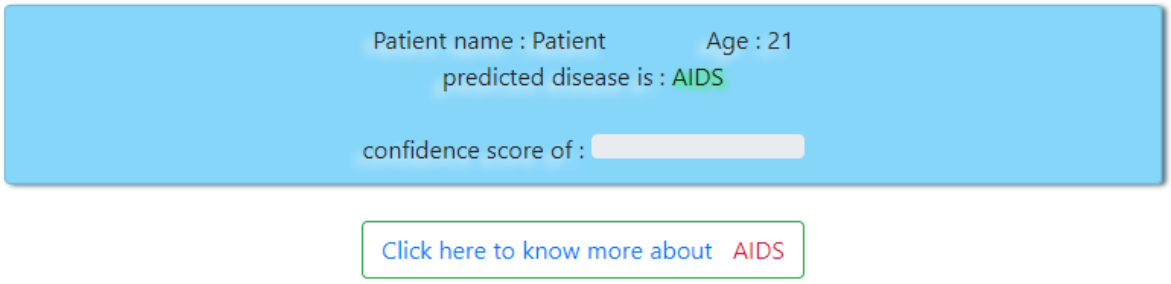


Figure 6.1: Symptoms Addition Module

6.2 Machine Learning Module

In our system we have used immense power of Machine learning algorithms to predict diseases for the end user on the basis of symptoms which they provide with the confidence score with which it is predicting. We have implemented six Machine Learning algorithms and trained them in so that we can use them in our future endeavours. The Algorithms implemented are as follows ::-

1. Logistic Regression
2. K - Nearest Neighbors
3. Kernel Support Vector Machine
4. Naive Bayes
5. Decision Tree
6. Random Forest

6.2.1 Logistic Regression

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression predicts the output of a categorical dependent variable

```
#!/usr/bin/env python
# coding: utf-8

#Importing Libraries
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression

#Read Dataset
def read_dataset(dataset):
    X_train = dataset.iloc[ : , 1 : ].values
    Y = dataset.iloc[ : , 0].values
    labelencoder = LabelEncoder()
    Y_train = labelencoder.fit_transform(Y)
    return X_train , Y_train , Y

#Training Prediction Model
def train_Logistic_regression_model(X_train , Y_train):
    classifier = LogisticRegression(random_state = 0)
    classifier.fit(X_train , Y_train)
    return classifier

#Get Custome Test Cases
def get_custom_test_cases(symptoms):
    input_symptoms=[]
    for x in range(0,len(symptoms)):
```

```
        input_symptoms.append(0)
    faiyaz_symptoms = ['joint_pain', 'vomiting', 'fatigue', 'yellowish_skin'
        , 'dark_urine', 'loss_of_appetite', 'abdominal_pain', '
        yellowing_of_eyes']
    for index, symptom in enumerate(symptoms):
        for faiyaz_symptom in faiyaz_symptoms:
            if symptom == faiyaz_symptom:
                input_symptoms[index] = 1
    return input_symptoms

#Get Names of Custom Inputs
def get_custom_symptoms_names(input_symptoms, symptoms):
    print('Passed Symptoms ::-')
    for i in range(0, len(input_symptoms)):
        if input_symptoms[i] == 1:
            print(symptoms[i])

#Predicting and Printing Disease
def predict_disease(model, input_symptoms, Y_train, Y):
    my_symptoms_pred = model.predict([input_symptoms])
    my_symptoms_pred_acc_score = model.predict_proba([input_symptoms])

    for num, disease in zip(Y_train, Y):
        if int(num) == int(my_symptoms_pred[0]):
            print('\nPredicted Disease :: ', disease)
            break
    else:
        print('No Disease')
    print('Accuracy Score ::-', my_symptoms_pred_acc_score.max()*100)


#Main Function
if __name__ == '__main__':
    headers = [*pd.read_csv('SHDPS_Training_1.csv', nrows=1)]
    dataset = pd.read_csv('SHDPS_Training_1.csv', usecols=[c for c in headers if
        c != 'Unnamed: 0'])
    symptoms = dataset.columns[1 : ]
    X_train, Y_train, Y = read_dataset(dataset)
    model = train_Logistic_regression_model(X_train, Y_train)
    input_symptoms = get_custom_test_cases(symptoms)
    get_custom_symptoms_names(input_symptoms, symptoms)
    predict_disease(model, input_symptoms, Y_train, Y)
```

6.2.2 K - Nearest Neighbour

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems. K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.

```
#!/usr/bin/env python
# coding: utf-8

#Importing Libraries
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.neighbors import KNeighborsClassifier

#Read Dataset
def read_dataset(dataset):
    X_train = dataset.iloc[ : , 1 : ].values
    Y = dataset.iloc[ : , 0].values
    labelencoder = LabelEncoder()
    Y_train = labelencoder.fit_transform(Y)
    return X_train , Y_train , Y

#Training Prediction Model
def train_KNN_model(X_train , Y_train):
    classifier = KNeighborsClassifier(n_neighbors = 30 , metric = 'minkowski' ,
        p = 2)
    classifier.fit(X_train , Y_train)
    return classifier

#Get Custom Test Cases
def get_custom_test_cases(symptoms):
    input_symptoms=[]
    for x in range(0,len(symptoms)):
        input_symptoms.append(0)
    faiyaz_symptoms = ['joint_pain' , 'vomiting' , 'fatigue' , 'yellowish_skin'
        , 'dark_urine' , 'loss_of_appetite' , 'abdominal_pain' , '
        yellowing_of_eyes']
    for index , symptom in enumerate(symptoms):
        for faiyaz_symptom in faiyaz_symptoms:
            if symptom == faiyaz_symptom:
                input_symptoms[index] = 1
    return input_symptoms

#Get Names of Custom Inputs
def get_custom_symptoms_names(input_symptoms , symptoms):
    print('Passed Symptoms ::-')
    for i in range(0 , len(input_symptoms)):
        if input_symptoms[i] == 1:
            print(symptoms[i])

#Predicting and Printing Disease
def predict_disease(model , input_symptoms , Y_train , Y):
    my_symptoms_pred = model.predict([input_symptoms])
    my_symptoms_pred_acc_score = model.predict_proba([input_symptoms])

    for num , disease in zip(Y_train , Y):
        if int(num) == int(my_symptoms_pred[0]):
            print('\nPredicted Disease :: ' , disease)
            break
    else:
        print('No Disease')
    print('Accuracy Score ::-' , my_symptoms_pred_acc_score.max()*100)

#Main Function
if __name__ == '__main__':
    headers = [*pd.read_csv('SHDPS_Training_1.csv', nrows=1)]
    dataset = pd.read_csv('SHDPS_Training_1.csv', usecols=[c for c in headers if
```

```
        c != 'Unnamed: 0'])
    symptoms = dataset.columns[1 : ]
    X_train , Y_train , Y = read_dataset(dataset)
    model = train_KNN_model(X_train , Y_train)
    input_symptoms = get_custom_test_cases(symptoms)
    get_custom_symptoms_names(input_symptoms , symptoms)
    predict_disease(model , input_symptoms , Y_train , Y)
```

6.2.3 Kernel - Support Vector Machine

SVM algorithms use a set of mathematical functions that are defined as the kernel. The function of kernel is to take data as input and transform it into the required form. Different SVM algorithms use different types of kernel functions. These functions can be different types. For example linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid. Introduce Kernel functions for sequence data, graphs, text, images, as well as vectors. The most used type of kernel function is RBF. Because it has localized and finite response along the entire x-axis.

```
#!/usr/bin/env python
# coding: utf-8

#Importing Libraries
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.svm import SVC

#Read Dataset
def read_dataset(dataset):
    X_train = dataset.iloc[ : , 1 : ].values
    Y = dataset.iloc[ : , 0].values
    labelencoder = LabelEncoder()
    Y_train = labelencoder.fit_transform(Y)
    return X_train , Y_train , Y

#Training Prediction Model
def train_SVC_model(X_train , Y_train):
    classifier = SVC(kernel = 'rbf' , probability = True , random_state = 0)
    classifier.fit(X_train , Y_train)
    return classifier

#Get Custom Test Cases
def get_custom_test_cases(symptoms):
    input_symptoms=[]
    for x in range(0,len(symptoms)):
        input_symptoms.append(0)
    faiyaz_symptoms = ['joint_pain' , 'vomiting' , 'fatigue' , 'yellowish_skin'
        , 'dark_urine' , 'loss_of_appetite' , 'abdominal_pain' , '
        yellowing_of_eyes']
    for index , symptom in enumerate(symptoms):
        for faiyaz_symptom in faiyaz_symptoms:
            if symptom == faiyaz_symptom:
                input_symptoms[index] = 1
    return input_symptoms

```

```
#Get Names of Custom Inputs
def get_custom_symptoms_names(input_symptoms , symptoms):
    print('Passed Symptoms ::-')
    for i in range(0 , len(input_symptoms)):
        if input_symptoms[i] == 1:
            print(symptoms[i])

#Predicting and Printing Disease
def predict_disease(model , input_symptoms , Y_train , Y):
    my_symptoms_pred = model.predict([input_symptoms])
    my_symptoms_pred_acc_score = model.predict_proba([input_symptoms])

    for num , disease in zip(Y_train , Y):
        if int(num) == int(my_symptoms_pred[0]):
            print('\nPredicted Disease :: ' , disease)
            break
    else:
        print('No Disease')
    print('Accuracy Score ::-' , my_symptoms_pred_acc_score.max()*100)

#Main Function
if __name__ == '__main__':
    headers = [*pd.read_csv('SHDPS_Training_1.csv', nrows=1)]
    dataset = pd.read_csv('SHDPS_Training_1.csv', usecols=[c for c in headers if
        c != 'Unnamed: 0'])
    symptoms = dataset.columns[1 : ]
    X_train , Y_train , Y = read_dataset(dataset)
    model = train_SVC_model(X_train , Y_train)
    input_symptoms = get_custom_test_cases(symptoms)
    get_custom_symptoms_names(input_symptoms , symptoms)
    predict_disease(model , input_symptoms , Y_train , Y)
```

6.2.4 Naive Bayes

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. Naïve Bayes has three different models i.e. Gaussian, Multinomial and Bernoulli Naïve. Gaussian and Multinomial Naive Bayes are the Two types used in our system for prediction.

```
#!/usr/bin/env python
# coding: utf-8

#Importing Libraries
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.naive_bayes import GaussianNB , MultinomialNB

#Read Dataset
def read_dataset(dataset_copy):
    X_train = dataset_copy.iloc[ : , 1 : ].values
    Y = dataset_copy.iloc[ : , 0].values
    labelencoder = LabelEncoder()
```

```
    Y_train = labelencoder.fit_transform(Y)
    return X_train , Y_train , Y

#Training Prediction Model
def train_naive_bayes_model(X_train , Y_train):
    classifier_1 = GaussianNB()
    classifier_2 = MultinomialNB()
    classifier_1.fit(X_train , Y_train)
    classifier_2.fit(X_train , Y_train)
    return classifier_1 , classifier_2

#Get Custom Test Cases
def get_custom_test_cases(symptoms):
    input_symptoms=[]
    for x in range(0,len(symptoms)):
        input_symptoms.append(0)
    faiyaz_symptoms = ['joint_pain' , 'vomiting' , 'fatigue' , 'yellowish_skin'
        , 'dark_urine' , 'loss_of_appetite' , 'abdominal_pain' , '
        yellowing_of_eyes']
    for index , symptom in enumerate(symptoms):
        for faiyaz_symptom in faiyaz_symptoms:
            if symptom == faiyaz_symptom:
                input_symptoms[index] = 1
    return input_symptoms

#Get Names of Custom Inputs
def get_custom_symptoms_names(input_symptoms , symptoms):
    print('Passed Symptoms ::-')
    for i in range(0 , len(input_symptoms)):
        if input_symptoms[i] == 1:
            print(symptoms[i])

#Predicting and Printing Disease
def predict_disease(gaussian_model , multinomial_model , input_symptoms ,
    Y_train , Y):
    my_symptoms_pred_1 = gaussian_model.predict([input_symptoms])
    my_symptoms_pred_acc_score_1 = gaussian_model.predict_proba([input_symptoms
        ])

    my_symptoms_pred_2 = multinomial_model.predict([input_symptoms])
    my_symptoms_pred_acc_score_2 = multinomial_model.predict_proba([
        input_symptoms])

    for num , disease in zip(Y_train , Y):
        if int(num) == int(my_symptoms_pred_1[0]):
            print('\nPredicted Disease (GaussianNB) :: ' , disease)
            break
    else:
        print('No Disease')
    print('Accuracy Score ::-' , my_symptoms_pred_acc_score_1.max()*100)

    for num , disease in zip(Y_train , Y):
        if int(num) == int(my_symptoms_pred_2[0]):
            print('\nPredicted Disease (MultinomialNB):: ' , disease)
            break
    else:
        print('No Disease')
    print('Accuracy Score ::-' , my_symptoms_pred_acc_score_2.max()*100)

#Main Function
if __name__ == '__main__':
```

```
    headers = [*pd.read_csv('SHDPS_Training_1.csv', nrows=1)]
    dataset = pd.read_csv('SHDPS_Training_1.csv', usecols=[c for c in headers if
        c != 'Unnamed: 0'])
    symptoms = dataset.columns[1 : ]
    X_train , Y_train , Y = read_dataset(dataset)
    gaussian_model , multinomial_model = train_naive_bayes_model(X_train ,
        Y_train)
    input_symptoms = get_custom_test_cases(symptoms)
    get_custom_symptoms_names(input_symptoms , symptoms)
    predict_disease(gaussian_model , multinomial_model , input_symptoms ,
        Y_train , Y)
```

6.2.5 Decision Tree

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

```
#!/usr/bin/env python
# coding: utf-8

#Importing Libraries
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier

#Read Dataset
def read_dataset(dataset):
    X_train = dataset.iloc[ : , 1 : ].values
    Y = dataset.iloc[ : , 0].values
    labelencoder = LabelEncoder()
    Y_train = labelencoder.fit_transform(Y)
    return X_train , Y_train , Y

#Training Prediction Model
def train_decision_tree_model(X_train , Y_train):
    classifier = DecisionTreeClassifier(criterion = 'entropy' , random_state =
        0)
    classifier.fit(X_train , Y_train)
    return classifier

#Get Custom Test Cases
def get_custom_test_cases(symptoms):
    input_symptoms=[]
```

```
    for x in range(0,len(symptoms)):
        input_symptoms.append(0)
    faiyaz_symptoms = ['joint_pain' , 'vomiting' , 'fatigue' , 'yellowish_skin'
        , 'dark_urine' , 'loss_of_appetite' , 'abdominal_pain' , '
        yellowing_of_eyes']
    for index , symptom in enumerate(symptoms):
        for faiyaz_symptom in faiyaz_symptoms:
            if symptom == faiyaz_symptom:
                input_symptoms[index] = 1
    return input_symptoms

#Get Names of Custom Inputs
def get_custom_symptoms_names(input_symptoms , symptoms):
    print('Passed Symptoms ::-')
    for i in range(0 , len(input_symptoms)):
        if input_symptoms[i] == 1:
            print(symptoms[i])


#Predicting and Printing Disease
def predict_disease(model , input_symptoms , Y_train , Y):
    my_symptoms_pred = model.predict([input_symptoms])
    my_symptoms_pred_acc_score = model.predict_proba([input_symptoms])

    for num , disease in zip(Y_train , Y):
        if int(num) == int(my_symptoms_pred[0]):
            print('\nPredicted Disease :: ' , disease)
            break
    else:
        print('No Disease')
    print('Accuracy Score ::-' , my_symptoms_pred_acc_score.max()*100)

#Main Function
if __name__ == '__main__':
    headers = [*pd.read_csv('SHDPS_Training_1.csv', nrows=1)]
    dataset = pd.read_csv('SHDPS_Training_1.csv', usecols=[c for c in headers if
        c != 'Unnamed: 0'])
    symptoms = dataset.columns[1 : ]
    X_train , Y_train , Y = read_dataset(dataset)
    model = train_decision_tree_model(X_train , Y_train)
    input_symptoms = get_custom_test_cases(symptoms)
    get_custom_symptoms_names(input_symptoms , symptoms)
    predict_disease(model , input_symptoms , Y_train , Y)
```

6.2.6 Random Forest

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based

on the majority votes of predictions, and it predicts the final output.

```
#!/usr/bin/env python
# coding: utf-8

#Importing Libraries
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier

#Read Dataset
def read_dataset(dataset):
    X_train = dataset.iloc[ : , 1 : ].values
    Y = dataset.iloc[ : , 0].values
    labelencoder = LabelEncoder()
    Y_train = labelencoder.fit_transform(Y)
    return X_train , Y_train , Y

#Training Prediction Model
def train_random_forest_model(X_train , Y_train):
    classifier = RandomForestClassifier(n_estimators = 10 , criterion = 'entropy
        ' , random_state = 0)
    classifier.fit(X_train , Y_train)
    return classifier

#Get Custom Test Cases
def get_custom_test_cases(symptoms):
    input_symptoms=[]
    for x in range(0,len(symptoms)):
        input_symptoms.append(0)
    faiyaz_symptoms = ['joint_pain' , 'vomiting' , 'fatigue' , 'yellowish_skin'
        , 'dark_urine' , 'loss_of_appetite' , 'abdominal_pain' , '
        yellowing_of_eyes']
    for index , symptom in enumerate(symptoms):
        for faiyaz_symptom in faiyaz_symptoms:
            if symptom == faiyaz_symptom:
                input_symptoms[index] = 1
    return input_symptoms

#Get Names of Custom Inputs
def get_custom_symptoms_names(input_symptoms , symptoms):
    print('Passed Symptoms ::-')
    for i in range(0 , len(input_symptoms)):
        if input_symptoms[i] == 1:
            print(symptoms[i])

#Predicting and Printing Disease
def predict_disease(model , input_symptoms , Y_train , Y):
    my_symptoms_pred = model.predict([input_symptoms])
    my_symptoms_pred_acc_score = model.predict_proba([input_symptoms])

    for num , disease in zip(Y_train , Y):
        if int(num) == int(my_symptoms_pred[0]):
            print('\nPredicted Disease :: ' , disease)
            break
    else:
        print('No Disease')
    print('Accuracy Score ::-' , my_symptoms_pred_acc_score.max()*100)

#Main Function
if __name__ == '__main__':
```

```
    headers = [*pd.read_csv('SHDPS_Training_1.csv', nrows=1)]
    dataset = pd.read_csv('SHDPS_Training_1.csv', usecols=[c for c in headers if
        c != 'Unnamed: 0'])
    symptoms = dataset.columns[1 : ]
    X_train , Y_train , Y = read_dataset(dataset)
    print('No. Of Symptoms in Dataset ::- ' , str(len(symptoms)))
    print('No of Diseases in Dataset ::- ' , )
    model = train_random_forest_model(X_train , Y_train)
    input_symptoms = get_custom_test_cases(symptoms)
    get_custom_symptoms_names(input_symptoms , symptoms)
    predict_disease(model , input_symptoms , Y_train , Y)
```

6.3 Live Consultation Module

Our system is designed to help patient in such tough times where whole world is suffering from pandemic an lockdown is imposed on different parts of the world. Our system aims to bring the Doctors and patients much closer while everyone are at their homes by connecting them virtually through our platform and patient can seek medical help through our website by making direct one to one live consultation with the doctors available / registered on our platform.

Our system will not only bridge the gap between doctors and patients but will also try to make the process more and more simplified. Not only that but the poor one's who cannot afford to consult doctors directly can also seek help through our platform since doctors who wants to do social service are most welcome on our platform and it doesn't even need much efforts from both sides since every process in online and can be done on the go.

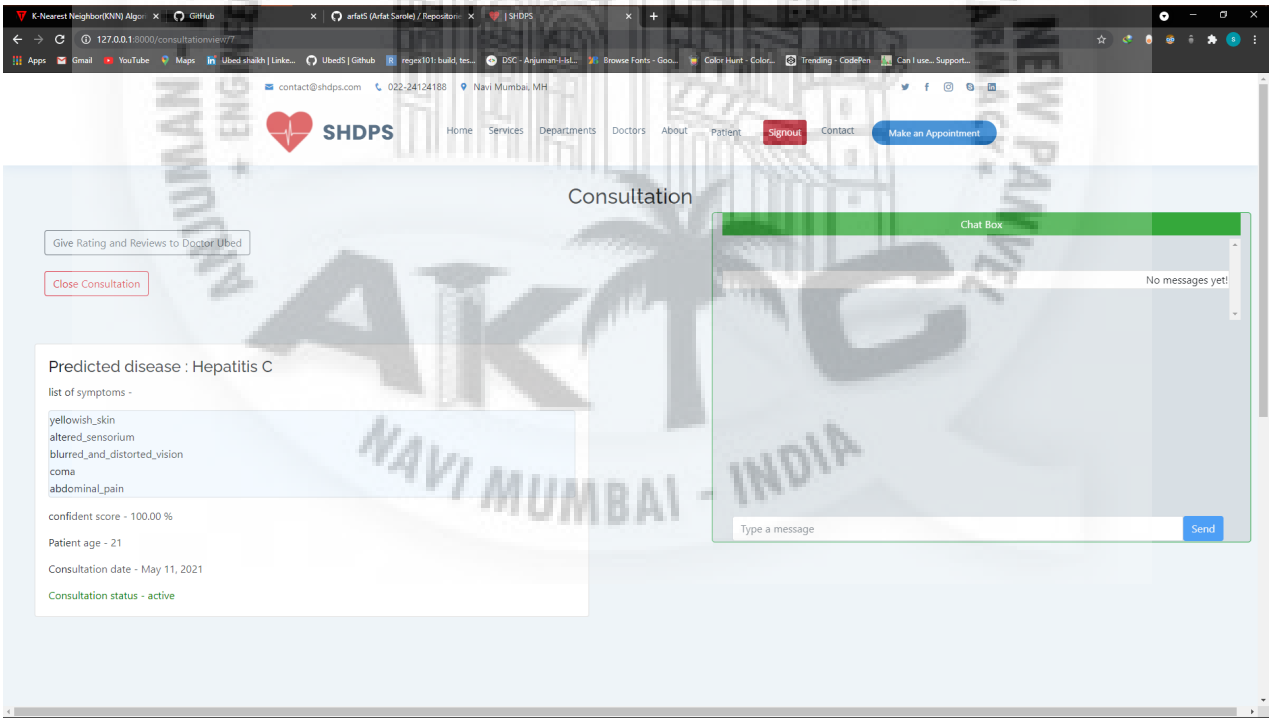


Figure 6.2: Live Consultation Module

```
def post(request):
    if request.method == "POST":
        msg = request.POST.get('msgbox', None)
        consultation_id = request.session['consultation_id']
        consultation_obj = consultation.objects.get(id=consultation_id)
        c = Chat(consultation_id=consultation_obj,sender=request.user, message=
            msg)
        if msg != '':
            c.save()
            print("msg saved"+ msg )
            return JsonResponse({ 'msg': msg })
    else:
        return HttpResponse('Request must be POST.')

def chat_messages(request):
   if request.method == "GET":
     consultation_id = request.session['consultation_id']
     c = Chat.objects.filter(consultation_id=consultation_id)
     return render(request, 'consultation/chat_body.html', {'chat': c})
```

Chapter 7

System Testing

We have tested a list of disease with our model in order to determine the best one some of the crucial and best results we have mentioned below.

7.1 Test Cases and Test Results

Test ID	Test Case Title	Test Condition	Obtained Disease + Accuracy Score	Expected Disease + Accuracy Score
T01	Disease Prediction	Hepatitis D	Hepatitis D , 94%	Hepatitis D , 98%
T02	Disease Prediction	Heart Attack	Heart Attack , 97%	Heart Attack , 90%
T03	Disease Prediction	Pneumonia	Pneumonia , 95%	Pneumonia , 91%

7.2 Sample of a Test Case

Title: Disease Prediction – Predicting Patient disease on the basis of provided symptoms
Description: A patient should be able to predict the disease from which he/she is suffering on the basis of provided symptoms.

Precondition: A registered user should be registered as a patient and should provide true symptoms which he/she is suffering from to our system.

Assumption: A supported browser is being used.

Test Steps:

1. Sign in as Patient.

2. Press Check Disease module on Patient UI page.
3. Enter the Symptoms using GUI.
4. Press Predict Button
5. Disease Will Be Predicted with Accuracy Score

Expected Result: A page displaying the Disease with Accuracy score with which the model is predicting to the user.

Actual Result:

A page displaying the Disease with Accuracy score with which the model is predicting to the user. It includes the disease which is predicted as well as the confidence score provided by the model. If the Accuracy score is greater than 70 then the model is quite confident with the prediction. The system automatically classifies the disease on the basis of medical fields and refers the speialist doctor registered on our platform in that disease i.e **Rheumatologist , Cardiologist , Orthopedist , Dermatologist**.

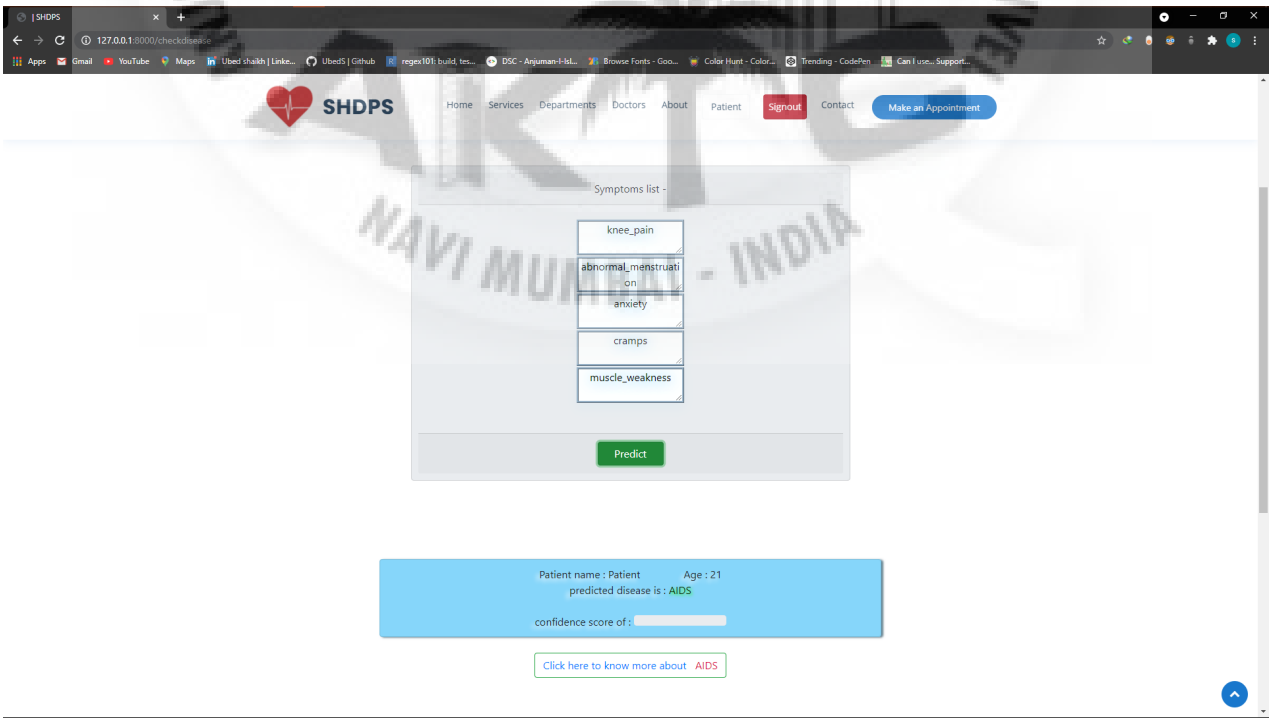


Figure 7.1: Disease Prediction

7.2.1 Software Quality Attributes

Smart Health Disease Prediction system provides enormous functionalities to solve real life problems and also the ultimate tools for problem solving. It must be made sure that the system is reliable in its operations and for securing the sensitive details.

If the internet service gets disrupted while sending information to the server, the information can be send again for prediction.

As the system is easy to handle and navigates in the most expected way with no delays. In that case the system program reacts accordingly and transverses quickly between its states.

Information transmission is securely transmitted to server without any changes in information. The integrity of user personal information is maintained at any cost.

The database is highly atomic and substantial ans also durable. In case of system failures the database can maintain its state without losing data.

Chapter 8

Screenshots of Project

8.1 Home Page

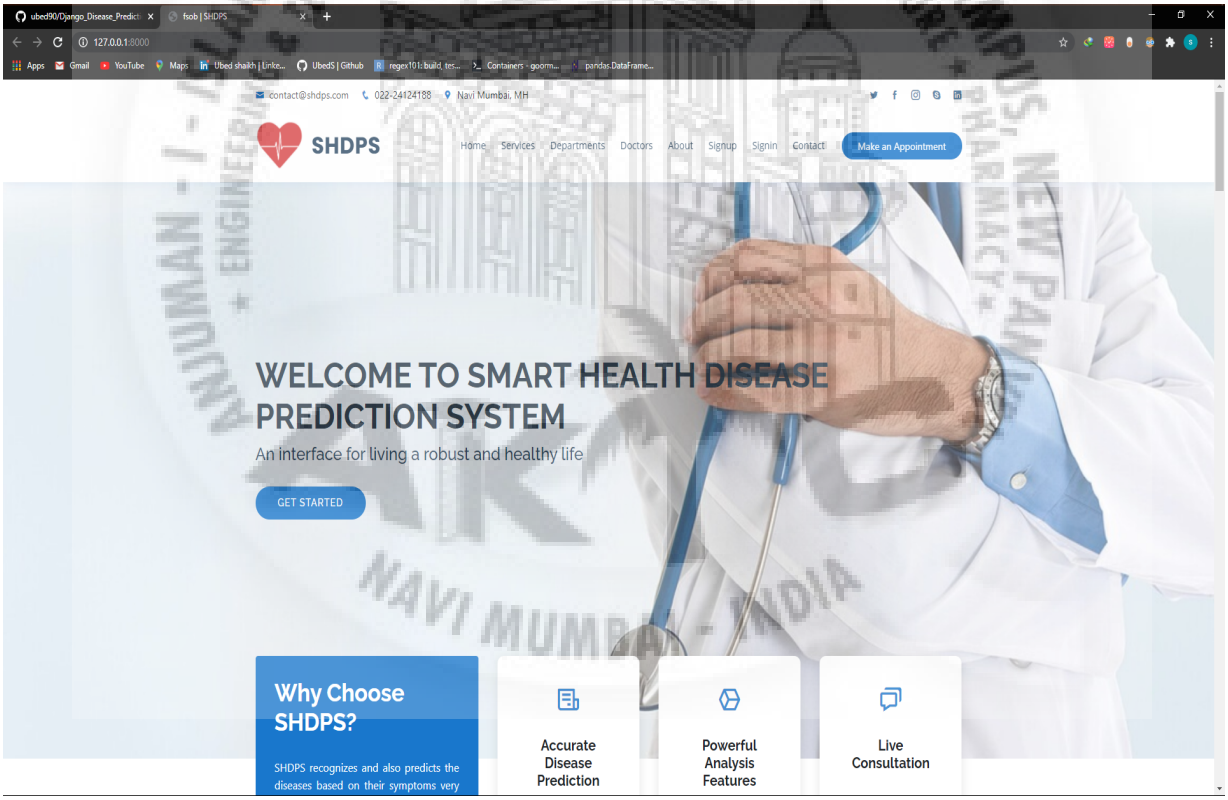


Figure 8.1: Screenshot of Home Page

8.2 Contact Page

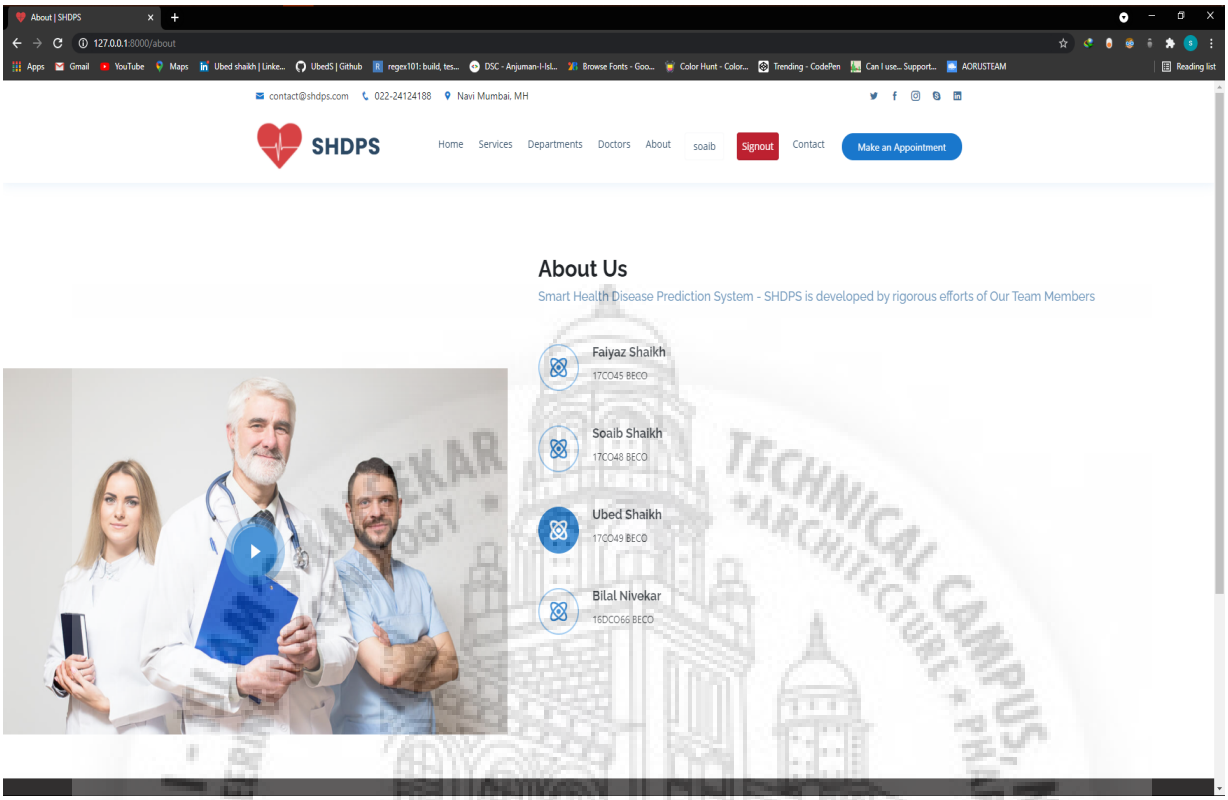


Figure 8.2: Screenshot of Contact Page

8.3 Dashboards

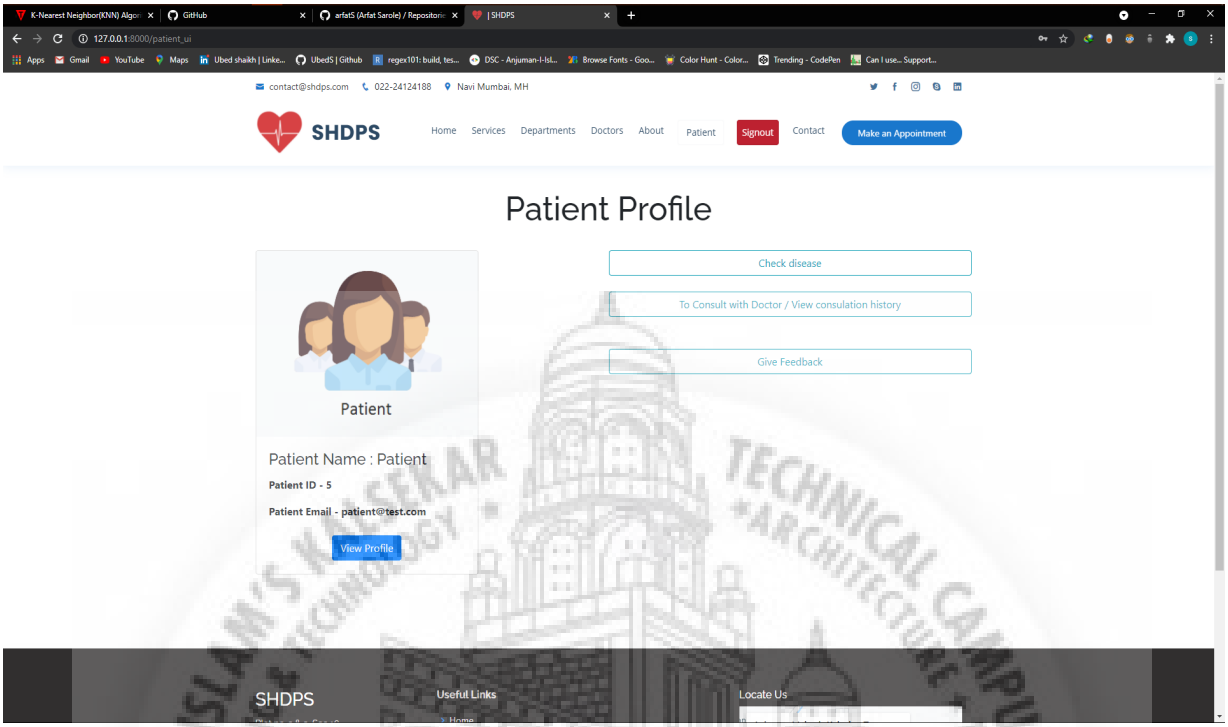


Figure 8.3: Screenshot of Patient Dashboard

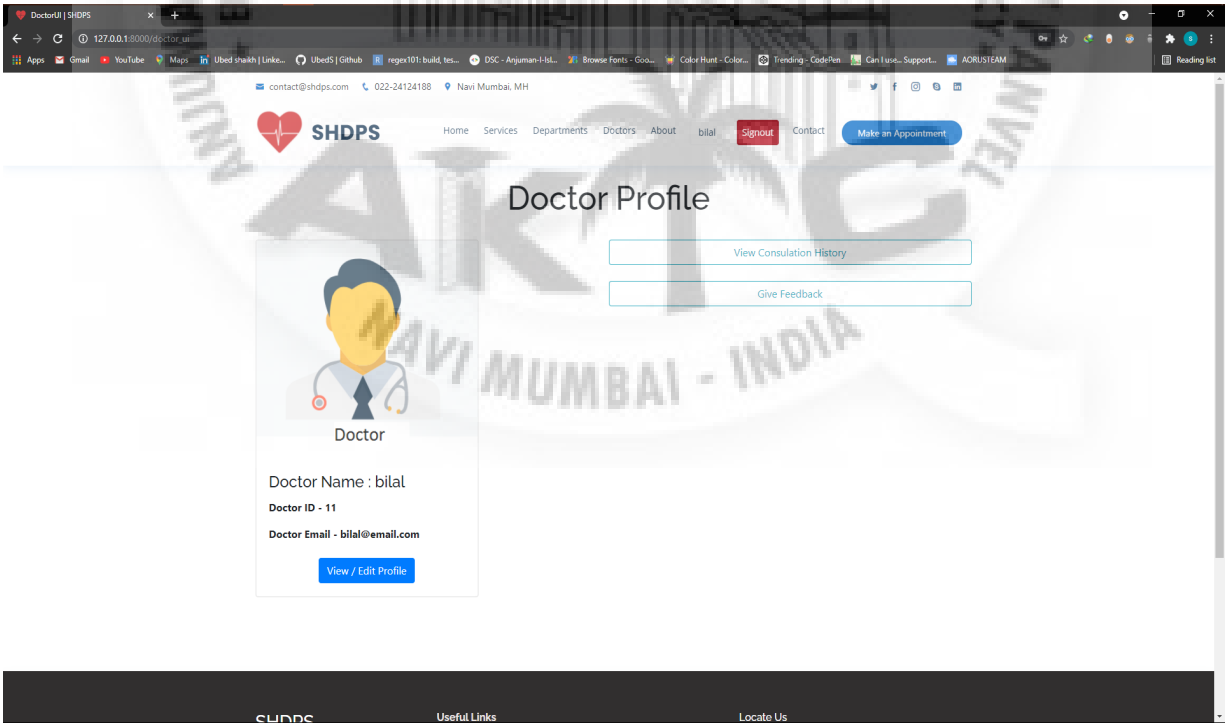


Figure 8.4: Screenshot of Doctor Dashboard

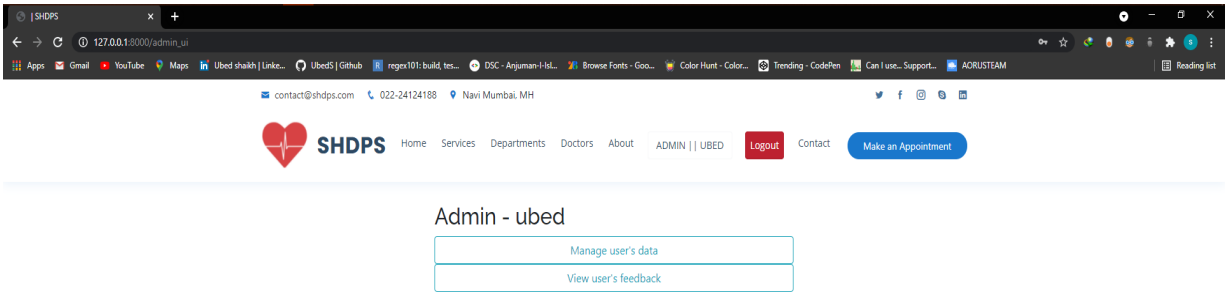


Figure 8.5: Screenshot of Admin Dashboard

8.4 Disease Prediction

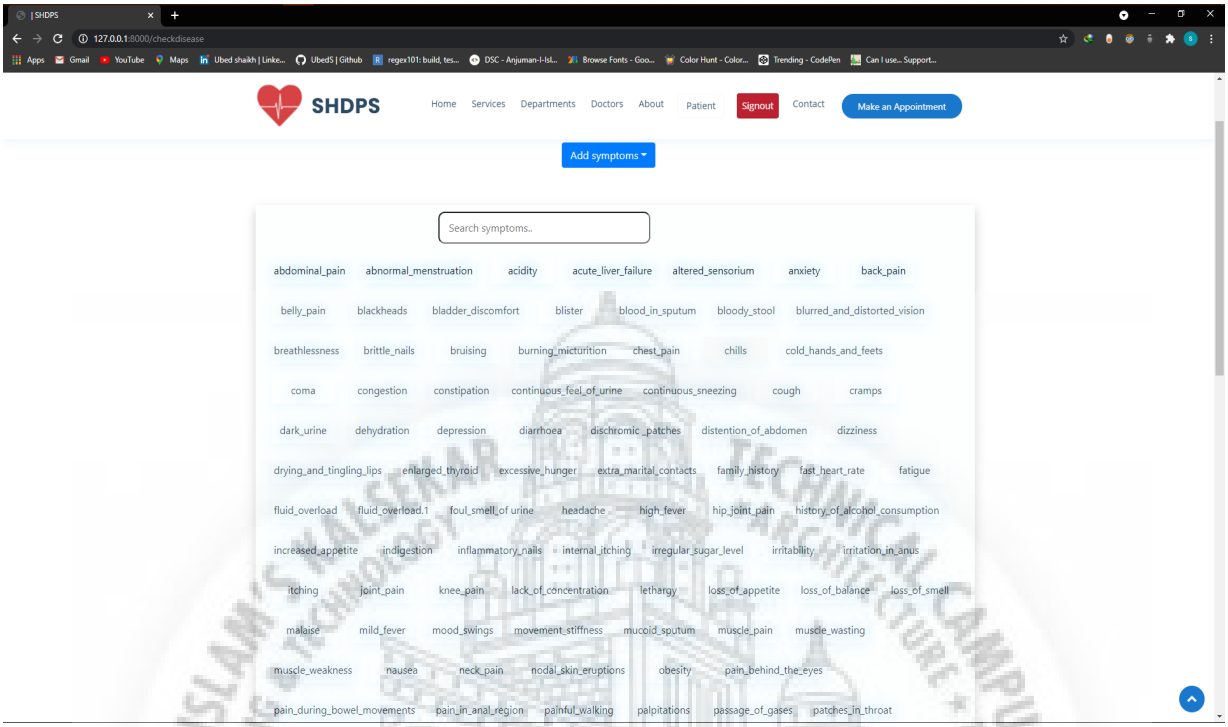


Figure 8.6: Screenshot of Symptoms Page

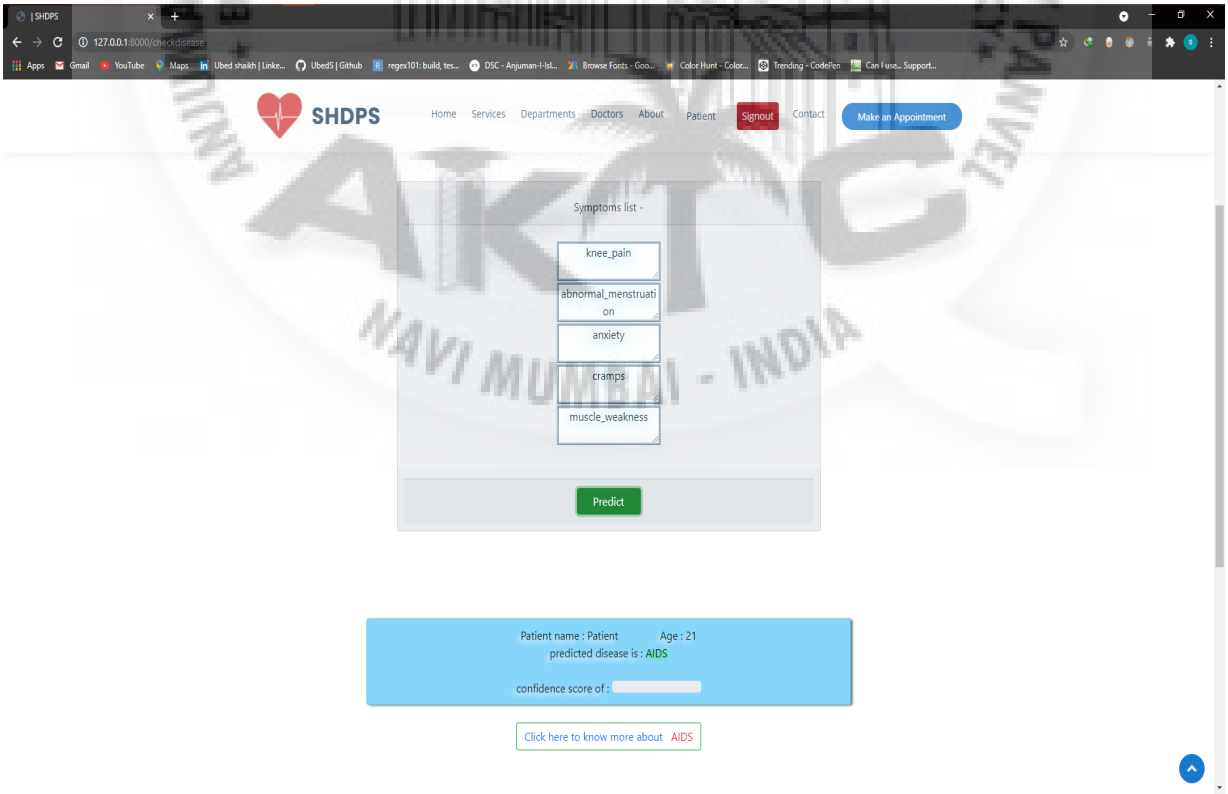


Figure 8.7: Screenshot of Disease Prediction Module

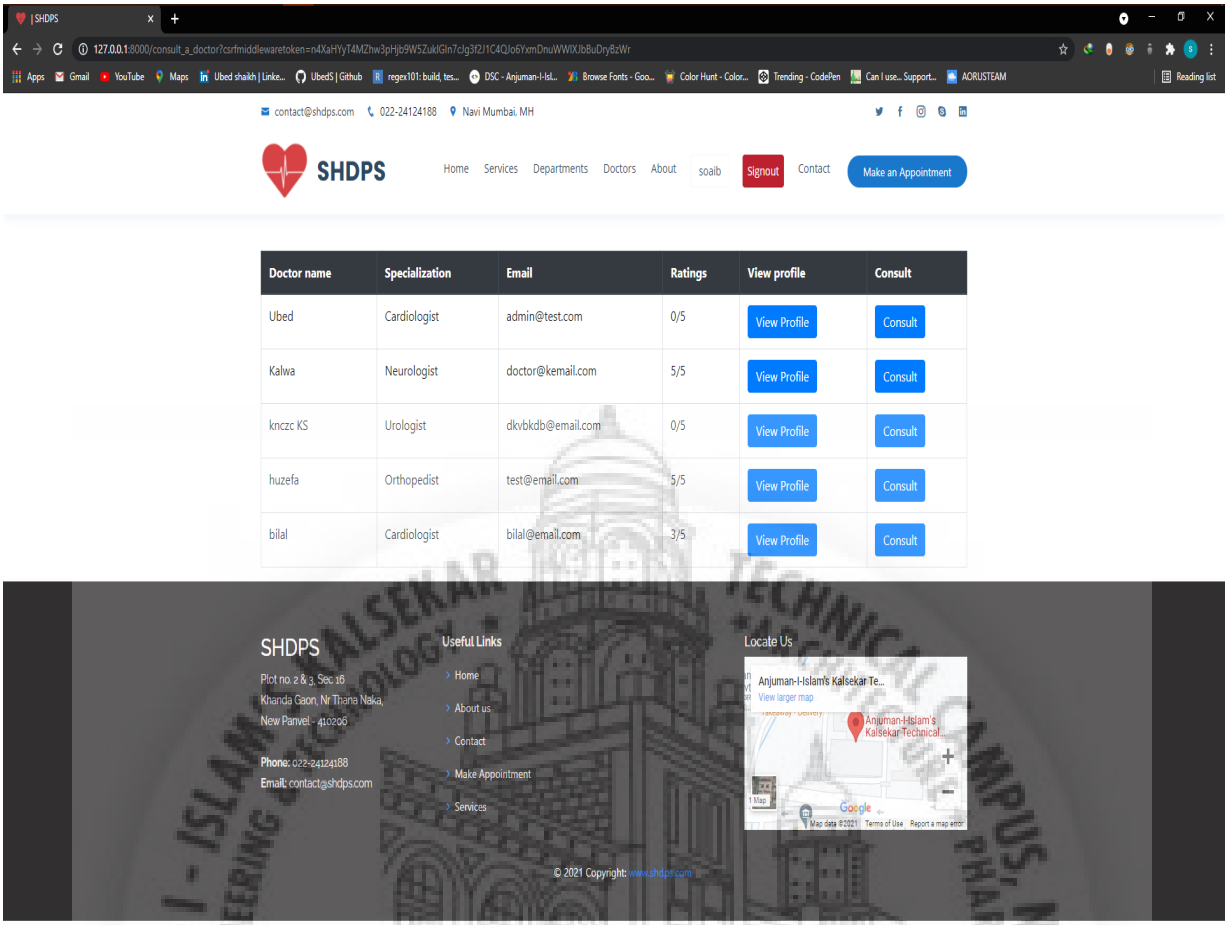


Figure 8.8: Screenshot of Available Doctors Page

8.5 Consultation History

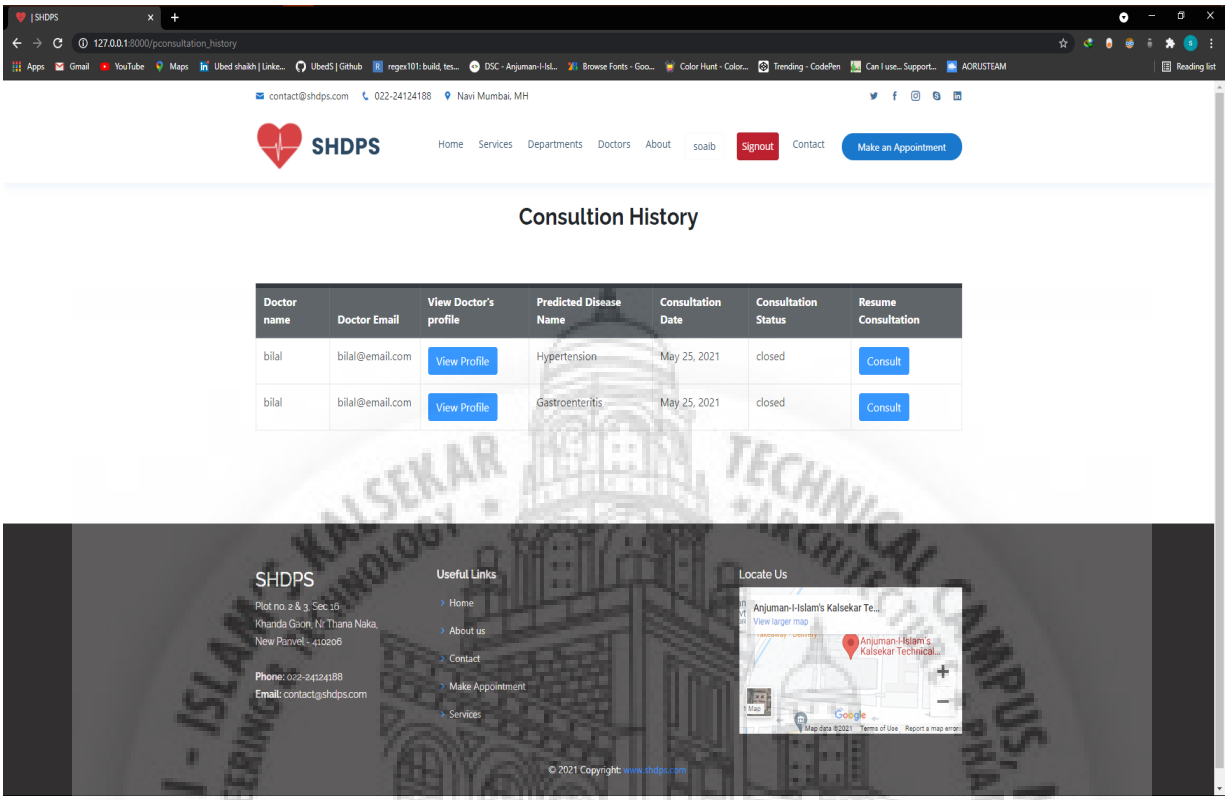


Figure 8.9: Screenshot of Patient Consultation History Page

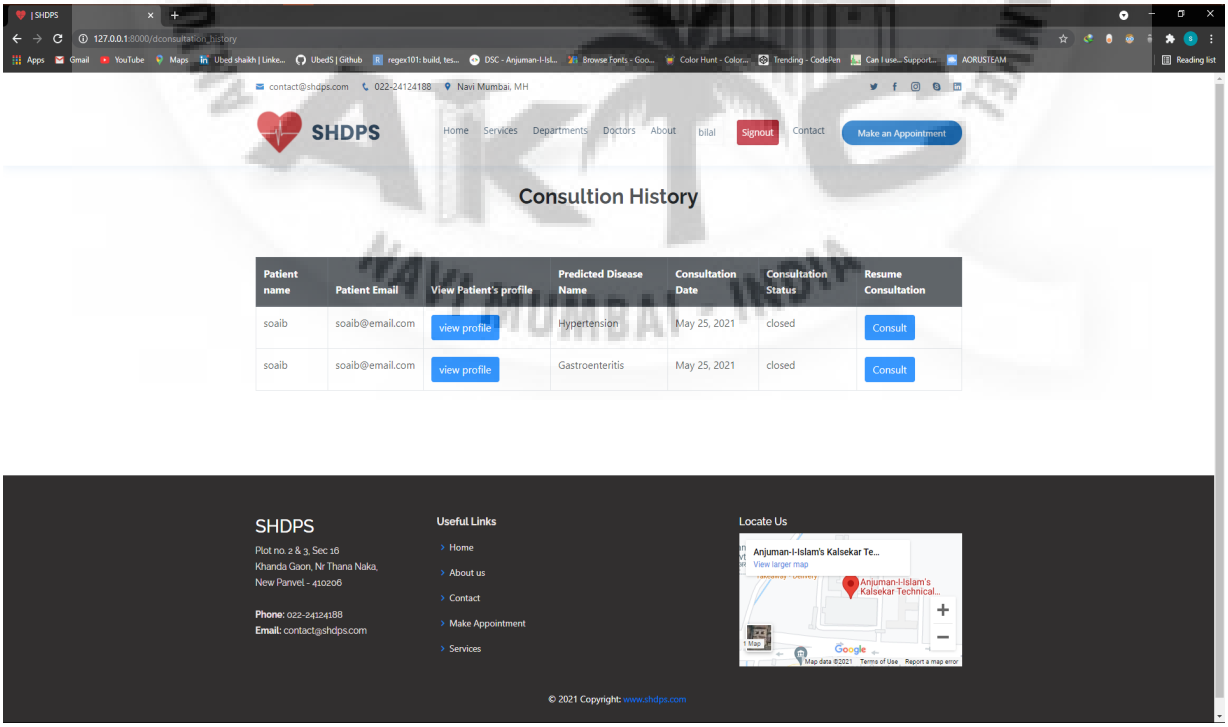


Figure 8.10: Screenshot of Doctor Consultation History Page

8.6 Live Consultation

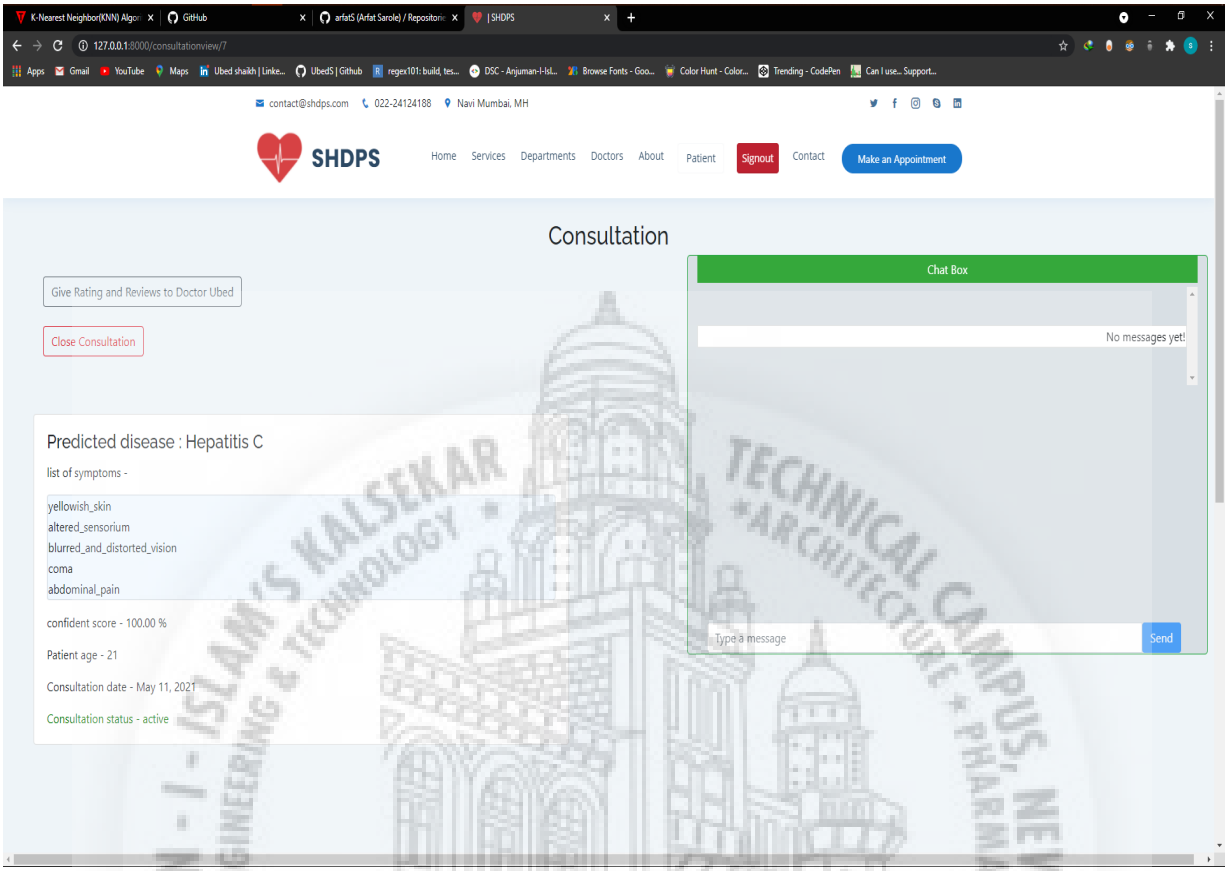


Figure 8.11: Screenshot of Live Consultation Page

Chapter 9

Conclusion and Future Scope

9.1 Conclusion

As we have seen the real importance and value of Health and lives in the current ongoing pandemic. Their is no wealth more than a Healthy life. As a part of serving the common people and solving their health related problems was our motto throughout the Development of this project.

We developed this Webapp for predicting and analysing the diseases using various Machine Learning Algorithms for early cure and ailment of diseases based on the symptoms provided by the user which indeed became very fruitful and successful by providing highest accuracy and accurate prediction. We also facilitated our users with live consultation with Doctors in ongoing pandemic so that they can get proper medical care from their home.

This project is a well planned Combo for Generic Health care with best optimal solutions be it in predicting a disease and analysing it. Also getting the information of the risks associated to other organs of body from predicted disease. This project also recommended Specialist Doctor for communication and getting cure as early as possible.

This project involved our immense and rigorous efforts for development and resolving the hurdles in day to day health problems of

people. We would like to make more Projects for the help of needy people and also for the upliftment of society in future.

9.2 Future Scope

We wish to keep working on our project to make it a marketable and leading product. Some of the additions that we feel are needed for it to happen are:

- Making it deployable on Cloud for global accessibility.
- Increasing the dataset by using our Disease Addition Program for predicting more diseases.
- Making User Interface more User Friendly.
- Adding more functionalities for Real Time Health Care using various parameters like BP, Sugar Level, Body Temperature and RBC WBC Readings, etc.
- Customizing this project with live Health Blogs using datascrapping.

References

[1] B. Nithya and V. Ilango,*"Predictive analytics in health care using machine learning tools and techniques,"* 2018

[2] A. Aldallal and A. Al-Moosa, *"Using Data Mining Techniques to Predict Diabetes and Heart Diseases,"* 2018 4th International Conference on Frontiers of Signal Processing (ICFSP), Poitiers, 2018

[3] Vijaya Shetty, Kathik GA, M Ashwin, *"Symptom Based Health Prediction using Data Mining,"* 2019

[4] Dhiraj Dahiwade, Gajanan Patle, Ektaa Meshram, *"Designing Disease Prediction Model Using Machine Learning Approach,"* 2019

[5] K. Lakshmi Priya, Mourya Sai Charan Reddy Kypa, Muchumarri Madhu Sudhan Reddy, G. Ram Mohan Reddy, *" A Novel Approach to Predict Diabetes by Using Naive Bayes Classifier,"* 2020

[6] R.Jane Preetha Princy, Saravanan Parthasarathy, P. Subha Hency Jose, *" Prediction of Cardiac Disease using Supervised Machine Learning Algorithms,"* 2020

[7] Aravindkumar.S, Ajith.A, *"Prediction of Chronic Disease by Machine Learning,"* 2019

[8] Sneha Grampurohit, Chetan Sagarnal, *"Disease Prediction using Machine Learning Algorithms,"* 2020

[9] Priyanka Sonar, K. JayaMalini, *"DIABETES PREDICTION USING DIFFERENT MACHINE LEARNING APPROACHES,"* 2019