COLLEGE OF ENGINEERING AND TECHNOLOGY

MULAVOOR P.O, MUVATTUPUZHA, KERALA – 686673

PUBLIC MENTAL HEALTH PREDICTION USING MACHINE LEARNING TECHNIQUES INNERHOUR: MENTAL HEALTH FRIEND

PROJECT REPORT

FATHIMA K S (ICE19CS048)

NANDANA P R (ICE19CS085)

NEERAJA SUBHASH (ICE19CS087)

PAVITHRA C A (ICE19CS092)

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY IN

COMPUTER SCIENCE AND ENGINEERING

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

AUGUST 2022

ILAHIA

COLLEGE OF ENGINEERING AND TECHNOLOGY

Mulavoor P.O, Muvattupuzha, Kerala – 686673

ICET
Dynamism, Growth & Selfless Service

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that the project report entitled **"PUBLIC MENTAL HEALTH PREDICTION USING MACHINE LEARNING TECHNIQUES"** is a bonafide record of the project report presented by FATHIMA K S (ICE19CS048), NANDANA P R (ICE19CS085), NEERAJA SUBHASH (ICE19CS087), PAVITHRA C A (ICE19CSE092) during the academic year 2021- 2022 towards the partial fulfilment of the requirement of the award of B. Tech Degree in Computer Science and Engineering of APJ Abdul Kalam Technological University, Thiruvananthapuram.

Dr. Vadhana Kumari S	**Ms. Chitra Rani P R**	**Dr. Sujith Kumar P.S**
Assistant Professor	Assistant Professor	Professor & HOD
CSE Department	CSE Department	CSE Department
Project Guide	**Project Coordinator**	**Head of the Department**

DECLARATION

We Fathima K S, Nandana P R, Neeraja subhash, Pavithra C A hereby declare that, this project report entitled “Public mental health prediction using machine learning techniques” is the bonafide work of mine carried out under the supervision of Mrs Chitra Rani. We declare that to the best of our knowledge, the work report here in does not form part of any project report or dissertation the basis of which a degree or award was conferred on an earlier occasion on any other candidate the content of this report is not being presented by any other student to this or any other university for the award of degree.

Signature:
Name of the Student: FATHIMA K S
Uni. Register No: ICE19CS048

Signature:
Name of the Student: NANDANA P R
Uni. Register No: ICE19CS085

Signature:
Name of the Student: NEERAJA SUBHASH
Uni. Register No: ICE19CS087

Signature:
Name of the Student: PAVITHRA C A
Uni. Register No: ICE19CS092

Signature:
Name of the guide: DR. VADHANA KUMARI S

Signature:
Name of the Coordinator: MS. CHITRA RANI P R

Countersigned with Name: Dr. SUJITH KUMAR P.S
HOD, Computer Science & Engineering
Ilahia College of Engineering and Technology

Date :

ACKNOWLEDGMENT

Apart from the efforts of our, the success of this project report depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We would like to show our heartfelt gratitude towards **Prof. Dr. ABDUL GAFUR M**, Principal, Ilahia College of Engineering and Technology for granting us the permission to work this project. Also, we would like to show our greatest gratitude towards our head of the department of computer science & Engineering, **Dr. Sujith Kumar P.S** and project guide **Dr. Vadhana Kumari S,** Our project Coordinators **Ms Chitra Rani,** for their valuable advice and guidance.

Finally, we express our gratitude and thanks to all our teachers and other faculty members of the Department of Computer Science & Engineering, for their sincere and friendly cooperation in completing this project.

Date :

Place :

FATHIMA K S
NANDANA P R
NEERAJA SUBHASH
PAVITHRA C A

ABSTRACT

Around the world 970 million individuals are affected with mental illness. In a fast-growing society, it is a hectic task to keep track on our mental health and wellness. As mental health is as important as someone's physical health it's is each one's responsibility to take care of it. Interpretation of public emotional wellness concerns utilizing data science, deep learning and machine learning techniques by observing life style, schedules, emotions and sentimental analysis will pave way to give the user an insight of their mental health. Our project focus on this issue and give prediction about user's mental health and thus suggest them a review about their mental health issue and lend a helping hand nearby who can be a therapist or counselor of their taste. High prevalence of mental health and the need for effective mental fitness care, blended with current advances in AI, has led to a growth in explorations of ways the sphere of system getting to know Machine Learning can assist inside the detection, prognosis and treatment of mental health issues.

CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

Terminology	Meaning
CSS	Cascading Style Sheet
ERD	Entity Relationship Diagram
SDLC	System Development Life Cycle
DFD	Data Flow Diagram
IT	Information Technology
ASP	Active Server Page
HTTP	Hyper Text Transfer Protocol
SQL	Structure Query Language
DBMS	Database Management System
HTML	Hypertext Markup Language
GUI	Graphical User Interface
IDE	Integrated Development Environment
XML	Extensible Markup Language
EES	Expert Embedded System

CHAPTER 1

INTRODUCTION

Around the world 970 million individuals are affected with mental illness. The increase of mental problems and the need for effective medical health care have led to an investigation of machine learning that can be applied in mental health problems. This paper presents a recent systematic review of machine learning approaches in predicting mental health problems. We will discuss the challenges, limitations and future directions for the application of machine learning in the mental health field. As the mental health is an important as someone's physical health it's is each one's responsibility to take care of it. Our project focus on this issue and give prediction about user's mental health.

Machine learning is a technique that aims to construct systems that can improve through experience by using advanced statistical and probabilistic techniques. It is believed to be significantly useful tool to help in predicting mental health. It is allowing many researches to acquire important information from the data, provide personalized experiences and develop automated intelligent systems. The widely used algorithms in the field of machine such as support vector machine, random forest, and artificial neural networks have been utilized to forecast and categorize the future events. Supervised learning in machine in machine learning in the most widely applied approach in many types of research, studies, and experiments, especially in predicting illness in the medical field. In supervised learning, the terms, attributes, and values should be reflected in all data instances.

In this paper, the main objective is to provide a systematic literature review, critical review, and summary of the machine learning techniques that are being used to predict, diagnose, and identify mental health problems. Moreover, this paper will propose future avenues for research on this topic. It would also give attention to the challenges and limitations of applying the machine learning techniques in this area.

CHAPTER 2

LITERATURE SURVEY

According to Yadav, S., Kaim, T., Gupta, S., Bharati, U., and Priyadarshi, P., the model could be integrated into existing government database or several public health entities to examine the risk factors and prevalence of depression. Early forecast and identification of depression from the data provided from varies surveys. This model only focus on predicting depression and it doesn't focus on predicting the specified type by analysing behavioural anomalies given by data set. This model boarded only to depression prediction as it can also predict several other mental illnesses as it use wide range of data set.

According to U. S. Reddy, A. V. Thota and A. Dharun, "machine learning techniques for stress prediction in working employees," analyse stress patterns in working adults and to narrow down the factors that strongly determine the stress levels. By using decision trees, prominent features that influence stress were identified as gender, family history and availability of health benefits in the workplace. This technique lack of accuracy in cross validation, which is the major aspect of forming a prediction model is to ensure its stability.

According to A.Gruenerbl, G.Bahle, J.Weppner,P .Olesky and P.Luknowicz,'Towards smartphone based monitoring of bipolar disorder', in proceedings of the second ACM workshop on mobile systems, application and services for healthcare. In this paper, the system, which based on smartphone sensing is able to recognize depressive and manic States and detect state changes of patients suffering from bipolar disorder.

According to U.Haetal, A Wearable EEG-HEG-HRV Multimodal system with simultaneous monitoring of tEs for mental health management'. The system can measure the brain activity from the various signal domain for improved accuracy on it enables the simultaneous monitoring of stimulation. This proposed wearable system can be used for an accurate, safe and effective closed loop stress management.

CHAPTER 3

PROPOSED SYSTEM

3.1 INTRODUCTION

Around the world 970 million individuals are affected with mental illness. The increase of mental problems and the need for effective medical health care have led to an investigation of machine learning that can be applied in mental health problems. This paper presents a recent systematic review of machine learning approaches in predicting mental health problems. We will discuss the challenges, limitations and future directions for the application of machine learning in the mental health field. As the mental health is an important as someone's physical health it's is each one's responsibility to take care of it. Our project focus on this issue and give prediction about user's mental health.

Machine learning is a technique that aims to construct systems that can improve through experience by using advanced statistical and probabilistic techniques. It is believed to be significantly useful tool to help in predicting mental health. It is allowing many researches to acquire important information from the data, provide personalized experiences and develop automated intelligent systems. The widely used algorithms in the field of machine such as support vector machine, random forest, and artificial neural networks have been utilized to forecast and categorize the future events. Supervised learning in machine in machine learning in the most widely applied approach in many types of research, studies, and experiments, especially in predicting illness in the medical field. In supervised learning, the terms, attributes, and values should be reflected in all data instances.

In this paper, the main objective is to provide a systematic literature review, critical review, and summary of the machine learning techniques that are being used to predict, diagnose, and identify mental health problems. Moreover, this paper will propose future avenues for research on this topic. It would also give attention to the challenges and limitations of applying the machine learning techniques in this area.

CHAPTER 4

4.1 REGISTRATION

Patients need to register to the system to access the features. Only registered users can login to the site. During registration details such as name, address, email, contact, location etc are stored in the database.

4.2 LOGIN

All three users-admin, therapist and patient need to login to the system to access their home page. Email id and password are the user credentials used. This is created during the registration of each request

4.3 MANAGE THERAPIST

Admin manages the details of therapist. This is done to prevent the submission of fake details to the system. Admin add the details of therapist like name, address, email, contact and location. This location is used during the searching process. These details are manually collected by the admin and added to the system. Admin can also delete a therapist if needed

4.4 MANAGE PATIENT

Admin can view all the details of the patients registered in the site. Admin can also delete their account if found to be fake or caught for some malpractices.

4.5 UPDATE PROFILE

Patients and therapists can update their personal details after login to the system.

4.6 ATTEND SURVEY

Registered patients can attend a survey containing objective type and descriptive type questions. Both the tests are used for analysing the mental health of patient. The answers of the descriptive type questions are used for sentimental analysis and this result is also used for analysis.

4.7 BOOK THERAPIST

After the test, patients can find out the nearest therapist based on their location. They can also book for the therapist.

4.8 VIEW BOOKING

Patients can view their booking details and booking history. Therapist can view which all patients are booked on each date.

4.9 VIEW HISTORY

Patients, therapist and admin can view the history of the questions and their answers given during each test of patients. This is used by the therapist for further treatments.

CHAPTER 5

SECURITY REQUIREMENTS

5.1 HARDWARE REQUIREMENTS

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, a hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

- Processor: 1.4GHz or above processors are considered good. They can handle heavier tasks.
- Memory (RAM): Minimum 2 GB; Recommended 4 GB or above.
- Storage: 1GB free disk space for workstations (2-5 GB Recommended)-1GB data storage capacity (2GB recommended).
- Minimum screen resolution is required is 738x844.

5.2 SOFTWARE REQUIREMENTS

The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from client’s point of view.

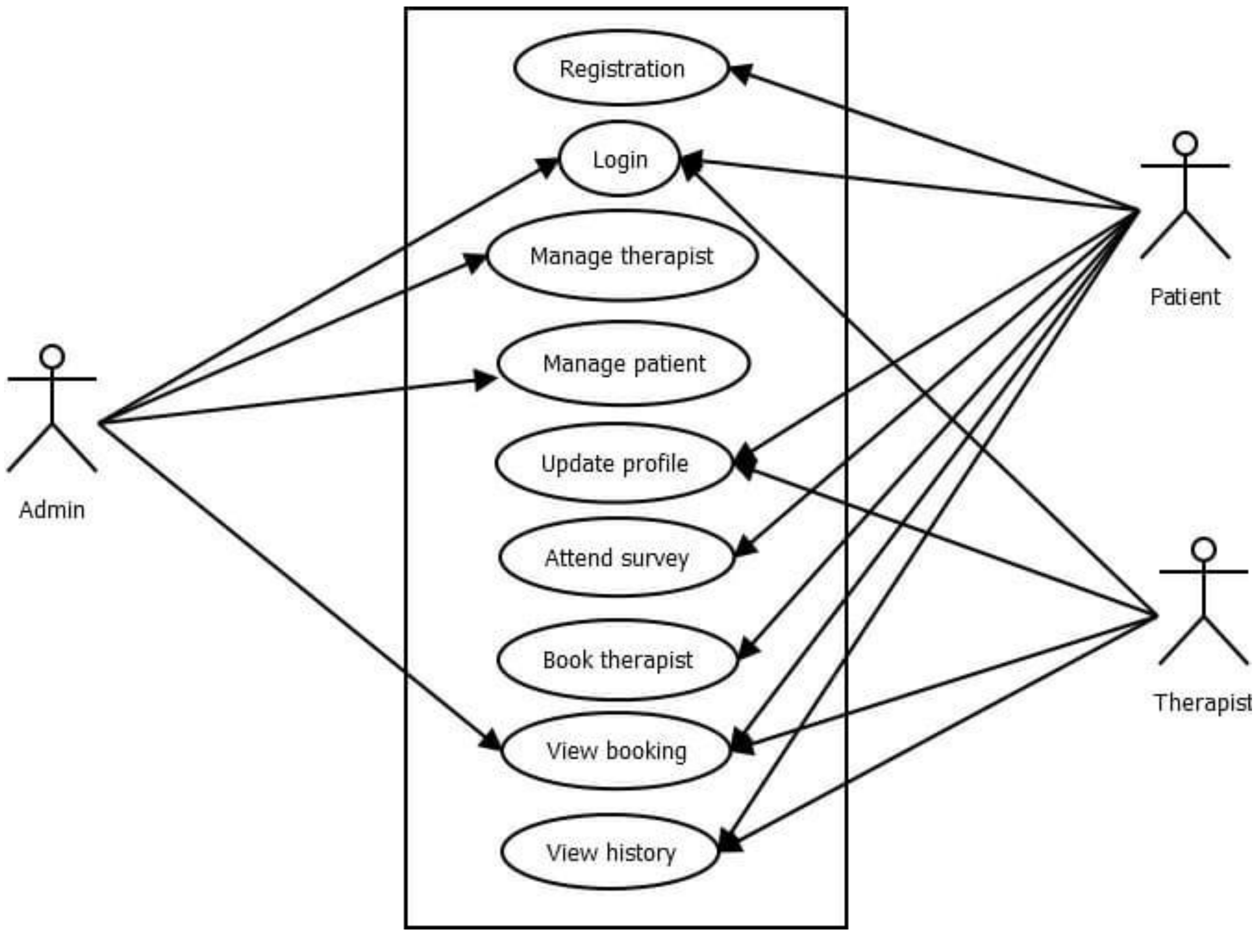
- Android version: 5 or above
- Languages used are HTML, CSS, JAVASCRIPT, Bootstrap in the frontend.

- Web-application development using python as backend and Django as python framework.
- MVT Architecture is used.
- SQL lite is used for database.

CHAPTER 6

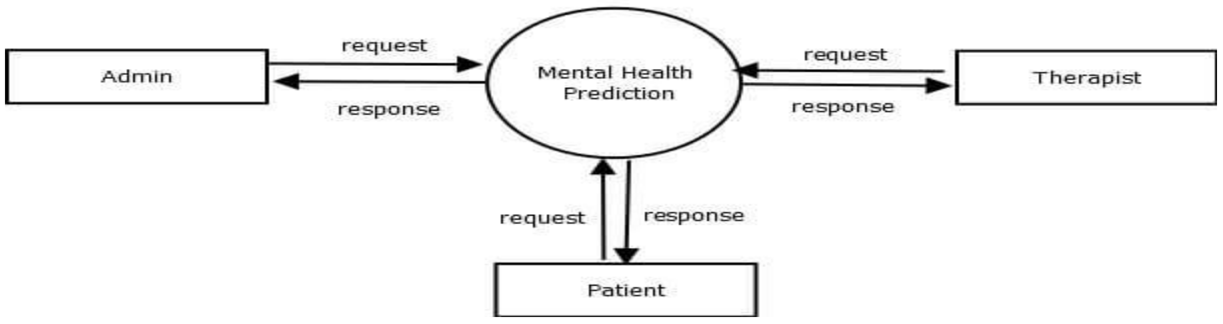
SOLUTION METHODOLOGY

6.1USE CASE DIAGRAM

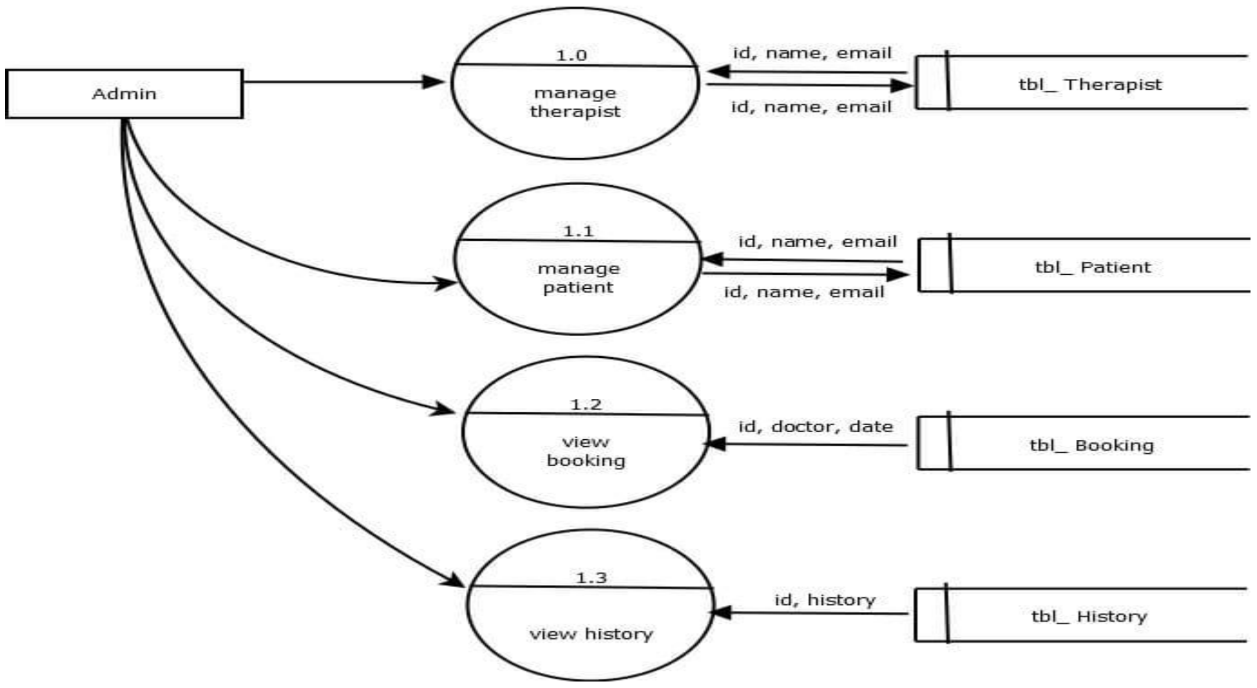


6.2 FLOW CHART

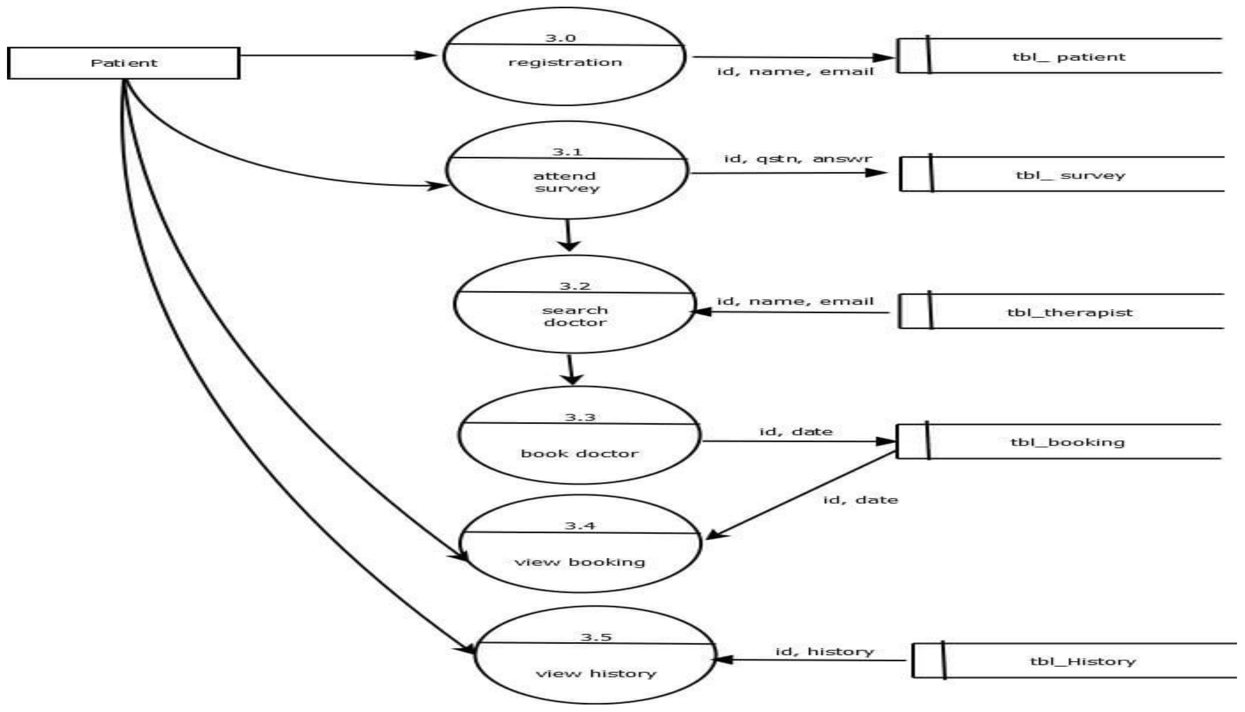
Level0



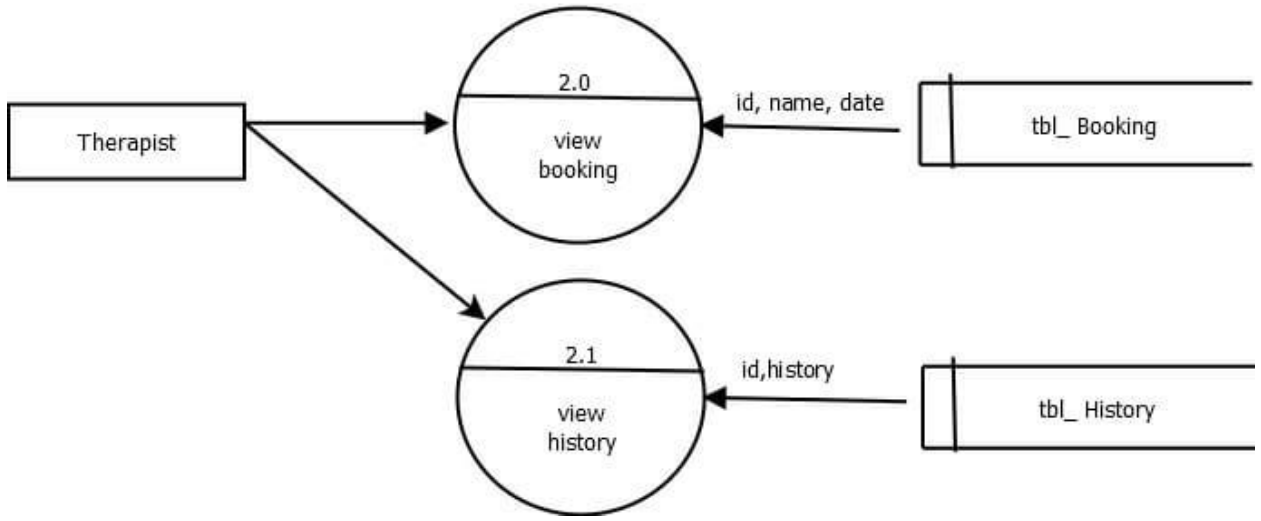
Level1 Admin



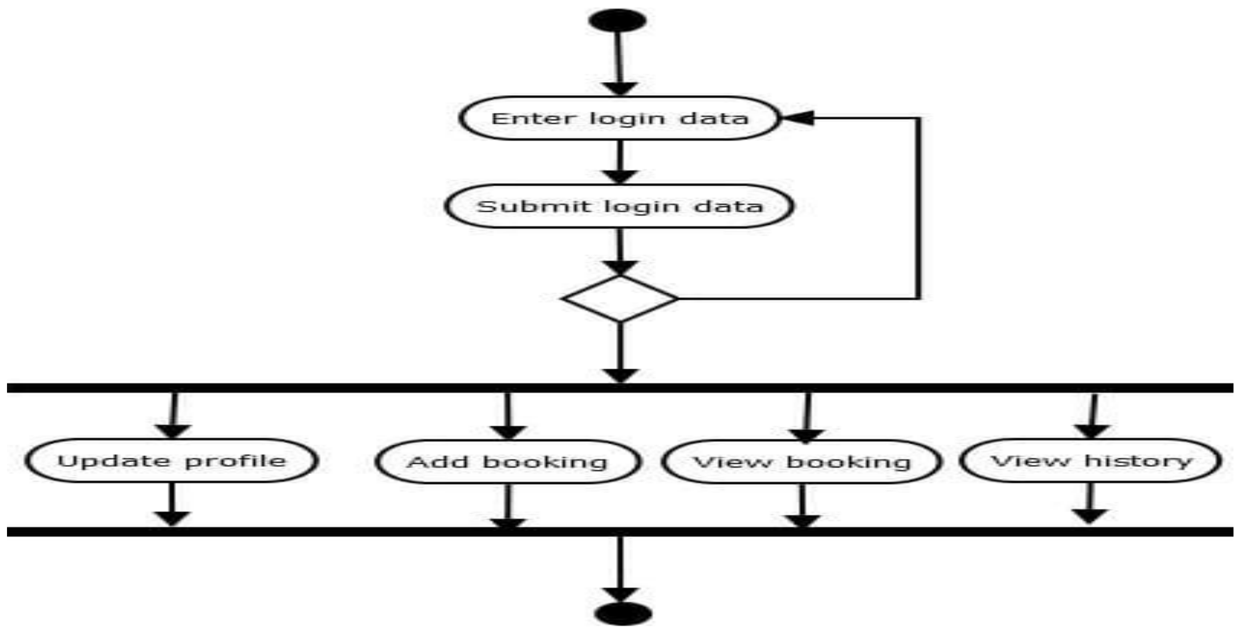
Level1 patient

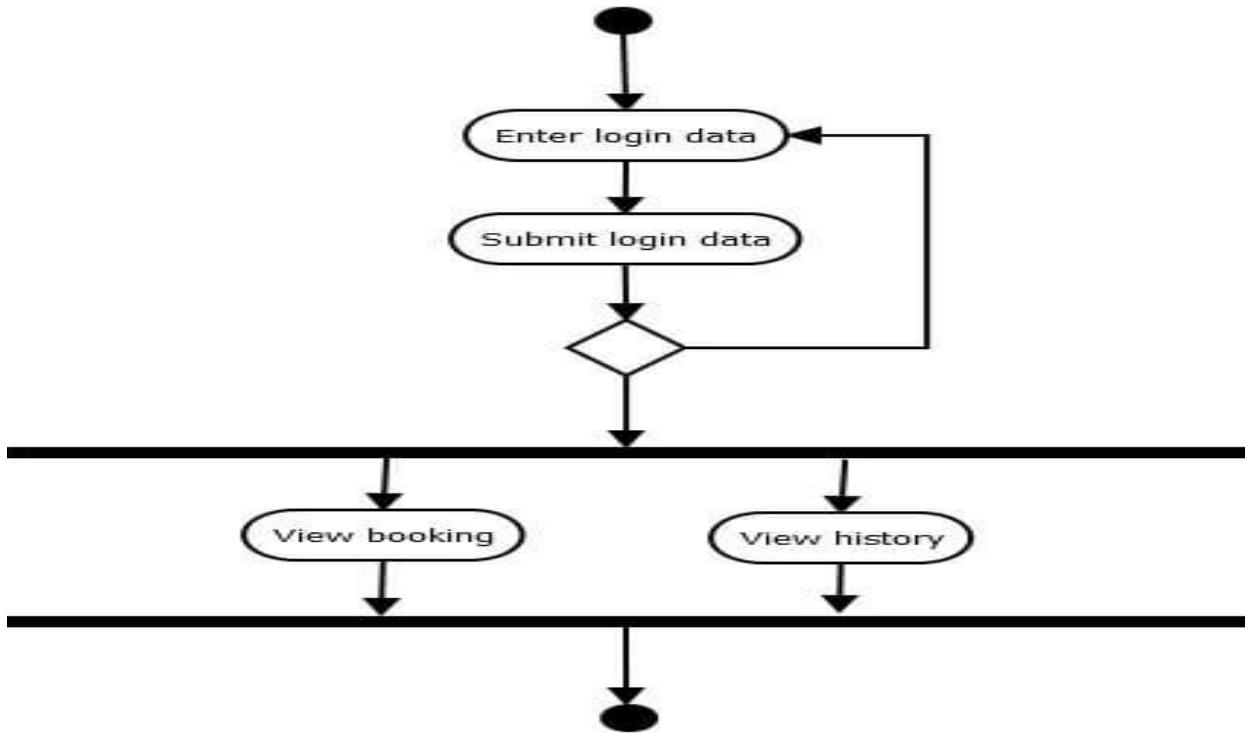


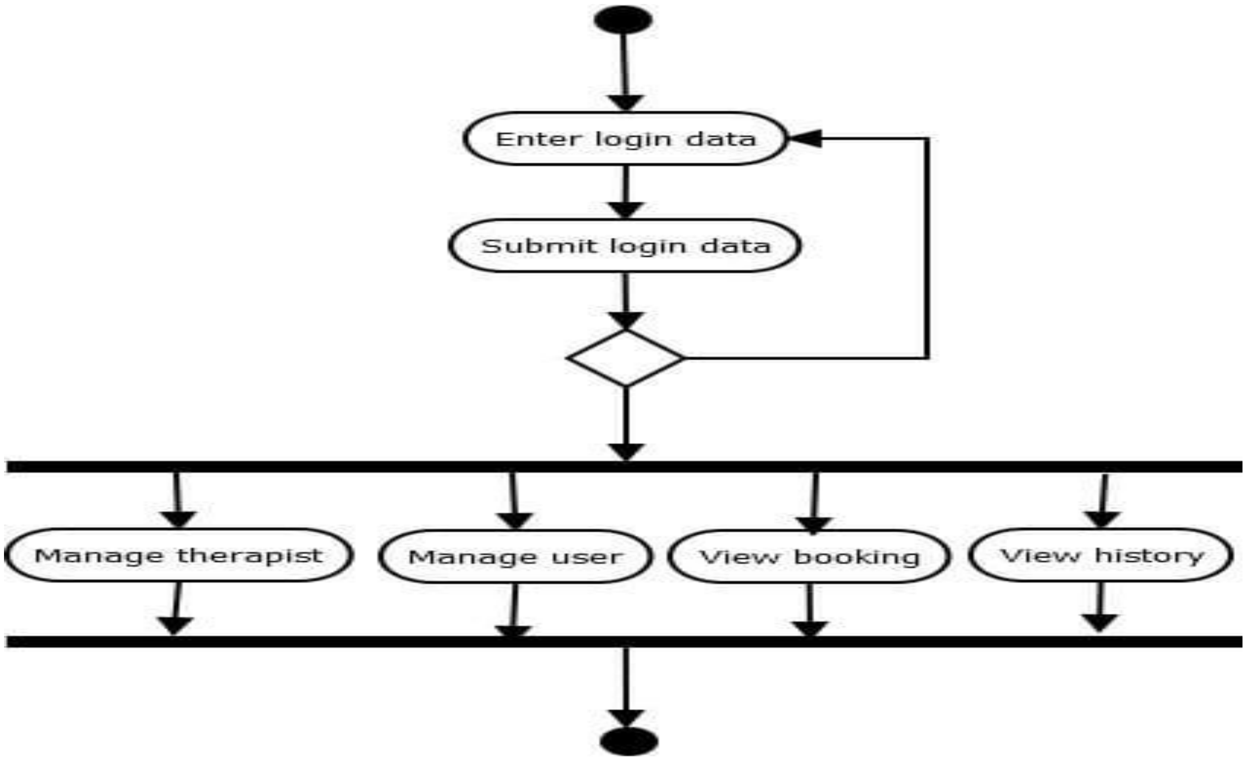
Level1 Therapist



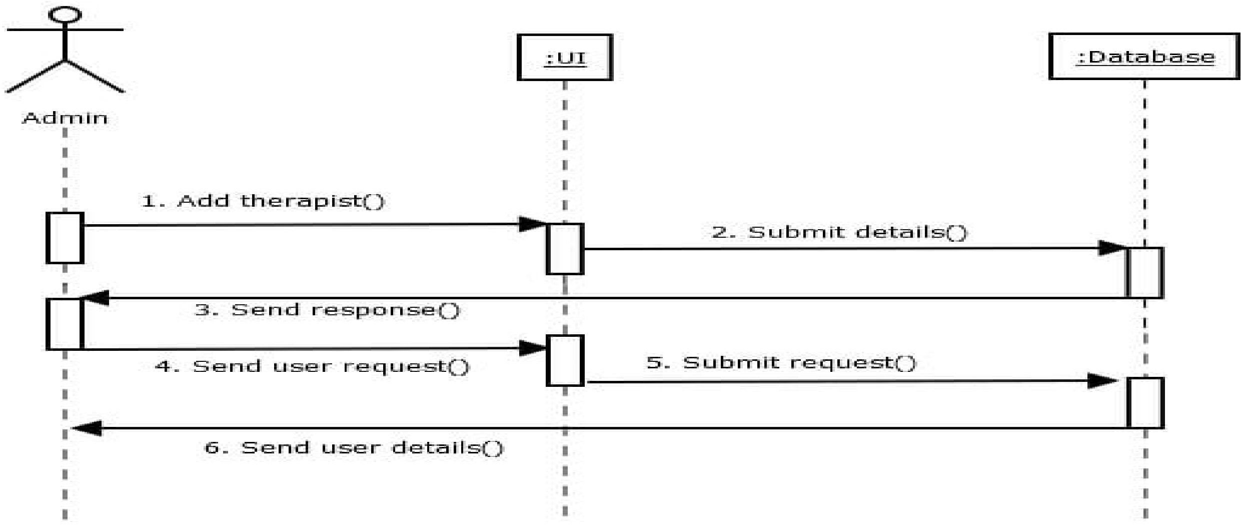
6.3 Activity Diagram

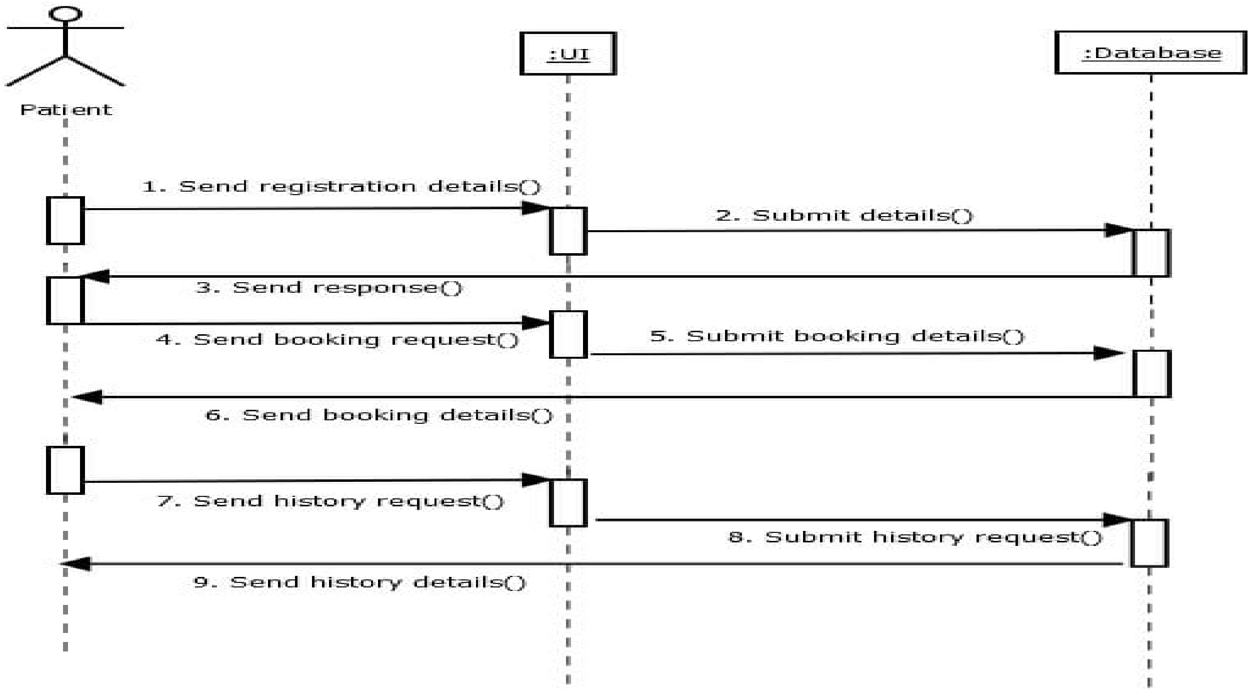


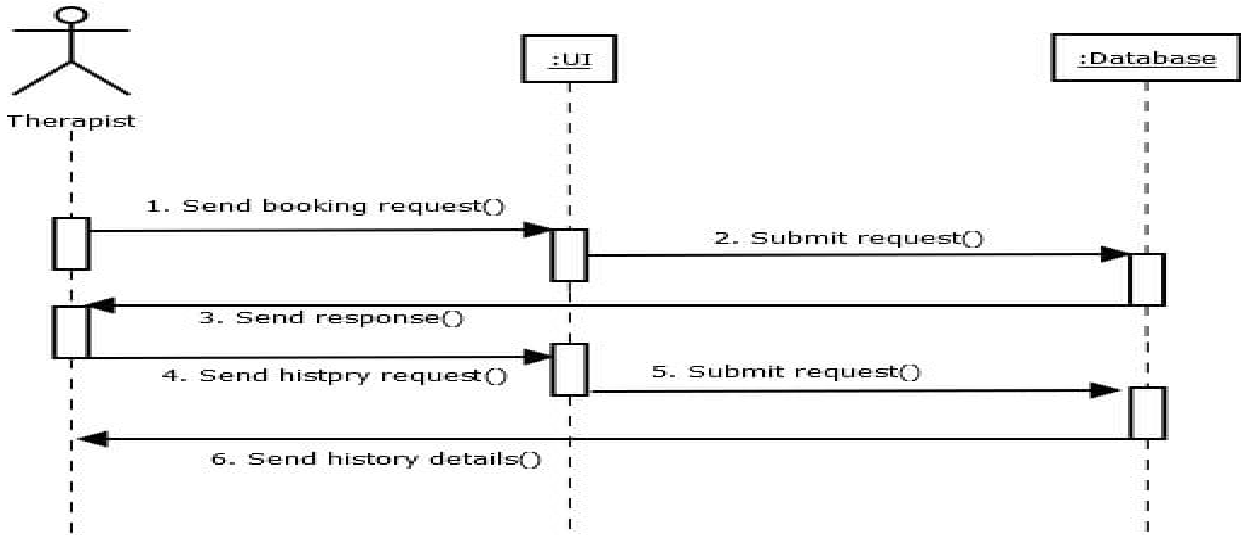



Enter login data
Submit login data
Manage therapist
Manage user
View booking
View history

6.4 Sequence Diagram




Patient
:UI
:Database
1. Send registration details()
2. Submit details()
3. Send response()
4. Send booking request()
5. Submit booking details()
6. Send booking details()
7. Send history request()
8. Submit history request()
9. Send history details()


Therapist
:UI
:Database
1. Send booking request()
2. Submit request()
3. Send response()
4. Send histpry request()
5. Submit request()
6. Send history details()

CHAPTER 7

IMPLEMENTATION

Common mental disorders such as depression, anorexia, dementia, posttraumatic stress disorder or schizophrenia affect millions of people around the world. Most people believe that mental disorders are not usual or happen to other people that have specific personal damage. When in fact, mental disorders are prevalent and familiar. Many families think they are not prepared to face the fact that some loved one has a mental problem. A mental disorder causes different interferences in the thinking and behaviour of the affected person. These interferences could vary from mild to severe, and could result in an inability to live routines in daily life and ordinary demands. Common mental disorders such as depression and anorexia affect millions of people around the world. They may be related to a single incident causing excessive stress on the person or by a series of different stressful events. It is also well known that mental disorders tend to increase in countries experiencing generalized violence or recurrent natural disasters. Mental health has always been an important and challenging issue. The modernized (hectic) lifestyle and workload take a toll over people over time making them more prone to mental disorders like mood disorder and anxiety disorder. Thus, the risk mental health problems increase in working professionals. Previous work they have addressed to utilize the data from mental health survey 2019 that contains the data of working professionals for both tech and non-tech company employees. We process data to find the features influencing the mental health of employees or features that can help to predict the mental health of the employee the feature can be either personal or professional. We apply multiple machine learning algorithms to find the model with the best accuracy. We take precision and recall as the measure to check the performance of different ML models. This information presents an opportunity for us to extend the use of ML algorithms to predict the mental health status of user and also assign a therapist if needed.

7.1 DATA PRE-PROCESSING

Variables with more than 50% missingness were removed from analysis (202 variables excluded). Redundant variables were also removed (134 excluded). Additionally, variables with no variance were removed (32 excluded) and those with variance near zero were combined into one variable, if possible, e.g., dust, mold, and pollen allergy collapsed into allergy. Ultimately, 85 variables were determined to be suitable for analysis. As most machine learning techniques require complete datasets, missing values were imputed with tree-based imputation with the R package mice.

7.2 DATASET EXTRACTION

The bug reports dataset is extracted from the repository. The bug reports were classified into five levels: lowest, low, medium, high and highest. Although software engineers usually follow a specific guideline on how to assign severity of reported bugs, however, the categorization process seems to be evaluated imprecisely. In this research work, we treat lowest and low severity as non-severe, while high and highest severity as severe bugs. The medium class, as it is investigated, is the default option for reporting a bug and it seems that a large number of reporters report any confusing bug as a medium. The datasets mainly consist of three features including bug ID, a short description (i.e., one-line summary) and the severity level of the bug. The description of each bug is represented as a short text (snippet). From the bug reports, the severity and the short description of the bug report were mainly used for the prediction process. Too many words are repetitive in the dataset where the number of distinct words is significantly fewer than the total number of all words exists within the dataset (765 out of 9016).

7.3 STATISTICAL ANALYSIS

All analyses were performed in R. First, a learning curve was plotted with the entire dataset in order to check if our study was sufficiently powered. Then, we split our data into a training-set (60% of the sample), a tune-set (10%), and a test-set (30%). Splitting data allows for more accurate determination in how the model will perform in a new dataset and helps alleviate overfitting, i.e., to fit the training data too closely to accurately predict other datasets. Stratified random sampling was used to ensure that the twin pairs would not get separated between the datasets, thus avoiding potential overfitting. Additionally, we preserved an equal distribution of the outcome between each set. Descriptive statistics were created for each set to determine the quality of the partition.

7.4 PERFORMANCE OF PREDICTIONS

The performance of predictions from considered models were determined by the area under the receiver operating characteristic curve (AUC). We created prediction models using several machine learning techniques: Support vector(sum), K- nearest neighbour (KNN), Decision Tree, Random Forest, Naïve Bayes. To determine which produced the best fitting model for a test set. Using cross validation, each technique trained multiple models using the training set and tested their performance on a subset of the training set. The model with the lowest error was then tested using the tune set. Once the performance in the tune set was deemed satisfactory, the final models were then fitted to the test set. Parameter tuning was guided in part by standard practice when available, however a majority of the tuning took place through the random search function in R package mlr. Random search was completed using cross-validation with 3 iterations Decision Tree, 50 times. Variable importance was calculated for tree-based model: random forest. Confidence intervals at 95% were created for each AUC by bootstrapping predictions 10,000 times. Positive Predictive and Negative predictive values were obtained for the best performing model.

7.5 MACHINE LEARNING ALGORITHMS

In this section, different machine learning algorithms for predicting the severity level of the bugs are described and utilized in this study. These algorithms are mainly concerned to deal with unstructured data such as text. In this study, machine learning algorithms namely: Support vector (SVM), K- nearest neighbour (KNN), Decision Tree, Random Forest, Naïve Bayes algorithms are used to classify severity levels of bug reports. These algorithms are described in the following subsections:

1) NAÏVE BAYES:

It is a probability-based classifier which applies Bayes theorem. For Naïve Bayes features should have strong independence. The Naïve Bayes classifier estimates the class conditional probability of class label with the assumption that all attributes are independent [34]. The NB has been widely used in the domain of text classification due to its simplicity and effectiveness. On the other hand, NBM is similar to the naïve Bayes classifier except that it considers a weight for each feature during probability calculations. The weight of each feature can be determined by specific distributions or as a parametric model. The parameters can be estimated based on the training data. For the case of NBM, the distribution of data is assumed as a multinomial model.

2) SUPPORT VECTOR MACHINE (SVM)

SVM takes data points as input and gives the output as a hyperplane. It divides the classes using a plane (hyperplane) also known as the decision boundary. Where the decision boundary must maximize the distance of the nearest element of each class. Decision boundary separates the points into different classes. Let n be number of data points

$(x1 * y1) .. ..(xn * yn)$ (1)

Here x is the real vector and y represents the class (0 or 1) .The maximum margin hyperplane can be defined as

$w*x - b$ (2)

Were w is the normal vector and is the offset of hyperplane along .

3) DECISION TREE:

The decision tree (DT) is a machine learning predictive approach used for classification and regression problems. It creates a prediction model by learning decision rules from data on a tree-like model. A DT model has a tree structure in which each node represents a test on a given variable and each branch represents the outcome of that test. Leaf nodes of the tree represent a class label (decision). The path from the root to leaf nodes represents a classification rule. Different DT based algorithms have been implemented including J48, random tree, random forest, Logistic model trees, and many others. Generally, decision tree classifiers have good accuracy. It is a typical inductive approach to learn knowledge from data. In this study, three DT algorithms were applied. These algorithms include J48, Radom Forest (RF), and Logistic Model Trees (LMT)

4) K-NEAREST NEIGHBOURS (KNN)

As it is a supervised learning algorithm, we will need labelled data for training the model. When classifying a data point, we look at K nearest neighbours of the data the majority will decide which class the data point will belong. Where nearest distance is in form of Euclidean distance, whose formula is

After this, the input data point will be assigned to the class which have the highest probability. The probability can be represented as

In KNN we have to be prudent when selecting the value of K. A smaller value of k usually leads to an irregular decision boundary and high value of k leads a smoother decision boundary.

5) RANDOM FOREST

It is an ensemble model which means it uses many machine learning algorithms to increase its performance as compared to other machine learning algorithms. Random forest randomly picks up a subset from the training data set and generate various decision trees. It will predict the class of test class objects using the decision trees.

7.6 EXPERIMENTAL SETTINGS

The validation approach used in our study to evaluate the performance of different classifiers is the k-fold cross validation approach. This approach was used to generate a test set and avoid the over-fitting problem. The cross-validation approach performs independent tests without requiring separate test data samples and without reducing the data samples used to build prediction models. In the k-fold cross-validation process, the original data is classified into k groups, so that each group is used once as a validation set and the remaining data as a training set. In this study, the 10-fold cross-validation was used to evaluate the performance of different classifiers. Stratified based sampling was used to generate training and testing sets. In each case, a prediction model was trained using 9 folds of the data and tested on the remaining fold and the results were averaged.

7.7 FRONT END

Simply put, the front end of the web-app is what the user experiences. You may be familiar with the terms “HTML, CSS, JS, BOOTSTRAP and PYTHON”; the front end is where the user experiences and interacts with these things. A front-end developer will be focused on what happens with the app, rather than what goes on behind the scenes, which is typically described as the backend. The web-app development process involves cooperation between front end and backend developers in order to finish as a complete web-application. There are typically two types of front-end developments: Native and hybrid. Native front-end developments are typically designed either for the iOS or Android platforms, whereas hybrids use both and can be compatible with all types of operating systems.

7.8 BACK END

The back end of a web-app is meant by the framework that we using and which is responsible for the working and storing of the data of the web-app. We use Django in back end. Django is an open-source framework for backend web applications based on Python one of the top web development languages. Its main goals are simplicity, flexibility, reliability, and scalability. Django has its own naming system for all functions and components. Django is an MVT web framework that is used to build web applications. The huge Django web-framework comes with so many “batteries included” that developers often get amazed as to how everything manages to work together. The principle behind adding so many batteries is to have common web functionalities in the framework itself instead of adding latter as a separate library.

7.9 PERFORMANCE EVALUTION

Once a classifier is trained successfully and a prediction model is generated, performance evaluation measures must be applied on a separate dataset called test set to evaluate the performance of the generated model. In our experiments, the performance of the generated prediction models is evaluated using performance measures. Each measure provides a different perspective of performance evaluation and a broader set of performance results to compare. The accuracy measures how correctly a classifier predicts class labels. It is calculated as the percentage of true positive and true negative rates to the number of all instance.

CHAPTER 8

DESIGN GOAL

Around the world 970 million individuals are affected with mental illness. In a fast growing society, it is a hectic task to keep track on our mental health and wellness. As mental health is as important as someone's physical health it's is each one's responsibility to take care of it. Interpretation of public emotional wellness concerns utilizing learning methods and machine learning techniques by observing sentiments of a person, schedules, emotions which will in turn pave a way to give the user an insight of their mental health.

Our project focus on this issue and give prediction about user's mental health and thus suggest them a review about their mental health status and lend a helping hand nearby by giving opportunity to book to a therapist or counsellor of their taste. High prevalence of mental health and the need for effective mental fitness care, blended with current advances in learning, has led to a growth in explorations of ways the sphere of system getting to know mental health status.

Machine Learning can assist us to dive inside the detection prognosis and treatment of mental health issues. The obtained results after the analysis provide us with the mental health status of user along with the sentimental analysis which is done using comparison with polarity parameters.

CHAPTER 9

RESULTS AND DISCUSSION

This is a web-app for predicting user’s mental health status using Machine Learning and if needed user can have a booking of a therapist. Input is the answers which are given by the user both objective and descriptive type questionnaire. Using Machine Learning Algorithms, our program predicts user’s mental health status. He/she/them can successfully sign up and create their own account. Then they can book their slots. After booking the slots they can attend a set of objective and descriptive questions.

OBJECTIVE QUESTIONS

1. You feel rested

2. You feel that too many demands are being made on you

3. You are irritable or grouchy

4. You have too many things to do

5. You feel lonely or isolated

6. You find yourself in a situation of conflict

7. You feel you are doing things you really like

8. You feel tired

9. You feel you may not manage to attain your goals

10. You feel calm

11. You have too many decisions to make

12. You feel frustrated

13. You are full of energy

14. You feel tense

15. Your problems seem to be piling up

16. You feel you are in a hurry

17. You feel safe and protected

18. You have many worries

19. You are under pressure from other people

20. You feel discouraged

21. You enjoy yourself

22. You are afraid of the future

23. You feel you are doing things because you have to not because you want tor

24. You feel criticized and judged

25. You are light-hearted

26. You feel mentally exhausted

27. You have trouble relaxing

28. You feel loaded down with responsibility

29. You have enough time for yourself

30. You feel under pressure from deadlines

DESCRIPTIVE QUESTIONS

How is your sleep?

How is your energy?

What do you prefer - to stay at home rather than going out and doing new things? Why?

What makes you irritated? Why?

How do you feel about yourself?

What makes you relaxed?

When both the objective and descriptive questions have been answered then the predicted results are available with respect to the responses provided by user. If in need of therapist the booking remains. In the test result, there will be

Sentiment	Negative/positive
Negativity	0-5
Positivity	0-5
Neutral	0-5

Following the test, He/she can connect to nearby therapist or counsellor of their taste. It also provides a daily routine plan for user. Each user's personality data is stored as history.

1. Feature extraction

There is an immense need to analyse and monitor a person's mental health as justified in our previous work. We utilize deep learning feature extraction algorithm like sentiment analysis to analyse mental health of person from a set of questions. Using polarity, the statement to the question is tested as positive or negative.

2. Mental health recognition

Registered patients can attend a survey which is objective and descriptive type. Both the tests are used for analysing the mental health of patient. The answers of the descriptive type questions are used for sentimental analysis and this result is also used for analysis.

3. Mental health classification

The mental health condition is analysed based on the sentiment analysis. Each objective type answer is tested using training set. Positivity is tested based on polarity. If the polarity is above

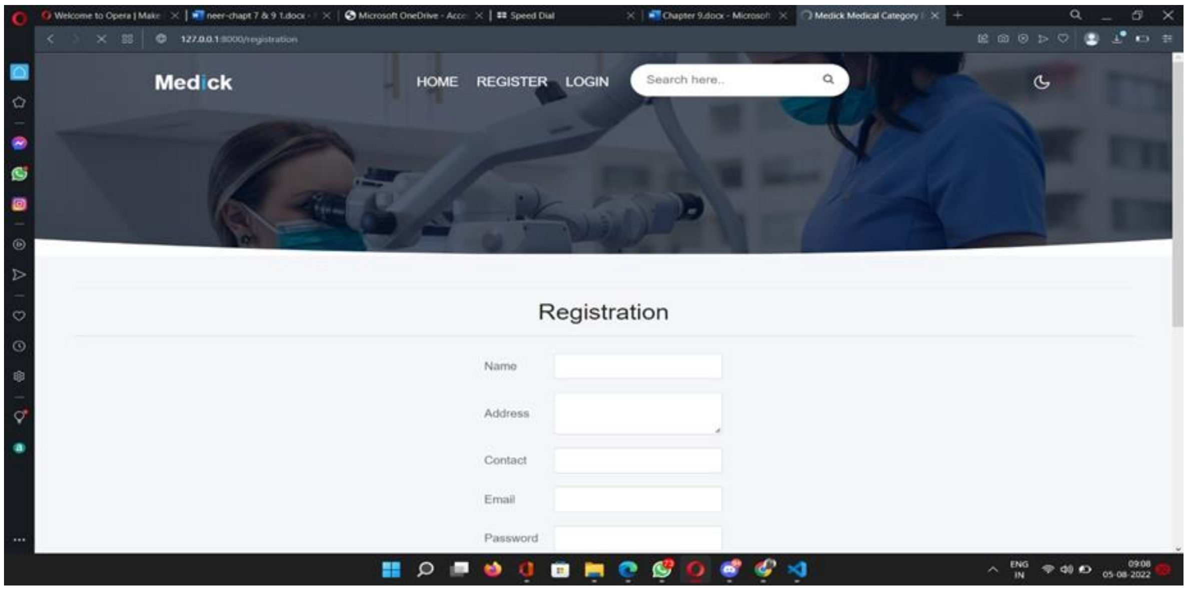
+0.5 the statement is positive, below -0.5 statement is negative and between is neutral statement. The descriptive type statement gives the level of mental condition as high, low and moderate.

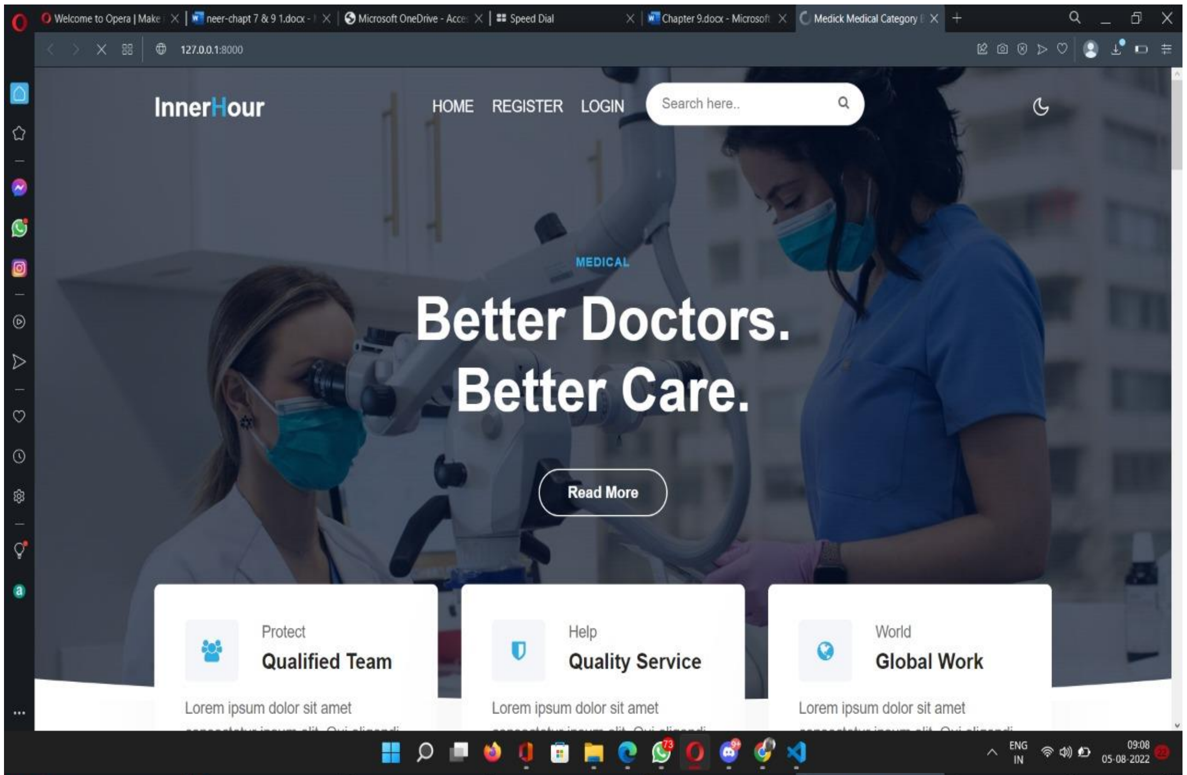
4. Recommendation

There is a set of activities provided irrespective of the mental condition of the user. Based on the test result output well have the sentiment of user and the negativity, positivity and the neutral measures also the mental health status (high/low). On the base of this result user also get a to-do-list for 30 days.

5. Easy handling

Well-designed application that runs in the background and provide the recommendation based on their social media activities.





InnerHour

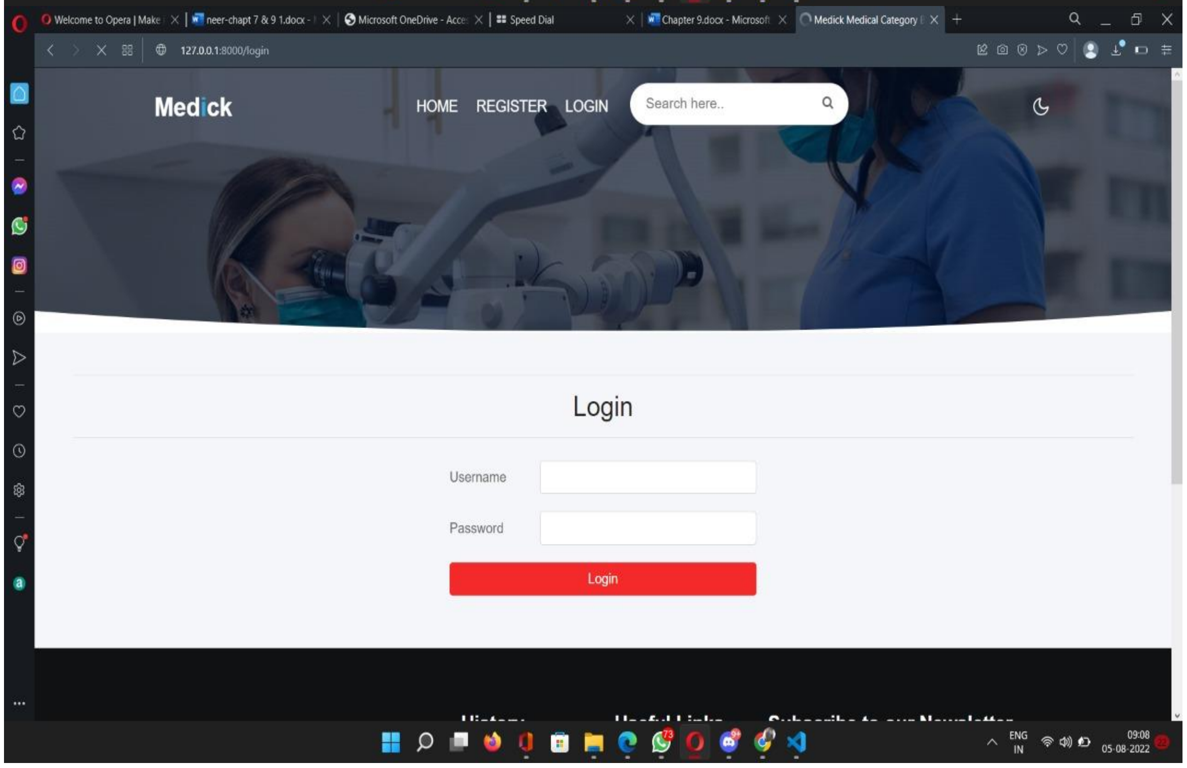
MEDICAL

Better Doctors.
Better Care.

Protect
Qualified Team
Lorem ipsum dolor sit amet

Help
Quality Service
Lorem ipsum dolor sit amet

World
Global Work
Lorem ipsum dolor sit amet



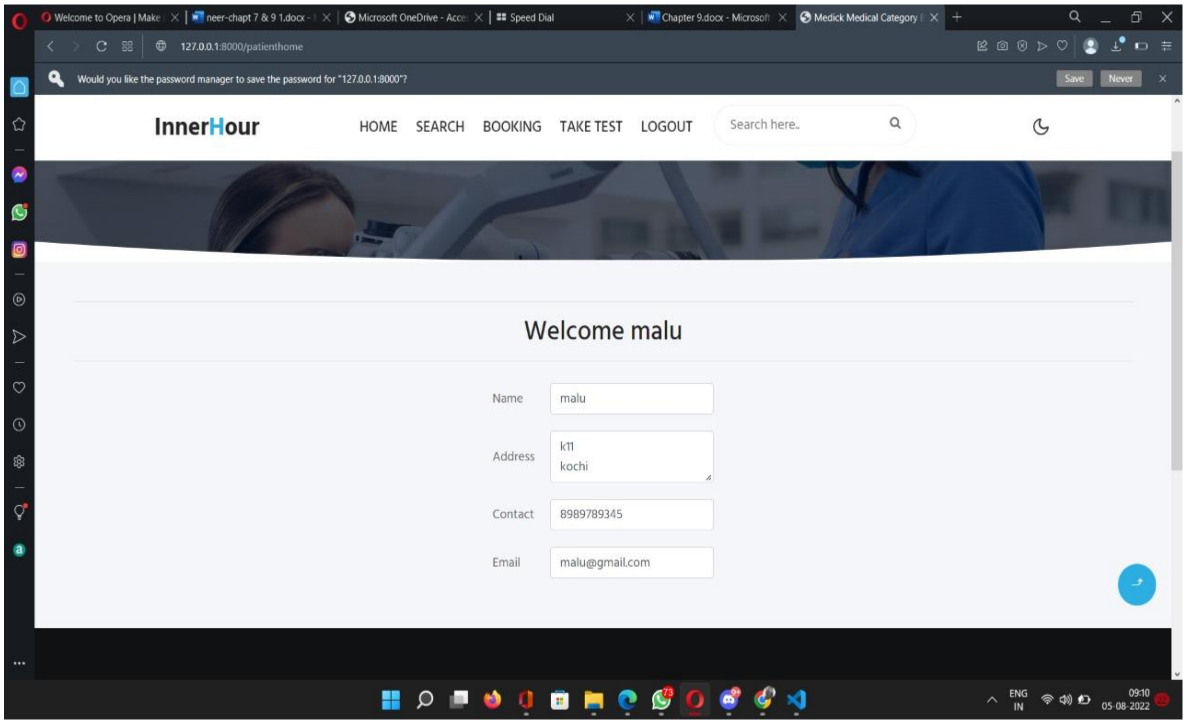
Medick

Login

Username

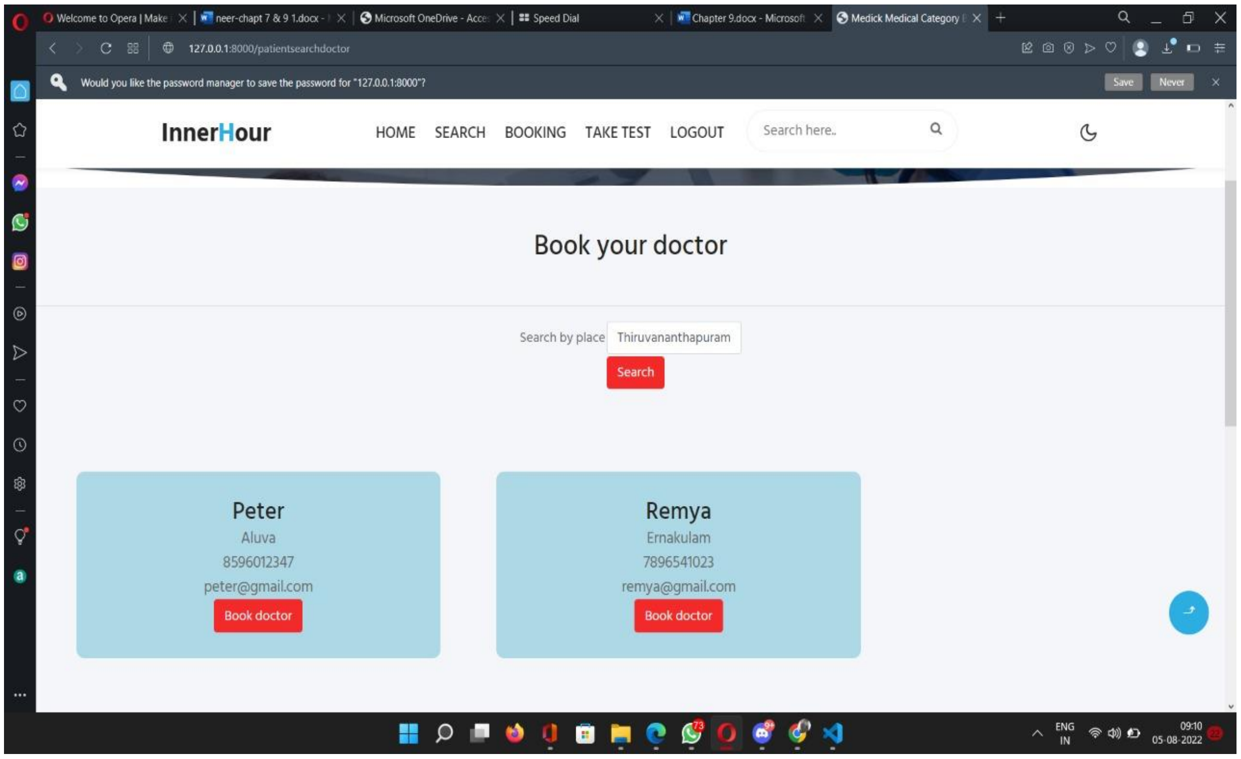
Password

Login



Welcome malu

Name	malu
Address	k11 kochi
Contact	8989789345
Email	malu@gmail.com



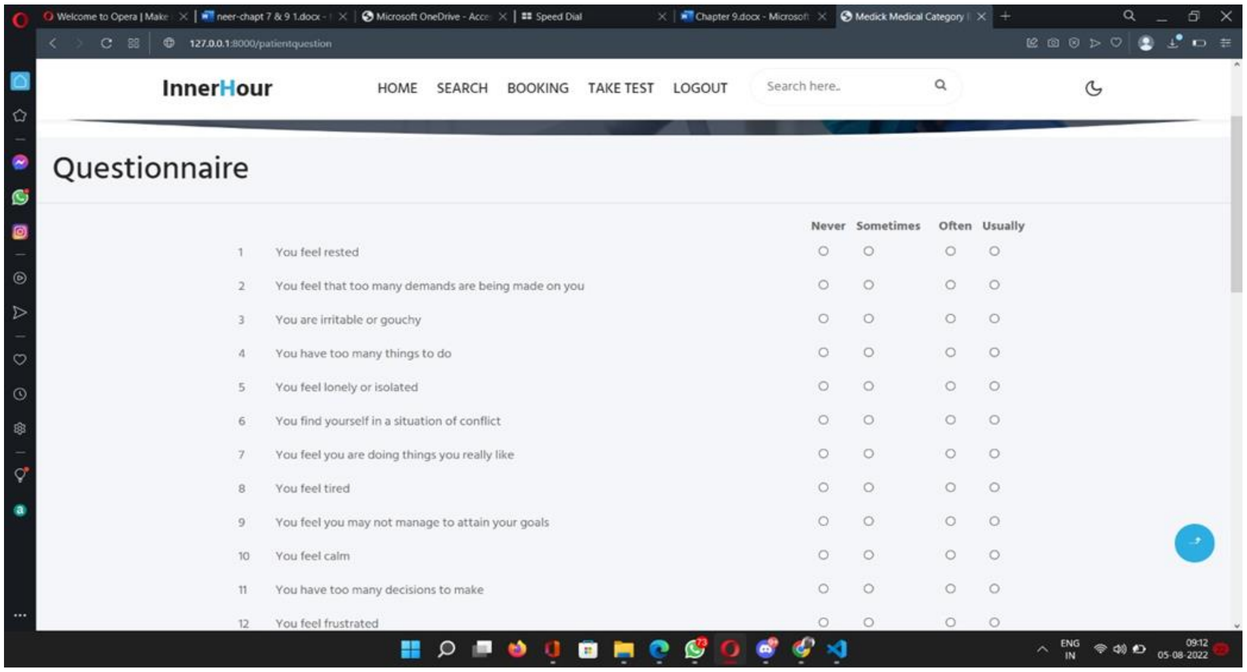
Book your doctor

Search by place: Thiruvananthapuram

Search

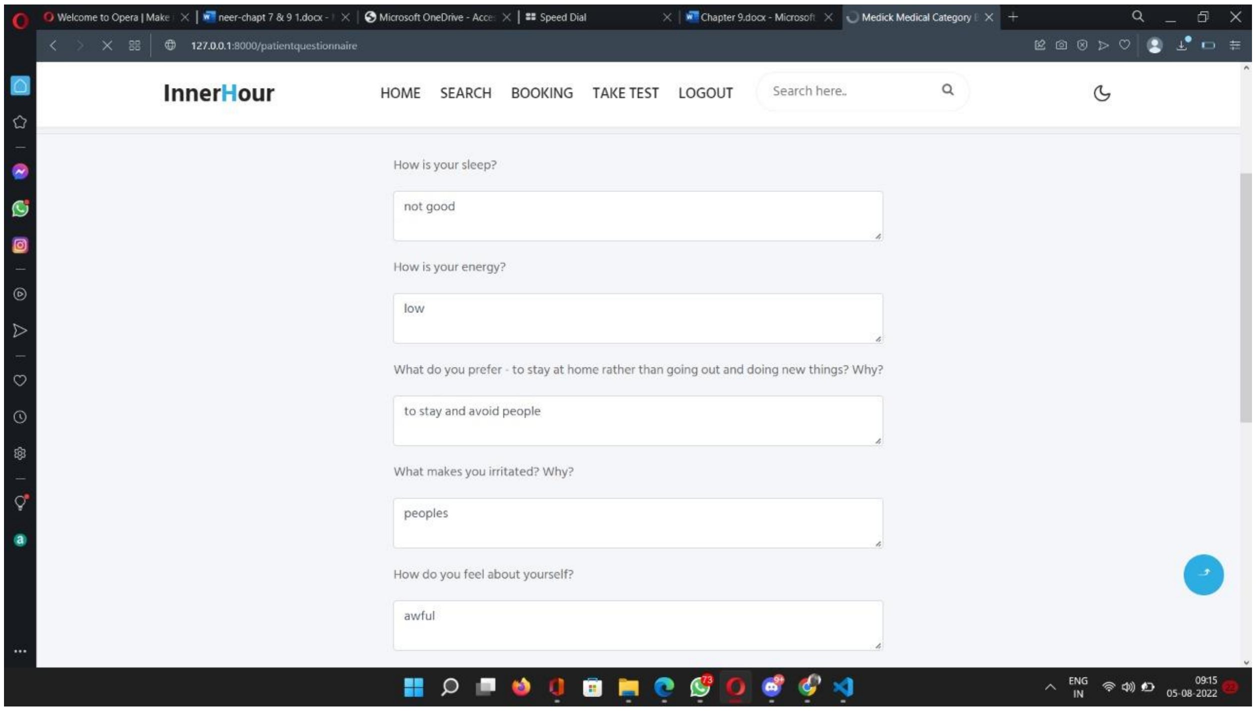
Peter
Aluva
8596012347
peter@gmail.com
Book doctor

Remya
Ernakulam
7896541023
remya@gmail.com
Book doctor



Questionnaire

		Never	Sometimes	Often	Usually
1	You feel rested	○	○	○	○
2	You feel that too many demands are being made on you	○	○	○	○
3	You are irritable or gouchy	○	○	○	○
4	You have too many things to do	○	○	○	○
5	You feel lonely or isolated	○	○	○	○
6	You find yourself in a situation of conflict	○	○	○	○
7	You feel you are doing things you really like	○	○	○	○
8	You feel tired	○	○	○	○
9	You feel you may not manage to attain your goals	○	○	○	○
10	You feel calm	○	○	○	○
11	You have too many decisions to make	○	○	○	○
12	You feel frustrated	○	○	○	○



How is your sleep?

not good

How is your energy?

low

What do you prefer - to stay at home rather than going out and doing new things? Why?

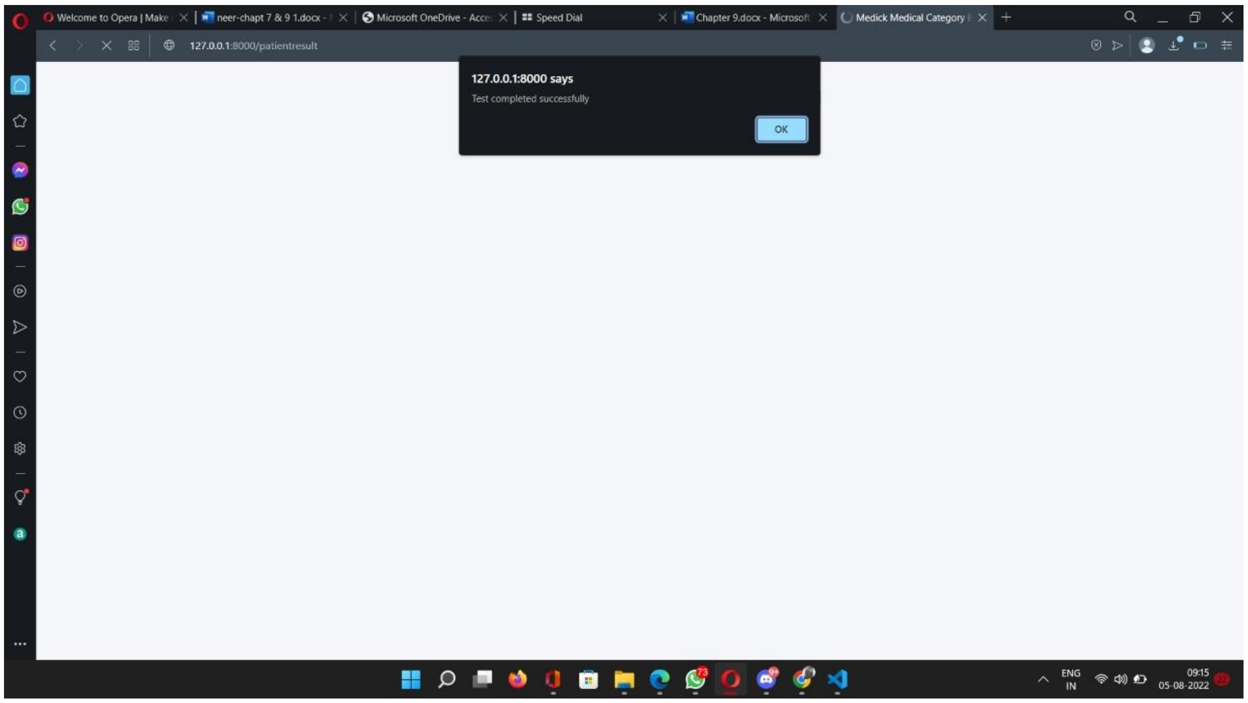
to stay and avoid people

What makes you irritated? Why?

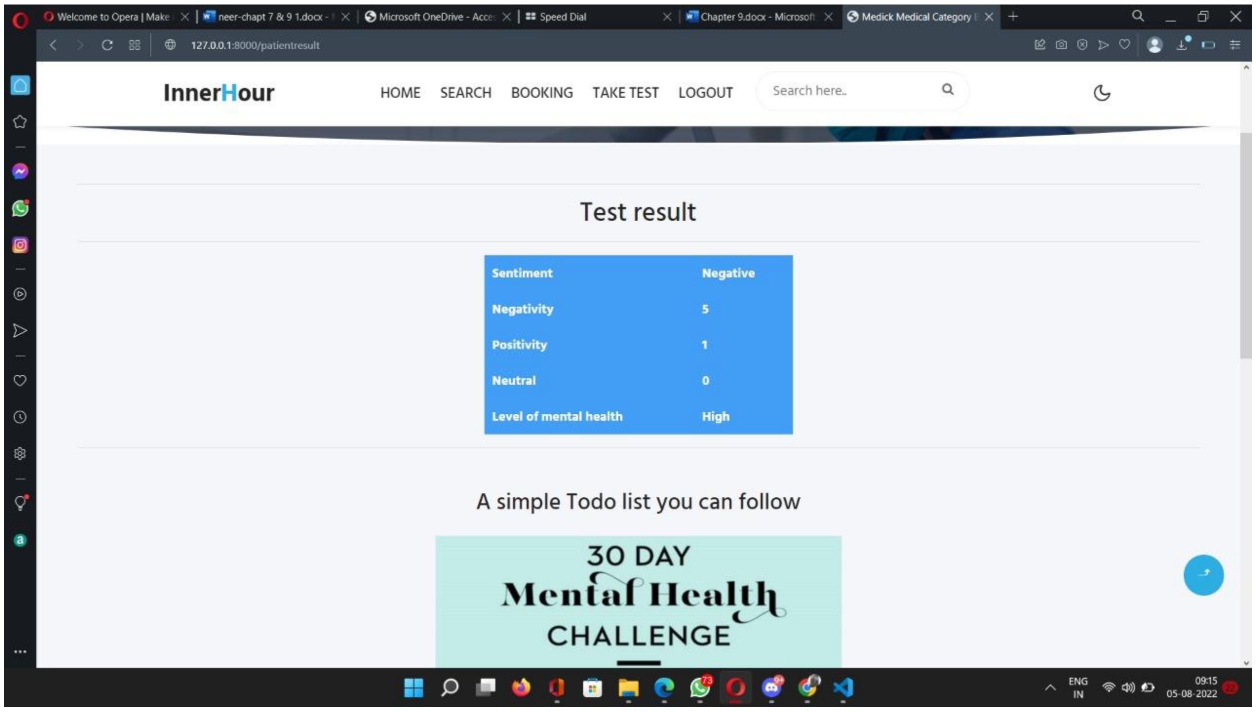
peoples

How do you feel about yourself?

awful


127.0.0.1:8000 says

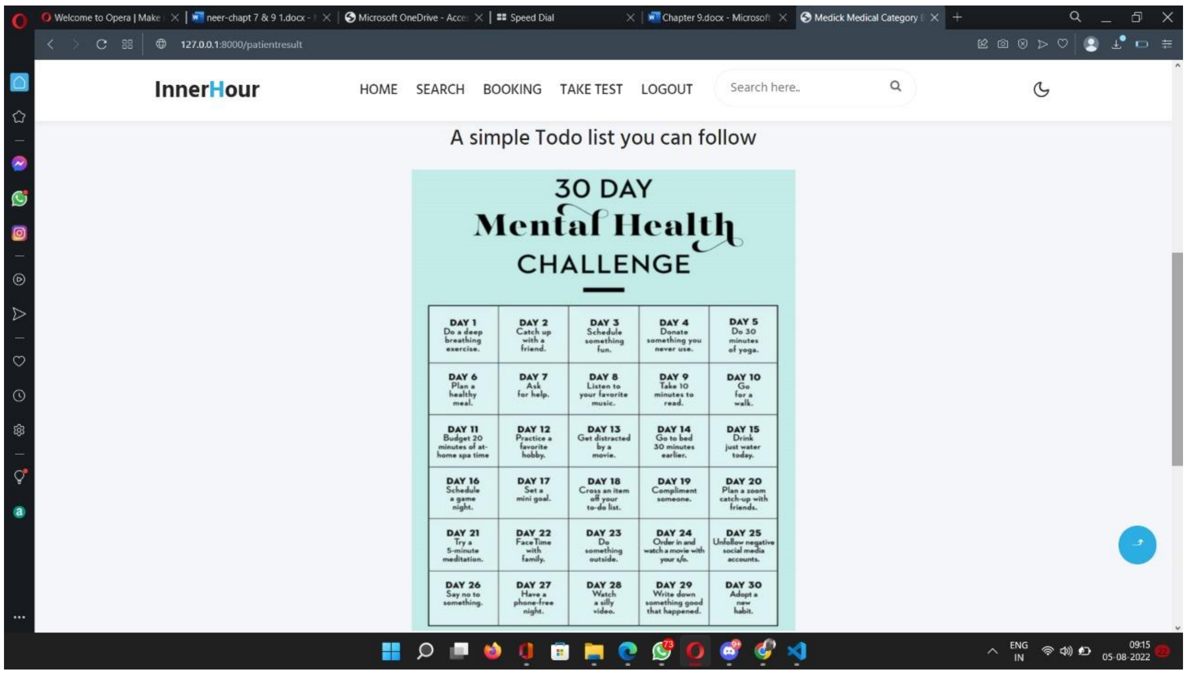
Test completed successfully



Test result

Sentiment	Negative
Negativity	5
Positivity	1
Neutral	0
Level of mental health	High

A simple Todo list you can follow



A simple Todo list you can follow

30 DAY Mental Health CHALLENGE

DAY 1 Do a deep breathing exercise.	DAY 2 Catch up with a friend.	DAY 3 Schedule something fun.	DAY 4 Donate something you never use.	DAY 5 Do 30 minutes of yoga.
DAY 6 Plan a healthy meal.	DAY 7 Ask for help.	DAY 8 Listen to your favorite music.	DAY 9 Take 10 minutes to read.	DAY 10 Go for a walk.
DAY 11 Budget 20 minutes of at-home spa time	DAY 12 Practice a favorite hobby.	DAY 13 Get distracted by a movie.	DAY 14 Go to bed 30 minutes earlier.	DAY 15 Drink just water today.
DAY 16 Schedule a game night.	DAY 17 Set a mini goal.	DAY 18 Cross an item off your to-do list.	DAY 19 Compliment someone.	DAY 20 Plan a zoom catch-up with friends.
DAY 21 Try a 5-minute meditation.	DAY 22 FaceTime with family.	DAY 23 Do something outside.	DAY 24 Order in and watch a movie with your s/o.	DAY 25 Unfollow negative social media accounts.
DAY 26 Say no to something.	DAY 27 Have a phone-free night.	DAY 28 Watch a silly video.	DAY 29 Write down something good that happened.	DAY 30 Adopt a new habit.

CHAPTER 10

CONCLUSION AND FUTURE SCOPE

Many different techniques and algorithms had been introduced and proposed to test and solve the mental health problems of user. There are still many solutions that can be refined .In addition, there are still many problems to be discover and tested using a wide variety of settings in machine learning for the mental health domain. As classifying the mental health data is generally a very challenging problem, the features used in the machine learning algorithms will significantly affect the performance of the classification. The existing studies and research show that machine learning can be a useful tool in helping understand psychiatric disorders. Besides that, it may also help distinguish and classify the mental health problems among patients for further treatment. Newer approaches that use data that arise from the integration of various sensor modalities present in advanced devices have proven to be a convenient resource to recognize the mood state and responses from patients among others. It is noticeable that most of the research and studies are still struggling to validate the results because of insufficiency of acceptable validated evidence, especially from the external sources. Besides that, most of the machine learning techniques might not have the same performance across all the problems. The performance of the machine learning models will vary depending on the data samples obtained and the features of the data. Moreover, machine learning models can also be affected by preprocessing activities such as data cleaning and parameter tuning in order to achieve optimal results. Hence, it is very important for researchers to investigate and analyze the data with various machine learning algorithms to choose the highest accuracy among the machine learning algorithms. Not only that, challenges and limitations faced by the researchers need to be managed with proper care to achieve satisfactory results that could improve the clinical practice and decision making.

REFERENCES

1] Rahul Katarya .,Saurav Maan .,"Predicting Mental health disorders using Machine Learning for employees in technical and non-technical companies", 2020 IEEE INTERNATIONAL CONFERENCE ON ADVANCES AND DEVELOPMENTS IN ELECTRICAL AND ELECTRONICS ENGINEERING (ICADEE 2020)

[2] A. Grünerbl et al., “Smartphone-based recognition of states and state changes in bipolar disorder patients,” IEEE J. Biomed. Heal. Informatics, vol. 19, no. 1, pp. 140–148, 2015, Doi: 10.1109/JBHI.2014.2343154.

[3] U. Ha et al., “A Wearable EEG-HEG-HRV Multimodal System with Simultaneous Monitoring of Tes for Mental Health Management,” IEEE Trans. Biomed. Circuits Syst., vol. 9, no. 6, pp. 758– 766, 2015, Doi: 10.1109/TBCAS.2015.2504959.

[4] Yadav, S., Kaim, T., Gupta, S., Bharti, U., & Priyadarshi, P. (2020). "Predicting Depression From Routine Survey Data using Machine Learning". IEEE 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN).

[5] U. S. Reddy, A. V. Thota, and A. Dharun, “Machine Learning Techniques for Stress Prediction in Working Employees,” 2018 IEEE Int. Conf. Compute . Intell. Compute . Res. ICCIC 2018, Doi: 10.1109/ICCIC.2018.8782395

CODE

```
import re

from django.shortcuts import render, redirect

from .models import Registration, Responsemaster, Therapist, Booking, Question,Responsechild, Questionnaire,Survey,Medicalhistory

from django.contrib.auth import authenticate

from django.contrib.auth.models import User

from django.contrib import messages

import json

import pandas as pd

import time

import numpy as np

import itertools

import matplotlib.pyplot as plt

from sklearn.metrics import confusion_matrix

from sklearn.feature_extraction.text import CountVectorizer

from sklearn import metrics

#from sklearn.metrics import roc_auc_score


tweets_data = []

x = []

y = []

vectorizer = CountVectorizer(stop_words='english')
```

```
def retrieveTweet(data_url):
 tweets_data_path = data_url
    tweets_file = open(tweets_data_path, "r")
    for line in tweets_file:
        try:
            tweet = json.loads(line)
            tweets_data.append(tweet)
        except:
            continue
def retrieveProcessedData(Pdata_url):
    sent = pd.read_excel(Pdata_url)
    for i in range(len(tweets_data)):
        if tweets_data[i]['id']==sent['id'][i]:
            x.append(tweets_data[i]['text'])
            y.append(sent['sentiment'][i])

def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)
fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

 def nbTrain():
    from sklearn.naive_bayes import MultinomialNB
    start_timenb = time.time()
    train_features = vectorizer.fit_transform(x)
       actual = y
    nb = MultinomialNB()
    nb.fit(train_features, [int(r) for r in y])

    test_features = vectorizer.transform(x)
    predictions = nb.predict(test_features)
    fpr, tpr, thresholds = metrics.roc_curve(actual, predictions, pos_label=1)
    nbscore = format(metrics.auc(fpr, tpr))
    nbscore = float(nbscore)*100
    nb_matrix = confusion_matrix(actual, predictions)
    plt.figure()
    plot_confusion_matrix(nb_matrix, classes=[-1,0,1], title='Confusion matrix For NB classifier')

    print("\n")
    print("Naive Bayes  Accuracy : \n", nbscore,"%")
    print(" Completion Speed", round((time.time() - start_timenb),5))
    print()
def datree():
    from sklearn import tree
    start_timedt = time.time()
    train_featurestree = vectorizer.fit_transform(x)
    actual1 = y
    test_features1 = vectorizer.transform(x)
    dtree = tree.DecisionTreeClassifier()

    dtree = dtree.fit(train_featurestree, [int(r) for r in y])

    prediction1 = dtree.predict(test_features1)
    ddd, ttt, thresholds = metrics.roc_curve(actual1, prediction1, pos_label=1)
    dtreescore = format(metrics.auc(ddd, ttt))
    dtreescore = float(dtreescore)*100
    print("Decision tree Accuracy : \n", dtreescore, "%")
    print(" Completion Speed", round((time.time() - start_timedt),5))
    print()

def Tsvm():
    from sklearn.svm import SVC
    start_timesvm = time.time()
    train_featuressvm = vectorizer.fit_transform(x)
    actual2 = y
    test_features2 = vectorizer.transform(x)
    svc = SVC()

    svc = svc.fit(train_featuressvm, [int(r) for r in y])
    prediction2 = svc.predict(test_features2)
    sss, vvv, thresholds = metrics.roc_curve(actual2, prediction2, pos_label=1)
    svc = format(metrics.auc(sss, vvv))
    svc = float(svc)*100
    print("Support vector machine Accuracy : \n", svc, "%")
    print(" Completion Speed", round((time.time() - start_timesvm),5))
    print()

def knN():
    from sklearn.neighbors import KNeighborsClassifier
    start_timekn = time.time()
    train_featureskn = vectorizer.fit_transform(x)
    actual3 = y
    test_features3 = vectorizer.transform(x)
    kn = KNeighborsClassifier(n_neighbors=2)



    kn = kn.fit(train_featureskn, [int(i) for i in y])
    prediction3 = kn.predict(test_features3)
    kkk, nnn, thresholds = metrics.roc_curve(actual3, prediction3, pos_label=1)
    kn = format(metrics.auc(kkk, nnn))
    kn = float(kn)*100

    print("Kneighborsclassifier Accuracy : \n", kn, "%")
    print(" Completion Speed", round((time.time() - start_timekn),5))
    print()

def RanFo():
    from sklearn.ensemble import RandomForestClassifier
    start_timerf = time.time()
    train_featuresrf = vectorizer.fit_transform(x)
    actual4 = y
    test_features4 = vectorizer.transform(x)
    rf = RandomForestClassifier(max_depth=2, random_state=0)



    rf = rf.fit(train_featuresrf, [int(i) for i in y])
    prediction4 = rf.predict(test_features4)
    rrr, fff, thresholds = metrics.roc_curve(actual4, prediction4, pos_label=1)
    kn = format(metrics.auc(rrr, fff))
    kn = float(kn)*100
    print("Random Forest Accuracy : \n", kn, "%")
    print(" Completion Speed", round((time.time() - start_timerf),5))
    print()
    print()



def runall():
    retrieveTweet('data/tweetdata.txt')
    retrieveProcessedData('processed_data/output.xlsx')
    nbTrain()
    datree()
    Tsvm()
    knN()
    RanFo()

def datreeINPUT(inputtweet):
    from sklearn import tree
    train_featurestree = vectorizer.fit_transform(x)
    dtree = tree.DecisionTreeClassifier()

    dtree = dtree.fit(train_featurestree, [int(r) for r in y])
    inputdtree= vectorizer.transform([inputtweet])
    predictt = dtree.predict(inputdtree)


    if predictt == 1:
        predictt = "Positive"
    elif predictt == 0:
        predictt = "Neutral"
    elif predictt == -1:
        predictt = "Negative"
    else:
        print("Nothing")


    print("\n*****************")
    print(predictt)
    print("*****************")
    return predictt
runall()
```

LOAD INDEX PAGE

```
#######################################################################


def index(request):
    """
        The function to load index page of the project.
        Parameters:
            HTTP request
          Returns:
            html page
    """
    return render(request, 'index.html')
#######################################################################
                              LOAD LOGIN PAGE
#######################################################################

def login(request):
    """
        The function to load index page of the project.
        -----------------------------------------------
        Parameters:
            HTTP request
     Returns:
            html page
    """
    if request.POST:
        # read the username and password given in UI
        email = request.POST['txtEmail']
        pwd = request.POST['txtPassword']
        # checking whether the client is admin
    # checking whether username and email exist in authenticate table
        user = authenticate(username=email, password=pwd)
        if user is None:
            #username or password is incorrect
            messages.info(request, 'Username or password incorrect')
        else:
            #username and password is correct
            user_data = User.objects.get(username=email)
            if user_data.is_superuser == 1:
                # if admin, goto admin interface
                return redirect("/adminhome")
            else:
                # if user, go to user interface
                if user_data.is_staff == 0:
                    request.session["email"] = email
                    r = Registration.objects.get(email=email)
                    request.session["id"] = r.id
                    request.session["name"] = r.name
                    return redirect("/patienthome")
                elif user_data.is_staff == 1:
                    request.session["email"] = email
                    r = Therapist.objects.get(email=email)
                    request.session["id"] = r.id
                    request.session["name"] = r.name
```

```
                    return redirect("/therapisthome")
                    return render(request, 'login.html')
######################################################################
                              LOAD REGISTRATION PAGE
######################################################################
def registration(request):
    """
        The function to load registration page of the project.
        -----------------------------------------------------
        Parameters:
            HTTP request
         Returns:
            html page
    """
    if request.POST:
        # read the values from UI
        name = request.POST['txtName']
        address = request.POST['txtAddress']
        phone = request.POST['txtContact']
        email = request.POST['txtEmail']
        pwd = request.POST['txtPassword']
        # check whether any duplicate entries occur
        user = authenticate(username=email, password=pwd)
        if user is None:
            # if no duplicate entries add the data to registration table
            try:
                r = Registration.objects.create(
                    name=name, address=address, phone=phone, email=email)
                r.save()
            except:
                messages.info(request, 'Sorry some error occured')
            else:
                # add the data to login table also
                try:
                    u = User.objects.create_user(
                        password=pwd,    username=email,    is_superuser=0,    is_active=1,    is_staff=0,
email=email)
                    u.save()
                except:
                    messages.info(request, 'Sorry some error occured')
                else:
                    messages.info(request, 'Registration successfull')
        else:
            # duplicate entries occur and registration is not possible
            messages.info(request, 'User already registered')
    return render(request, 'registration.html')
######################################################################
                                LOADADMINHOMEPAGE
######################################################################
```

```
def adminhome(request):
    """
        The function to load admin page of the admin.
        ----------------------------------------------
        Parameters:
            HTTP request
        Returns:
            html page
    """
    return render(request, "adminhome.html")
#######################################################################
                                    LOAD PATIENT PAGE
#######################################################################
def adminpatient(request):
    """
        The function to view patients.
        ------------------------------------------------------
        Parameters:
            HTTP request
         Returns:
            html page
    """
    # fetch and load all therapist from database
    patient_data = Registration.objects.all()
    return render(request, 'adminpatient.html', {"patient": patient_data})
######################################################################
                          LOAD DOCTOR HOME PAGE
######################################################################
def therapisthome(request):
    """
        The function to load index page of the therapist.
        -----------------------------------------------
        Parameters:
            HTTP request
        Returns:
            html page
    """
    return render(request, "therapisthome.html")
######################################################################
                          LOAD DOCTOR BOOKING PAGE
######################################################################
def therapistbooking(request):
    """
        The function to load booking page of the therapist.
        -----------------------------------------------
        Parameters:
            HTTP request
        Returns:
            html page
    """
    id=request.session["id"]
    rid=Therapist.objects.get(id=id)
    booking=Booking.objects.filter(therapistid=rid)
    return render(request, "therapistbooking.html",{"booking":booking})
##########################################################################
                          LOAD DOCTOR PATIENT PAGE
##########################################################################
def therapistpatient(request):
    """
        The function to load patient page of the therapist.
        ---------------------------------------------
       Parameters:
            HTTP request
        Returns:
            html page
    """
    id=request.GET.get("id")
    booking=Booking.objects.get(id=id)
    responsemaster=Responsemaster.objects.get(bookingid=id)
    print(responsemaster)
    rid=responsemaster.id
    print(rid)
    responsechild=Responsechild.objects.filter(respid=rid)
    survey=Survey.objects.get(bookingid=id)
    questionnaire=Questionnaire.objects.filter(surveyid=survey.id)
    return
render(request,      "therapistpatient.html",{"booking":booking,"responsemaster":responsemaster,
"survey":survey,"questionnaire":questionnaire,"responsechild":responsechild})
##########################################################################
                                LOAD PATIENT HOME PAGE
##########################################################################

def patienthome(request):
    """
        The function to load index page of the patient.
        ---------------------------------------------
        Parameters:
            HTTP reques
         Returns:
            html page
    """
    id = request.session["id"]
    profile_data = Registration.objects.filter(id=id)
    return render(request, "patienthome.html", {"profile_data": profile_data})
##########################################################################
                          LOAD PATIENT QUESTION PAGE
##########################################################################
```

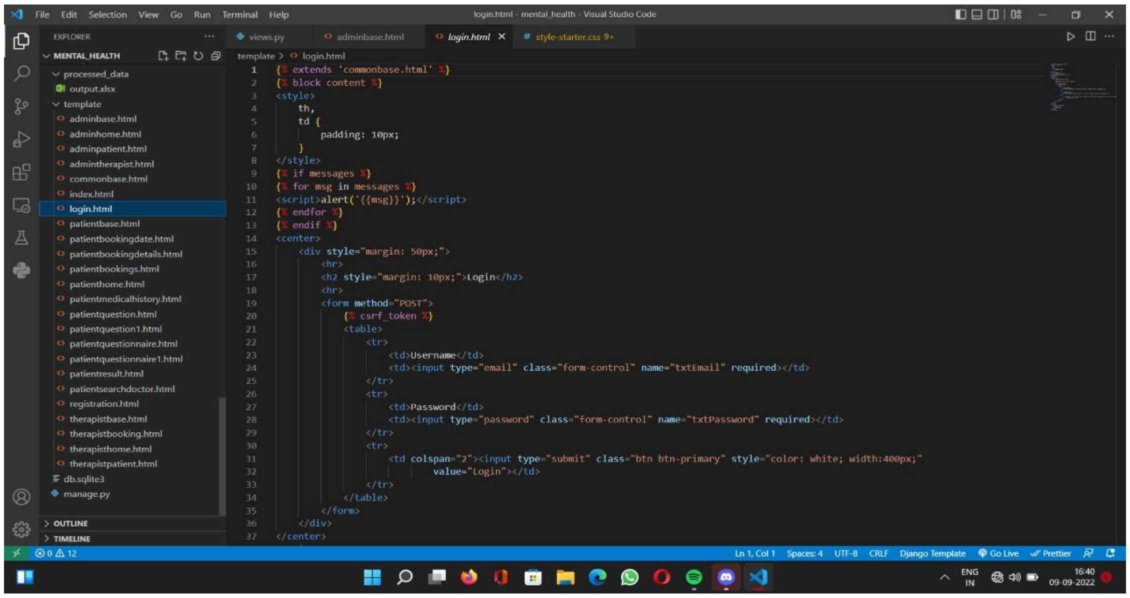
```
def patientquestion(request):
    """
        The function to load question page of the patient.
        ----------------------------------------------
        Parameters:
            HTTP request
        Returns:
            html page
    """
    id = request.session["bookingid"]
    print(id)
    regid= Booking.objects.get(id=id)
    question = Question.objects.all()
    print(question)
    if request.POST:
        raw_score=0
        rr=Responsemaster.objects.create(bookingid=regid)
        rr.save()
        print(rr)
        res=Responsemaster.objects.get(id=rr.id)
        r=res.id
        print(r)
        for i in question:
                print(i.id)
                nm="point"+str(i.id)
                name=request.POST[nm]
                print(name)
                res=Responsechild.objects.create(respid=rr,question=i.question,response=name)
                res.save()
                if i.id==1 or i.id==7 or i.id==10 or i.id==13 or i.id==17 or i.id==21 or i.id==25 or
i.id==29:
                if name=="1":
                    name="4"
                elif name=="2":
                    name="3"
                elif name=="3":
                    name="2"
                elif name=="4":
                    name="1"
                raw_score+=int(name)
                print(raw_score)
                psq=(raw_score-30)/90
                print(psq)
        level=""
        if psq>=0 and psq<0.34:
            level="Low"
        elif psq>=0.34 and psq<=0.46:
            level="Moderate"
        elif psq>0.46:
            level="High"
        print(level)
        resp=Responsemaster.objects.get(id=r)
        resp.level=level
        resp.save()
        request.session["level"]=level
        return redirect("/patientquestionnaire")
    return render(request, "patientquestion.html", {"question": question})
######################################################################
#                         LOAD PATIENT QUESTION PAGE
######################################################################
def patientquestion1(request):
    """
        The function to load question page of the patient.
        ---------------------------------------------
        Parameters:
            HTTP request
        Returns:
            html page
    """

    question = Question.objects.all()
    print(question)
    if request.POST:
        raw_score=0

        for i in question:
                print(i.id)
                nm="point"+str(i.id)
                name=request.POST[nm]
                print(name)

                if i.id==1 or i.id==7 or i.id==10 or i.id==13 or i.id==17 or i.id==21 or i.id==25 or i.id==29:
                        if name=="1":
                            name="4"
                        elif name=="2":
                            name="3"
                        elif name=="3":
                            name="2"
                        elif name=="4":
                            name="1"
                raw_score+=int(name)
                print(raw_score)
                psq=(raw_score-30)/90
                print(psq)
        level=""
        if psq>=0 and psq<0.34:
            level="Low"
        elif psq>=0.34 and psq<=0.46:
            level="Moderate"
        elif psq>0.46:
            level="High"
        print(level)
        request.session["level"]=level
        return redirect("/patientquestionnaire1")
    return render(request, "patientquestion1.html", {"question": question})
########################################################################
#                          LOAD PATIENT QUESTIONNAIRE
########################################################################
def patientquestionnaire1(request):
    """
        The function to load questionnaire
        ---------------------------------------------
        Parameters:
            HTTP request
        Returns:
            html page
    """
    if request.POST:
        sleep=request.POST.get("txtSleep")
```

```
 energy=request.POST.get("txtEnergy")
 prefer=request.POST.get("txtPrefer")
 irritate=request.POST.get("txtIrritate")
 yourself=request.POST.get("txtYourself")
 relaxed=request.POST.get("txtRelaxed")
 pos=0
 neu=0
 neg=0
 sleepsenti=datreeINPUT(sleep)
 print(sleepsenti)

 if sleepsenti=="Positive":
     pos+=1
 elif sleepsenti=="Negative":
     neg+=1
 elif sleepsenti=="Neutral":
     neu+=1
if datreeINPUT(energy)=="Positive":
     pos+=1
 elif datreeINPUT(energy)=="Negative":
     neg+=1
 elif datreeINPUT(energy)=="Neutral":
     neu+=1
 if datreeINPUT(prefer)=="Positive":
     pos+=1
 elif datreeINPUT(prefer)=="Negative":
     neg+=1
 elif datreeINPUT(prefer)=="Neutral":
     neu+=1
if datreeINPUT(irritate)=="Positive":
     pos+=1
 elif datreeINPUT(irritate)=="Negative":
     neg+=1
 elif datreeINPUT(irritate)=="Neutral":
     neu+=1

 if datreeINPUT(yourself)=="Positive":
     pos+=1
 elif datreeINPUT(yourself)=="Negative":
     neg+=1
 elif datreeINPUT(yourself)=="Neutral":
     neu+=1

  if datreeINPUT(relaxed)=="Positive":
     pos+=1
 elif datreeINPUT(relaxed)=="Negative":
     neg+=1
 elif datreeINPUT(relaxed)=="Neutral":
```

```
            neu+=1
        print(pos,neg,neu)
          big=""
         if int(pos)>int(neu) and int(pos)>int(neg) :
            big="Positive"
         elif int(neu)>int(pos) and int(neu)>int(neg):
            big="Neutral"
         else:
            big="Negative"
         print(big)
         request.session["sentiment"]=big
         request.session["positive"]=pos
         request.session["negative"]=neg
         request.session["neutral"]=neu
         messages.info(request, 'Test completed successfully')
         return redirect("/patientresult")
         return render(request, "patientquestionnaire1.html")
#########################################################################
                         LOAD PATIENT BOOKING
#########################################################################
def patientbookings(request):
    """
        The function to view booking
        ---------------------------------------------
        Parameters:
            HTTP request
        Returns:
            html page
    """
    id = request.session["id"]
    rid=Registration.objects.get(id=id)
    booking=Booking.objects.filter(regid=rid)
    return render(request, "patientbookings.html",{"booking":booking})
######################################################################
#                        LOAD PATIENT SEARCH PAGE
######################################################################
def patientbookingdetails(request):
    """
        The function to search page of the patient
        ---------------------------------------------
        Parameters:
            HTTP request
        Returns:
            html page
    """
    id=request.GET.get("id")
    booking=Booking.objects.get(id=id)
    responsemaster=Responsemaster.objects.get(bookingid=id)
    print(responsemaster)
    rid=responsemaster.id
    print(rid)
    responsechild=Responsechild.objects.filter(respid=rid)
    survey=Survey.objects.get(bookingid=id)
    questionnaire=Questionnaire.objects.filter(surveyid=survey.id)
    return render(request,
"patientbookingdetails.html",{"booking":booking,"responsemaster":responsemaster,
"survey":survey,"questionnaire":questionnaire,"responsechild":responsechild})
######################################################################
#                   LOAD PATIENT MEDICAL HISTORY
######################################################################
def patientmedicalhistory(request):
    """
        The function to load medical history
        ---------------------------------------------
        Parameters:
            HTTP request
        Returns:
            html page
    """
    if request.POST:
        history=request.POST.get("txtHistory")
        heridity=request.POST.get("txtHeridity")
        try:
            id = request.session["bookingid"]

            regid= Booking.objects.get(id=id)
            s=Medicalhistory.objects.create(bookingid=regid,history=history,heriditary=heridity)
            s.save()
        except:
            messages.info(request, 'Sorry some error occured')
        else:
            messages.info(request, 'Booking successfully created')
            return redirect('/patientresult')
    return render(request, 'patientmedicalhistory.html')
```

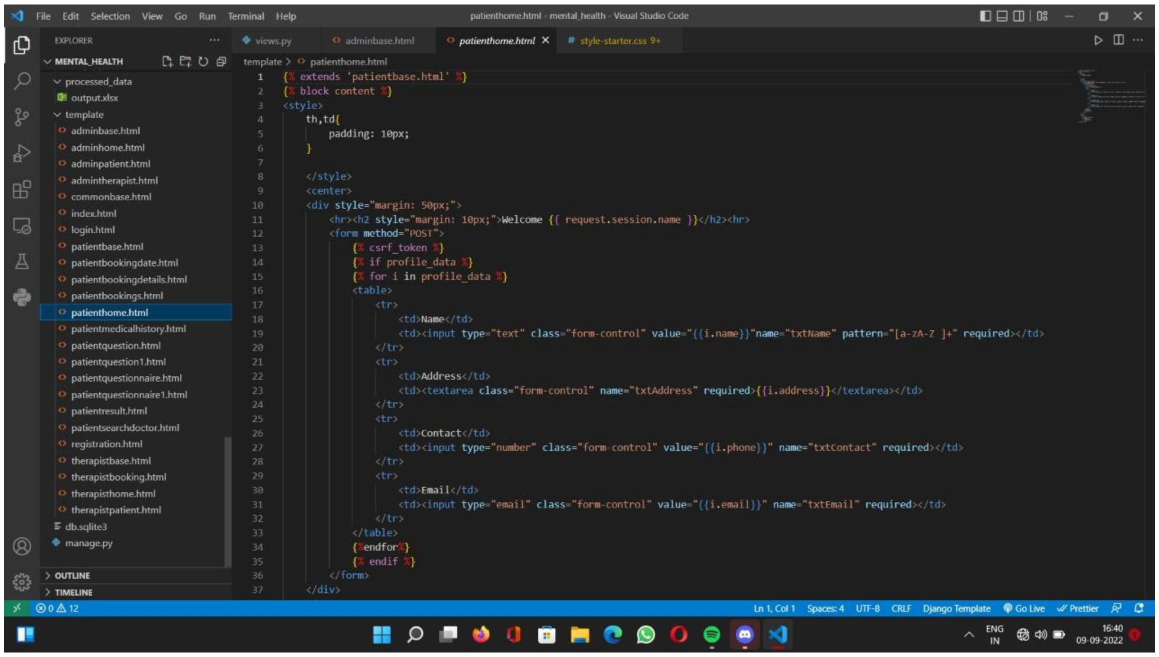
APPENDIX

1. LOGIN PAGE



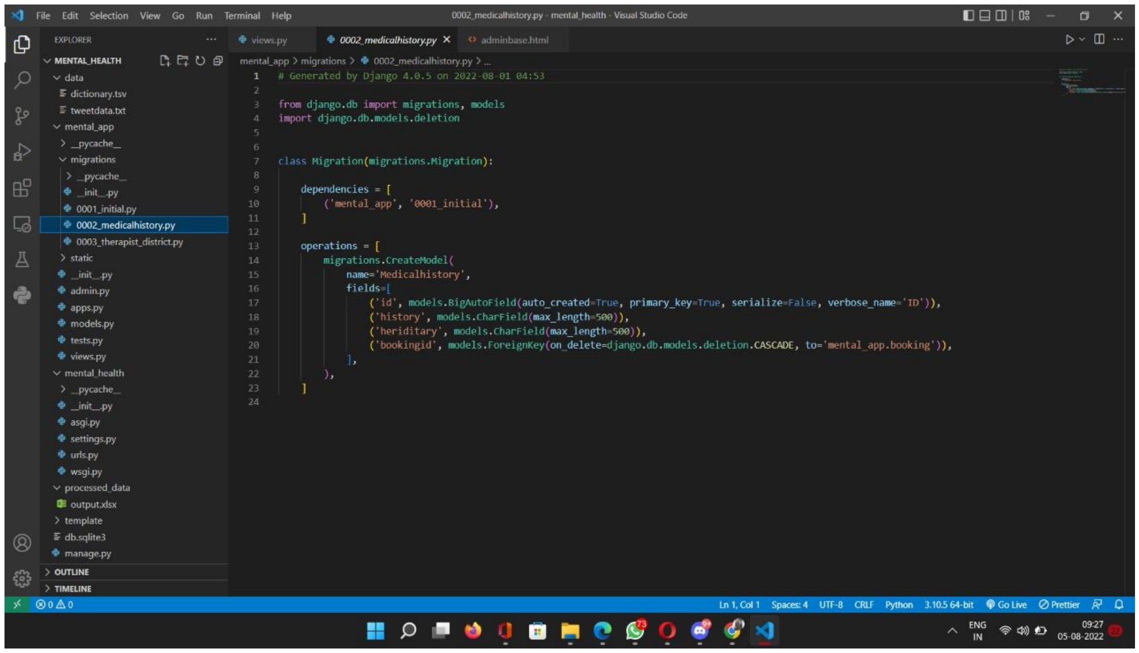
```
{% extends 'commonbase.html' %}
{% block content %}
<style>
    th,
    td {
        padding: 10px;
    }
</style>
{% if messages %}
{% for msg in messages %}
<script>alert('{{msg}}');</script>
{% endfor %}
{% endif %}
<center>
    <div style="margin: 50px;">
        <hr>
        <h2 style="margin: 10px;">Login</h2>
        <hr>
        <form method="POST">
            {% csrf_token %}
            <table>
                <tr>
                    <td>Username</td>
                    <td><input type="email" class="form-control" name="txtEmail" required></td>
                </tr>
                <tr>
                    <td>Password</td>
                    <td><input type="password" class="form-control" name="txtPassword" required></td>
                </tr>
                <tr>
                    <td colspan="2"><input type="submit" class="btn btn-primary" style="color: white; width:400px;"
                            value="Login"></td>
                </tr>
            </table>
        </form>
    </div>
</center>
```

2. PATIENT HOME



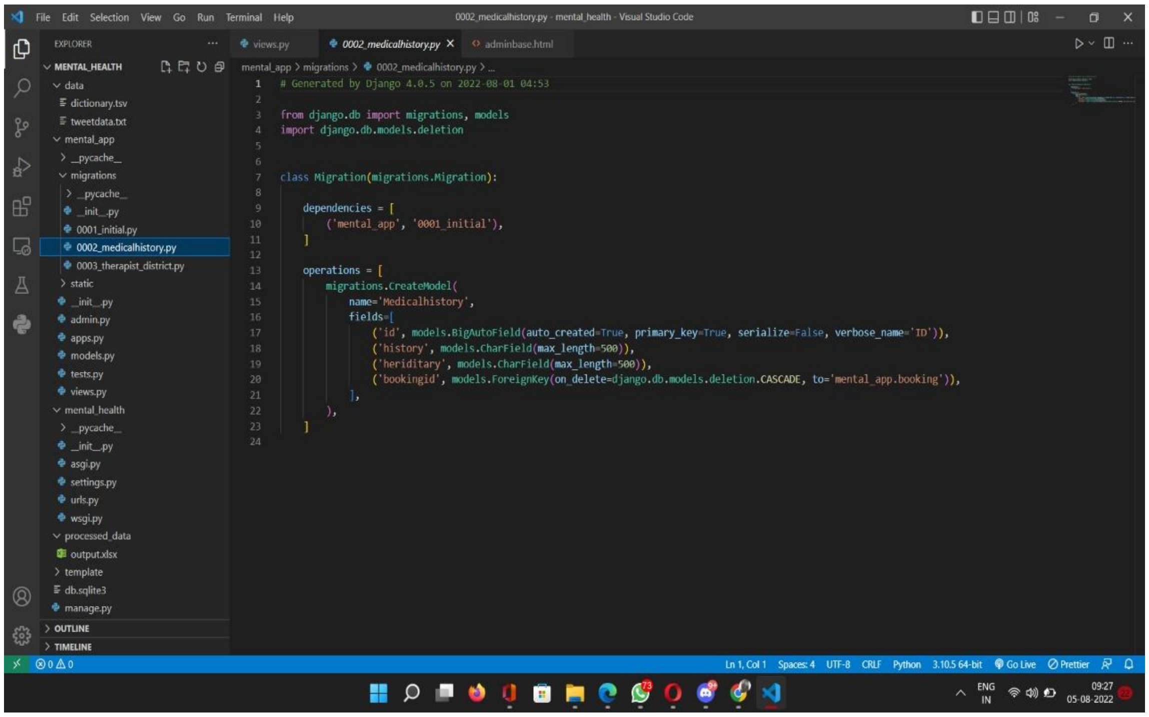
```
{% extends 'patientbase.html' %}
{% block content %}
<style>
    th,td{
        padding: 10px;
    }

    </style>
    <center>
    <div style="margin: 50px;">
        <hr><h2 style="margin: 10px;">Welcome {{ request.session.name }}</h2><hr>
        <form method="POST">
            {% csrf_token %}
            {% if profile_data %}
            {% for i in profile_data %}
            <table>
                <tr>
                    <td>Name</td>
                    <td><input type="text" class="form-control" value="{{i.name}}"name="txtName" pattern="[a-zA-Z ]+" required></td>
                </tr>
                <tr>
                    <td>Address</td>
                    <td><textarea class="form-control" name="txtAddress" required>{{i.address}}</textarea></td>
                </tr>
                <tr>
                    <td>Contact</td>
                    <td><input type="number" class="form-control" value="{{i.phone}}" name="txtContact" required></td>
                </tr>
                <tr>
                    <td>Email</td>
                    <td><input type="email" class="form-control" value="{{i.email}}" name="txtEmail" required></td>
                </tr>
            </table>
            {%endfor%}
            {% endif %}
        </form>
    </div>
```

3. PATIENT MEDICAL HISTORY



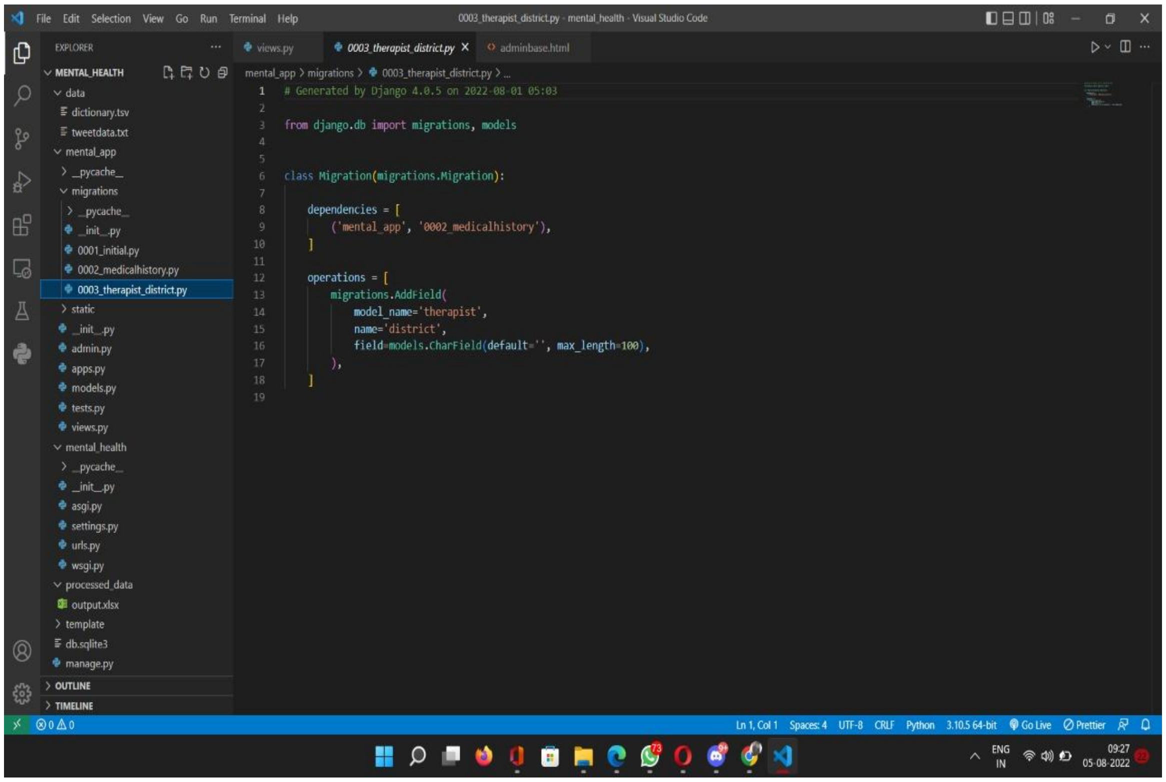
```python
# Generated by Django 4.0.5 on 2022-08-01 04:53

from django.db import migrations, models
import django.db.models.deletion


class Migration(migrations.Migration):

    dependencies = [
        ('mental_app', '0001_initial'),
    ]

    operations = [
        migrations.CreateModel(
            name='Medicalhistory',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('history', models.CharField(max_length=500)),
                ('heriditary', models.CharField(max_length=500)),
                ('bookingid', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, to='mental_app.booking')),
            ],
        ),
    ]
```

4. PATIENT MEDICAL HISTORY



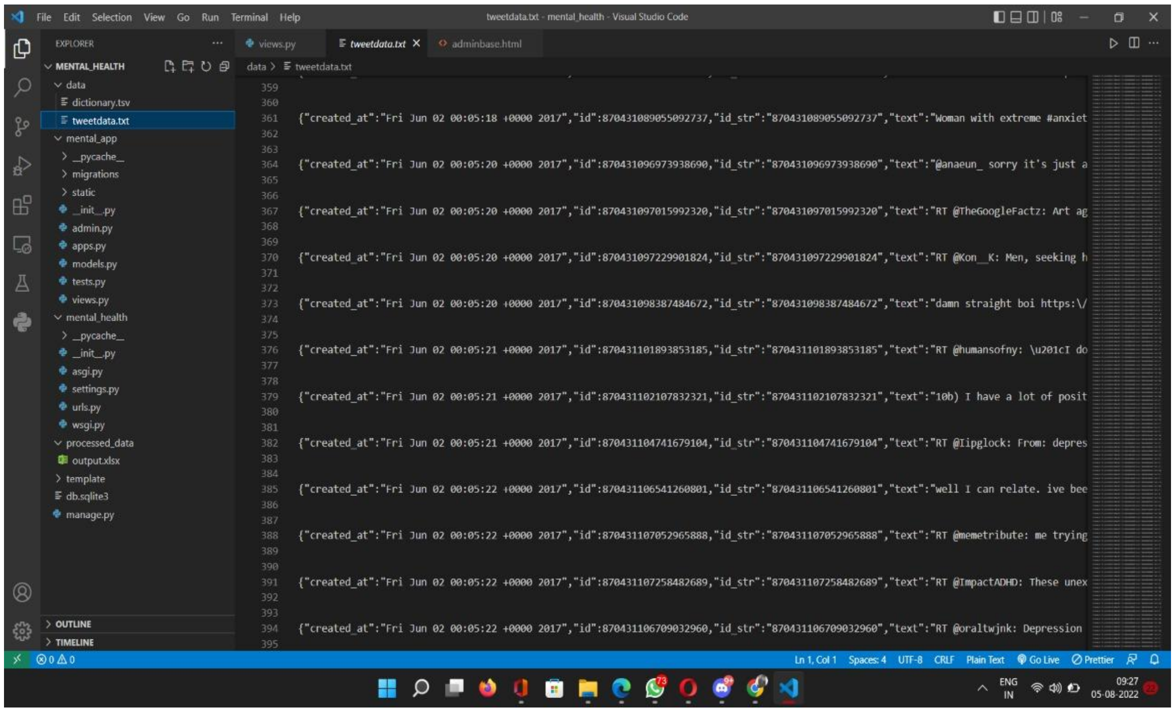
```python
# Generated by Django 4.0.5 on 2022-08-01 04:53

from django.db import migrations, models
import django.db.models.deletion


class Migration(migrations.Migration):

    dependencies = [
        ('mental_app', '0001_initial'),
    ]

    operations = [
        migrations.CreateModel(
            name='Medicalhistory',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('history', models.CharField(max_length=500)),
                ('heriditary', models.CharField(max_length=500)),
                ('bookingid', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, to='mental_app.booking')),
            ],
        ),
    ]
```

5. THERAPIST DISTRICT



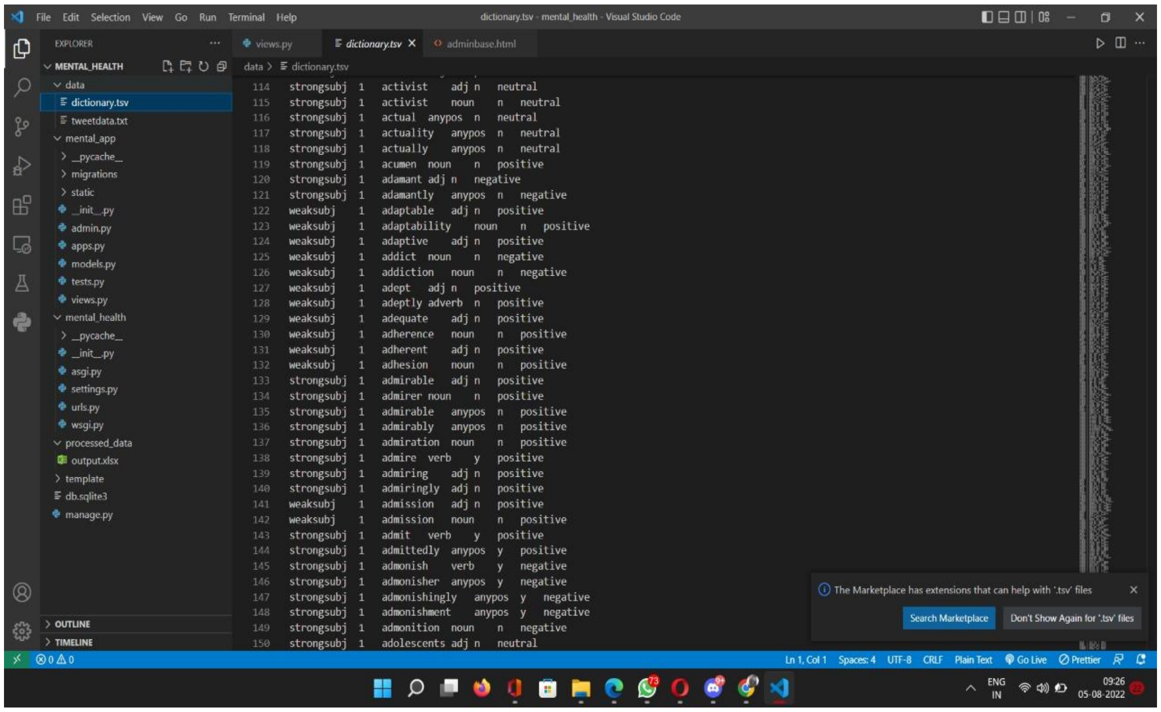
```python
# Generated by Django 4.0.5 on 2022-08-01 05:03

from django.db import migrations, models


class Migration(migrations.Migration):

    dependencies = [
        ('mental_app', '0002_medicalhistory'),
    ]

    operations = [
        migrations.AddField(
            model_name='therapist',
            name='district',
            field=models.CharField(default='', max_length=100),
        ),
    ]
```

6. DATA COMPARISON



```
{"created_at":"Fri Jun 02 00:05:18 +0000 2017","id":870431089055092737,"id_str":"870431089055092737","text":"Woman with extreme #anxiet


{"created_at":"Fri Jun 02 00:05:20 +0000 2017","id":870431096973938690,"id_str":"870431096973938690","text":"@anaeun_ sorry it's just a


{"created_at":"Fri Jun 02 00:05:20 +0000 2017","id":870431097015992320,"id_str":"870431097015992320","text":"RT @TheGoogleFactz: Art ag


{"created_at":"Fri Jun 02 00:05:20 +0000 2017","id":870431097229901824,"id_str":"870431097229901824","text":"RT @Kon__K: Men, seeking h


{"created_at":"Fri Jun 02 00:05:20 +0000 2017","id":870431098387484672,"id_str":"870431098387484672","text":"damn straight boi https:\/


{"created_at":"Fri Jun 02 00:05:21 +0000 2017","id":870431101893853185,"id_str":"870431101893853185","text":"RT @humansofny: \u201cI do


{"created_at":"Fri Jun 02 00:05:21 +0000 2017","id":870431102107832321,"id_str":"870431102107832321","text":"10b) I have a lot of posit


{"created_at":"Fri Jun 02 00:05:21 +0000 2017","id":870431104741679104,"id_str":"870431104741679104","text":"RT @Iipglock: From: depres


{"created_at":"Fri Jun 02 00:05:22 +0000 2017","id":870431106541260801,"id_str":"870431106541260801","text":"well I can relate. ive bee


{"created_at":"Fri Jun 02 00:05:22 +0000 2017","id":870431107052965888,"id_str":"870431107052965888","text":"RT @memetribute: me trying


{"created_at":"Fri Jun 02 00:05:22 +0000 2017","id":870431107258482689,"id_str":"870431107258482689","text":"RT @ImpactADHD: These unex


{"created_at":"Fri Jun 02 00:05:22 +0000 2017","id":870431106709032960,"id_str":"870431106709032960","text":"RT @oraltwjnk: Depression
```

7. MENTAL HEALTH EVALUATION



```
strongsubj  1   activist    adj n   neutral
strongsubj  1   activist    noun    n   neutral
strongsubj  1   actual  anypos  n   neutral
strongsubj  1   actuality   anypos  n   neutral
strongsubj  1   actually    anypos  n   neutral
strongsubj  1   acumen  noun    n   positive
strongsubj  1   adamant adj n   negative
strongsubj  1   adamantly   anypos  n   negative
weaksubj    1   adaptable   adj n   positive
weaksubj    1   adaptability    noun    n   positive
weaksubj    1   adaptive    adj n   positive
weaksubj    1   addict  noun    n   negative
weaksubj    1   addiction   noun    n   negative
weaksubj    1   adept   adj n   positive
weaksubj    1   adeptly adverb  n   positive
weaksubj    1   adequate    adj n   positive
weaksubj    1   adherence   noun    n   positive
weaksubj    1   adherent    adj n   positive
weaksubj    1   adhesion    noun    n   positive
strongsubj  1   admirable   adj n   positive
strongsubj  1   admirer noun    n   positive
strongsubj  1   admirable   anypos  n   positive
strongsubj  1   admirably   anypos  n   positive
strongsubj  1   admiration  noun    n   positive
strongsubj  1   admire  verb    y   positive
strongsubj  1   admiring    adj n   positive
strongsubj  1   admiringly  adj n   positive
weaksubj    1   admission   adj n   positive
weaksubj    1   admission   noun    n   positive
strongsubj  1   admit   verb    y   positive
strongsubj  1   admittedly  anypos  y   positive
strongsubj  1   admonish    verb    y   negative
strongsubj  1   admonisher  anypos  y   negative
strongsubj  1   admonishingly   anypos  y   negative
strongsubj  1   admonishment    anypos  y   negative
strongsubj  1   admonition  noun    n   negative
strongsubj  1   adolescents adj n   neutral
```

The Marketplace has extensions that can help with '.tsv' files