PES1PG21CA145_jagadish_plag

by Jagadish Bob

Submission date: 06-Feb-2023 09:35AM (UTC+0530)
Submission ID: 2007251527
File name: PES1PG21CA145_jagadish_plag.pdf (6.33M)
Word count: 6927
Character count: 36147

Abstract

Mental issue is a main problem everywhere globally bipolar and sleep apnea are some foremost issues. Bipolar and sleep apnea kinds of disorders are separate hospital-based issues that could co-arise in many people and both of which could have a giant effect on human basic fitness and well-being bipolar disease is grouped by the aid of high temper swings including series of mania and melancholy whilst sleep apnea is a sleep problem this is indicated by way of recurrent episodes of interrupted respiratory for the duration of sleep.

Bipolar and sleep apnea detection helps a sick person to check whether the sick person has bipolar or sleep apnea disorder this gives results at earlier stages if the disorder is detected at an earlier stage it can be effectively diagnosed and treated.

-

Chapter 1

Introduction

The process of making-decision is made more difficult by the fact that medical information frequently contains information that has been withheld discovering previously hidden trends in medical information and performing data analysis both need the use of computer vision data analysis is a popular practise in many different fields including the business world government the transportation industry the medical field and the marketing field 6 there are many methods for making this analysis more efficient and machine learning is one of them data mining is a subset of machine learning that works with massive volumes of theyll-formatted data there are three primary uses for algorithms in the medical field illness prediction detection and diagnosis bipolar illness may be diagnosed and treated more effectively if it is diagnosed and treated at an earlier stage and these approaches are designed to do just that in past years medical research focused on a wide range of automatically generated characteristics gleaned from structural information structured data unstructured data and semi-structured data are all contained in the database structured data is the most common kind of data there are several types of structured data including electronic health records laboratory records and data from the results of medical testing 2 reference required as a consequence a substantial amount of information about bipolar disorder can be handled making it possible to forecast a persons propensity for developing the illness in the future when medical data is not supplied in the appropriate format two processes that need to be carried out are known as imputation and data cleaning it is impossible to perform disease prediction as a result of the poor formatting of the data which occasionally results in an inaccurate prognosis of the condition.

The disease diagnosis based on symptoms was something that they had already completed the nave Bayes classification algorithm was applied to the arranged data that they had in order to make predictions regarding illness within the scope of this study they perform operations on medically formatted data the concept of the convolution neural network which is a sort of network that can automatically provide the needed result by extracting features from enormous datasets is the foundation of deep learning when working with structured data was designed to strip the information of its important attribute values and produce disease prediction based on the dataset this was done in order to improve accuracy the utilization of structured data for the purpose of forecasting bipolar disorder as theyll as the risk of acquiring bipolar disorder is the primary objective of this work.

1.1 Project Description

The project aims to develop a tool for the detection of the phenomenon of bipolar and sleep apnea in individuals the tool will utilize a combination of diagnostic methods including physical exams medical history sleep studies and psychological evaluations the first step of the project will involve conducting a thorough literature review to gather information on current characteristic methods and best practices for the detection of the phenomenon of these two conditions this will be used to inform the design and to make a diagnostic tool next the project will involve recruiting a diverse number of participants who will undergo comprehensive tests using the characteristic the results of the assessment will be analyzed to determine the tools sensitivity specificity and overall accuracy in detecting the co-occurrence of bipolar disorder and sleep apnea.

Once the diagnostic tool has been validated the project will involve conducting it in a clinic and evaluating its effectiveness in practice this will include assessing the impact of the tool on patient results such as improved quality of life and reduced symptoms of both conditions, in conclusion, this project aims to develop a comprehensive diagnostic tool for the detection of co-occurrence of bipolar disorder and sleep apnea in individuals that can be implemented in a clinical place and increase patient outcomes the project includes a thorough literature review development and validation of the tool and implementation and evaluation of it in a clinical setting.

1.1.1 Problem Statement

The problem statement of bipolar and sleep apnea could result in a complicated and punishing state of affairs for the affected person and the healthcare specialists the symptoms of both ailments might be severe and incapacitating impairing a person's capability to feature in day-by-day existence including activity school and social activities bipolar sickness can exacerbate the symptoms of sleep apnea and vice versa.

1.1.2 Proposed Solution

The objective of this project is to determine the presence or absence of bipolar and sleep apnea in the subject additionally if a patient has type 1 or type 2 bipolar disorder assume that they are likely to enter a bipolar disorder phase based on the medical report the user provided the required input information next the historical data set is loaded because most medical records contain missing information reliable estimation becomes difficult to fill up the gaps interpolation and data cleaning are necessary using an approach that combines data cleansing and computing they must turn unstructured data back into unstructured data when the calculation is complete this approach aims to identify individuals with sleep apnea as early as possible and provide them with appropriate treatment to alleviate symptoms and improve overall health and theyll-being.

1.1.3 Purpose

The main purpose of developing this application is to make people provide automatic results of disorders this will help patient to easily see their report online and know about their disorder stage secondly doctors would be easier to know what their patients are suffering from disorders and which stage it is it helps to take necessary action to cure the disorders this project presents a smart theyb and machine learning-based bipolar prediction methodology where they get all information about the type of bipolar patients to suffer from. In summary the purpose of detecting the co-occurrence of bipolar disorder and sleep apnea is to improve the overall health and theyll-being of individuals affected by these conditions by identifying them early providing appropriate treatment reducing the risk of complications increasing functionality and reducing healthcare costs.

1.1.4 Scope and Limitations

The scope of detecting co-occurrence of bipolar disorder and sleep apnea includes developing and implementing a comprehensive diagnostic tool and checking its effectiveness in practice the limitations include limited availability of resources difficulty in recruiting a diverse sample of participants limited access to sleep specialists difficulty in determining the best course of treatment and challenges in adherence to treatment limited availability of resources the diagnostic tool may require advanced equipment and trained personnel which may not be readily available in all healthcare settings. -

Chapter 2

Literature Survey

2.1 Domain Study

Domain study of bipolar and sleep apnea detection would involve a thorough literature review study design collection information analysis evaluation working and dissemination of results to gain information on the occurrence of the two conditions and its diagnostic tool and help in improving the treatment of these conditions.

- **Literature review:** doing a thorough study on the existing materials at the co-prevalence of bipolar disorder and sleep apnea such as analyzing the hazard symptomatic strategies and remedy selections for those problems
- **Study design:** Developing a research study that aims to investigate the diagnostic tool and its effectiveness in detecting the co-occurrence of these two conditions. The study design could be observational, cross-sectional, or case-control study.
- **Data collection:** Recruiting a diverse sample of participants and collecting data using the diagnostic tool, as well as other relevant data such as medical history, sleep studies, and psychological evaluations.
- **Data analysis:** Analyzing the data collected to determine the diagnostic tool's sensitivity, specificity, and overall accuracy in detecting the co-occurrence of bipolar disorder and sleep apnea.
- **Implementation and evaluation:** It Implement the diagnostic tool in a clinical setting and evaluate its effectiveness in practice, such as the impact on patient outcomes, including improved quality of life and reduced symptoms of both conditions.

2.1.1 Existing Systems

It is essential to be able to accurately diagnose the patient through physical tests and clinical judgment computer-aided decision support systems may be required for reliable choices healthcare outcomes are large volumes of data related to patient selection patient feedback treatments follow-up visits medications and other concerns you need to put in a lot of planning to achieve the goals due to increasing data legitimate technology is required to collect and process data it is essential to use the correct disease prediction from the ever-increasing medical data however processing a lot of data is also crucial.

2.1.2 Drawbacks of Existing Systems

Drawbacks of existing systems are issues with diagnosis limited access to specialised care limited patient education limited knowledge on the prevalence of the illnesses and the difficulty of treatment are some of the difficulties of identifying the occurrence of bipolar disease and sleep apnea these shortcomings may lead to inconsistent and erroneous diagnoses which could have an adverse effect on the efficacy of the diagnosis and therapy.

2.2 Tools and Technologies

2.2.1 Tools

- **Django:** Python-based the django software foundation dsf is the organization responsible for the progress in development and maintenance of the open-source framework known as django it foundation djangos rich built-in functionality is largely responsible for its current widespread adoption many popular companies and apps like google instagram discus spottily you tube and pinterest use django to make website python is commonly used in website making by placing all html files in the templates directory you can quickly render html pages as templates and static files are supported similarly all style related files including css and js are stored in the fixed directory more customization and flexibility is possible due to this feature the tasks on the front end of this project are done using django additionally django has more functionality than competing frameworks some of which are described below.

- **Anaconda Navigator:** It is a graphical interface platform that permits the user to run program manage environments and packages and use conda as a language-neutral package addiction and environmental manager ruby python and java scala javascript c and fortran usually installed alongside the anaconda package of python and r it is a free and open-source program.

2.2.2 Technologies

- **HTML 5:** HTML stands for hypertext mark-up language is used to format arrange and display content on the internet it is the inspiration behind all internet sites to describe the structure and format of the material on an internet web page HTML uses a set of preset tags along with for paragraphs headers photos and links these tags are used to enclose and layout the text photographs movies as well as many other element to make a website HTML documents can be viewed using a web browser like chrome firefox or safari they are typically saved with the HTML or HTML report extension html5 is the latest version of HTML it was released in 2014 and it builds on the previous versions of HTML. HTML 401 and XHTML 10 by introducing new features and capabilities that make it easier to create and structure web content.

- **CSS 3:** The display of a page created in an HTML it can be described using CSS cascading style sheets and which is a language for stylesheets it is used to separate a webpages display from its information making it simpler to change and manage a websites design CSS is a method for applying styles to HTML tags on a web page with colors fonts layouts and animations without changing the underlying HTML code developers can adjust the layout fonts and colors of a webpage by utilizing CSS.

- **Bootstrap4:** For the purpose of developing responsive web applications and designs bootstrap is a framework that is both free and open source the ability of a computer software to function properly on portable electronic gadgets like mobile phones and tablets is referred to as responsiveness every component of the HTML document is stacked if the page is resized minimized or otherwise reduced in size t bootstraps default setup utilises 12 columns each of which has the same width and is separated evenly from the next column as a result each columns width is the same to create layouts and designs according to the requirements you may alter the default parameters and use tag bootstrap includes a grid structure that can be customised for devices of many shapes and sizes including very tiny small medium extra-large and massive ones as a result the programme may operate effectively on any gadget in addition it has some wonderful tables buttons and forms among other things the most current bootstrap version 4 contains a lot of additional features compared to earlier versions the Django and bootstrap 4 are both used in the front-end development of this project

- **Python :** Python is a high-level interpreted programming language and is broadly used for a diffusion of purposes which includes internet development medical computing records analysis artificial intelligence and extra one of the main advantages of python is its readability because the language uses indentation to suggest code blocks and has an exceptionally easy syntax this makes it smooth to learn and recognize each novice and skilled programmer python has a large and energetic network which has developed a huge variety of libraries and frameworks for numerous obligations which includes numpy and scipy for medical computing pandas for facts evaluation and django and flask for web improvement python also affords some built-in records types along with lists dictionaries and units that make it smooth to paintings with information the language also has some superior capabilities such as decorators and turbines that allow for more powerful and efficient programming python can be used for an extensive range of applications which includes net development medical computing records evaluation synthetic intelligence and extra it may be

run on a ramification of structures which include windows mac and linux and it can be integrated with different languages such as c and c to create greater complex programs.

2.3 Feasibility Study

The procedure used to determine whether an assignment is viable or not is referred to as the viability period one step in this process is the feasibility study in other words it involves identifying the specific needs of each unique user mobilizing valuable resources and managing those resources over time the purpose of this investigation is to determine whether the developed machine is practical or not once the requirement is given development moves to the design phase the design process involves a number of possibilities the most important task of the developer in developing the target is to select the final solution from all the available options is known as Feasibility Study.

Technical Feasibility

It examines the hardware and software used and generates a report on the technical feasibility of the system. For example, it analyzes whether the system can operate under different conditions and meet users' expectations by delivering intended and expected results. In addition, it displays the system reaction time as well as the speed of the process. Developers now have to make sure that the programming languages they choose are simple enough for a non-technical person to understand. Developers should ensure that the option they choose is versatile, meaning both the front-end and back-end are compatible, functional and support all scenarios. Whichever option you choose, it's still true.

Economic Feasibility

Economic feasibility for the detection of co-occurring bipolar and sleep apnea disorder involves analyzing the costs and advantages of different diagnostic and treatment options one important factor to consider is the cost of diagnostic testing such as polysomnography sleep study and psychiatric evaluations these tests can be costly especially if they are not covered by insurance additionally treatment for both conditions can also be expensive such as the cost of medications and therapy sessions on the other hand early detection and treatment of co-occurring bipolar disorder and sleep apnea can have significant benefits in terms of improved patient results such as reduction in symptoms and improved quality of life it can also lead to cost savings in the long run by avoiding hospitalization and other associated costs it is also important to consider the costs of not detecting and treating co-occurring bipolar and sleep apnea the untreated symptoms of these disorders can lead to significant impairment in daily functioning and work productivity as well as increased healthcare costs.

Operational Feasibility

Operational feasibility for the detection of co-occurring bipolar disorder and sleep apnea disorder involves evaluating whether the proposed diagnostic and treatment options can be effectively implemented within the existing healthcare system and resources it includes the availability and accessibility of diagnostic testing and treatment options and ability to integrate the detection and treatment within existing healthcare systems and protocols practicality of implementing the proposed options and the potential for resistance to change among healthcare providers and patients. The word operational should make it

clear that it has nothing to do with how the system performs, behaves, or operates. Here, there are a number of factors to keep in mind, some of which are listed below.

- In what ways will the system be altered in the coming months?
- In addition to the introduction of new talent, what type of professionalism is required?
- Is there anybody on the current team who has these qualities?
- Are they going to obtain training if they don't already have it?
- The technology will be put into service after it has been fully designed and tested. A basic understanding of how to operate the system is essential for everyone who wants to utilise it. There should be at least a basic degree of instruction provided to the user.

-

Chapter 3

Hardware and Software Requirements

3.1 Hardware Specification

Hardware Specification	
Specification	**Desired Value**
Processor	Intel I3
CPU Speed	2Ghz or Better
Memory (RAM)	4 GB
Hard Disk	120 GB

Table 3.1: Hardware Specification

3.2 Software Specification

Software Specification	
Specification	**Desired Value**
Operating System	windows/Linux
Development Tools	Anaconda navigator IDE
IDE	jupyter notebook
Web Server	Django
Web Browser	chrome,firefox,brave etc..

Table 3.2: Software Specification

-

Chapter 4

Software Requirements Specification

4.1 Users

- Doctor

  A doctor can use this system for the detection and management of bipolar and sleep apnea disorder detection. A new doctor can register for the first time and after the registration is complete the doctor can log in to the system. the doctor can select the disorder either bipolar or sleep apnea. In bipolar tests, the doctor should take a clinical test for the patient and enter the required information. after which the prediction will happen. in sleep apnea, the doctor needs to upload the EEG data after the data has been uploaded the result will be generated. the doctor can even send the report to the patient directly through the mail. there is user-friendly interface so that a non technical person can understand the system and use it.

- Patient

  A patient can take the primary test for bipolar disorder, the patient can view the report of bipolar. A new patient should register into the system after the registration is complete. the patient can log in to the system and there will be two options in the menu either to take the bipolar test or to view the report. report section will be empty for new patients. if the patient wishes to take the bipolar test he will be redirected to the bipolar test page where there are 5 questions that the patient needs to ansitr if he gets 3 out of 5 then the system prompts the user to consult the doctor at an earlier stage then there is report section he the patient can view the report generated by the doctor.

4.2 Functional Requirements

Data Gathering Statistics gathering is a important step in the bipolar and sleep apnea disorder the use of machine learning algorithms project the information which you accumulate have to be relevant to the problem handy and it ought to be of excessive high-quality one way to acquire records for this project is to apply publicly available datasets for example the bipolar and sleep apnea sickness dataset on the Kaggle itbsite is a broadly used dataset for this kind of assignment any other way to collect statistics is to gather it the self this obtain this statistics additionally you have to be privy to any moral issues when right now you have gathered the information is critical to preprocess it and easy it up earlier than using it for education the model normal the facts collecting process can be a challenging but important step in building an poitrful bipolar and sleep apnea disease detection the usage of machine gaining knowledge of algorithms an excellent quality facts is the first step to a terrific acting model so its vital to make an effort to gather and preprocess the facts carefully.

Data Pre-processing Handling lack of data and if there is any missing data inside the dataset is necessary to decide a way to manage them this should encompass losing the rows with missing data or imputing the missing data with estimates primarily based on the many different records value normalization this is system studying models regularly require that the input statistics are in a selected variety which includes among 0 and 1 if the information isn't already in this variety it is going to be important to normalize or scale the information many system mastering models require that the input records is in a certain layout such as numerical if the records are in a one of a kind layout it'll want to be converted but in this dataset their arent any such values.

Model Construction and Model Training Pre-trained models are system studying fashions that have been executed on a big dataset and may be used for brand spanking new studying or trouble-fixing situations these models can be used to extract data and skills which have been received at some point of their preceding schooling and follow them to new obligations.

Deploy model in the Django Deploying a machine learning model in Django involves converting the model to a format that can be used in Django creating a new django app to host the model defining views and URL patterns to handle the model's predictions creating a template to display the predictions and adding the app to the installed apps list in the projects settings.py file after that you can deploy the Django project on a itb server

Bipolar Test module A patient can take the bipolar test by answer a set of questions. the system can detect bipolar disorder and it prompts the patient to meet the doctor for further treatment.

Doctor Module The doctor module will build the account and review the data together with the patient's characteristics to see whether the illness has been impacted.

Patient Module The patient module allows login and this module maintains all the patient information. a patient can view the report and take the test in the module.

4.3 Non-Functional Requirements

Reliability:

Reliability is a key non-functional requirement in terms of the detection and control of the co-occurring bipolar disease and sleep apnea disease it refers to the devices capability to feature correctly and consistently over the years assembly reliability necessities is vital because it ensures that the machine might be reliable and honest for sufferers healthcare companies and researchers along with excessive availability fault tolerance recoverability consistency repeatability and robustness assembly reliability requirements is essential as it guarantees that the device might be dependable and honest for patients healthcare carriers and researchers to meet reliability necessities builders may additionally use strategies such as redundancy load balancing and failover mechanisms.

Availability:

As the patient can access this itbsite any time a patient can access this itbsite 24/7 and this itbsite can be examined from any device as this is made easy to use the main aim is to keep the itbsite working and available 24/7 for the patient.

Security:

A patient's account is extremely susceptible to cyberattacks, thus by using the proper login information it can prevent hacking as both the health practitioner and the patient must check in and log in to use the itbsite, which offers excellent security.

Easy to use:

Any person with the knowledge can use this itbsite as it is made user friendly due to user friendly so the users can easily use the itbsite.

Performance:

If multiple users are using the itbsite the itbsite should run efficentlty and there should not be any delays in the process and working of the itbsite.

-

Chapter 5

System Design

5.1 Architecture Diagram

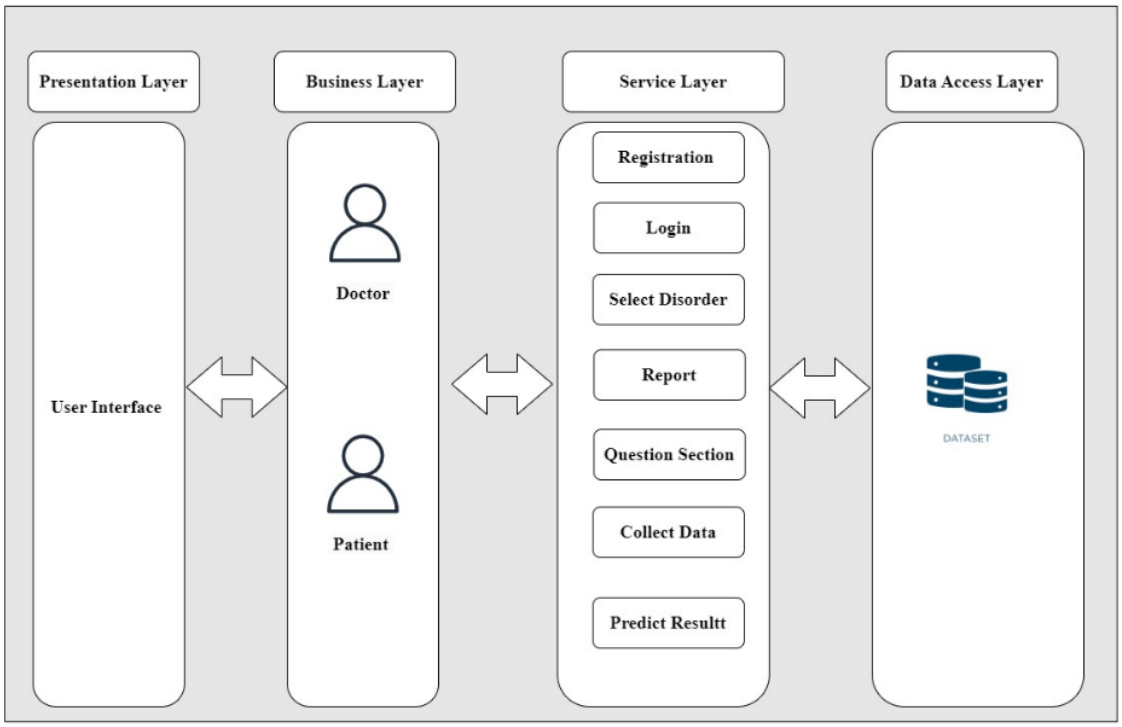


Figure 5.1: Architecture Diagram

An architecture diagram for a bipolar and sleep apnea detection gadget might usually include the subsequent additives:

- Data acquisition: This aspect could be answerable for accumulating and obtaining facts from diverse sources, such as clinical facts, surveys, sleep research, and interviews.
- Data preprocessing: This aspect could be chargeable for cleansing, transforming, and organizing the facts to make them ready for analysis.
- Machine Learning Model: This issue could be answerable for education and deploying the machine learning model so that it will be used to come across bipolar ailment and sleep apnea disorder.

- Inference Engine: This thing would be accountable for making predictions about the use of the deployed system and gaining knowledge of the version.
- Data visualization: This aspect could be accountable for showing the results of the predictions in a consumer-friendly format, which includes graphs or charts.
- User Interface: This aspect might provide a user-pleasant interface for sufferers, healthcare carriers, and researchers to engage with the system and access the predictions.
- Authentication and Authorization: This component would be accountable for ensuring that the best legal customers have access to the device and facts.
- Monitoring and Logging: This element could be chargeable for monitoring the system's overall performance, tracking errors and exceptions, and producing logs.

This is a widespread architecture diagram, the precise architecture will depend on the device being developed and the wishes of the stakeholders.In summary, An structure diagram for a bipolar and sleep apnea detection gadget would generally include the following additives: Data acquisition, Data preprocessing, Machine Learning Model, Inference Engine, Data visualization, User Interface, Database, Authentication and Authorization, Monitoring and Logging. It affords an excessive-level view of the system's additives and the way they interact with every different.

5.2 Data Flow Diagram

5.2.1 Context Diagram

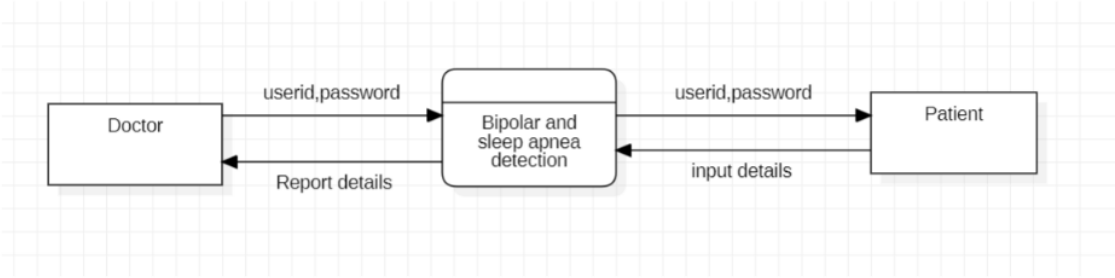


Figure 5.2: Context Diagram

A context diagram is an excessive-level diagram that provides a visible representation of a machine's outside interactions and inputs/outputs, it's far used to show the relationship between the machine and its surroundings, A context diagram for a bipolar and sleep apnea detection system could commonly consist of the subsequent additives: Patients, Healthcare vendors, Researchers, Data resources, Bipolar and sleep apnea detection machine, Output. It affords a high-degree view of the gadget's outside interactions and inputs/outputs and suggests how the device suits the bigger surroundings.

-

Chapter 6

Detailed Design

6.1 Use Case Diagram

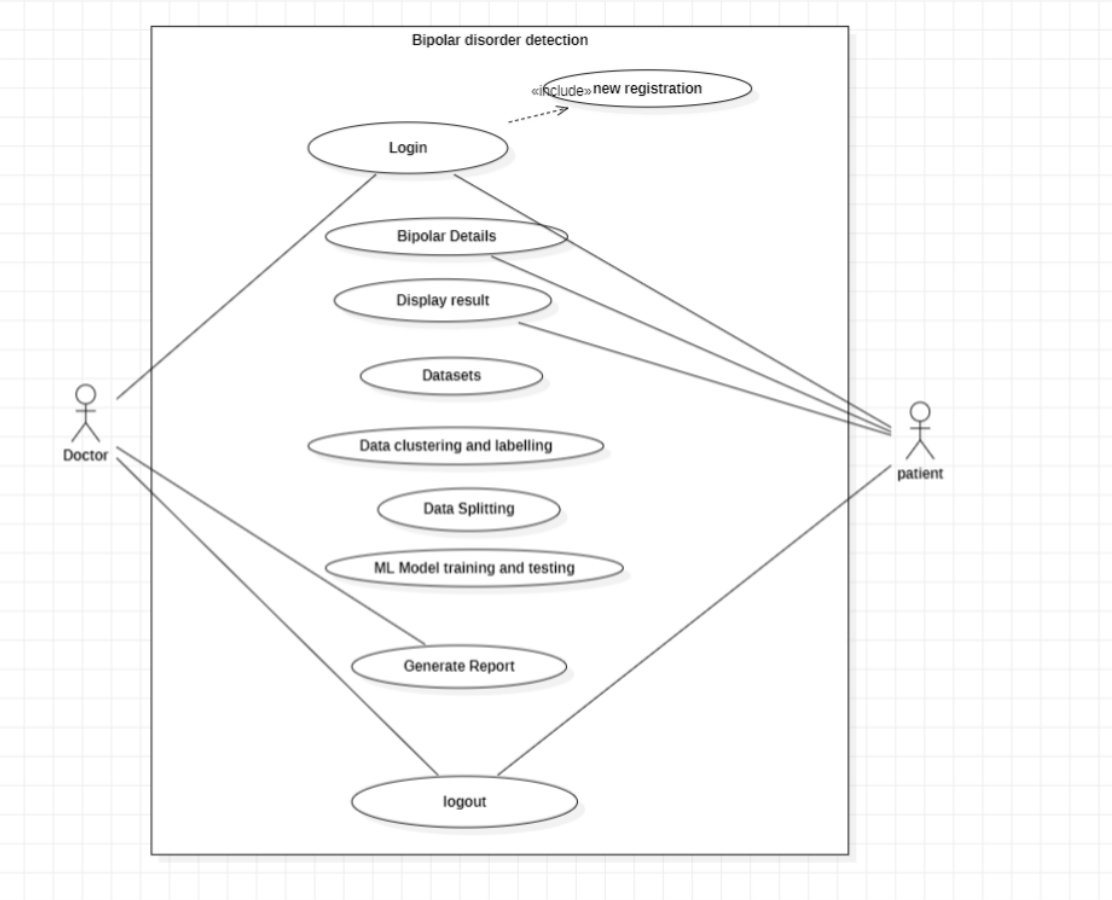


Figure 6.1: Use Case Diagram

The above figure is a use case diagram, it contains two actors doctor and a patient.

- **Doctor :** The doctor can login. view patient details and generate reports for the patient.
- **patient :** The patient can login and take the bipolar test he can see the results of the bipolar test

and he can see the report.

6.2 Process Flow Diagram

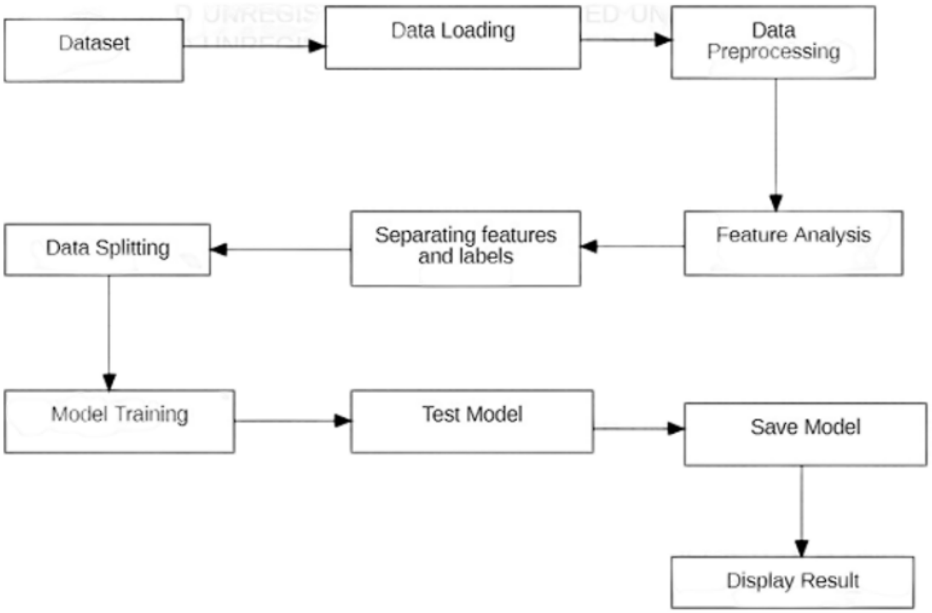


Figure 6.2: Process Flow Diagram

The system goes with the flow diagram and represents the sequential waft of the events passed off from beginning to stop that is from input to outputit starts off evolved with the historic records because of this the prerequisite information that is required to enter the facts thats one of the critical steps in the device mastering model the accumulated statistics are loaded into the device which is accompanied by means of pre-processing where the null redundant values are removed and the lacking values are delivered with the mean value so that the uniformly inside the values are maintained subsequently comes the function evaluation wherein the relevant fields are chose which might be possible for the model and the encoding approaches are performed where the explicit values are transformed into numerical values and grouped so that there is a clear distinction between the values and there can be a clear distinction among the columns and the information might be supervised and subsequently comes the records splitting wherein the datasets are split into schooling and testing values and the algorithms such as random wooded area and choice tree classifiers are implemented and based totally on the accuracy rating and the confusion matrix that is received from the implementation is recorded for predicting the consequences and version which is prepared is stored in the database in order that each time the user input the exclusive records then the model outputs the result based totally on the events it executed from the previous history and is expecting the output in turn it saves the records of the version and will produce the most accurate end result subsequent time consumers enter the facts and end result might be extra accurate every time the model comes to be an increasing number of intelligent and examine from the past enter dataset the

version may be made powerful with the more gui primarily based interface the usage of html and css the usage of the flask framework and the the pickle library module is used to keep the output statistics. -

Chapter 7

Implementation

7.1 Sample Code / Pseudo Code

```
In [1]: import numpy as np
        import pandas as pd

        from sklearn.metrics import accuracy_score,classification_report,confusion_matri

        import warnings
        warnings.filterwarnings("ignore")

In [2]: data=pd.read_csv("C:/Users/Jagadish BOB/Desktop/BipolarCassification/bipolar .csv

In [3]: data.head()
```

Out[3]:

	Group	Type	Age	Right_answers	Audio_prosody	Combined_channel	Face_video	Body_video
0	Bipolar	BD I	47	40	9	11	9	11
1	Bipolar	BD I	49	49	13	13	11	12
2	Bipolar	BD I	45	43	9	11	13	10
3	Bipolar	BD I	53	44	10	10	12	12
4	Bipolar	BD II	50	50	14	13	11	12

Table 7.1: Bipolar disorder

Importing the necessary libraries and loading the dataset.

In [5]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
data.Group=le.fit_transform(data.Group)
data.Type=le.fit_transform(data.Type)
data.head()
```

Out[5]:

	Group	Type	Age	Right_answers	Audio_prosody	Combined_channel	Face_video	Body_video
0	1	0	47	40	9	11	9	11
1	1	0	49	49	13	13	11	12
2	1	0	45	43	9	11	13	10
3	1	0	53	44	10	10	12	12
4	1	1	50	50	14	13	11	12

In [18]:
```
from sklearn.model_selection import train_test_split
x= data[['Age', 'Right_answers', 'Audio_prosody', 'Combined_channel', 'Face_videc
y=data.Type
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

In [19]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(X)

X=scaler.transform(X)
print(X)
```

```
[[0.33333333 0.45614035 0.61702128 ... 0.6        0.57894737 0.23529412]
 [0.33333333 0.49122807 0.80851064 ... 0.75       0.63157895 0.70588235]
 [0.33333333 0.42105263 0.68085106 ... 0.6        0.63157895 0.35294118]
 ...
 [0.33333333 0.22807018 0.87234043 ... 0.8        0.78947368 0.70588235]
 [0.33333333 0.19298246 0.82978723 ... 0.9        0.73684211 0.64705882]
 [0.33333333 0.66666667 0.82978723 ... 0.85       0.68421053 0.70588235]]
```

In [20]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,shuffle=1
```

In [21]:
```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
```

Out[21]:
```
DecisionTreeClassifier()
```

In [22]:
```
y_pred = model.predict(X_test)
```

Table 7.2: Bipolar diorder

Using Label Encoder to preprocess the categorical data and splitting the data set into x and y where 80 percent of dataset will be used for training and the remaining 20 percent for testing purpose.Creating an instance of DecisionTreeClassifier and performing the training for the same with the xtrain and train.

```
In [22]: y_pred = model.predict(X_test)

In [23]: acc_rc = accuracy_score(y_true=y_test, y_pred= y_pred)
         print("Overall accuracy of DC model using test-set is : %f" %(acc_rc*100))

         Overall accuracy of DC model using test-set is : 81.060606

In [24]: print(classification_report(y_pred, y_test))

                      precision    recall  f1-score   support

                   0       0.94      0.82      0.88        74
                   1       0.77      0.83      0.80        41
                   2       0.12      0.50      0.20         2
                   3       0.43      0.43      0.43         7
                   4       1.00      1.00      1.00         8

            accuracy                           0.81       132
           macro avg       0.65      0.72      0.66       132
        weighted avg       0.85      0.81      0.83       132
```

Table 7.3: Bipolar diorder

Performing the prediction trained by DecisionTreeClassifier model and checking the accuracy rate of bipolar disorder with DecisionTreeClassifier algorithm.

sleep apnea disorder

```
In [1]: import os
        import math
        import time
        import pandas as pd
        import numpy as np
        import seaborn as sns
        import missingno as msno
        import sklearn.ensemble as ske
        from sklearn import tree, linear_model
        from sklearn.feature_selection import SelectFromModel
        import joblib
        from sklearn.naive_bayes import GaussianNB
        from sklearn.metrics import confusion_matrix,f1_score
        import matplotlib.pyplot as plt
        from sklearn.neighbors import KNeighborsClassifier
        from sklearn.svm import SVC
        from sklearn.linear_model import LogisticRegression
        from sklearn.model_selection import train_test_split
        from sklearn.neural_network import MLPClassifier
        from sklearn.tree import DecisionTreeClassifier
```

Table 7.4: sleep apnea disorder

```
In [3]: data = pd.read_csv('./EEG-Brainwave-Sensor-Dataset/EEG dataset.csv', sep=',')

In [4]: data
```

Out[4]:

	Id	F1	F2	F3	F4	F5	F6	F7	F8	F9	...	F77	F78	F79	F80	F81	F8
0	1	1.000000	0.760188	0.760727	1.000000	1.000000	0.619948	0.760190	0.619946	0.760204	...	0.760201	0.760300	1.000000	0.620375	5.7974	5.792
1	2	0.760193	0.760200	1.000000	0.760203	0.760206	0.760194	0.760198	1.000000	0.760197	...	0.760219	0.760310	0.760312	0.760298	0.3977	0.395
2	3	0.760190	1.000000	1.000000	0.760193	1.000000	1.000000	1.000000	1.000000	1.000000	...	1.000000	1.000000	0.760275	1.000000	33.0799	32.887
3	4	0.760194	0.760202	1.000000	0.760188	0.760199	0.760200	0.760201	0.620025	0.760199	...	0.760261	1.000000	0.760253	1.000000	4.3899	4.418
4	5	1.000000	0.760193	0.619927	1.000000	1.000000	1.000000	0.760205	0.619962	1.000000	...	1.000000	1.000000	0.760188	0.760211	9.6461	9.683
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
211	212	0.996808	0.973875	0.977583	0.989293	0.974798	0.989429	0.970077	0.997414	0.980088	...	0.991694	0.986618	0.991613	0.973607	6.2080	15.561
212	213	0.999523	1.000000	0.980014	0.958475	0.982399	0.992096	0.992679	0.975254	0.980730	...	0.995139	0.996653	0.989380	1.000000	8.0515	15.278
213	214	0.983421	0.965538	0.987046	1.000000	0.965775	0.945798	0.964697	0.967879	0.986044	...	0.991904	1.000000	0.981772	0.998529	3.4924	4.450
214	215	0.984973	0.992781	0.991488	0.984010	0.973622	0.980828	0.980499	0.970921	0.994212	...	0.991062	0.996968	1.000000	0.972931	3.5942	10.957
215	216	0.949342	0.941573	0.961897	0.948350	0.975833	0.957784	0.940625	0.967572	0.960908	...	0.978802	0.972240	0.994135	0.980072	0.7369	0.390

216 rows × 87 columns

Table 7.5: sleep apnea disorder

Import all the required libraries and load the dataset.

```
In [25]: print('Testing  key  feature:')
         fsel = ske.ExtraTreesClassifier().fit(X, y)
         model = SelectFromModel(fsel, prefit=True)
         X_new = model.transform(X)
         nb_features = X_new.shape[1]

         X_train, X_test, y_train, y_test = train_test_split(X, y ,test_size=0.3)
         features = []
         indices = np.argsort(fsel.feature_importances_)[::-1][:nb_features]
         a =np.argsort(fsel.feature_importances_)[::-1][:nb_features]
         for f in sorted(np.argsort(fsel.feature_importances_)[::-1][:nb_features]):
             features.append(data.columns[f])

         '''
         Train Models:
         (alg1=Random Forest, alg2=AdaBoost, alg3=Gausian Naive Bayes, alg4=MLP, alg5=Decision Tree, alg6=Support Vector Machine,
         alg7=Logistic Reggression)

         '''
         algorithms = {
                 "RF": ske.RandomForestClassifier(n_estimators=10),
                 "AB": ske.AdaBoostClassifier(n_estimators=10),
                 "GNB": GaussianNB(),
```

Table 7.6: sleep apnea disorder

Splitting the Sleep apnea dataset into x and y.

```
FS=f1_score(y_test, res, average='macro')
plt.show()
print ('F1 Score is  : %f %%'% (FS*100))
print ('Confusion Matrix')
print (mt)
print("False positive rate : %f %%" % ((mt[0][1] / float(sum(mt[0])))*100))
print('False negative rate : %f %%' % ( (mt[1][0] / float(sum(mt[1])))*100))

winner = max(results, key=results.get)
print('\nWinner algorithm is %s with a %f %% success' % (winner, results[winner]*100))
clf = algorithms[winner]
res = clf.predict(X_test)
mt = confusion_matrix(y_test, res)
print("False positive rate : %f %%" % ((mt[0][1] / float(sum(mt[0])))*100))
print('False negative rate : %f %%' % ( (mt[1][0] / float(sum(mt[1])))*100))
```

```
Testing  key  feature:
Algorithm Test:
RF : 86.153846 %
AB : 46.153846 %
GNB : 43.076923 %
MLP : 72.307692 %
DT : 84.615385 %
LR : 80.000000 %

Best algorithm is RF with a 86.153846 % success
```

Table 7.7: sleep apnea disorder

The accuracy rate of sleep apnea disorder.

Doctor module

```python
from django.contrib import messages


from django.views.generic.detail import DetailView
from patient.models import PatientReg
from django.contrib.auth.models import User, auth
from django.shortcuts import render,redirect
from django.http import HttpResponse
import pandas as pd

from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import numpy as np
from reportlab.platypus import Paragraph, SimpleDocTemplate, Table, TableStyle, Image,Spacer
#creating the reportlab pdf library here.
import time
from reportlab.lib.enums import TA_JUSTIFY

from reportlab.lib.pagesizes import letter
from reportlab.lib import colors
from reportlab.lib.styles import getSampleStyleSheet
#creating the CNN library from here

import datetime


#importing the smtp
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders
#creating the reportlab pdf library here.
import time
from reportlab.lib.enums import TA_JUSTIFY

from reportlab.lib.pagesizes import letter

```

Table 7.8: Doctor module

```python
from reportlab.lib.styles import getSampleStyleSheet, ParagraphStyle
from reportlab.lib.units import inch
# Create your views here.
def reg(request):
    return render(request,"doctor/reg.html")
def index(request):
    return render(request,"doctor/index.html")
def home(request):
    if request.method == 'POST':
        email=request.POST['email']
        password=request.POST['password']
        user = auth.authenticate(username=email,password=password)
        if user is not None:
            auth.login(request,user)
            return render(request,"doctor/option.html")
        else :
            messages.info(request,'invalid username or password!')
            return render(request,"doctor/log.html")

    else :

        return render(request,"doctor/log.html")
    return render(request,'doctor/log.html')

def login(request):

    #suppose if we post the data
    if request.method == 'POST':
        email=request.POST['email']
        password=request.POST['password']
        user = auth.authenticate(username=email,password=password)
        if user is not None:
            auth.login(request,user)
            return render(request,"doctor/option.html")
        else :
            messages.info(request,'invalid username or password')
            return render(request,"doctor/log.html")

```

Table 7.9: Doctor module

```
        return render(request,"doctor/log.html")
    #return render(request,'doctor/home.html')
#def register(request):
#    return render(request,'doctor/register.html')

def login2(request):
    if request.method == 'POST':
        email=request.POST['email']
        password=request.POST['password']
        user = auth.authenticate(username=email,password=password)
        if user is not None:
            auth.login(request,user)
            return render(request,"doctor/option.html")
        else :
            messages.info(request,'invalid username or password')
            return render(request,"doctor/log.html")

    else :

        return render(request,"doctor/log.html")
    return render(request,'doctor/log.html')

def register(request):
    if request.method == 'POST':
        first_name=request.POST['first_name']
        last_name=request.POST['last_name']
        email1=request.POST['email']
        email2=request.POST['email2']
        password1=request.POST['password']
        password2=request.POST['password2']
        if password1 == password2 and email1 == email2:
            if User.objects.filter(username=email1):
                #print("Username is taken")
                messages.info(request,'Username is taken')
                return redirect('register')
            else:

                user = User.objects.create_user(username=email1, password = password1, email = email1, first_name=first_name,last_name=last_name)
```

Table 7.10: Doctor module

```
def predBipolar(request):
    a = request.POST['Age']
    b = request.POST['Right_answers']
    c = request.POST['Audio_prosody']
    d = request.POST['Combined_channel']
    e = request.POST['Face_video']
    f = request.POST['Body_video']

    g = request.POST['Positive_valence']
    h = request.POST['Negative_valence']
    i = request.POST['Dominant']
    j = request.POST['Submissive']
    pemail1 = request.POST['pemail']
    docname1 = request.POST['docname']
    reportof = request.POST['reportof']

    lists =[a,b,c,d,e,f,g,h,i,j]
    df = pd.read_csv(r"static/database/Bipolar.csv")
    X_train = df[['Age', 'Right_answers', 'Audio_prosody', 'Combined_channel', 'Face_video','Body_video','Positive_valence','Negative_valence','Dominant','Submissive']]

    Y_train = df[['Type']]
    tree = DecisionTreeClassifier(max_leaf_nodes=6, random_state=0)

    tree.fit(X_train, Y_train)
    prediction = tree.predict([[a,b,c,d,e,f,g,h,i,j]])

    return render(request,'doctor/predictBipolar.html',{"data":prediction,"lists":lists,"a1":a,"b1":b,"c1":c,"d1":d,"e1":e,"f1":f,"g1":g,"h1":h,"i1":i,"j1":j,"pemail1":pe

def bipolarSv(request):
    a1 = request.POST['Age']
    b = request.POST['Right_answers']
    c = request.POST['Audio_prosody']
    d1 = request.POST['Combined_channel']
    e = request.POST['Face_video']
    f = request.POST['Body_video']
    g = request.POST['Positive_valence']
    h = request.POST['Negative_valence']
    i = request.POST['Dominant']
```

Table 7.11: Doctor module

```
s.sendmail(fromaddr, toaddr, text)

print("Msg sent successful")
s.quit()
#saving the data

if len(BipolarReport.objects.filter(patientemail=pemail1)) == 1:
    a = BipolarReport.objects.get(patientemail = pemail1)
    a.docname = docname1
    a.reportof = reportof1
    a.reportnm = filename
    a.Age = a1
    a.Right_answers = b
    a.Audio_prosody = c
    a.Combined_channel = d1
    a.Face_video = e
    a.Body_video = f
    a.Positive_valence = g
    a.Negative_valence = h
    a.Dominant = i
    a.Submissive = j
    a.riskvalue = detail
    a.save()
else:


    d = BipolarReport(patientemail=pemail1,docname=docname1,reportof=reportof1,reportnm=filename,Age=a1,Right_answers=b,Audio_prosody=c,Combined_channel=d1,Face_video
    d.save()




return render(request,'doctor/sendSuccess.html')
```

Table 7.12: Doctor module

This code defines several views for a web application using the Django web framework. The first two views, "reg" and "index", simply render the "doctor/reg.html" and "doctor/index.html" templates, respectively. The "home" view handles a POST request when a user submits a login form. It attempts to authenticate the user using the email and password provided in the form. If the authentication is successful, the user is logged in and the "doctor/option.html" template is rendered. If the authentication is not successful, a message is added to the request and the "doctor/log.html" template is rendered. The "login" view is similar to the "home" view, but it is not clear from the code how it is different.

Patient module

```
from __future__ import unicode_literals
from django.shortcuts import render,redirect
from django.http import HttpResponse
from .models import PatientReg

from doctor.models import BipolarReport

import sqlite3
from django.contrib.auth.models import auth
from django.contrib import messages
loginUser = ""
loginFlag = False

# Create your views here.
def home(request):
    return render(request,'patient/log.html')
def register(request):
    return render(request,'patient/reg.html')
def login2(request):
    return render(request,'patient/log.html')

def pregister(request):
    if request.method == 'POST':
        full_name = request.POST['fname']
        pemail2 = request.POST['pemail']
        ppassword = request.POST['Ppassword']
        phoneno = request.POST['phone']
        address = request.POST['address']
        #report1 = "COPD.pdf"
        #reportof1 = "Copd"
        #doctornm1 = "kaisher"
        new_reg = PatientReg(pname=full_name,pemail=pemail2,pphone=phoneno,password=ppassword,paddress=address)
        new_reg.save()
        '''a = PatientReg.objects.get(pemail=pemail2)
        a.report = report1
        a.reportof = reportof1
        a.doctornm = doctornm1
        a.save()'''
```

Table 7.13: Patient module

```
def bipolarReport(request):
    email = request.POST['pemail']
    a = BipolarReport.objects.get(patientemail = email)
    docname1 = a.docname
    reportof1= a.reportof
    reportnm1 = a.reportnm
    Age = a.Age
    Right_answers = a.Right_answers
    Audio_prosody = a.Audio_prosody
    Combined_channel = a.Combined_channel
    Face_video = a.Face_video
    Body_video = a.Body_video
    Positive_valence = a.Positive_valence
    Negative_valence = a.Negative_valence
    Dominant = a.Dominant
    Submissive = a.Submissive
    riskvalue1 = a.riskvalue


    return render(request,'patient/bipolarReport.html',{"docname1":docname1,"reportof":reportof1,"download":reportnm1,"a1":Age,
                                                        "b1":Right_answers,"c1":Audio_prosody,"d1":Combined_channel,
                                                        "e1":Face_video,"f1":Body_video,"g1":Positive_valence,"h1":Negative_valence,"i1":Dominant,"j1":Submissive,"data":riskv



def login(request):
    #email = request.POST['email']
    #password = request.POST['password']

    '''preg =PatientReg.objects.all()
    if(preg.pemail == email and preg.password == password):

        return render(request,'patient/test.html')
    else:
        return render(request,'patient/home.html')'''
```

Table 7.14: Patient module

```
    else:
        return render(request,'patient/home.html')'''
    global loginFlag,loginUser
    if request.method == 'POST':
        username = request.POST['email']
        password2 = request.POST['password']

        print(username,password2)
        message = ""

        if len(PatientReg.objects.filter(pemail=username)) == 1 and len(PatientReg.objects.filter(pemail=username))  == 1:
            message = message + "Login successful"
            #mail = username
            #a= PatientReg.objects.exclude(pemail = "username")
            #mail = a.pname
            a=PatientReg.objects.get(pemail = username)
            fname = a.pname
            email = a.pemail

            flag = 0
            flag2 = 0
            flag3 = 0
            flag4 = 0
            flag5 = 0

            if len(BipolarReport.objects.filter(patientemail=username)) == 1:
                flag5 = 1
            else:
                flag5 = 0

            #copd1 = copd.objects.get(patientemail=username)
            #report = copd1.reportnm

            return render(request,'patient/reportpage.html',{"b":fname,"flag":flag,"flaglung":flag2,"flagdia":flag3,"flagBio":flag5,"flagheart":flag4,"email":email})
        else:
            #pass_hash = str(PatientReg.objects.filter(pemail=username)[0]).split(";")[4]
            #decrypt_text = pass_hash
```

Table 7.15: Patient module

```
        else:
            #pass_hash = str(PatientReg.objects.filter(pemail=username)[0]).split(";")[4]
            #decrypt_text = pass_hash
            #message = message + "Wrong Password Entered"
            messages.info(request,'invalid username or password')
            return render(request,"patient/log.html")


        print(message)
        context = {"message":message}
        #return render(request,'RTO/login.html',context)
        return render(request,'patient/log.html',context)

    else:
        return render(request,'patient/log.html')
```

Table 7.16: Patient module

This is a Django web application code that handles various patient-related actions such as registration, login, and viewing of a bipolar report. The code includes several views (i.e. functions that handle specific actions in response to specific URLs) such as home, register, pregister, bipolarReport, and login. These views handle rendering different HTML templates, handling form data, and querying the database to retrieve information. The code uses the PatientReg model to handle patient registration and the BipolarReport model to handle the retrieval of a bipolar report. It also uses the django.contrib.auth.models import for authentication purposes.

Django manage.py

manage.py

C: > Users > Jagadish BOB > Desktop > Project > BipolarCassification > manage.py

```
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys


def main():
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'emrnew.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)


if __name__ == '__main__':
    main()

```

Table 7.17: Django manage.py

This code is a script for the command-line utility of Django, a web framework for Python. It sets the default environment variable 'DJANGO SETTINGS MODULE' to 'emrnew.settings' and imports the execute from command line function from Django's management module. The execute from command line function is then called with the command line arguments passed to the script. If there is an import error, it will raise an error message indicating that Django may not be installed or available on the PYTHONPATH. If the script is run as the main program, the main() function is called.

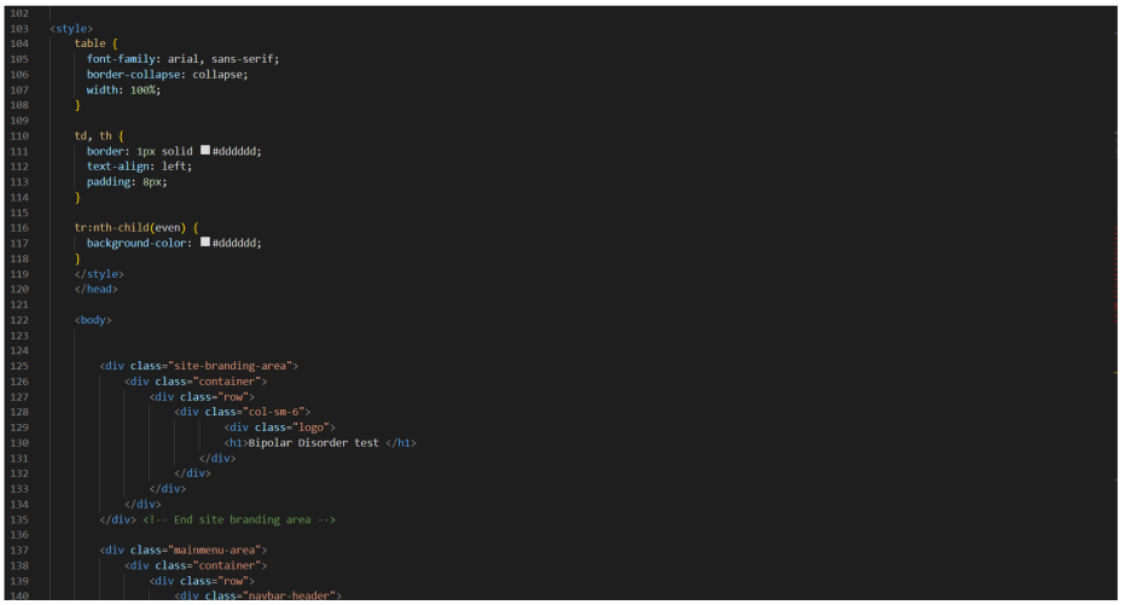
Bipolar test

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <title>Bipolar</title>

        <!-- Google Fonts -->
        <link href='http://fonts.googleapis.com/css?family=Titillium+Web:400,200,300,700,600' rel='stylesheet' type='text/css'>
        <link href='http://fonts.googleapis.com/css?family=Roboto+Condensed:400,700,300' rel='stylesheet' type='text/css'>
        <link href='http://fonts.googleapis.com/css?family=Raleway:400,100' rel='stylesheet' type='text/css'>

        <!-- Bootstrap -->
        <link rel="stylesheet" href="http://maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css">

        <!-- Font Awesome -->
        <link rel="stylesheet" href="http://maxcdn.bootstrapcdn.com/font-awesome/4.3.0/css/font-awesome.min.css">

        <!-- Custom CSS -->
        <link rel="stylesheet" href="css/owl.carousel.css">
        <link rel="stylesheet" href="./static/styles/style.css">
        <link rel="stylesheet" href="css/responsive.css">

<script language="javascript">
            function check_bipolar(){
                //var itemid=document.getElementById("itemid").value;
                var count1=0,count2=0,count3=0,count4=0,count5=0;
                if(document.getElementById('r1').checked == true) {  count1++;}
                if(document.getElementById('r2').checked == true) {  count1++;}
                if(document.getElementById('r3').checked == true) {  count1++;}
                if(document.getElementById('r4').checked == true) {  count1++;}
                if(document.getElementById('r5').checked == true) {  count1++;}
                if(document.getElementById('r6').checked == true) {  count1++;}
                if(document.getElementById('r7').checked == true) {  count1++;}
                if(document.getElementById('r8').checked == true) {  count1++;}
                if(document.getElementById('r9').checked == true) {  count1++;}
                if(document.getElementById('r10').checked == true) {  count1++;}
```

Table 7.18: Bipolar test

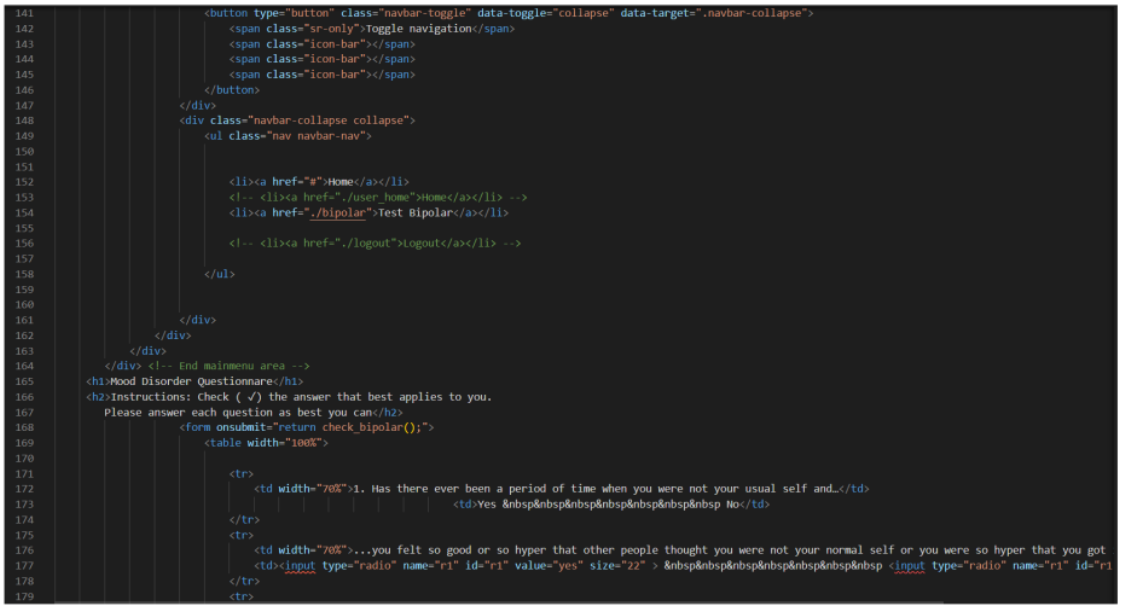
```
                if(document.getElementById('r5th').checked == true) {  count5++;}

                alert(count1);
                alert(count2);
                alert(count3);
                alert(count4);
                alert(count5);
                var check=0;

                if(count1>=7)
                {
                    check++;
                }

                if(count2==1)
                {
                    check++;
                }
                if(count3==1)
                {
                    check++;
                }
                if(count4==1)
                {
                    check++;
                }
                if(count5==1)
                {
                    check++;
                }
                if(check>=3)
                {

                    alert("Bipolar");
                    window.location.replace('./bipolar_msg');
                //   window.location.href='./user_signup';
                }
                else{
```

Table 7.19: Bipolar test



```
<style>
    table {
      font-family: arial, sans-serif;
      border-collapse: collapse;
      width: 100%;
    }

    td, th {
      border: 1px solid #dddddd;
      text-align: left;
      padding: 8px;
    }

    tr:nth-child(even) {
      background-color: #dddddd;
    }
    </style>
    </head>

    <body>


        <div class="site-branding-area">
            <div class="container">
                <div class="row">
                    <div class="col-sm-6">
                            <div class="logo">
                            <h1>Bipolar Disorder test </h1>
                        </div>
                    </div>
                </div>
            </div>
        </div> <!-- End site branding area -->

        <div class="mainmenu-area">
            <div class="container">
                <div class="row">
                    <div class="navbar-header">
```

Table 7.20: Bipolar test



```
                        <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
                            <span class="sr-only">Toggle navigation</span>
                            <span class="icon-bar"></span>
                            <span class="icon-bar"></span>
                            <span class="icon-bar"></span>
                        </button>
                    </div>
                    <div class="navbar-collapse collapse">
                        <ul class="nav navbar-nav">


                            <li><a href="#">Home</a></li>
                            <!-- <li><a href="./user_home">Home</a></li> -->
                            <li><a href="./bipolar">Test Bipolar</a></li>

                            <!-- <li><a href="./logout">Logout</a></li> -->

                        </ul>


                    </div>
                </div>
            </div>
        </div> <!-- End mainmenu area -->
    <h1>Mood Disorder Questionnare</h1>
    <h2>Instructions: Check ( ✓) the answer that best applies to you.
       Please answer each question as best you can</h2>
                <form onsubmit="return check_bipolar();">
                    <table width="100%">

                        <tr>
                            <td width="70%">1. Has there ever been a period of time when you were not your usual self and…</td>
                                                    <td>Yes         No</td>
                        </tr>
                        <tr>
                            <td width="70%">...you felt so good or so hyper that other people thought you were not your normal self or you were so hyper that you got
                            <td><input type="radio" name="r1" id="r1" value="yes" size="22" >          <input type="radio" name="r1" id="r1
                        </tr>
                        <tr>
```

Table 7.21: Bipolar test

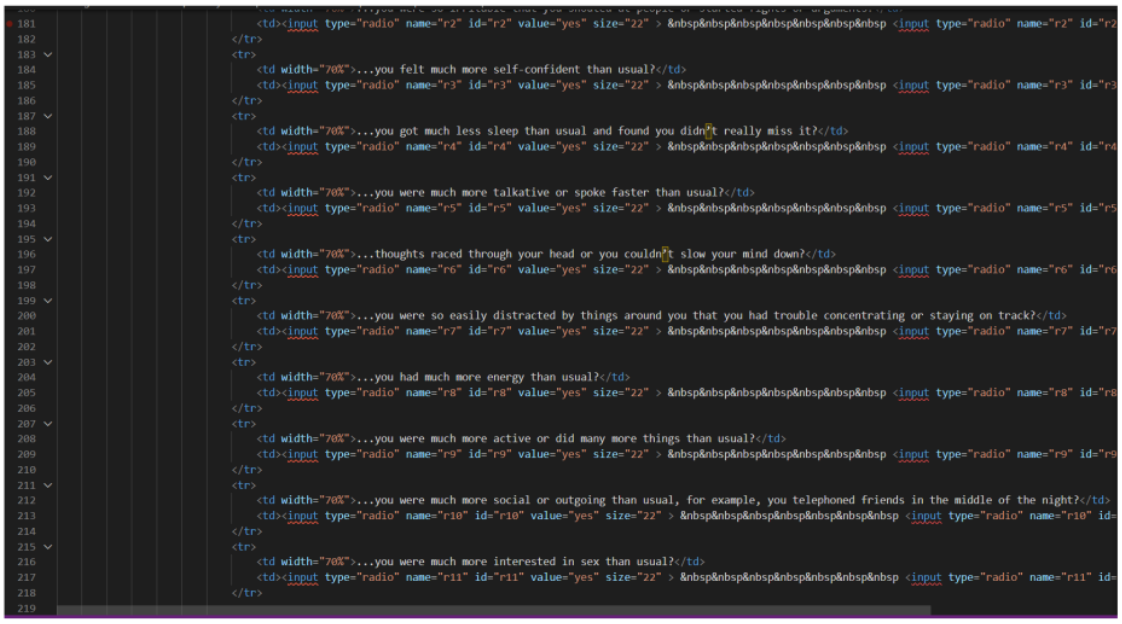


Table 7.22: Bipolar test

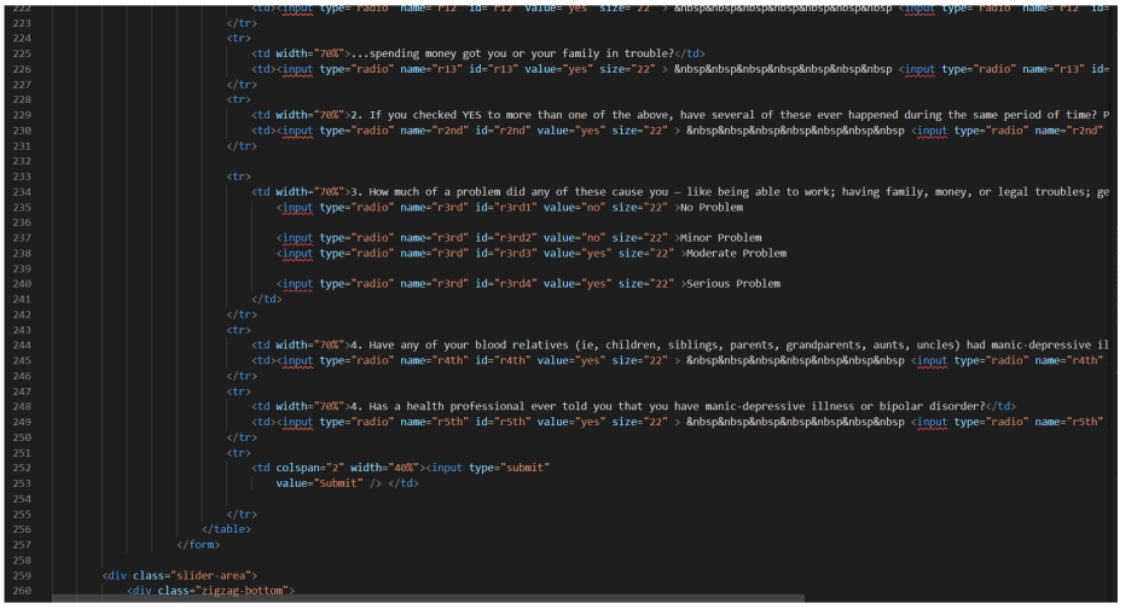


Table 7.23: Bipolar test

A patient can take the bipolar test.This code is written in HTML and is for a website's home page. It includes links to various CSS and JS files, including bootstrap and jQuery. The page displays a title "Bipolar disorder Analysis System" and includes a clock and a message welcoming a user and mentioning that the site is a tool for predicting reports in real-time. There is also a sign-in section on the page. The code uses Django's template language to load static files.

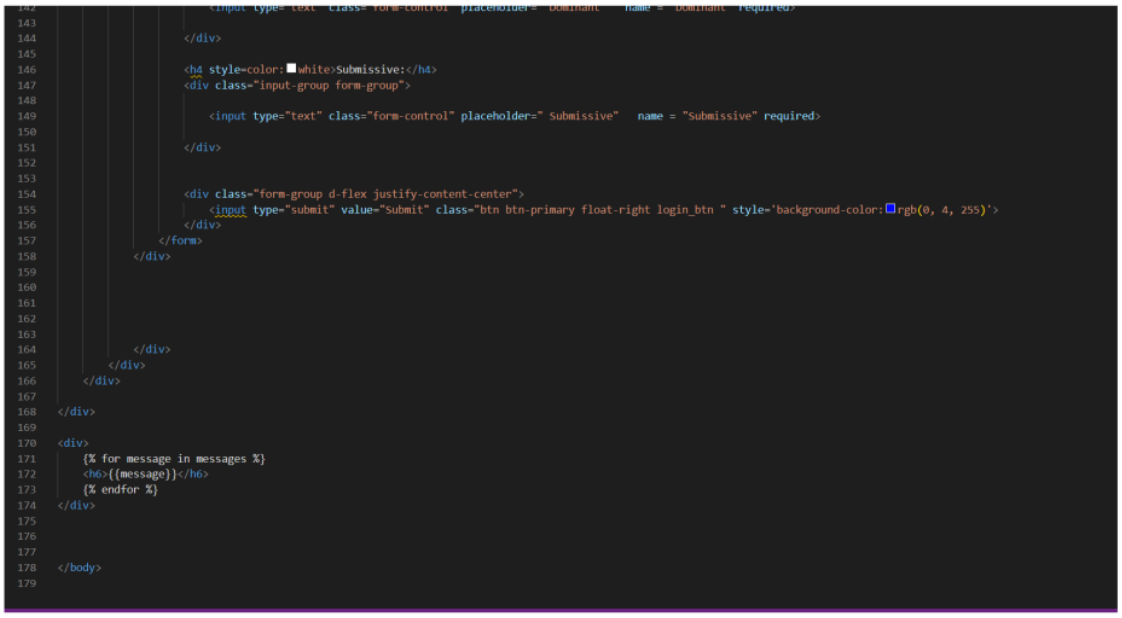
Report generation

```
<div class="d-flex justify-content-center h-100 float-md-center option-doctor">



    <div class="card"  style="width: 50rem; height: 90rem;">
        <div class="card-header">
            <h3>Fill the details</h3>

        </div>

        <div class="card-body" >
            <form action="predBipolar" method="POST">
                {% csrf_token %}

                <h4 style=color:white>Doctor name:</h4>
                <div class="input-group form-group">

                    <input type="text" class="form-control" placeholder="Doctor name"   name = "docname" required>

                </div>

                <h4 style=color:white>Patient email:</h4>
                <div class="input-group form-group">

                    <input type="text" class="form-control" placeholder="Patient email:"   name = "pemail" required>

                </div>

                <h4 style=color:white>Report of:</h4>
                <div class="input-group form-group">

                    <input type="text" class="form-control" placeholder="Report of"   name = "reportof" value = "Bipolar Disease" readonly>

                </div>


                <h4 style=color:white>Age:</h4>
                <div class="input-group form-group">
```

Table 7.24: Report generation

```
                    <input type="text" class="form-control" placeholder="Age"   name = "Age" required>

                </div>
                <h4 style=color:white>Right_answers:</h4>
                <div class="input-group form-group">

                    <input type="text" class="form-control" placeholder="Right_answers"   name = "Right_answers" required>

                </div>
                <h4 style=color:white>Audio_prosody:</h4>
                <div class="input-group form-group">

                    <input type="text" class="form-control" placeholder="Audio_prosody"   name = "Audio_prosody" required>

                </div>
                <h4 style=color:white>Combined_channel:</h4>
                <div class="input-group form-group">

                    <input type="text" class="form-control" placeholder="Combined_channel"   name = "Combined_channel" required>

                </div>
                <h4 style=color:white>Face_video:</h4>
                <div class="input-group form-group">

                    <input type="text" class="form-control" placeholder=" Face_video"   name = "Face_video" required>

                </div>
                <h4 style=color:white>Body_video:</h4>
                <div class="input-group form-group">

                    <input type="text" class="form-control" placeholder=" Body_video"   name = "Body_video" required>

                </div>

                <h4 style=color:white>Positive_valence:</h4>
                <div class="input-group form-group">

                    <input type="text" class="form-control" placeholder=" Positive_valence"   name = "Positive_valence" required>
```

Table 7.25: Report generation

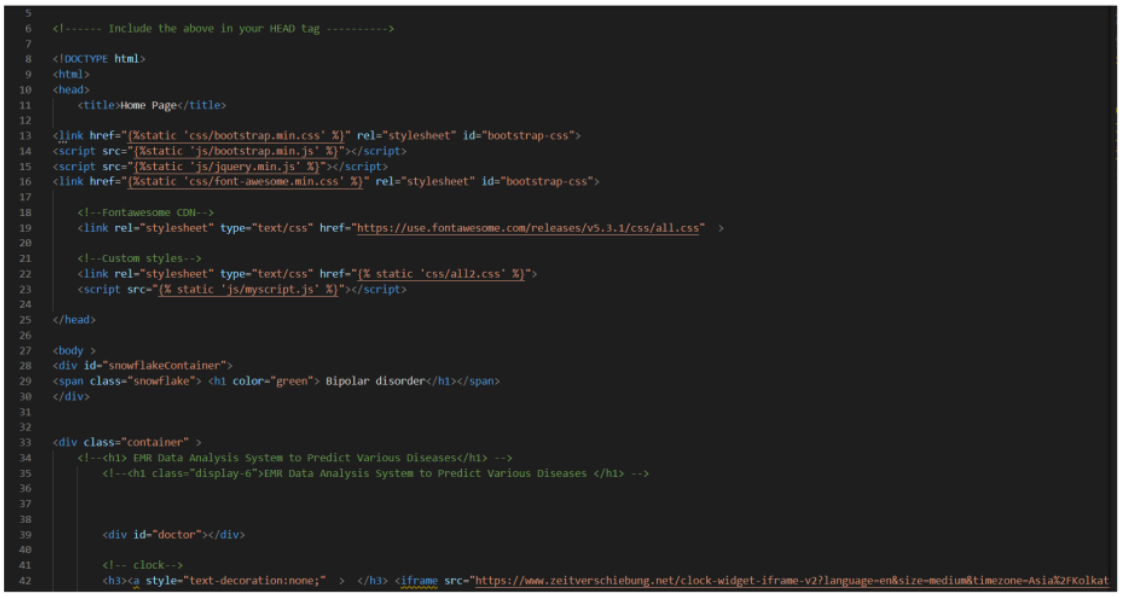


```
                    <input type="text" class="form-control" placeholder=" Dominant"   name = "Dominant" required>

                </div>

                <h4 style=color:white>Submissive:</h4>
                <div class="input-group form-group">

                    <input type="text" class="form-control" placeholder=" Submissive"   name = "Submissive" required>

                </div>


                <div class="form-group d-flex justify-content-center">
                    <input type="submit" value="Submit" class="btn btn-primary float-right login_btn " style='background-color:rgb(0, 4, 255)'>
                </div>
            </form>
        </div>




        </div>
    </div>
</div>

</div>

<div>
    {% for message in messages %}
    <h6>{{message}}</h6>
    {% endfor %}
</div>


</body>

```

Table 7.26: Report generation

This code is a html template for a webpage that has a form to fill in details of a patient's bipolar disorder report. The template uses the Django web framework,After the doctor take the details of the patient he can generate a report for future use.

Doctor Homepage



```

<!------ Include the above in your HEAD tag ---------->

<!DOCTYPE html>
<html>
<head>
    <title>Home Page</title>

<link href="{%static 'css/bootstrap.min.css' %}" rel="stylesheet" id="bootstrap-css">
<script src="{%static 'js/bootstrap.min.js' %}"></script>
<script src="{%static 'js/jquery.min.js' %}"></script>
<link href="{%static 'css/font-awesome.min.css' %}" rel="stylesheet" id="bootstrap-css">

    <!--Fontawesome CDN-->
    <link rel="stylesheet" type="text/css" href="https://use.fontawesome.com/releases/v5.3.1/css/all.css"  >

    <!--Custom styles-->
    <link rel="stylesheet" type="text/css" href="{% static 'css/all2.css' %}">
    <script src="{% static 'js/myscript.js' %}"></script>

</head>

<body >
<div id="snowflakeContainer">
<span class="snowflake"> <h1 color="green"> Bipolar disorder</h1></span>
</div>


<div class="container" >
    <!--<h1> EMR Data Analysis System to Predict Various Diseases</h1> -->
        <!--<h1 class="display-6">EMR Data Analysis System to Predict Various Diseases </h1> -->



        <div id="doctor"></div>

        <!-- clock-->
        <h3><a style="text-decoration:none;"  >  </h3> <iframe src="https://www.zeitverschiebung.net/clock-widget-iframe-v2?language=en&size=medium&timezone=Asia%2FKolkat
```

Table 7.27: Doctor Homepage

```
<div id="doctor"></div>

<!-- clock-->
<h3><a style="text-decoration:none;"  >  </h3> <iframe src="https://www.zeitverschiebung.net/clock-widget-iframe-v2?language=en&size=medium&timezone=Asia%2FKolkat

<!--First container -->
<div class="d-flex justify-content-center h-100 float-md-left option-doctor">
    <div class="card" style="width: 40rem;">
        <div class="card-header">
        <div class="row align-items-center remember" align="center">

            <h3>Bipolar disorder Analysis System </h3>
        </div>
        </div>

        <div class="card-body" >

            <div class="alert alert-success" role="alert">
                <h4 class="alert-heading">Welcome doctor</h4>
                <p>We make a tool to predict the report in real time which help you make the report based on the data what you inserted .</p>
                <hr>
                <p class="mb-0">We are woking hard to add more disease to predic through the machine learning.</p>
                 <hr>
            </div>


        </div>

    </div>
</div>



<div class="d-flex justify-content-center h-100 float-md-right option-doctor">
```

Table 7.28: Doctor Homepage

```
        <div class="card-body" >
            <form action="login" method="POST">
                {% csrf_token %}
                {% for message in messages %}
                    <h6 style=color:red;>{{message}}</h6>
                {% endfor %}

                <div class="input-group form-group">

                    <div class="input-group-prepend">
                        <span class="input-group-text" ><i class="fa fa-user icon" style='color:blue'></i></span>
                    </div>
                    <input type="email" class="form-control" placeholder="Email id" id="EMail" name = "email" required>

                </div>
                <div class="input-group form-group" >
                    <div class="input-group-prepend"  >
                        <span class="input-group-text" style='color:red'><i class="fas fa-key" style='color:blue'></i></span>
                    </div>
                    <input type="password" class="form-control" placeholder="password" name = "password" required>
                </div>
                <div class="row align-items-center remember">
                    <input type="checkbox">Remember Me
                </div>
                <div class="form-group">
                    <input type="submit" value="Login" class="btn float-right login_btn" style='background-color:red'>
                </div>
            </form>
        </div>

        <div class="card-footer">
            <div class="d-flex justify-content-center links" style='background-color:green'>
                Don't have an account?<a href="register">Sign Up</a>
            </div>
```

Table 7.29: Doctor Homepage

This code is written in HTML and is for a website's home page. It includes links to various CSS and JS files, including bootstrap and jQuery. The page displays a title "Bipolar disorder Analysis System" and includes a clock and a message welcoming a user and mentioning that the site is a tool for predicting reports in real-time. There is also a sign-in section on the page. The code uses Django's template language to load static files.

7.2 Screenshots

Doctor Homepage

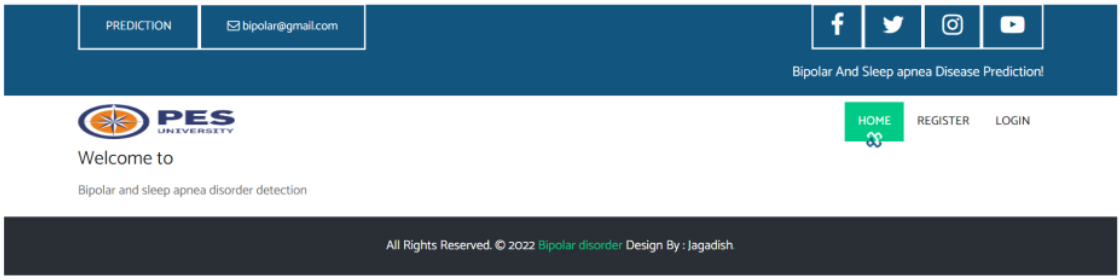


Table 7.30: Doctor Homepage

This is the homepage for Doctor. where a new doctor can register or an existing doctor can login.

Register new user

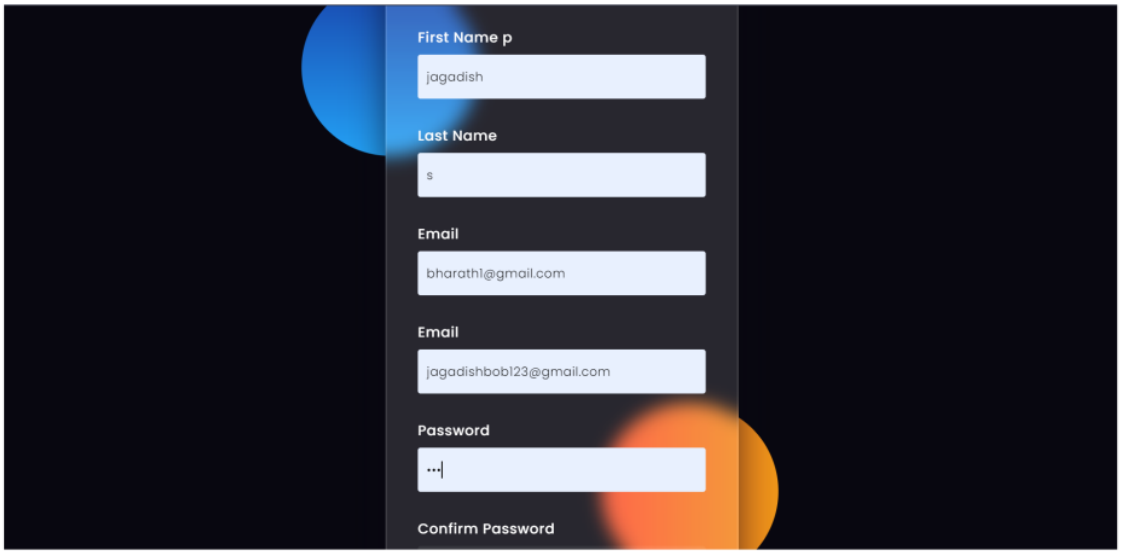


Table 7.31: Register new user

A new doctor can register by giving the required credentials.

Doctor login

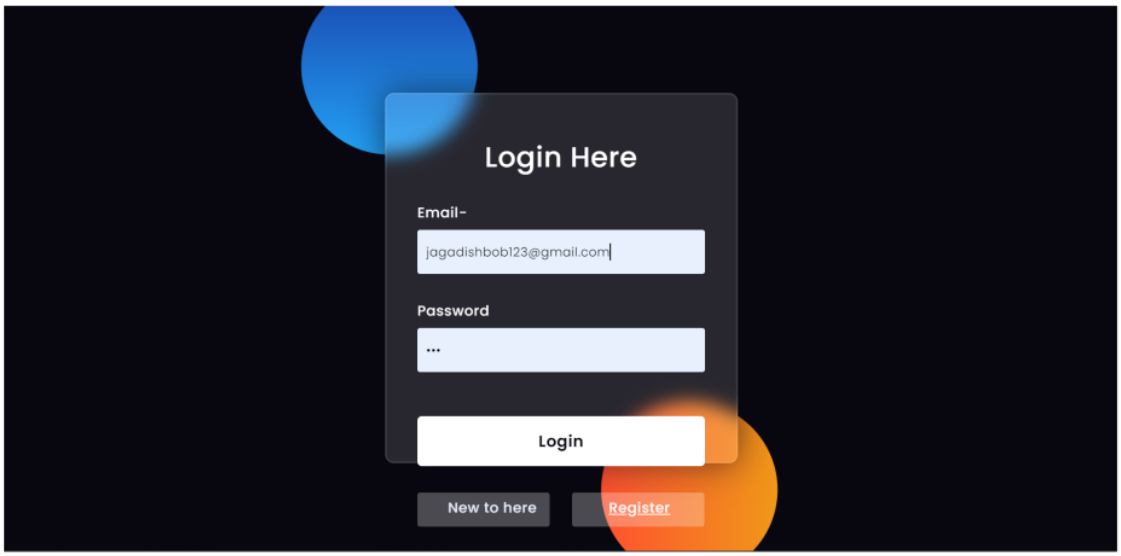


Table 7.32: Doctor login

After the registration is done the doctor can login and logout.

Select Disorder

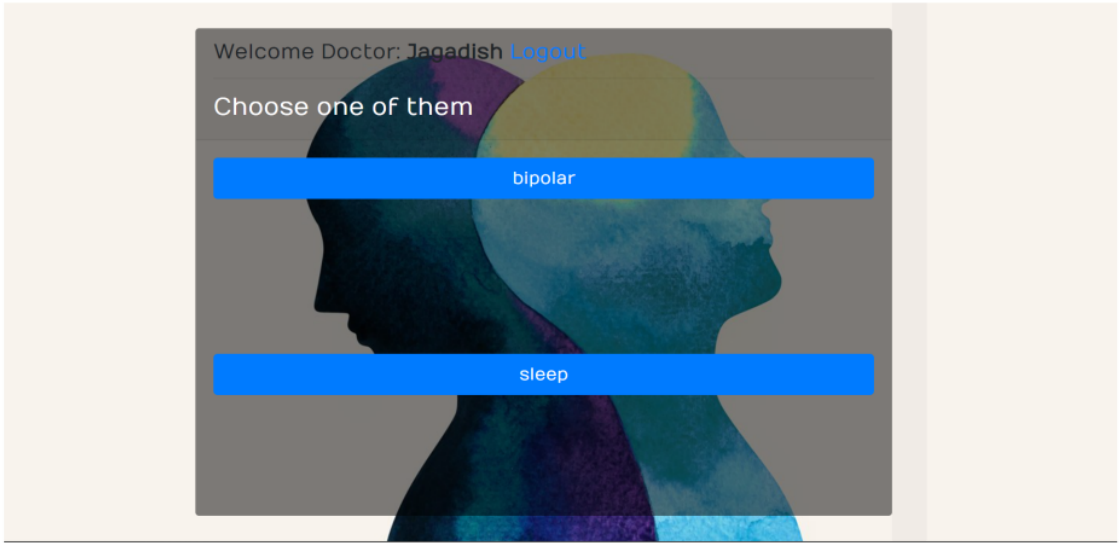


Table 7.33: Select Disorder

Doctor can select the Disorder.

Bipolar Test

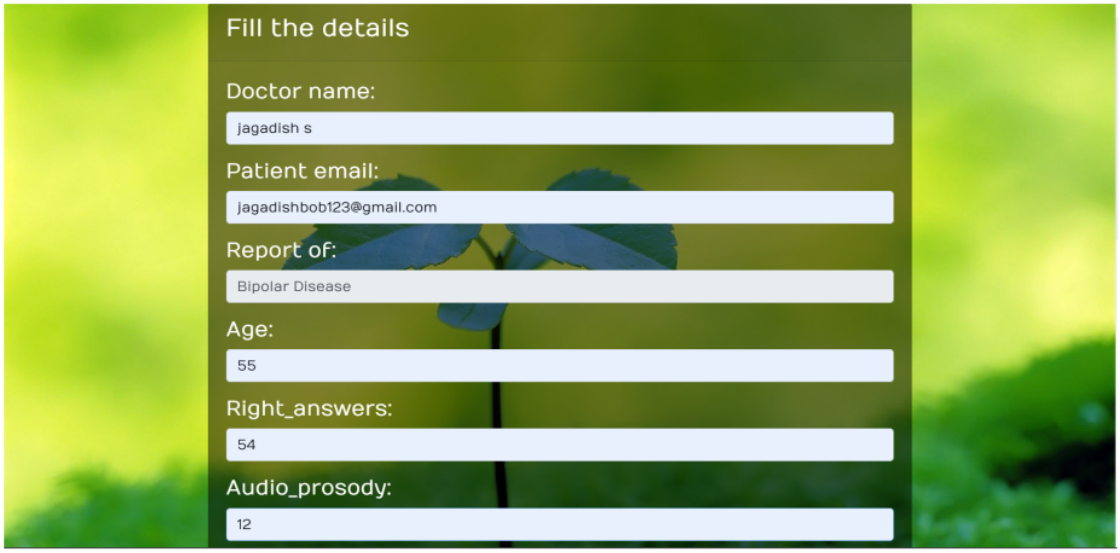


Table 7.34: Bipolar Test

After the Doctor has examined the patient the doctor can fill the input fields and predict the results.

Sleep apnea

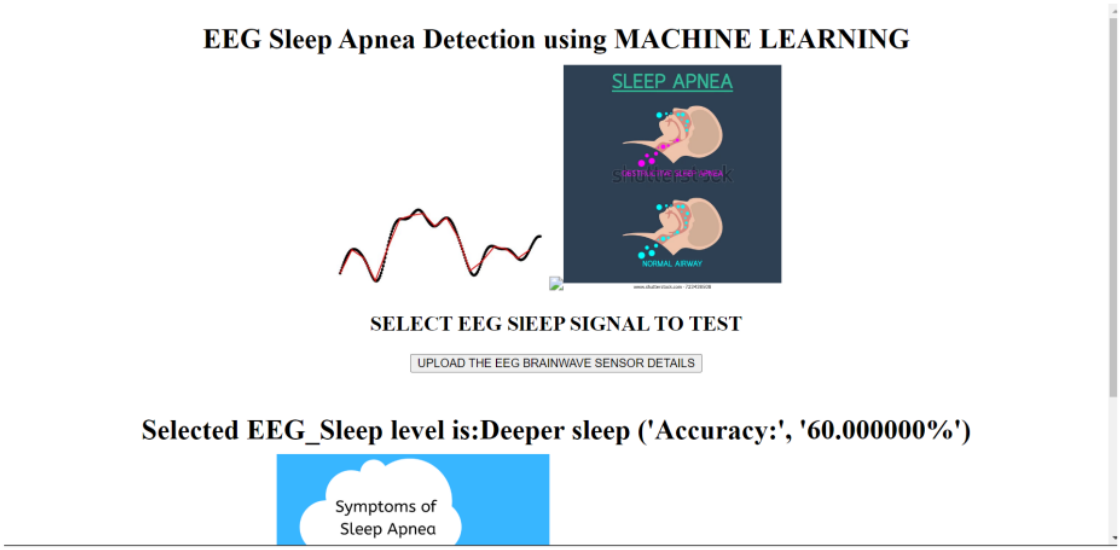


Table 7.35: Sleep apnea

Uploading the csv file which containts the sleep apnea test data it can predict the disorder.

Patient login

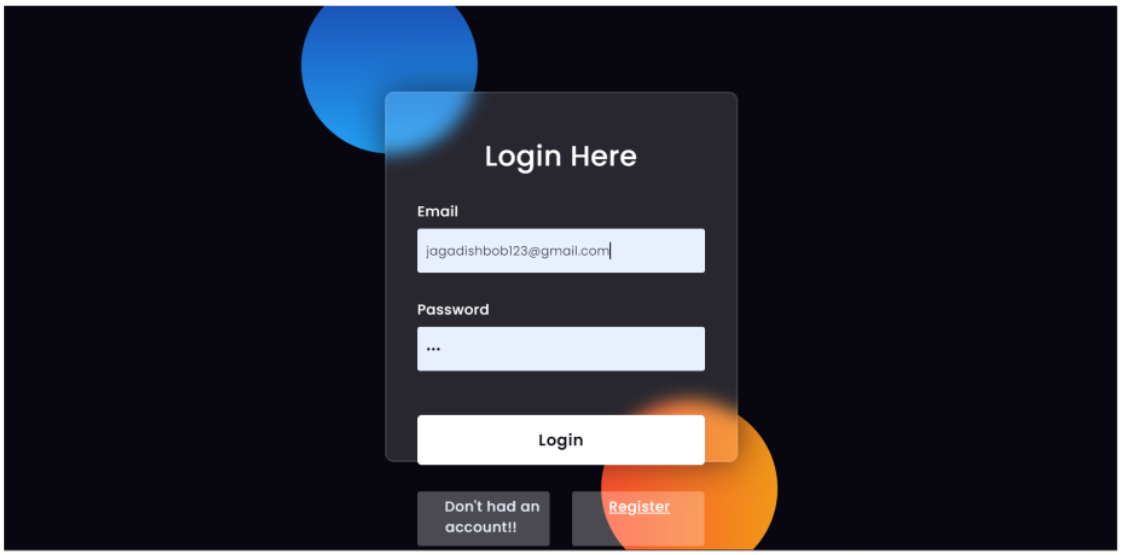


Table 7.36: Patient login

After the registration is done the patient can login and logout.

Patient Homepage

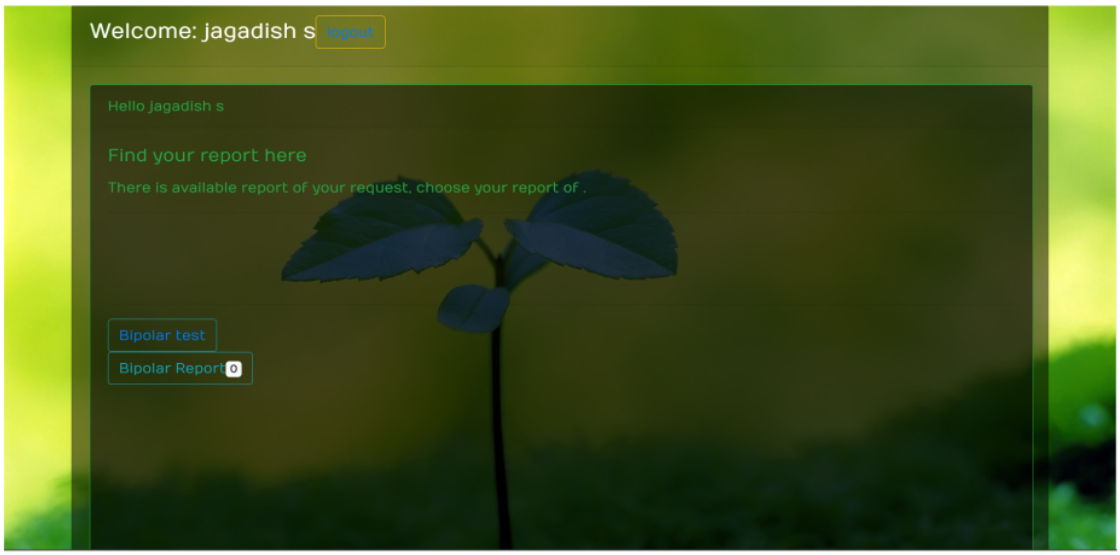


Table 7.37: Patient Homepage

This is the homepage for patient. where the patient can take the bipolar test and the patient can view the report generated by the doctor.

Bipolar initial test

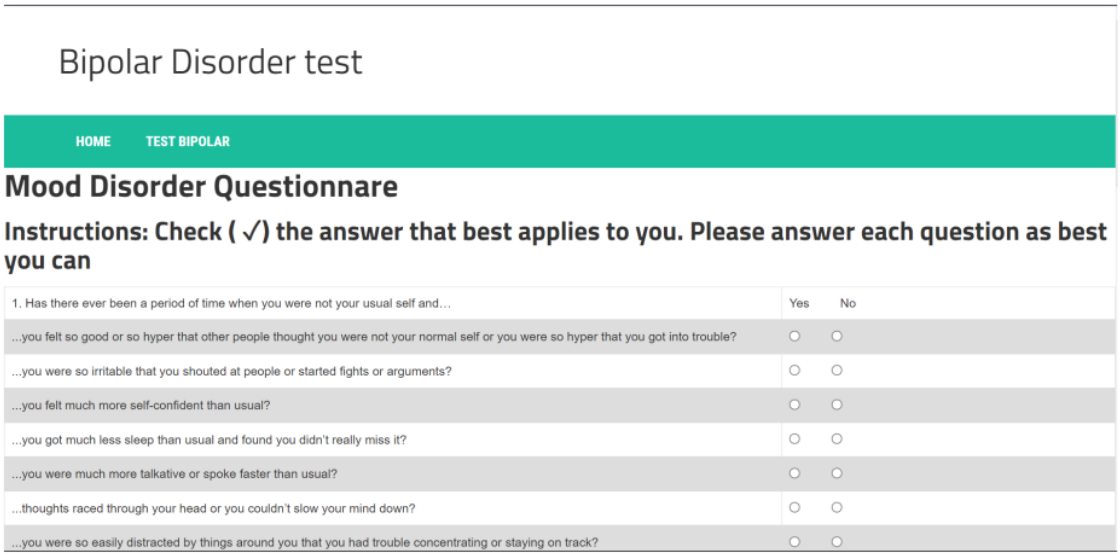


Table 7.38: Bipolar initial test

Patient can take the bipolar test before going to the hospital.

Report

Table 7.39: Report

Patient can view the report generated by the doctor.

-

Chapter 8

Testing

8.1 Testing

Test Case : Doctor Module Registration Form

Test cases are carried out in order to validate the data entered in the doctor registration form

MANUAL TESTING					
MODULE NAME					
TEST ID		TC 1			
TEST NAME		Doctor Module Registration Form			
TEST DESCRIPTION		Test cases to validate Doctor Module Registration form			
STEP#	TEST STEPS	TEST DATA	EXPECTED RESULT	ACTUAL RESULT	STATUS
1	Register Doctor	First name, last Name, incorrect Email ID, Password	Invalid Email ID	Invalid Email ID	PASS
2	Register Doctor	First name, last Name, Correct Email ID, Password	Account created	Account created	PASS
3	Register Doctor	First name, last Name, Email ID, Password length less than 8 characters	Invalid password ID	Invalid password ID	PASS
4	Register Doctor	First name, last name, Email ID, Password length greater than 8 characters	Account created	Account created	PASS
5	Register Doctor	First name, last name, Email ID, Password, incorrect password	Password Mismatched	Password Mismatched	PASS
6	Register Doctor	First name, last name, Email ID, incorrect Email ID, Password	EMail ID Mismatched	EMail ID Mismatched	PASS

Table 8.1: Test Case : Doctor Module Registration Form

Test Case : Login Page

Test cases are carried out in order to validate the information entered on the login page for the doctor and patient.

MANUAL TESTING					
MODULE NAME					
TEST ID		TC 2			
TEST NAME		Login Page			
Test cases to validate Login Page.		enter —			
STEP#	TEST STEPS	TEST DATA	EXPECTED RESULT	ACTUAL RESULT	STATUS
1	Login	Incorrect Email, Password	Invalid credentials	Invalid credentials	PASS
2	Login	Email, Incorrect Password	Invalid credentials	Invalid credentials	PASS
3	Login	Email, Password	Logged in	Logged in	PASS

Table 8.2: Test Case : Login Page

Test Case: Bipolar Patient Information page

Test cases are carried out in order to validate the details entered by the doctor in the Biploar information page about patients.

MANUAL TESTING					
MODULE NAME					
TEST ID		TC 3			
TEST NAME		Bipolar patient details page validation			
Test cases to validate Bipolar patient details page.		enter —			
STEP#	**TEST STEPS**	**TEST DATA**	**EXPECTED RESULT**	**ACTUAL RESULT**	**STATUS**
1	Bioplar Patient information page	Missed any one fields	Please fill out this field	Please fill out this field	PASS
2	Bioplar Patient information page	All input fields filled	Succesfully submitted	Succesfully submitted	PASS

Table 8.3: Test Case : Bipolar Patient Information page

Test Case: Sleep Apnea Detection Page

Test cases are carried out in order to check whether the results displayed in the page is according to the flow of process.

MANUAL TESTING					
MODULE NAME					
TEST ID		TC 4			
TEST NAME		Sleep Apnea Detection Page			
Test cases to perform sleep apnea detection		enter —			
STEP#	**TEST STEPS**	**TEST DATA**	**EXPECTED RESULT**	**ACTUAL RESULT**	**STATUS**
1	Upload EEG Data	csv file	Displays result	Displays result	PASS
2	Upload EEG Data	CSV file with invalid data	No result	No result	PASS

Table 8.4: Test Case : Sleep Apnea Detection Page

Test Case: Sleep Apnea Detection Page

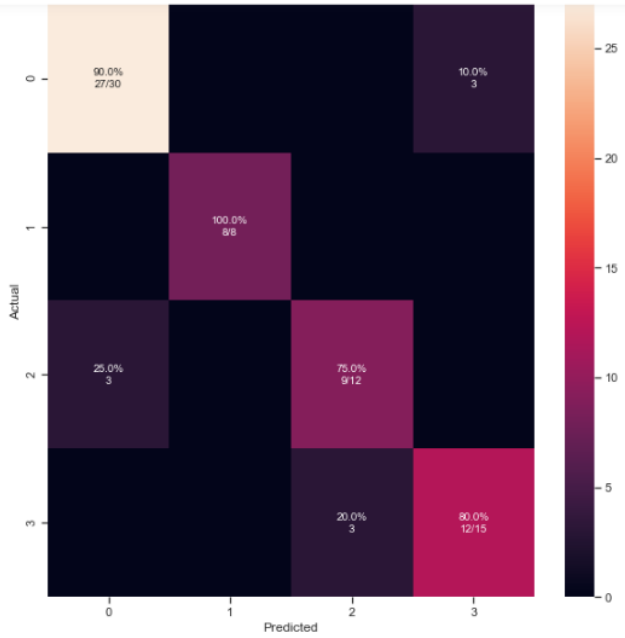
Test cases are carried out in order to check whether the results displayed in the page is according to the flow of process.

MANUAL TESTING					
MODULE NAME					
TEST ID		TC 4			
TEST NAME		Sleep Apnea Detection Page			
Test cases to perform sleep apnea detection		enter —			
STEP#	**TEST STEPS**	**TEST DATA**	**EXPECTED RESULT**	**ACTUAL RESULT**	**STATUS**
1	Upload EEG Data	csv file	Displays result	Displays result	PASS
2	Upload EEG Data	CSV file with invalid data	No result	No result	PASS

Table 8.5: Test Case : Sleep Apnea Detection Page

8.2 Model Testing

8.2.1 Confusion matrix




```
F1 Score is  : 86.250000 %
Confusion Matrix
[[27  0  0  3]
 [ 0  8  0  0]
 [ 3  0  9  0]
 [ 0  0  3 12]]
False positive rate : 0.000000 %
False negative rate : 0.000000 %

Winner algorithm is RF with a 86.153846 % success
False positive rate : 0.000000 %
False negative rate : 0.000000 %
```

Table 8.6: Confusion Matrix

```
In [23]: acc_rc = accuracy_score(y_true=y_test, y_pred= y_pred)
         print("Overall accuracy of DC model using test-set is : %f" %(acc_rc*100))

         Overall accuracy of DC model using test-set is : 81.060606

In [24]: print(classification_report(y_pred, y_test))
```

	precision	recall	f1-score	support
0	0.94	0.82	0.88	74
1	0.77	0.83	0.80	41
2	0.12	0.50	0.20	2
3	0.43	0.43	0.43	7
4	1.00	1.00	1.00	8
accuracy			0.81	132
macro avg	0.65	0.72	0.66	132
weighted avg	0.85	0.81	0.83	132

Table 8.7: Accuracy rate

Chapter 9

Conclusion

The main purpose of developing this application is to make people provide automatic results of disorders. This will help patients to easily see their reports online and know about their disorder stage. Secondly, doctors would be easier to know what their patients are suffering from disorders and which stage it is. It helps to take necessary action to cure the disorders.

Chapter 10

Future Work

This application may be expanded in the future to include the ability to anticipate additional illnesses, such as diabetes and lung cancer. In the future, they will be able to predict Bipolar disorder using supervised learning algorithms. Online therapy will eventually be included to this application. In the future, adding AI algorithms will assist to forecast illnesses more precisely and enhance the outcomes. In order for the findings to more reliably predict illnesses in the future, more diseases should be present, and the list of other biomarkers should also be enlarged. Even other chronic diseases like arthritis, cardiopathy, and cancer should be monitored for early detection phases. This is the main hope for screening against cardiopathy and cancer in the near future. Because of the natural environment and people's eating habits, there are a great number of opportunities for people to be afflicted by a variety of diseases. Therefore, if they wish to protect themselves, they should use this machine learning technique to take a measurement of the current state of health. It is helpful in the prevention of diseases as theyll as in the treatment of their various stages.

PES1PG21CA145_jagadish_plag

ORIGINALITY REPORT

4%	2%	1%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Visvesvaraya Technological University, Belagavi Student Paper	1%
2	www.coursehero.com Internet Source	1%
3	Daniel Rubio. "Beginning Django", Springer Science and Business Media LLC, 2017 Publication	1%
4	publications.aston.ac.uk Internet Source	<1%
5	Submitted to University of Sheffield Student Paper	<1%
6	www.scribd.com Internet Source	<1%
7	eprints.utar.edu.my Internet Source	<1%
8	Submitted to Colorado State University, Global Campus Student Paper	<1%

9	Submitted to National College of Ireland Student Paper	<1%
10	www.slideshare.net Internet Source	<1%
11	1library.net Internet Source	<1%
12	wiki.smu.edu.sg Internet Source	<1%
13	s3-eu-west-1.amazonaws.com Internet Source	<1%
14	www.cse.iitd.ac.in Internet Source	<1%
15	"Knowledge Innovation Through Intelligent Software Methodologies, Tools and Techniques", IOS Press, 2020 Publication	<1%
16	Vahid Mohsenin. "Sleep-Related Breathing Disorders and Risk of Stroke", Stroke, 2001 Publication	<1%

Exclude quotes	On	Exclude matches	< 5 words
Exclude bibliography	On		