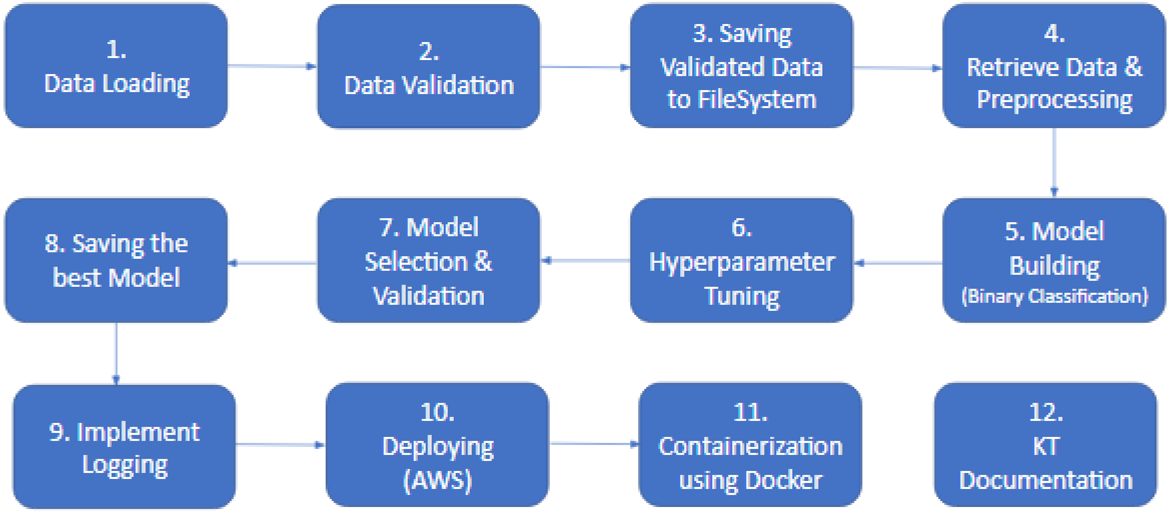
Problem Statement

To build a classification methodology to predict the chances of Epileptic Disorder based on the given training data.

The dataset contains a hashed patient ID column, 178 EEG readings over one second, and a Y output variable describing the status of the patient at that second. When a patient is having a seizure, Y is denoted as 1 while all other numbers are other statuses which means that the person is not having the disorder. So when we turn our Y variable into a binary variable, this problem becomes a binary classification problem.

Architecture


1. Data Loading
2. Data Validation
3. Saving Validated Data to FileSystem
4. Retrieve Data & Preprocessing
5. Model Building (Binary Classification)
6. Hyperparameter Tuning
7. Model Selection & Validation
8. Saving the best Model
9. Implement Logging
10. Deploying (AWS)
11. Containerization using Docker
12. KT Documentation

Data Description

The client will send data in multiple sets of files in batches at a given location. Data will contain “unnamed” column which is the patient id and 178 columns of 178 EEG readings over one second. The last column, Y is output variable describing the status of the patient at that second.

"Y" column will has five unique values 1 , 2 , 3, 4, 5.

5 - eyes open, means when they were recording the EEG signal of the brain the patient had their eyes open
4 - eyes closed, means when they were recording the EEG signal the patient had their eyes closed

3 - Yes they identify where the region of the tumor was in the brain and recording the EEG activity from the healthy brain area
2 - They recorder the EEG from the area where the tumor was located
1 - Recording of seizure activity

All patients falling in classes 2, 3, 4, and 5 are subjects who did not have epileptic seizure. Hence the ‘2,3,4,5’ classes can be classified as ‘With no disorder’ and has been changed to class ‘0’.

Only subjects in class 1 have epileptic seizure.

So in summary,

"1" represents “Patient has the Epileptic disorder”.

"0" represents “Patient does not hase the Epileptic disorder”.

Number of Columns and their datatype to be described and adhered.

Data Validation

In this step, we perform different sets of validation on the given set of training files.

1. Number of Columns - We validate the number of columns present in the files, and if it doesn't match with the value given in the schema file, then the file is moved to "Bad_Data_Folder."

2. The datatype of columns - The datatype of columns is given in the schema file. This is validated when we insert the files into Database. If the datatype is wrong, then the file is moved to "Bad_Data_Folder".

3. Null values in columns - If any of the columns in a file have all the values as NULL or missing, we discard such a file and move it to "Bad_Data_Folder".

Model Training

1) Data Reading - The pre-processed data is read as a CSV file to be used for model training.

2) Data Preprocessing

a) Check for null values in the columns. If present, impute the null values using the KNN imputer. No null values present in the current data provided for training & testing. So this step is not needed.

b) Check for the duplicate columns. If present, remove the duplicate columns. No duplicate column present in the given dataset.

c) Check for distribution of data for Mean and Median values for columns. Data is distributed normally with respect to mean. But the data is skewed with respect to standard deviation.

d) Check for the duplicate columns. If present, remove the duplicate columns. No duplicate column present in the given dataset.

3) Outlier Detection – Large number of outliers for each feature are present. Removal of outliers will lead to significant data loss.

4) Tain & Test Split – Perform 75:25 Train & Test split.

5) Standardization – Scale down the data using standard scalar to address the data skewness observed at median values.

6) Multi Collinearity – Check if the data has collinearity among the among the independent variables using the VIF function. It has been found out the all the features have high collinearity.

Feature having minimum VIF is X1 and VIF score is 77.54073101770086

Feature having maximum VIF is X162 and VIF score is 948.8775562775533

7) Principal Componenet Analysis – Since a strong co-linearity exists between all the features, perform

8) Class Imbalance – Class imbalance seen with 70% data belonging to class '0' and 30% data belonging to class '1'. Class Imbalance issue has been addressed 'Using Weights' as well as 'Applying SMOTE' on the train data. The method that gives better results will be taken.

9) Model Training & Selection – We are using two algorithms, "Random Forest Classifier" and "Support Vector Classifoer". Both the algorithms are passed with the best parameters derived from RandomizedSearchCV. We calculate the AUC scores for both models and select the model with the best score. Model with the best score is saved for use in prediction.

Prediction Data Description

Client will send the data in form of csv files in batches at a given location OR will upload the data in the form of csv file at the UI. Data will contain 'Unnamed columns with Patient ID' and 178 columns of different EEG reading for each patient.

Number of Columns, Name of the Columns and their datatype.

Data Validation

In this step, we perform different sets of validation on the given set of training files.

1. Number of Columns - We validate the number of columns present in the files, and if it doesn't match with the value given in the schema file, then the file is moved to "Bad_Data_Folder."

2. The datatype of columns - The datatype of columns is given in the schema file. This is validated when we insert the files into Database. If the datatype is wrong, then the file is moved to "Bad_Data_Folder".

3. Null values in columns - If any of the columns in a file have all the values as NULL or missing, we discard such a file and move it to "Bad_Data_Folder".

Prediction

1) Data Reading - The pre-processed data is read as a CSV file to be used for prediction.

2) Data Preprocessing

a) Check for null values in the columns. If present, impute the null values using the KNN imputer. No null values present in the current data provided for training & testing.

b) Check for the duplicate columns. If present, remove the duplicate columns. No duplicate column present in the given dataset.

c) Check for distribution of data for Mean and Median values for columns. Data is distributed normally with respect to mean. But the data is skewed with respect to median.

d) Check for the duplicate columns. If present, remove the duplicate columns. No duplicate column present in the given dataset.

3) Prediction – The final selected model is loaded and is used to predict the data for that cluster.

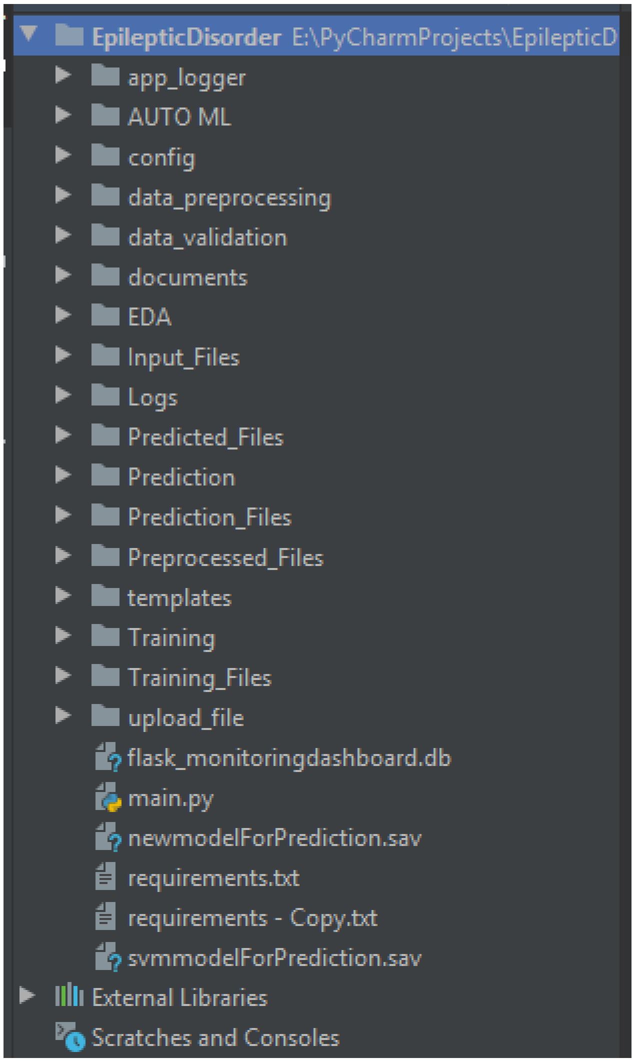
4) Once the prediction is made for all patients, the predictions are saved in a CSV file at a given location and the location is returned to the client.

Deployment Readyness

We will be deploying the model to the AWS platform.

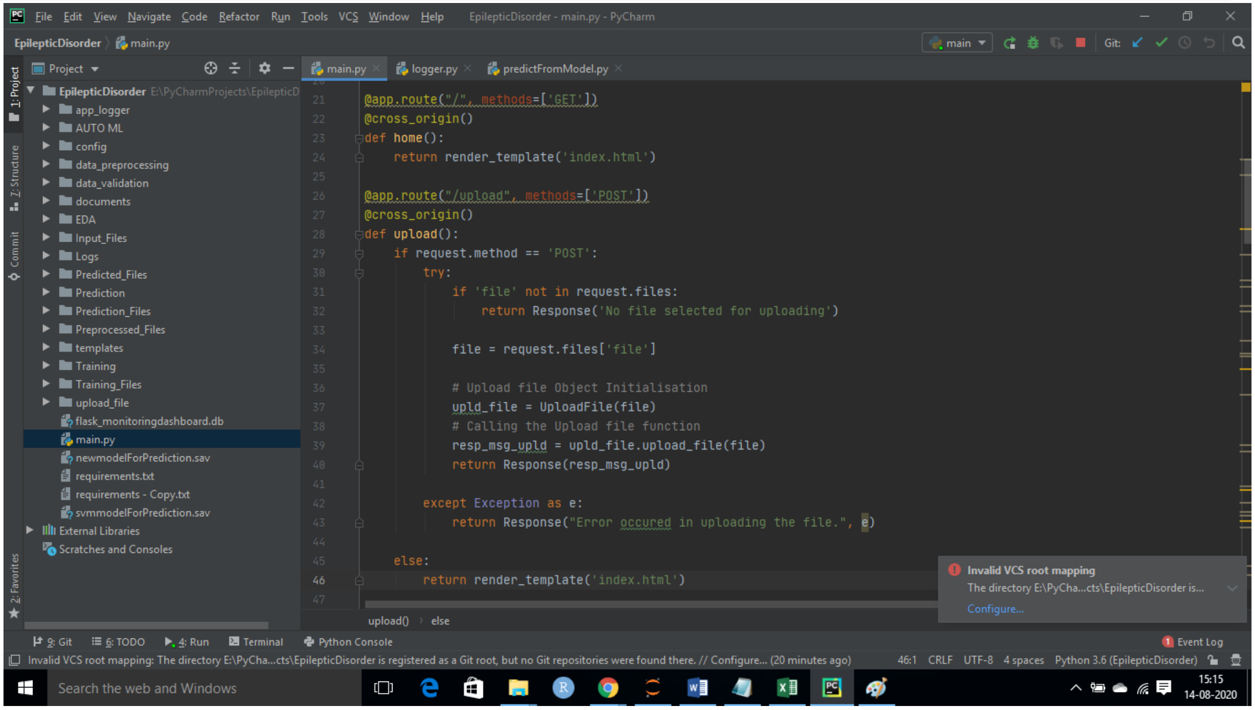
This is a workflow diagram for the prediction of using the trained model.

Now let's see the Epileptic disorder project folder structure.



```
▼ EpilepticDisorder  E:\PyCharmProjects\EpilepticD
    ▶ app_logger
    ▶ AUTO ML
    ▶ config
    ▶ data_preprocessing
    ▶ data_validation
    ▶ documents
    ▶ EDA
    ▶ Input_Files
    ▶ Logs
    ▶ Predicted_Files
    ▶ Prediction
    ▶ Prediction_Files
    ▶ Preprocessed_Files
    ▶ templates
    ▶ Training
    ▶ Training_Files
    ▶ upload_file
      flask_monitoringdashboard.db
      main.py
      newmodelForPrediction.sav
      requirements.txt
      requirements - Copy.txt
      svmmodelForPrediction.sav
▶ External Libraries
  Scratches and Consoles
```

requirements.txt file consists of all the packages that you need to deploy the app in the cloud.

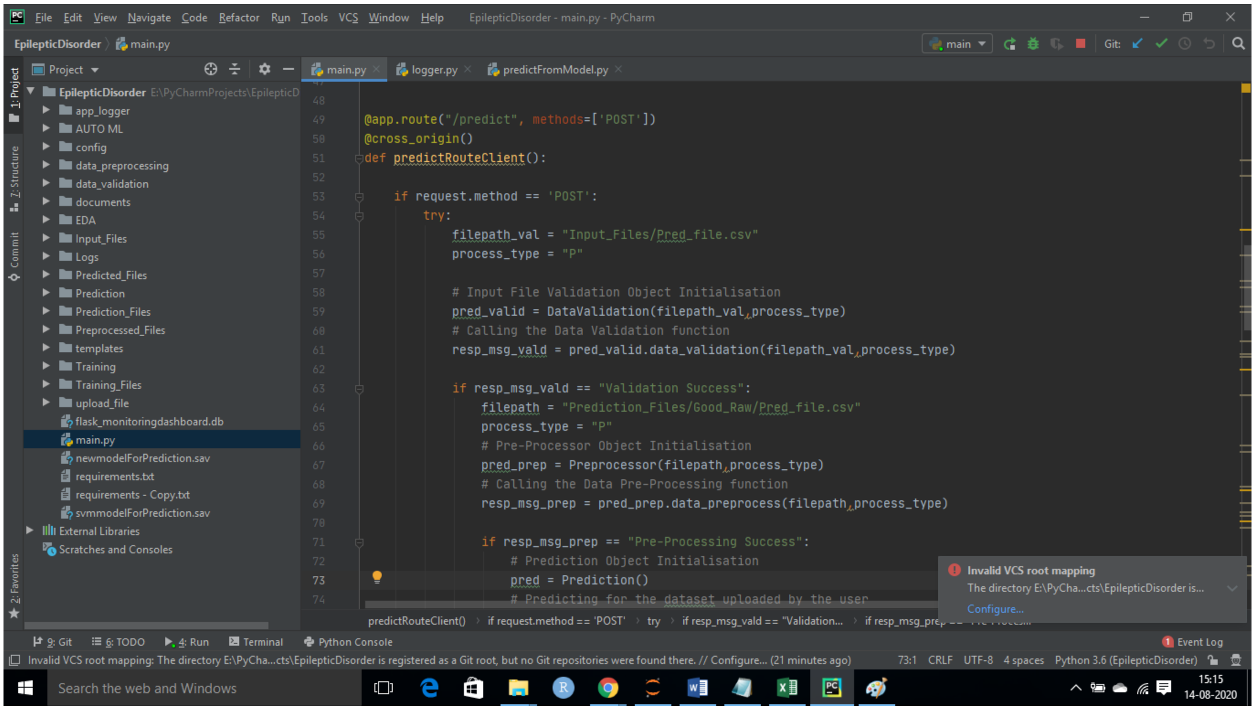


```python
@app.route("/", methods=['GET'])
@cross_origin()
def home():
    return render_template('index.html')

@app.route("/upload", methods=['POST'])
@cross_origin()
def upload():
    if request.method == 'POST':
        try:
            if 'file' not in request.files:
                return Response('No file selected for uploading')

            file = request.files['file']

            # Upload file Object Initialisation
            upld_file = UploadFile(file)
            # Calling the Upload file function
            resp_msg_upld = upld_file.upload_file(file)
            return Response(resp_msg_upld)

        except Exception as e:
            return Response("Error occured in uploading the file.", e)

    else:
        return render_template('index.html')
```

Invalid VCS root mapping
The directory E:\PyCha...cts\EpilepticDisorder is...
Configure...

Invalid VCS root mapping: The directory E:\PyCha...cts\EpilepticDisorder is registered as a Git root, but no Git repositories were found there. // Configure... (20 minutes ago)

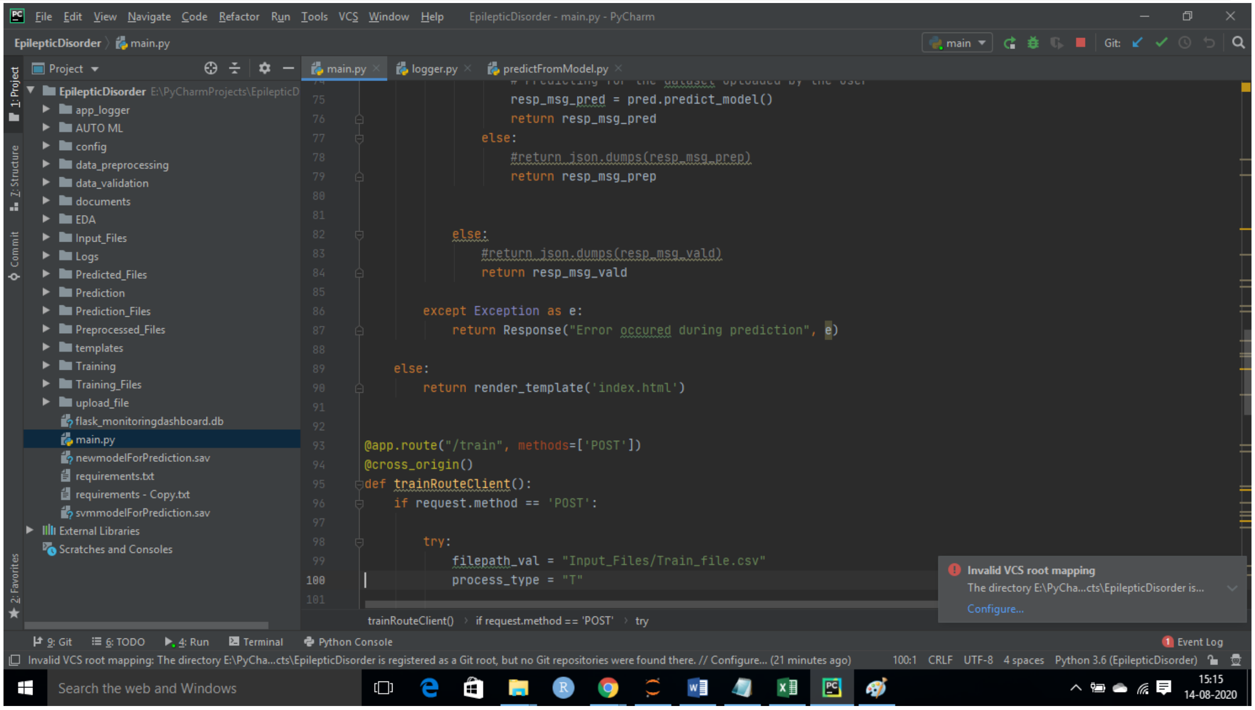
main.py is the entry point of our application, where the flask server starts. Here we will be decoding a base64 to an image, and then we will be making predictions.



```python
@app.route("/predict", methods=['POST'])
@cross_origin()
def predictRouteClient():

    if request.method == 'POST':
        try:
            filepath_val = "Input_Files/Pred_file.csv"
            process_type = "P"

            # Input File Validation Object Initialisation
            pred_valid = DataValidation(filepath_val,process_type)
            # Calling the Data Validation function
            resp_msg_vald = pred_valid.data_validation(filepath_val,process_type)

            if resp_msg_vald == "Validation Success":
                filepath = "Prediction_Files/Good_Raw/Pred_file.csv"
                process_type = "P"
                # Pre-Processor Object Initialisation
                pred_prep = Preprocessor(filepath,process_type)
                # Calling the Data Pre-Processing function
                resp_msg_prep = pred_prep.data_preprocess(filepath,process_type)

                if resp_msg_prep == "Pre-Processing Success":
                    # Prediction Object Initialisation
                    pred = Prediction()
                    # Predicting for the dataset uploaded by the user
```

Invalid VCS root mapping
The directory E:\PyCha...cts\EpilepticDisorder is...
Configure...

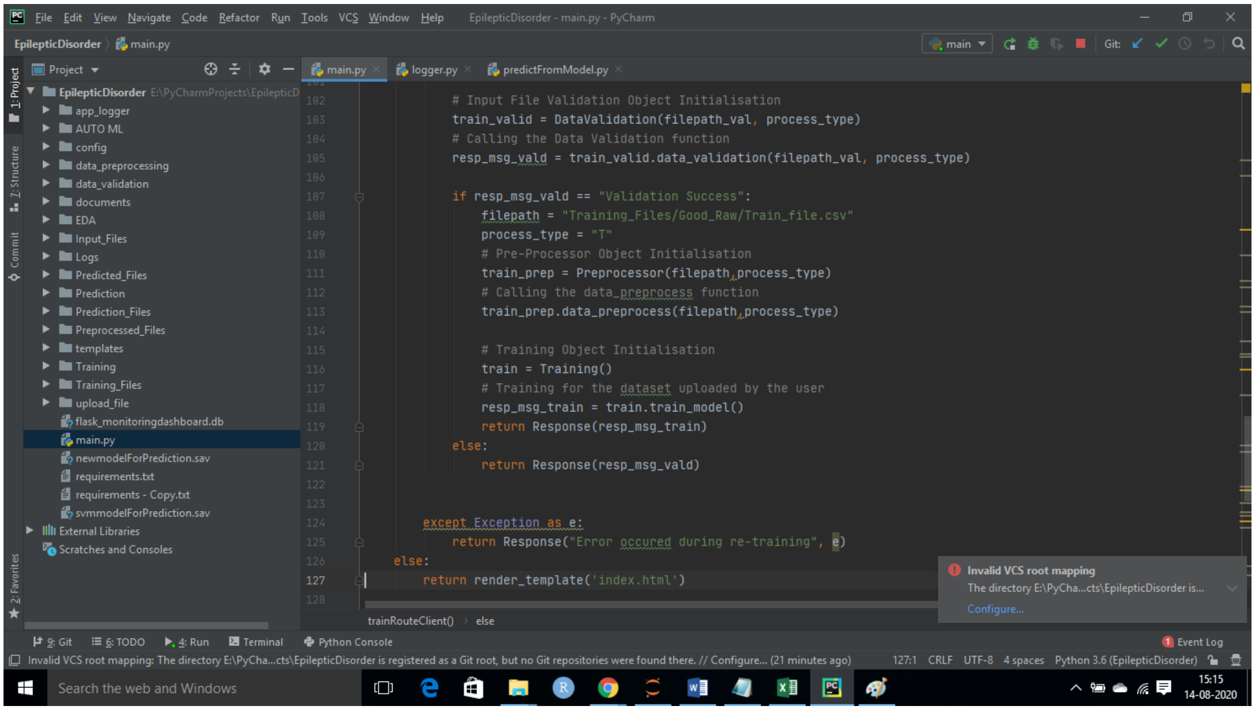
Invalid VCS root mapping: The directory E:\PyCha...cts\EpilepticDisorder is registered as a Git root, but no Git repositories were found there. // Configure... (21 minutes ago)


File Edit View Navigate Code Refactor Run Tools VCS Window Help
EpilepticDisorder - main.py - PyCharm
EpilepticDisorder > main.py
main
Git:
Project
main.py
logger.py
predictFromModel.py
EpilepticDisorder E:\PyCharmProjects\EpilepticD
app_logger
AUTO ML
config
data_preprocessing
data_validation
documents
EDA
Input_Files
Logs
Predicted_Files
Prediction
Prediction_Files
Preprocessed_Files
templates
Training
Training_Files
upload_file
flask_monitoringdashboard.db
main.py
newmodelForPrediction.sav
requirements.txt
requirements - Copy.txt
svmmodelForPrediction.sav
External Libraries
Scratches and Consoles
trainRouteClient() > if request.method == 'POST' > try
9: Git
6: TODO
4: Run
Terminal
Python Console
Event Log
100:1 CRLF UTF-8 4 spaces Python 3.6 (EpilepticDisorder)
Search the web and Windows
15:15
14-08-2020

```
                    # Predicting for the dataset uploaded by the user
                    resp_msg_pred = pred.predict_model()
                    return resp_msg_pred
                else:
                    #return json.dumps(resp_msg_prep)
                    return resp_msg_prep


            else:
                #return json.dumps(resp_msg_vald)
                return resp_msg_vald

        except Exception as e:
            return Response("Error occured during prediction", e)

    else:
        return render_template('index.html')


@app.route("/train", methods=['POST'])
@cross_origin()
def trainRouteClient():
    if request.method == 'POST':

        try:
            filepath_val = "Input_Files/Train_file.csv"
            process_type = "T"
```

Invalid VCS root mapping
The directory E:\PyCha...cts\EpilepticDisorder is...
Configure...

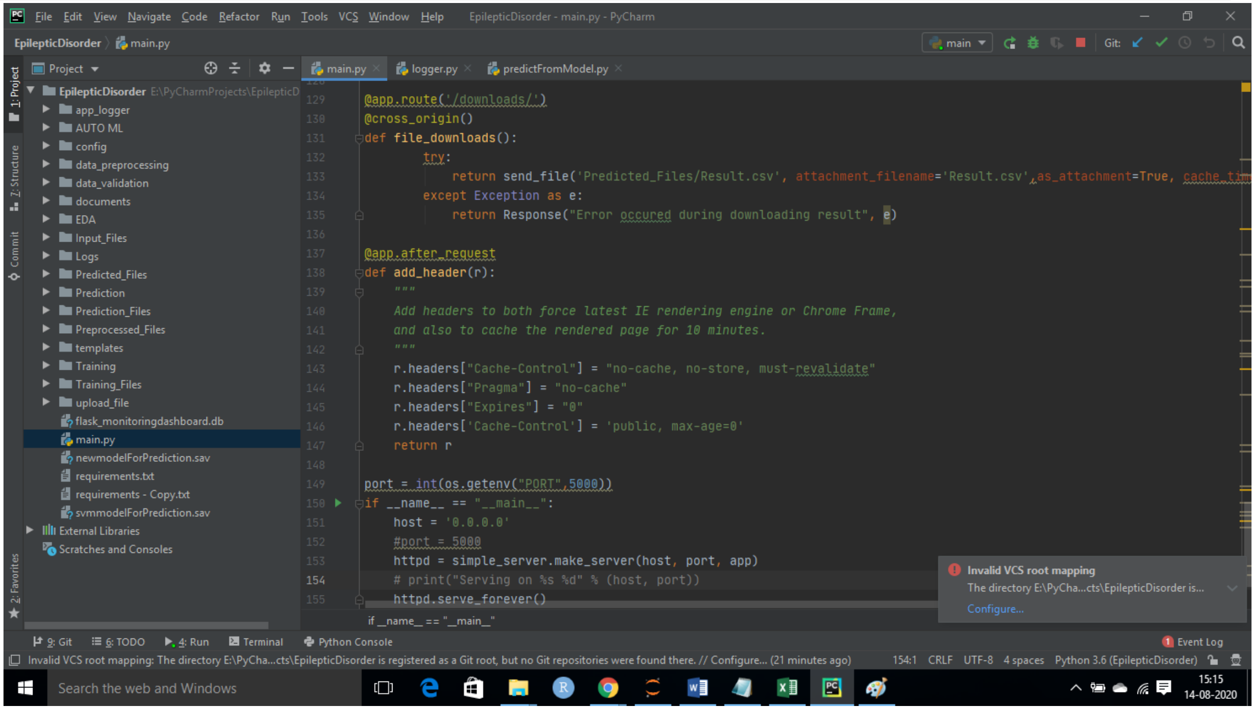
Invalid VCS root mapping: The directory E:\PyCha...cts\EpilepticDisorder is registered as a Git root, but no Git repositories were found there. // Configure... (21 minutes ago)



```
            # Input File Validation Object Initialisation
            train_valid = DataValidation(filepath_val, process_type)
            # Calling the Data Validation function
            resp_msg_vald = train_valid.data_validation(filepath_val, process_type)

            if resp_msg_vald == "Validation Success":
                filepath = "Training_Files/Good_Raw/Train_file.csv"
                process_type = "T"
                # Pre-Processor Object Initialisation
                train_prep = Preprocessor(filepath,process_type)
                # Calling the data_preprocess function
                train_prep.data_preprocess(filepath,process_type)

                # Training Object Initialisation
                train = Training()
                # Training for the dataset uploaded by the user
                resp_msg_train = train.train_model()
                return Response(resp_msg_train)
            else:
                return Response(resp_msg_vald)


        except Exception as e:
            return Response("Error occured during re-training", e)
    else:
        return render_template('index.html')
```

Invalid VCS root mapping
The directory E:\PyCha...cts\EpilepticDisorder is...
Configure...

Invalid VCS root mapping: The directory E:\PyCha...cts\EpilepticDisorder is registered as a Git root, but no Git repositories were found there. // Configure... (21 minutes ago)



```
@app.route('/downloads/')
@cross_origin()
def file_downloads():
        try:
            return send_file('Predicted_Files/Result.csv', attachment_filename='Result.csv',as_attachment=True, cache_tim
        except Exception as e:
            return Response("Error occured during downloading result", e)

@app.after_request
def add_header(r):
    """
    Add headers to both force latest IE rendering engine or Chrome Frame,
    and also to cache the rendered page for 10 minutes.
    """
    r.headers["Cache-Control"] = "no-cache, no-store, must-revalidate"
    r.headers["Pragma"] = "no-cache"
    r.headers["Expires"] = "0"
    r.headers['Cache-Control'] = 'public, max-age=0'
    return r

port = int(os.getenv("PORT",5000))
if __name__ == "__main__":
    host = '0.0.0.0'
    #port = 5000
    httpd = simple_server.make_server(host, port, app)
    # print("Serving on %s %d" % (host, port))
    httpd.serve_forever()
```

Invalid VCS root mapping
The directory E:\PyCha...cts\EpilepticDisorder is...
Configure...

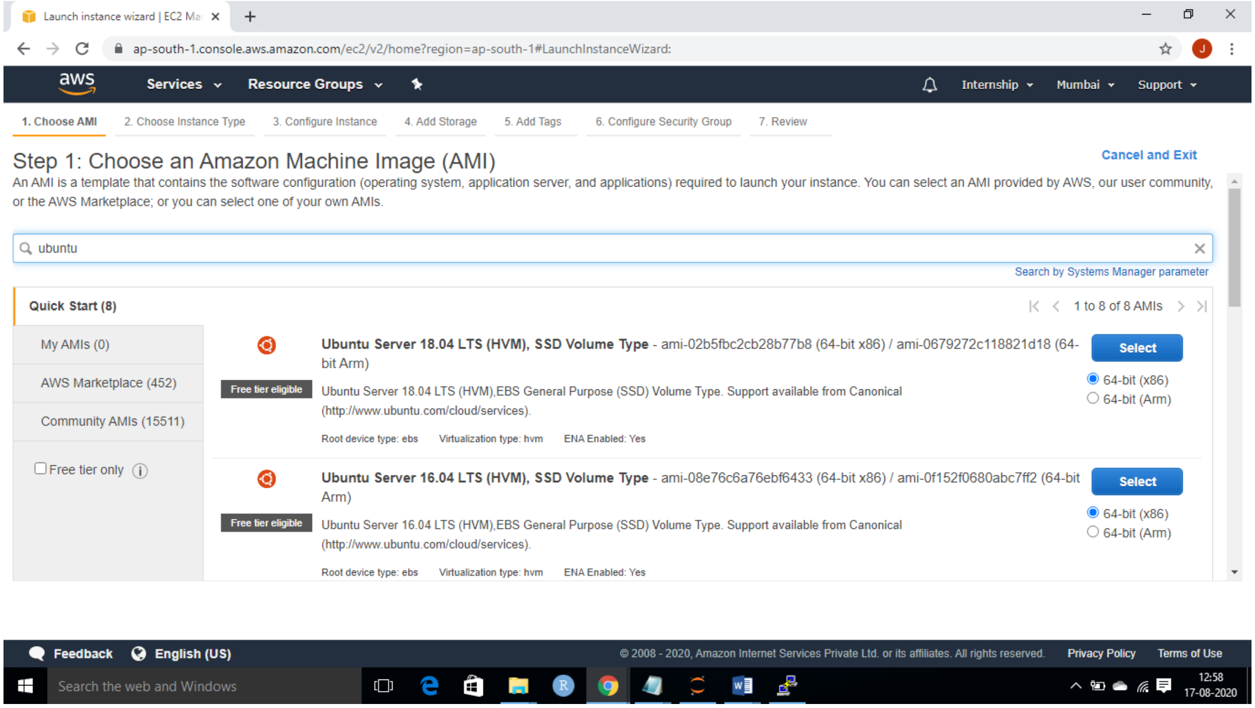
```
Invalid VCS root mapping: The directory E:\PyCha...cts\EpilepticDisorder is registered as a Git root, but no Git repositories were found there. // Configure... (21 minutes ago)    154:1  CRLF  UTF-8  4 spaces  Python 3.6 (EpilepticDisorder)
```

Deployment & Dokerization

We will be deploying the model to the AWS platform.

Steps for Deployment

1. Log in to AWS management console by clicking on ‘Sign in to the console’.
2. Search for EC2, click on ‘launch Instance’.
3. Search for Ubuntu free tier and then select that option


Launch instance wizard | EC2 Ma
ap-south-1.console.aws.amazon.com/ec2/v2/home?region=ap-south-1#LaunchInstanceWizard:
aws
Services
Resource Groups
Internship
Mumbai
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Cancel and Exit
Search by Systems Manager parameter
Select
Feedback
English (US)
© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Search the web and Windows
12:58
17-08-2020

Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

ubuntu

Quick Start (8)

My AMIs (0)

AWS Marketplace (452)

Community AMIs (15511)

□ Free tier only

1 to 8 of 8 AMIs

Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-02b5fbc2cb28b77b8 (64-bit x86) / ami-0679272c118821d18 (64-bit Arm)

Free tier eligible

Ubuntu Server 18.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

● 64-bit (x86)
○ 64-bit (Arm)

Ubuntu Server 16.04 LTS (HVM), SSD Volume Type - ami-08e76c6a76ebf6433 (64-bit x86) / ami-0f152f0680abc7ff2 (64-bit Arm)

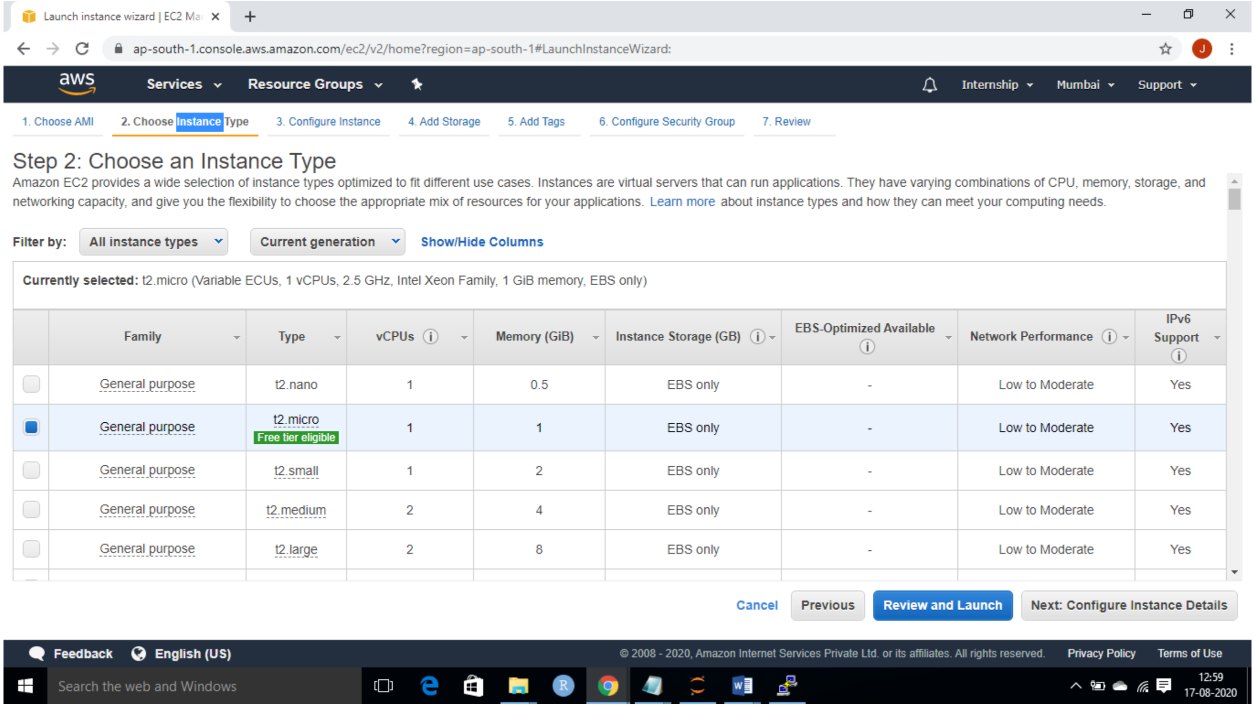
Free tier eligible

Ubuntu Server 16.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

● 64-bit (x86)
○ 64-bit (Arm)

4. Choose instance type (t2:micro)
5. Click on launch


Launch instance wizard | EC2 Ma
ap-south-1.console.aws.amazon.com/ec2/v2/home?region=ap-south-1#LaunchInstanceWizard:
aws
Services
Resource Groups
Internship
Mumbai
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Cancel
Previous
Review and Launch
Next: Configure Instance Details
Feedback
English (US)
© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Search the web and Windows
12:59
17-08-2020

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.

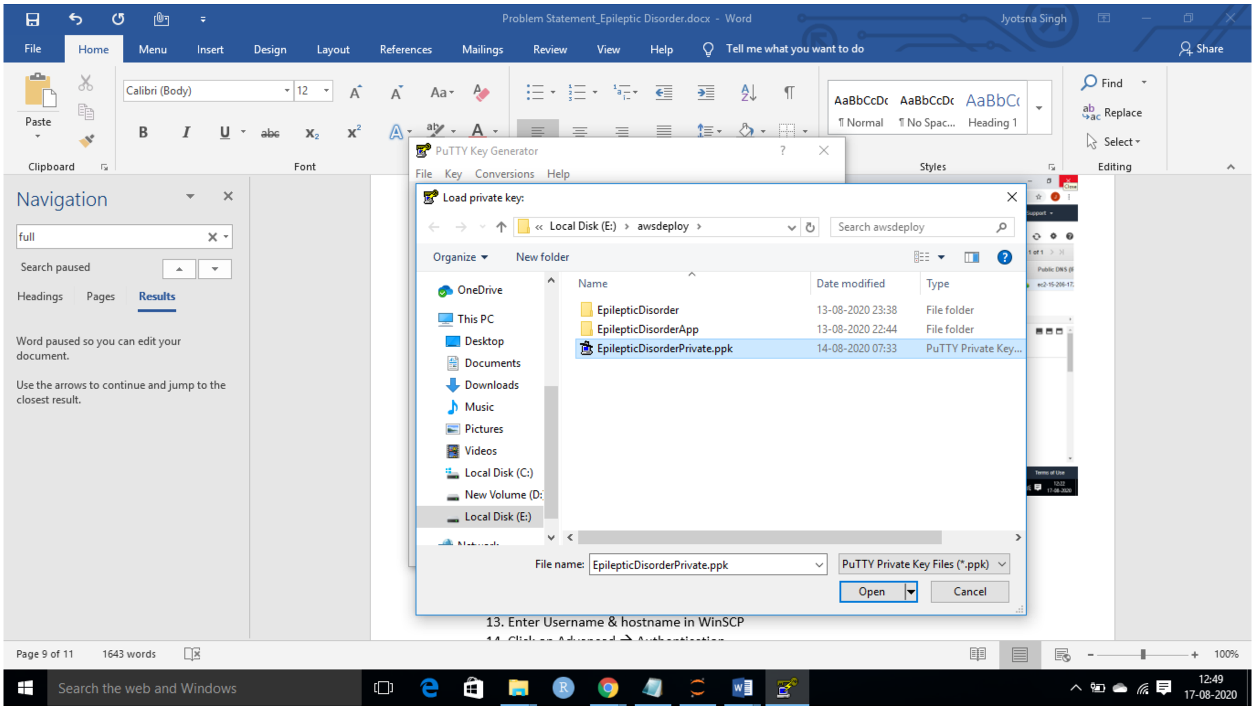
Filter by: All instance types Current generation Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
□	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
☑	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
□	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
□	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
□	General purpose	t2.large	2	8	EBS only	-	Low to Moderate	Yes

6. Create a new key pair and insert name. Download and save the (.pem file)
7. Click on launch instance and give it a name.

8. Download ‘Putty’ for widows and ‘Puttygen’.
9. Go to Puttygen, load the .pem file and click on OK

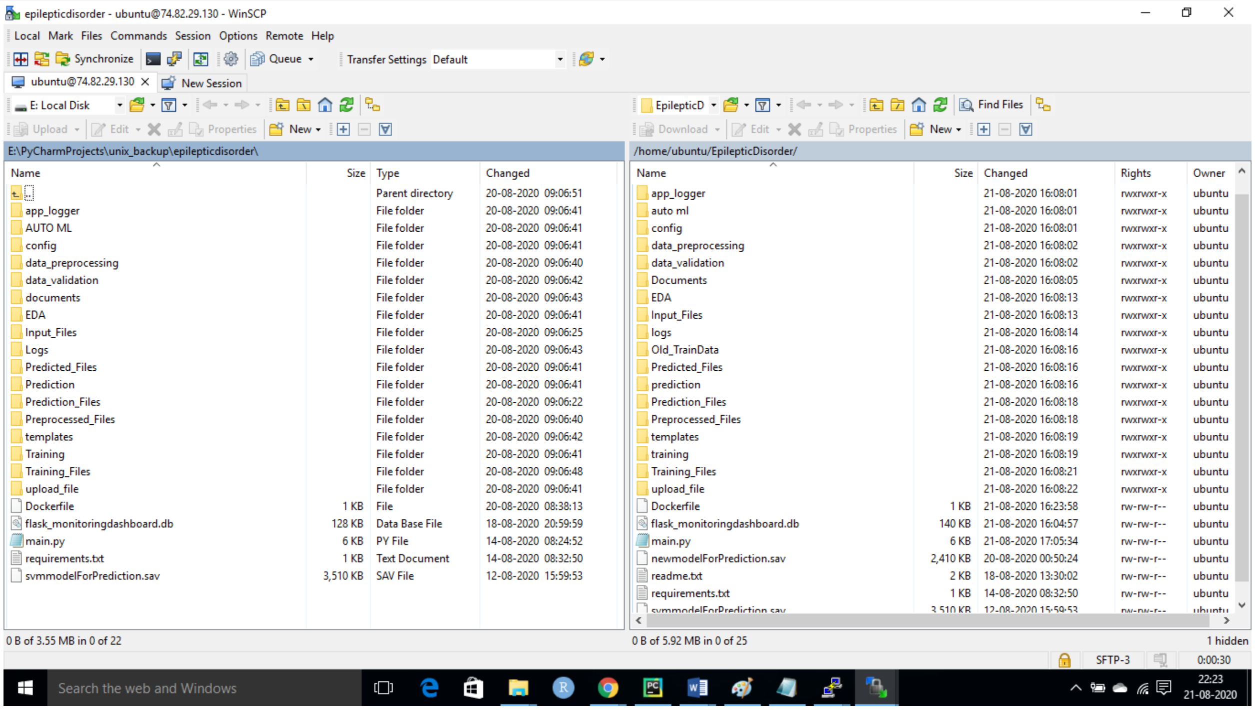


Name	Date modified	Type
EpilepticDisorder	13-08-2020 23:38	File folder
EpilepticDisorderApp	13-08-2020 22:44	File folder
EpilepticDisorderPrivate.ppk	14-08-2020 07:33	PuTTY Private Key...

File name: EpilepticDisorderPrivate.ppk

PuTTY Private Key Files (*.ppk)

10. Save the Private key(.ppk) file
11. Download WinSCP & open it (it is needed to deploy code in AWS EC2 instance)
12. Go to AWS console, click on actions → Connect (You will get host name)
13. Enter Username & hostname in WinSCP
14. Click on Advanced → Authentication
15. Upload the (.ppk) file & click on Open & then click on OK
16. Click on Login. Select the option Yes


epilepticdisorder - ubuntu@74.82.29.130 - WinSCP
Local Mark Files Commands Session Options Remote Help
Synchronize
Queue
Transfer Settings Default
ubuntu@74.82.29.130
New Session
E: Local Disk
Upload
Edit
Properties
New
EpilepticD
Find Files
Download
Search the web and Windows

E:\PyCharmProjects\unix_backup\epilepticdisorder\

Name	Size	Type	Changed
..		Parent directory	20-08-2020 09:06:51
app_logger		File folder	20-08-2020 09:06:41
AUTO ML		File folder	20-08-2020 09:06:41
config		File folder	20-08-2020 09:06:41
data_preprocessing		File folder	20-08-2020 09:06:40
data_validation		File folder	20-08-2020 09:06:42
documents		File folder	20-08-2020 09:06:43
EDA		File folder	20-08-2020 09:06:41
Input_Files		File folder	20-08-2020 09:06:25
Logs		File folder	20-08-2020 09:06:43
Predicted_Files		File folder	20-08-2020 09:06:41
Prediction		File folder	20-08-2020 09:06:41
Prediction_Files		File folder	20-08-2020 09:06:22
Preprocessed_Files		File folder	20-08-2020 09:06:40
templates		File folder	20-08-2020 09:06:42
Training		File folder	20-08-2020 09:06:41
Training_Files		File folder	20-08-2020 09:06:48
upload_file		File folder	20-08-2020 09:06:41
Dockerfile	1 KB	File	20-08-2020 08:38:13
flask_monitoringdashboard.db	128 KB	Data Base File	18-08-2020 20:59:59
main.py	6 KB	PY File	14-08-2020 08:24:52
requirements.txt	1 KB	Text Document	14-08-2020 08:32:50
svmmodelForPrediction.sav	3,510 KB	SAV File	12-08-2020 15:59:53

0 B of 3.55 MB in 0 of 22

/home/ubuntu/EpilepticDisorder/

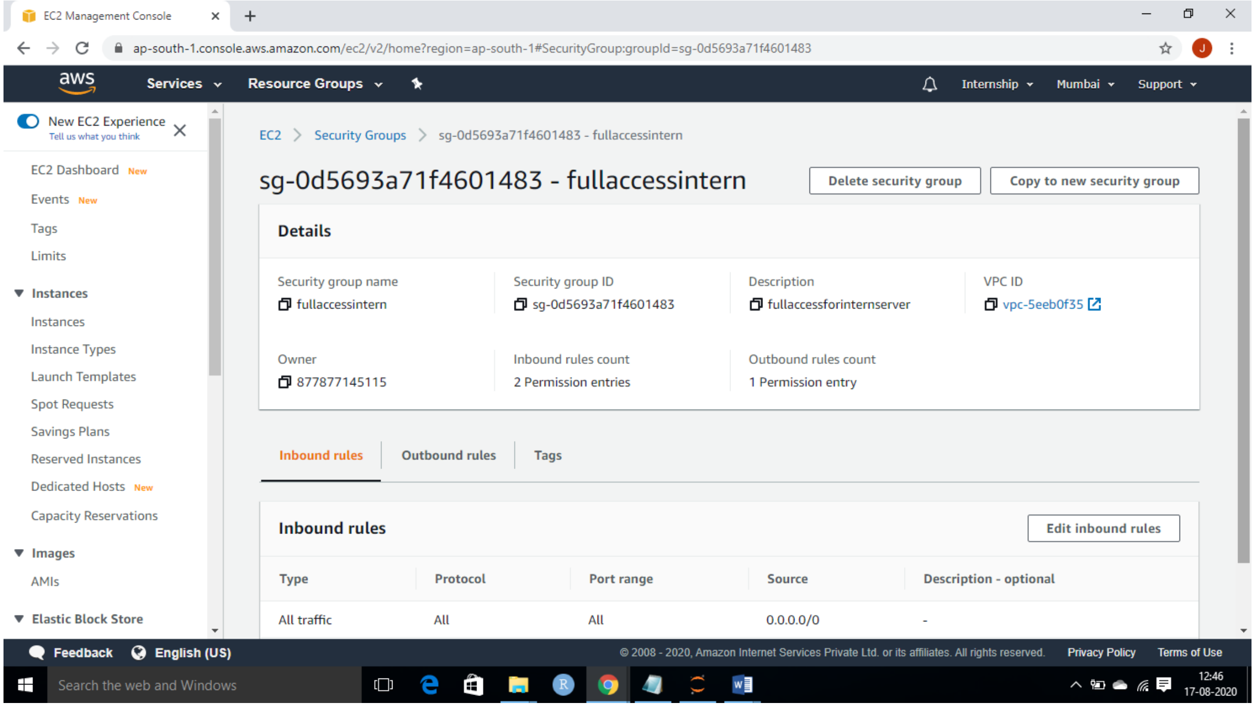
Name	Size	Changed	Rights	Owner
app_logger		21-08-2020 16:08:01	rwxrwxr-x	ubuntu
auto ml		21-08-2020 16:08:01	rwxrwxr-x	ubuntu
config		21-08-2020 16:08:01	rwxrwxr-x	ubuntu
data_preprocessing		21-08-2020 16:08:02	rwxrwxr-x	ubuntu
data_validation		21-08-2020 16:08:02	rwxrwxr-x	ubuntu
Documents		21-08-2020 16:08:05	rwxrwxr-x	ubuntu
EDA		21-08-2020 16:08:13	rwxrwxr-x	ubuntu
Input_Files		21-08-2020 16:08:13	rwxrwxr-x	ubuntu
logs		21-08-2020 16:08:14	rwxrwxr-x	ubuntu
Old_TrainData		21-08-2020 16:08:16	rwxrwxr-x	ubuntu
Predicted_Files		21-08-2020 16:08:16	rwxrwxr-x	ubuntu
prediction		21-08-2020 16:08:16	rwxrwxr-x	ubuntu
Prediction_Files		21-08-2020 16:08:18	rwxrwxr-x	ubuntu
Preprocessed_Files		21-08-2020 16:08:18	rwxrwxr-x	ubuntu
templates		21-08-2020 16:08:19	rwxrwxr-x	ubuntu
training		21-08-2020 16:08:19	rwxrwxr-x	ubuntu
Training_Files		21-08-2020 16:08:21	rwxrwxr-x	ubuntu
upload_file		21-08-2020 16:08:22	rwxrwxr-x	ubuntu
Dockerfile	1 KB	21-08-2020 16:23:58	rw-rw-r--	ubuntu
flask_monitoringdashboard.db	140 KB	21-08-2020 16:04:57	rw-rw-r--	ubuntu
main.py	6 KB	21-08-2020 17:05:34	rw-rw-r--	ubuntu
newmodelForPrediction.sav	2,410 KB	20-08-2020 00:50:24	rw-rw-r--	ubuntu
readme.txt	2 KB	18-08-2020 13:30:02	rw-rw-r--	ubuntu
requirements.txt	1 KB	14-08-2020 08:32:50	rw-rw-r--	ubuntu
svmmodelForPrediction.sav	3,510 KB	12-08-2020 15:59:53	rw-rw-r--	ubuntu

0 B of 5.92 MB in 0 of 25 1 hidden

SFTP-3 0:00:30

22:23
21-08-2020

17. Save the entire code along with main.py , requirements.txt, model.pkl file into the Ubuntu server
18. Install all dependencies & libraries
    a. Open Putty & insert host name
    b. Name and Saved session
    c. Click on SSH → Authentication
    d. Put private key (.ppk file) and save
    e. Select and click on open (Select saved session name)
19. Login to Ubuntu
    a. Insert cmd pwd → to check the directory
    b. Insert cmd ls → to check files in the directory
    c. Sudo apt-get update &&sudo apt-get install python 3-pip
    d. Enter Y
20. Go to AWS dashboard console
    a. Click on Security Group
    b. Create Security group
    c. Insert Security Group name: fullaccessintern , Description: fullaccessinternserver , VPX: default
    d. Click on Add Rule
    e. Select type – 'All traffic', Source – 'anywhere'
    f. Click on create
    g. Check and verify the created security group



sg-0d5693a71f4601483 - fullaccessintern

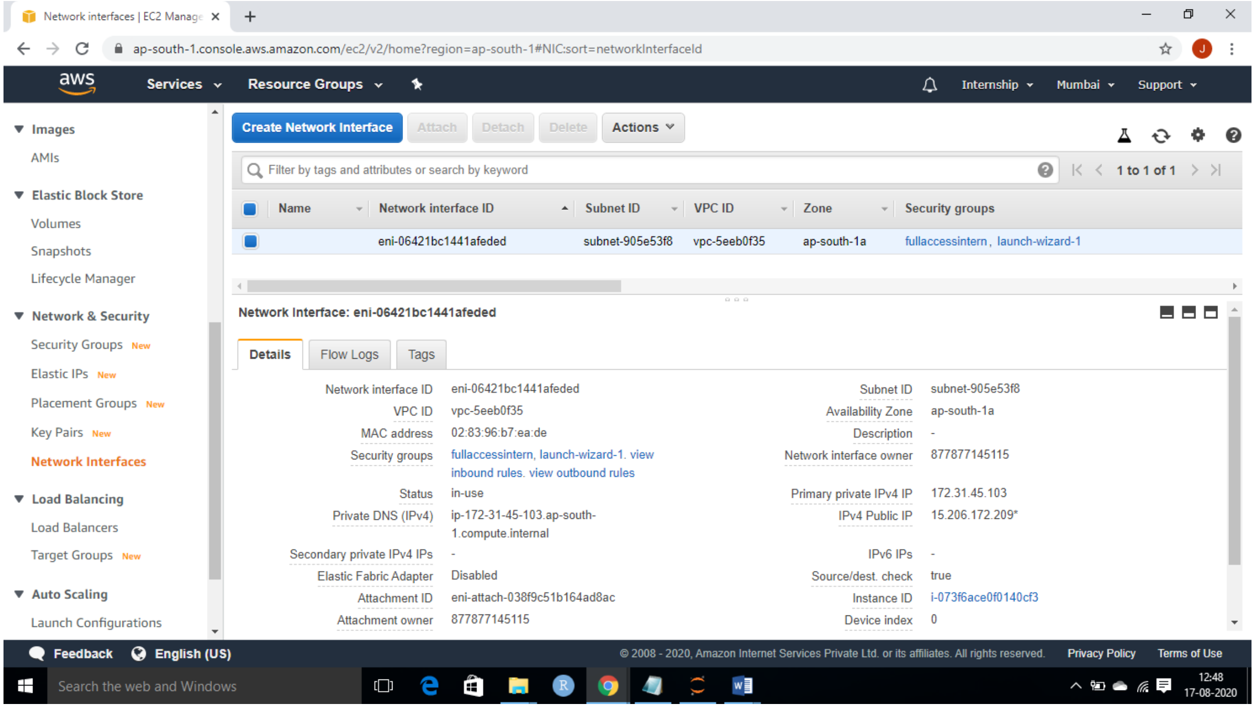
Details

Security group name fullaccessintern	Security group ID sg-0d5693a71f4601483	Description fullaccessforinternserver	VPC ID vpc-5eeb0f35
Owner 877877145115	Inbound rules count 2 Permission entries	Outbound rules count 1 Permission entry	

Inbound rules

Type	Protocol	Port range	Source	Description - optional
All traffic	All	All	0.0.0.0/0	-

21. Click on Network & Security
    a. Click on Security groups
    b. Check group ID in network Interface
    c. Select matched instance ID in network interfaces & group ID with Security groups
    d. Left click on above selected matched row and click on change security groups
    e. Selected the ‘Full Access’ created in *
    f. Click on save

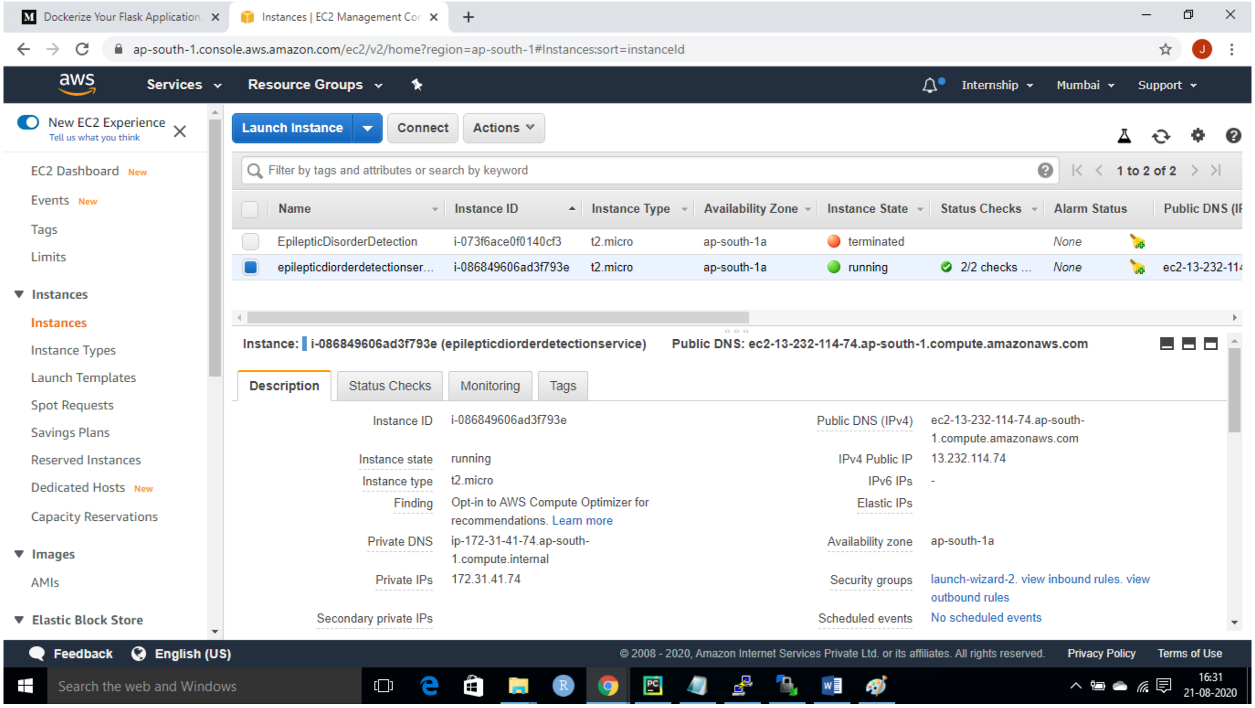


Name	Network interface ID	Subnet ID	VPC ID	Zone	Security groups
	eni-06421bc1441afeded	subnet-905e53f8	vpc-5eeb0f35	ap-south-1a	fullaccessintern , launch-wizard-1

Network Interface: eni-06421bc1441afeded

Network interface ID	eni-06421bc1441afeded	Subnet ID	subnet-905e53f8
VPC ID	vpc-5eeb0f35	Availability Zone	ap-south-1a
MAC address	02:83:96:b7:ea:de	Description	-
Security groups	fullaccessintern, launch-wizard-1. view inbound rules. view outbound rules	Network interface owner	877877145115
Status	in-use	Primary private IPv4 IP	172.31.45.103
Private DNS (IPv4)	ip-172-31-45-103.ap-south-1.compute.internal	IPv4 Public IP	15.206.172.209*
Secondary private IPv4 IPs	-	IPv6 IPs	-
Elastic Fabric Adapter	Disabled	Source/dest. check	true
Attachment ID	eni-attach-038f9c51b164ad8ac	Instance ID	i-073f6ace0f0140cf3
Attachment owner	877877145115	Device index	0

22. Click on instance and see if your instance state is running

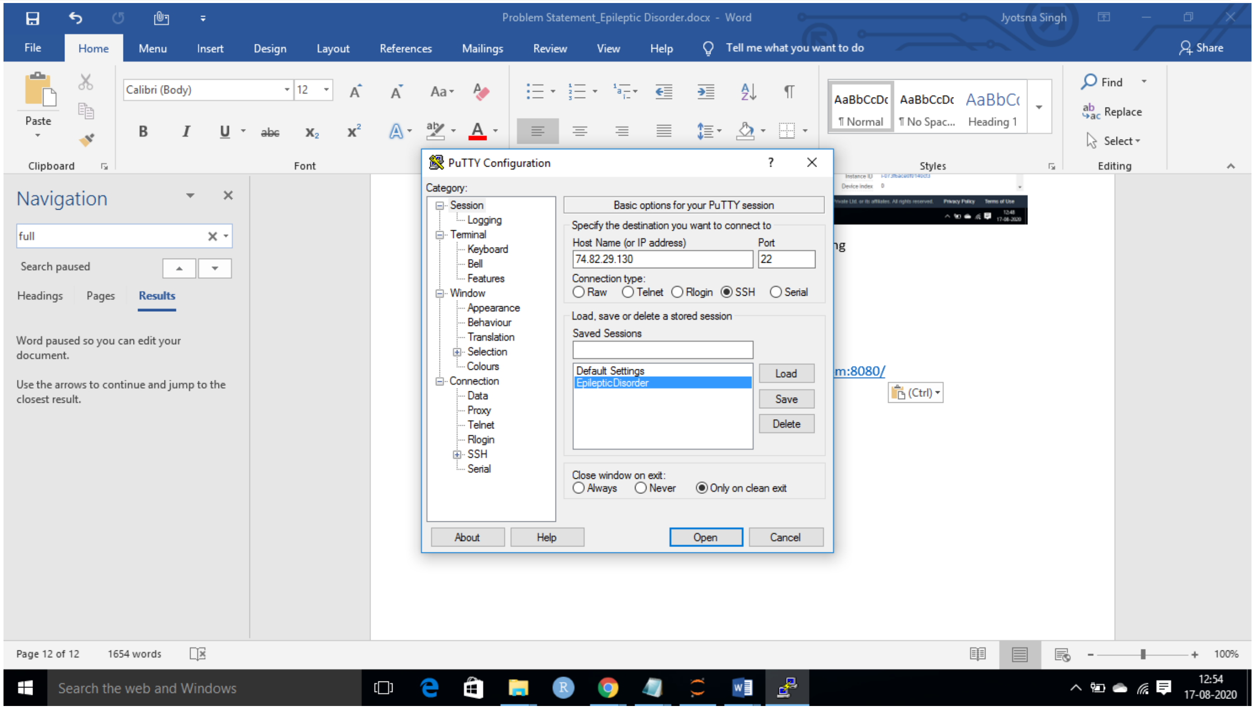


Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IF
EpilepticDisorderDetection	i-073f6ace0f0140cf3	t2.micro	ap-south-1a	terminated		None	
epilepticdiorderdetectionser...	i-086849606ad3f793e	t2.micro	ap-south-1a	running	2/2 checks ...	None	ec2-13-232-114

Instance: i-086849606ad3f793e (epilepticdiorderdetectionservice)   Public DNS: ec2-13-232-114-74.ap-south-1.compute.amazonaws.com

Instance ID	i-086849606ad3f793e	Public DNS (IPv4)	ec2-13-232-114-74.ap-south-1.compute.amazonaws.com
Instance state	running	IPv4 Public IP	13.232.114.74
Instance type	t2.micro	IPv6 IPs	-
Finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more	Elastic IPs	
Private DNS	ip-172-31-41-74.ap-south-1.compute.internal	Availability zone	ap-south-1a
Private IPs	172.31.41.74	Security groups	launch-wizard-2. view inbound rules. view outbound rules
Secondary private IPs		Scheduled events	No scheduled events

23. Go back to cmd prompt of putty



PuTTY Configuration

Category:

Session
- Logging

Terminal
- Keyboard
- Bell
- Features

Window
- Appearance
- Behaviour
- Translation
- Selection
- Colours

Connection
- Data
- Proxy
- Telnet
- Rlogin
- SSH
- Serial

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 74.82.29.130

Port: 22

Connection type:
○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

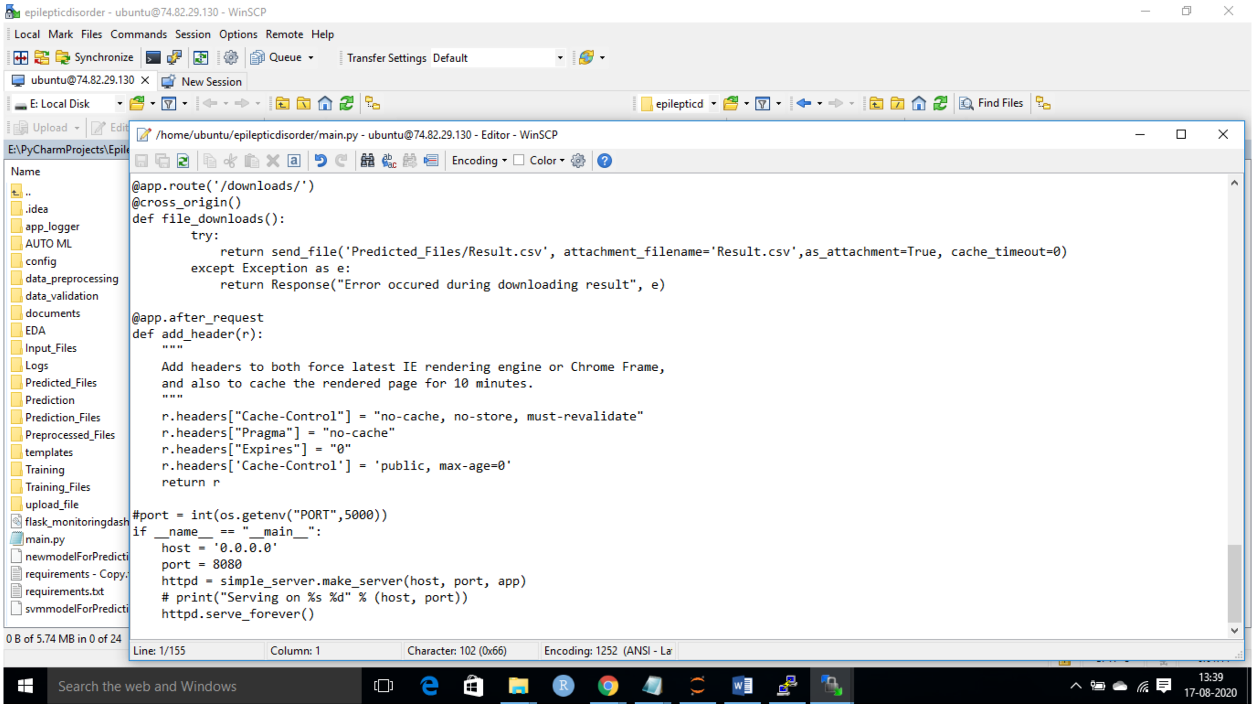
Load, save or delete a stored session

Saved Sessions

Default Settings
EpilepticDisorder

Close window on exit:
○ Always ○ Never ● Only on clean exit

24. Check code in ‘main.py’ file to have host and port name as given below

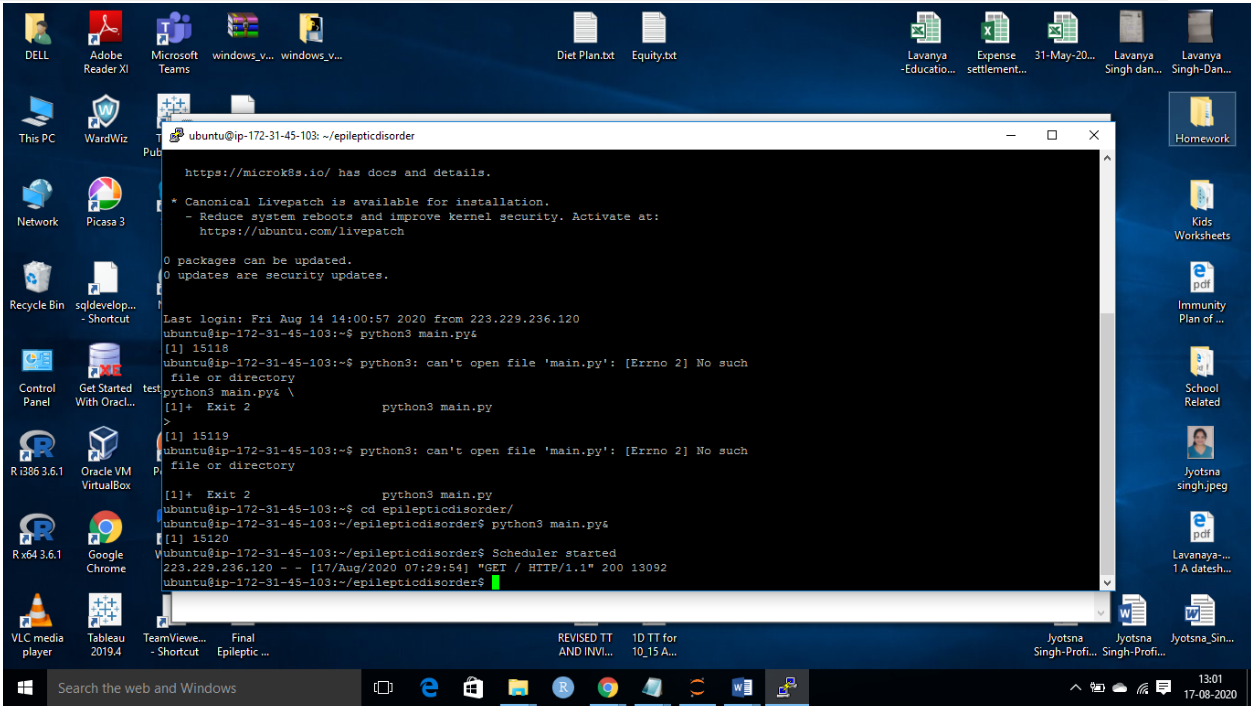


/home/ubuntu/epilepticdisorder/main.py - ubuntu@74.82.29.130 - Editor - WinSCP

```
@app.route('/downloads/')
@cross_origin()
def file_downloads():
        try:
            return send_file('Predicted_Files/Result.csv', attachment_filename='Result.csv',as_attachment=True, cache_timeout=0)
        except Exception as e:
            return Response("Error occured during downloading result", e)

@app.after_request
def add_header(r):
    """
    Add headers to both force latest IE rendering engine or Chrome Frame,
    and also to cache the rendered page for 10 minutes.
    """
    r.headers["Cache-Control"] = "no-cache, no-store, must-revalidate"
    r.headers["Pragma"] = "no-cache"
    r.headers["Expires"] = "0"
    r.headers['Cache-Control'] = 'public, max-age=0'
    return r

#port = int(os.getenv("PORT",5000))
if __name__ == "__main__":
    host = '0.0.0.0'
    port = 8080
    httpd = simple_server.make_server(host, port, app)
    # print("Serving on %s %d" % (host, port))
    httpd.serve_forever()
```

Line: 1/155 Column: 1 Character: 102 (0x66) Encoding: 1252 (ANSI - La

25. Login to Ubuntu server using Putty and go to the code and run the application using cmd: python3 main.py&. It says ‘Schedular started’.



ubuntu@ip-172-31-45-103: ~/epilepticdisorder

```
  https://microk8s.io/ has docs and details.

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

0 packages can be updated.
0 updates are security updates.


Last login: Fri Aug 14 14:00:57 2020 from 223.229.236.120
ubuntu@ip-172-31-45-103:~$ python3 main.py&
[1] 15118
ubuntu@ip-172-31-45-103:~$ python3: can't open file 'main.py': [Errno 2] No such
 file or directory
python3 main.py& \
[1]+  Exit 2                  python3 main.py
>
[1] 15119
ubuntu@ip-172-31-45-103:~$ python3: can't open file 'main.py': [Errno 2] No such
 file or directory

[1]+  Exit 2                  python3 main.py
ubuntu@ip-172-31-45-103:~$ cd epilepticdisorder/
ubuntu@ip-172-31-45-103:~/epilepticdisorder$ python3 main.py&
[1] 15120
ubuntu@ip-172-31-45-103:~/epilepticdisorder$ Scheduler started
223.229.236.120 - - [17/Aug/2020 07:29:54] "GET / HTTP/1.1" 200 13092
ubuntu@ip-172-31-45-103:~/epilepticdisorder$
```

Deployment & Running App using Docker Container

26. Install Python

    Step1: sudo apt-get install software-properties-common

    Step2: sudo apt-add-repository universe

    Step3: sudo apt-get update

    Step4: sudo apt-get install python3-pip

27. Install Docker in AWS EC2 instance

    Step1:sudo apt-get remove docker docker-engine docker.io containerd runc

    Step2:sudo apt-get update

    Step3:sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common

    Step4:curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

    Step5:sudo apt-key fingerprint 0EBFCD88

    Step6:uname -a

    Step7:sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable test"

    Step8:sudo apt-get update

    Step9:sudo apt-get install docker-ce docker-ce-cli containerd.io

    Step10:uname -a

    Step11:apt-cache madison docker-ce

Step12:uname -a

Step13:sudo apt-get install docker-ce=5:19.03.12~3-0~ubuntu-bionic docker-ce-cli=5:19.03.12~3-0~ubuntu-bionic containerd.io

28. Testing Docker installation

    Step1:sudo docker run hello-world

    Step2:sudo docker images

29. Create Dockerfile. Below is the spinet of the docker file.

    ```
    FROM python:3.6
    RUN sudo apt-get update -y
    RUN sudo apt-get install python3-pip
    WORKDIR /home/ubuntu/epilepticdisorder
    RUN pip3 install -r requirements.txt
    RUN pip3 install Flask==1.1.2
    RUN pip3 install Flask-Cors==3.0.8
    RUN pip3 install Flask-MonitoringDashboard==3.0.6
    RUN pip3 install pandas==1.1.0
    RUN pip3 install scipy==1.5.2
    RUN pip3 install sklearn==0.0
    RUN pip3 install imbalanced-learn==0.7.0
    RUN pip3 install imblearn==0.0
    RUN pip3 uninstall scikit-learn==0.23.2
    RUN pip3 install scikit-learn==0.22.1
    EXPOSE 8080
    CMD python3 main.py
    ```

30. Create Docker Image using below command

    ```
    sudo docker image build -t epilepticaldisorder:1.0 .
    ```

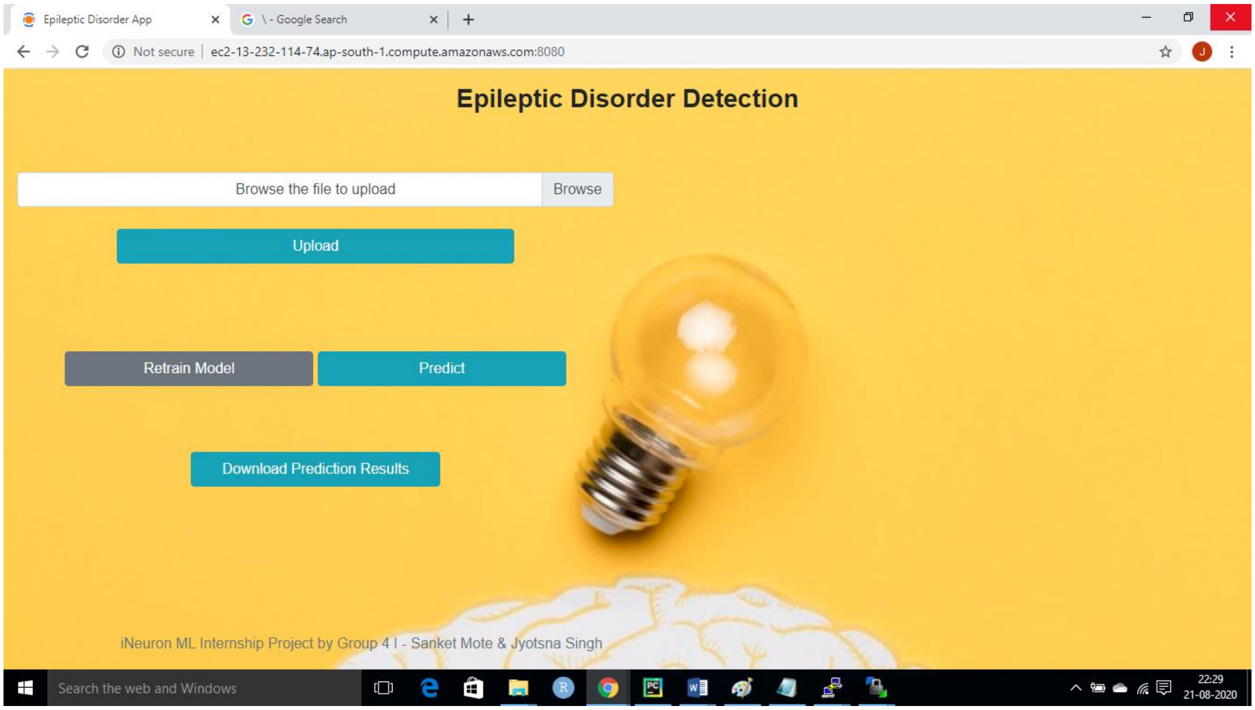
31. Validate the image is present and get the image ID.

    ```
    ubuntu@ip-172-31-41-74:~/EpilepticDisorder$ sudo docker images

    REPOSITORY           TAG           IMAGE ID         CREATED           SIZE
    epilepticaldisorder  1.0           12a0552a19de     24 seconds ago    1.86GB
    python               3.6           46ff56815c7c     3 days ago        874MB
    ```

32. Run the docker image

    ```
    sudo docker run -p 8080:8080 12a0552a19de33.
    ```

The Epileptic Disorder App is available at the below URL: http://ec2-13-232-114-74.ap-south-1.compute.amazonaws.com:8080/


Epileptic Disorder App
\ - Google Search
Not secure | ec2-13-232-114-74.ap-south-1.compute.amazonaws.com:8080
Epileptic Disorder Detection
Browse the file to upload
Browse
Upload
Retrain Model
Predict
Download Prediction Results
iNeuron ML Internship Project by Group 4 I - Sanket Mote & Jyotsna Singh
Search the web and Windows
22:29
21-08-2020