**STROKE PREDCTION USING MACHINE LEARNING**

**A Mini Project Report submitted for the partial fulfilment of the requirement for the award of the degree of**

# MASTER OF COMPUTER APPLICATIONS

**Submitted**

**By**

**M Venkatesh**

**(Reg. No. 21691F00I7)**

**B UmaPriya**

**(Reg. No. 21691F00H7)**

**H Yugundhar**

**(Reg. No. 21691F00J7)**

## Under the Guidance of

**Dr. R.MARUTHAMUTHU, M.C.A, Ph.D.**

**Associate Professor**

**Department of Computer Applications**



### DEPARTMENT OF COMPUTER APPLICATIONS MADANAPALLE INSTITUTE OF TECHNOLOGY AND SCIENCE MADANAPALLE

(UGC AUTONOMOUS)

(Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu)

[www.mits.ac.in](http://www.mits.ac.in/)

## 2019 – 2022

**MADANAPALLE INSTITUTE OF TECHNOLOGY & SCIENCE**

### MADANAPALLE

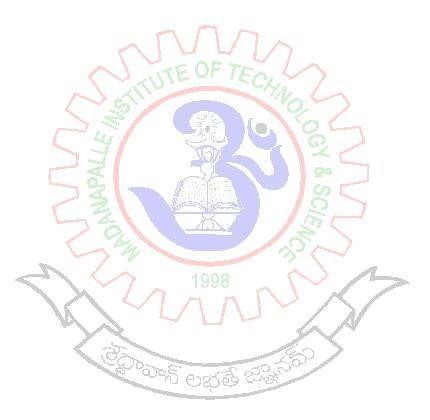
(UGC AUTONOMOUS)

#### (Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu)

DEPARTMENT OF COMPUTER APPLICATIONS

**BONAFIDE CERTIFICATE**

This is to certify that the MiniProject work entitled “**STROKE PREDDECTIION USING MACHINE LEARNING”** is a bonafide work carried out by **M.VENKATESH**(**Regd. No. 21691F00I7), B.UMAPRIYA**(**Regd. No. 21691F00H7), H.YUGUNDHAR**(**Regd. No.**

**21691F00J7),** submitted in partial fulfilment of the requirements for the award of the degree of **Master of Computer Applications** in **Madanapalle Institute of Technology and Science,** Madanapalle, affiliated to **Jawaharlal Nehru Technological University Ananthapuramu** during the **academic year 2021-2022.**

-

-

**PROJECT GUIDE HEAD OF THE DEPARTMENT**

### Dr. M.MARUTHA Dr. N. Naveen Kumar

Associate Professor Associate Professor

Department of Computer Applications Department of Computer Applications

### DECLARATION

We, **M. Venkatesh** (**RollNo.21691F00D5), B. Umapriya (RollNo.21691F00F2), H. Yugundhar (Roll No.21691F00E7)** hereby declare that the project entitles **“STROKE PREDECTION USING MACHINE LEARNING**” is done by me under the guidance of **Dr. M. Maruthamuth, Associate Professor.** submitted in partial fulfilment of the requirements for the award of degree of Master of Computer Applications at MADANAPALLE INSTITUTE OF TECHNOLOGY & SCIENCE, Madanapalle, affiliated to **Jawaharlal Nehru Technological University Anantapur, Anantapuramu** during the academic year 2021-2022. This work has not been submitted by anybody towards the award of any degree.

**Date:**

**Place: Madanapalle. [Signature of the Students]**

M.Venkatesh (Regd.No:21691F00I7)

B.Umapriya(Regd.No:21691F00H7)

H.Yugundhar(Regd.No:21691F00J7)

### 

### ACKNOWLEDGEMENT

First, I must thank the almighty who has granted me knowledge, wisdom, strength and courage to serve in this world and to carry out my project work in a successful way.

Next, I have to thank my parents who encouraged me to undergo this MCA course and do this project in a proper way.

I express my sincere thanks to **Dr. N. Vijaya Bhaskar Choudhry**, Secretary & Correspondent, **Madanapalle Institute of Technology & Science** for his continuous encouragement towards practical education and constant support in all aspects which includes the provision of very good infrastructure facilities in the institute.

It is my duty to thank our Principal, **Dr. C. Yuvaraj,** Madanapalle Institute of Technology & Science, for his guidance and support at the time of my course and project.

I also wish to express my thanks to our Vice Principal, **Dr. P. Ramanathan** for his continuous support throughout my MCA career.

It is my foremost duty to thank the Guide and Head of the Department **Dr N. Naveen Kumar, Associate Professor**, who gave me constant support during the project time and continuous encouragement towards the completion of the project successfully.

.

## ABSTRACT

Stroke is a medical disorder in which the blood arteries in the brain are ruptured, causing damage to the brain. Strokes have rapidly raised globally even at not only in older ages but also in juvenile ages.Stroke prediction is found to be a compound task which requires enormous amount of data pre-processing also there is a need to automatize the prophecy process for the early exposure of symptoms related to stroke so that it can be averted at an early stage. When the supply of blood and other nutrients to the brain is interrupted, symptoms might develop.

According to the World Health Organization (WHO), stroke is the greatest cause of death and disability globally. Early recognition of the various warning signs of a stroke can help reduce the severity of the stroke. Different machine learning (ML) models have been developed to predict the likelihood of a stroke occurring in the brain.

This research uses a range of physiological parameters and machine learning algorithms, such as Logistic Regression (LR), Decision Tree (DT) Classification, Random Forest (RF) KNN, SVM. Classification to train five different models for reliable prediction. Random Forest was the best performing algorithm for this task with an accuracy of approximately 92 percent.

.

**Keywords**

Brain Stroke; Analyzing Dataset; Stroke Predictions.

**Index**

|  |  |  |
| --- | --- | --- |
| Chapter | Contents | Page No. |
| I. | **Introduction**  1.1 About the Project | **1** |
| II | **System Analysis**  2.1 Existing System  2.2 Proposed System  2.3 Hardware configuration  2.4 software configuration  2.5 Feasibility Study  2.5.1Technical Feasibility  2.5.2 Economic Feasibility  2.5.3 Operational Feasibility  2.5.4 Social Feasibility | **2**  **2**  **3**  **3**  **4**  **4**  **4**  **4** |
| III | **System Design**  3.1 Module Description  3.2 UML Diagram  3.2.1 Class Diagram  3.2.2 Use Case Diagram  3.2.3 Activity Diagram  3.2.4 Sequence Diagram  3.2.5ER Diagram | **5**  **5**  **6**  **7**  **8**  **9**  **10** |
| IV | **System Implementation**  4.1 Language Selection  4.2 Algorithm  4.3 Flowchart  4.4 Screenshots  4.5 Sample code | **11**  **12-13**  **14**  **15-16**  **17-20** |
| V | **System Testing**  5.1 Testing Description  5.2 Testing strategy  5.2.1 Unit Testing  5.2.2 Integration Testing  5.2.3 Validation Testing | **21**  **21**  **21**  **21**  **21** |
| VI | **Conclusion** | **23** |
| VII | **References** | **24** |

**CHAPTER -I**

## INTRODUCTION

### 1.1 About the Project

* Stroke occurs when the blood flow to various areas of the brain is disrupted or diminished, resulting in the cells in those areas of the brain not receiving the nutrients and oxygen they require and dying. A stroke is a medical emergency that requires urgent medical attention. Early detection and appropriate management are required to prevent further damage to the affected area of the brain and other complications in other parts of the body. A stroke is a brain attack, cutting off crucial blood flow and oxygen to the brain. Stroke arises when a blood artery feeding the brain gets clogged or bursts.
* Identifying stroke and taking medical action immediately can not only lengthen life but also help to prevent heart disease in the future.The World Health Organization (WHO) estimates that fifteen million people worldwide suffer from strokes each year, with one person dying every four to five minutes in the affected population. Stroke is the sixth leading cause of mortality in the United States according to the Centers for Disease Control and Prevention (CDC) .
* It is India’s fourth leading cause of death. Strokes are classified as ischemic or hemorrhagic. In a chemical stroke, clots obstruct the drainage. in a hemorrhagic stroke, a weak blood vessel bursts and bleeds into the brain. Stroke may be avoided by leading a healthy and balanced lifestyle that includes abstaining from unhealthy behaviors, such as smoking and drinking, keeping a healthy body mass index (BMI) and an average glucose level, and maintaining an excellent heart and kidney function.
* The algorithms included in ML are beneficial as they allow for accurate prediction and proper analysis. The majority of previous stroke-related research has focused on, among other things, the prediction of heart attacks. Brain stroke has been the subject of very few studies. The main motivation of this paper is to demonstrate how ML may be used to forecast the onset of a brain stroke.

## CHAPTER –II SYSTEM ANALYSIS

### 2.1 EXISTING SYSTEM

In the existing system, implementation of machine learning algorithms is bit complex to build due to the lack of information about the data visualization and Relief feature selection. Mathematical calculations are used in existing system for Multiple Linear Regression(MLR) model building that may takes the lot of time and complexity. To overcome all this, we use machine learning packages available in the scikit-learn library**.** The usage of different algorithms like Logistic Regression , Random Forest , Decision Tree ,SVM, and KNN five alternative models, the accuracy measures namely accuracy score and Confusion matrix are used to compare them.

* **LIMITATIONS OF EXISTING SYSTEM**
* It has low reliability
* Low Accuracy
* Time Consuming
* High Complexity

### 2.2 PROPOSED SYSTEM

* In our predicted model, ten features have been evaluated to make this comparison more unique. Our introduced algorithms were conducted based on the all features, Relief selected features the obtained outcomes were compared to other works to show the percentage of improvement, while decrease in performance also noted in one occasion. The highest increment was noticed for one approach as opposed to previous works which was about percentage improvement were calculated for 12 attributes. This project is used to determine whether or not a patient is at risk of having a brain stroke, we used a Decision Tree (DT), Support vector machine, Logistic Regression ,‍k-nearest neighbors (KNN). Algorithm in a based Python environment.
* **ADVANTAGES OF PROPOSED SYSTEM**
* High Accuracy
* Reduce Time Complexity
* Reduce doctor risk

### 2.3. HARDWARE SPECIFICATION

The selection of hardware is very important in the existence and proper working of any software. In the selection of hardware, the size and the capacity requirements are also important.

RAM : 4GB +

HDD or SSD : 500GB+

Processor : i3 and above

### 2.4. SOFTWARE SPECIFICATION

Operating system : Windows 7 or above

IDE : PyCharm

Libraries : Pandas , NumPy , Matplotlib

#### 2.5 Feasibility Study

Feasibility study is measure the software product in terms of how much beneficial product development will be for the organization in practical point of view .It is used to analyze whether the software product will be right in terms of development, implementation, and contribution of project to the organization.

Strokes have rapidly raised globally even at not only in older ages but also in juvenile ages.Stroke prediction is found to be a compound task which requires enormous amount of data pre-processing also there is a need to automatize the prophecy process for the early exposure of symptoms related to stroke so that it can be averted at an early stage. When the supply of blood and other nutrients to the brain is interrupted, symptoms might develop.

#### 2.5.1 Technical Feasibility

Our system “Stroke Detection” is technically feasible since all the required tools are easily available .Although these tools available easily we need to do efforts to use it in effective manner. The Required Technical aspects are keenly done here with proper guidance.

#### 2.5.2 Economic Feasibility

In this project the economic feasibility is very low .All the tools that are required to complete the project are available in open source manner. The process will give accurate results.

##### 2.5.3 Operational Feasibility

Proposed project is beneficial only if it can be turned into information system that will meet the operating requirements. So our project is simpler to operate and can be used in any platform. It is also not costly to operate.

##### 2.5.4 Social Feasibility

The acceptability of a log analysis system depends on the social and cultural values of the participant populations. A careful analysis and articulation of these issues and their identification can improve both acceptability and effectiveness.

## CHAPTER-III SYSTEM DESIGN

**3.1 Module Description**

## Data Collection

Data collection is the process of gathering and storing information for analysis and use in machine learning. In the context of stroke prediction using machine learning techniques, data collection would involve gathering and storing information about patients who have had strokes, as well as information about patients who have not had strokes.

The data would then be used to train a machine learning model to predict the likelihood of a patient having a stroke based on their individual characteristics. The more data that is collected, the more accurate the model is likely to be.

This dataset is used to predict whether the particular person is likely to get stroke or not by taking into considerations like gender ,age , existing diseases in the body , smoking status of the individual . The dataset contains data of “5112 individuals” and 12 features set.

## Data Preprocessing

Data preprocessing is required to remove unwanted noise and outliers from the dataset that could lead the model to depart from its intended training. This stage addresses everything that prevents the model from functioning more efficiently. The collection of the relevant dataset, the data must be cleaned and prepared for model development. The dataset used has twelve characteristics. The column id is omitted since its presence has no bearing on model construction. The dataset is then inspected for null values and filled if any are detected. The null values in the column BMI are filled using the data column’s mean in this case.

Label encoding converts the dataset’s string literals to integer values that the computer can comprehend. As the computer is frequently trained on numbers, the strings must be converted to integers. The gathered dataset has five columns of the data type string. All strings are encoded during label encoding, and the whole dataset is transformed into a collection of numbers. The dataset used for stroke prediction is very imbalanced.

## Prediction Module

The prediction module is the component of a machine learning system that uses the trained model to make predictions on new data. In the context of stroke prediction, the prediction module would take in information about a patient, such as their demographic information, medical history, and lifestyle factors, and use the trained model to predict the likelihood of that patient having a stroke. The prediction module could be designed to output a probability of the patient having a stroke, or it could be designed to output a binary decision (e.g., "high risk" or "low risk"). The exact output of the prediction module would depend on the specific design of the machine learning system and the goals of the project.

* The prediction module in a stroke prediction project using machine learning techniques would involve the following steps:
  + - * 1. **Input**
        2. **Processing**.
        3. **Output**
* The prediction module is the final step in a machine learning pipeline, which includes data collection, preprocessing, training and validation. The prediction module uses the trained model generated from the previous steps to make predictions on new data.

## Data Visualization

The Data Visualization module was written to provide a simple, high-level visualization of your unique form data, letting you - very easily! - visualize waht you need through Line Charts, Area Charts, Pie Charts, Bar Charts and Column Charts. In some cases, this can provide merely a novel insight into the information in your form: in other cases, genuinely useful information.

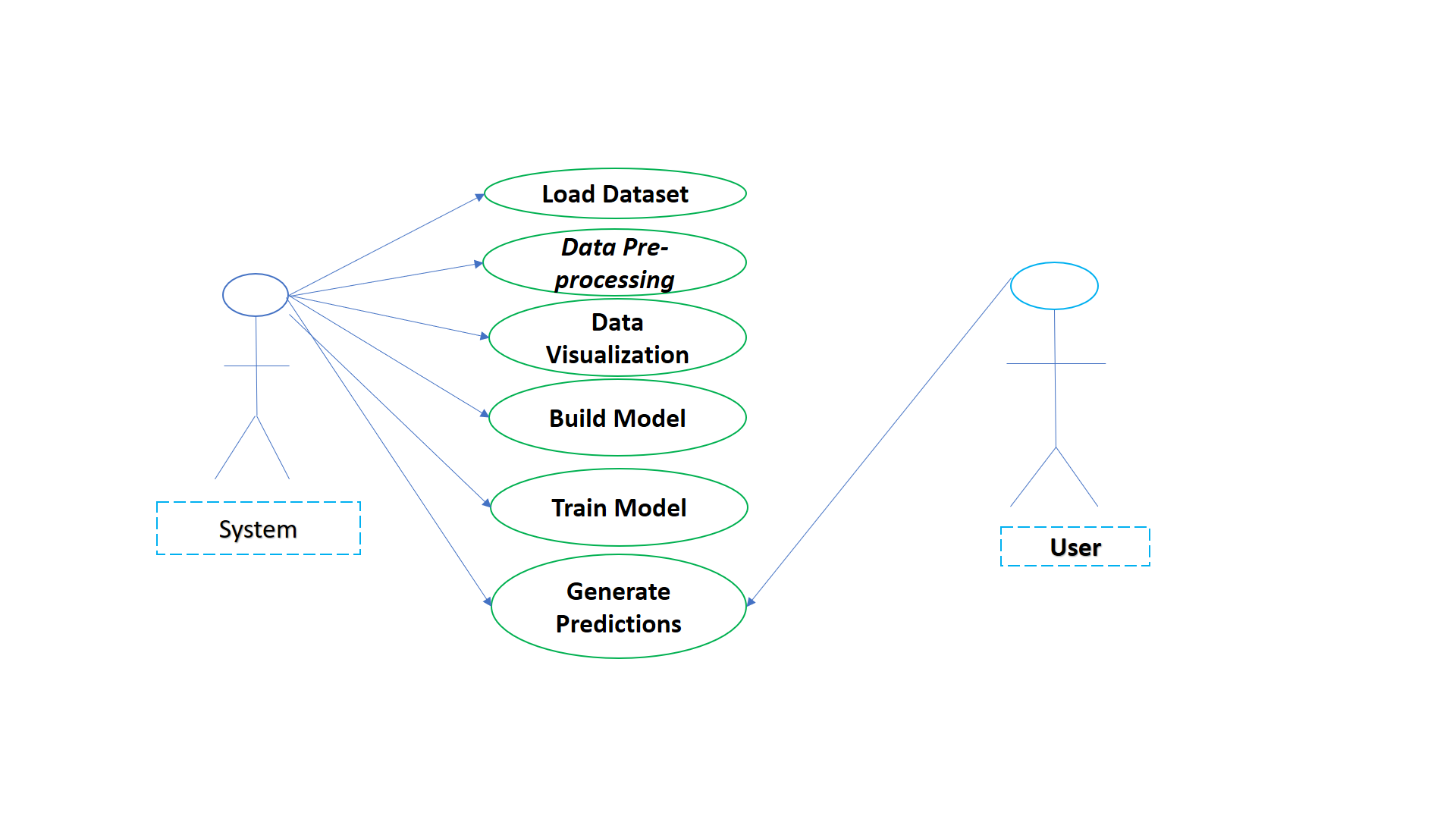
### 3.2 UML Diagrams

UML stands for Unified Modelling Language. UML is a standardized general-purpose Modelling language in the field of object-oriented software engineering. The main aim of the UML diagrams is for creating models of object-oriented computer software. The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the software system, as well as for business modelling and other non-software systems. The Unified Modelling Language uses mostly graphical notations to express the design of software projects and their flow. Various UML diagrams are used in this project to understand the flow of the system easily. The system flow is depicted using Use case diagram, Sequence diagram, Activity diagram etc. which are shown below.

#### .1 USE CASE DIAGRAM

A use case diagram in the Unified Modelling Language (UML) is a type of behavioral diagram. It is created from a Use-Case analysis. Its purpose is to represent a graphical overview of the functionality of the system in terms of actors, their goals and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed or assigned for which actor in the system.

This diagram shows the entire functionality of each process.

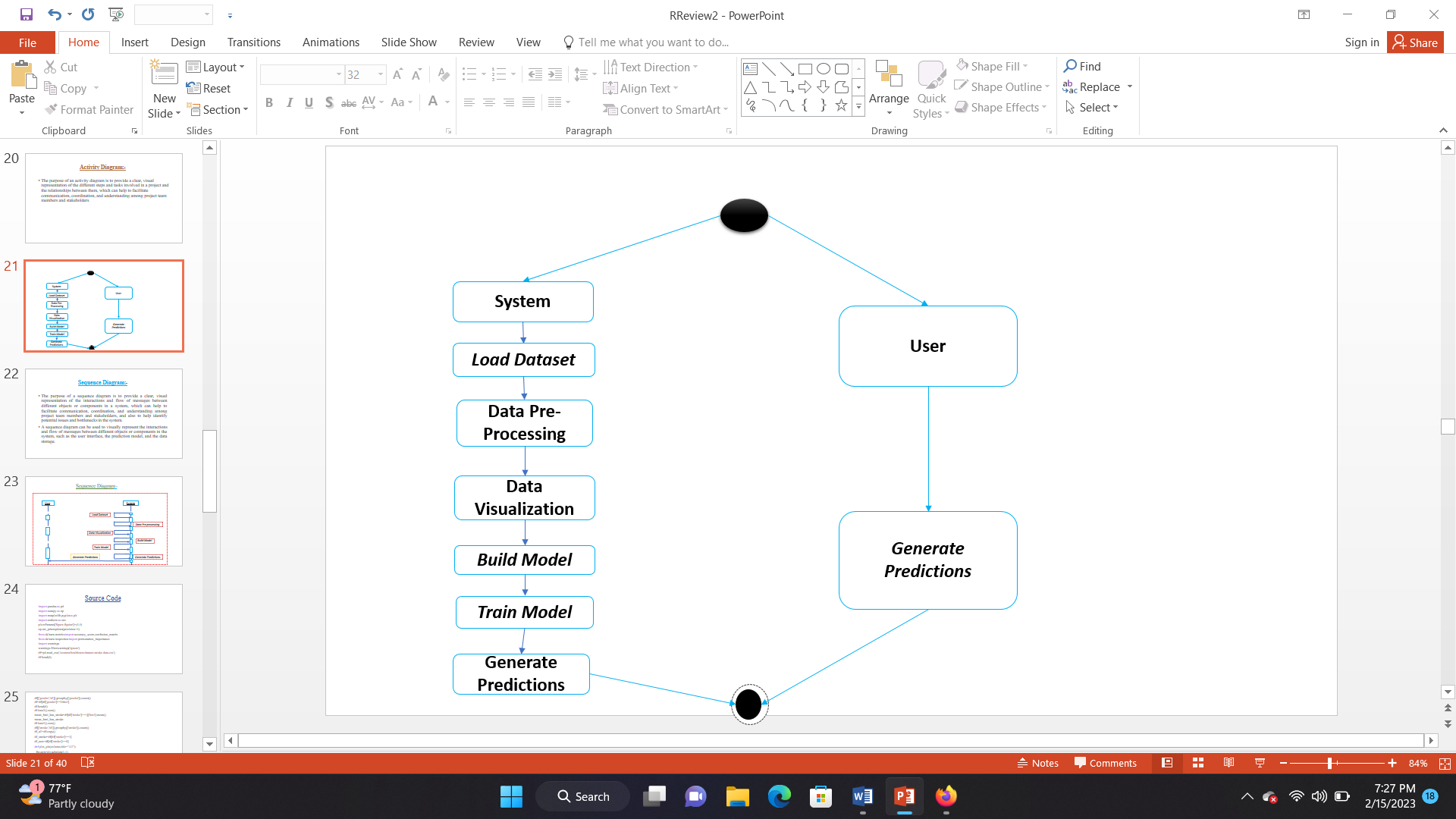


**Fig.1.UseCase Diagram**

.3 ACTIVITY DIAGRAM

Activity diagrams are graphical representations about the workflow of the stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational workflows of components in a system.

Following diagram defines the activities of each and every entity in the form of an activity diagram in an understandable manner.

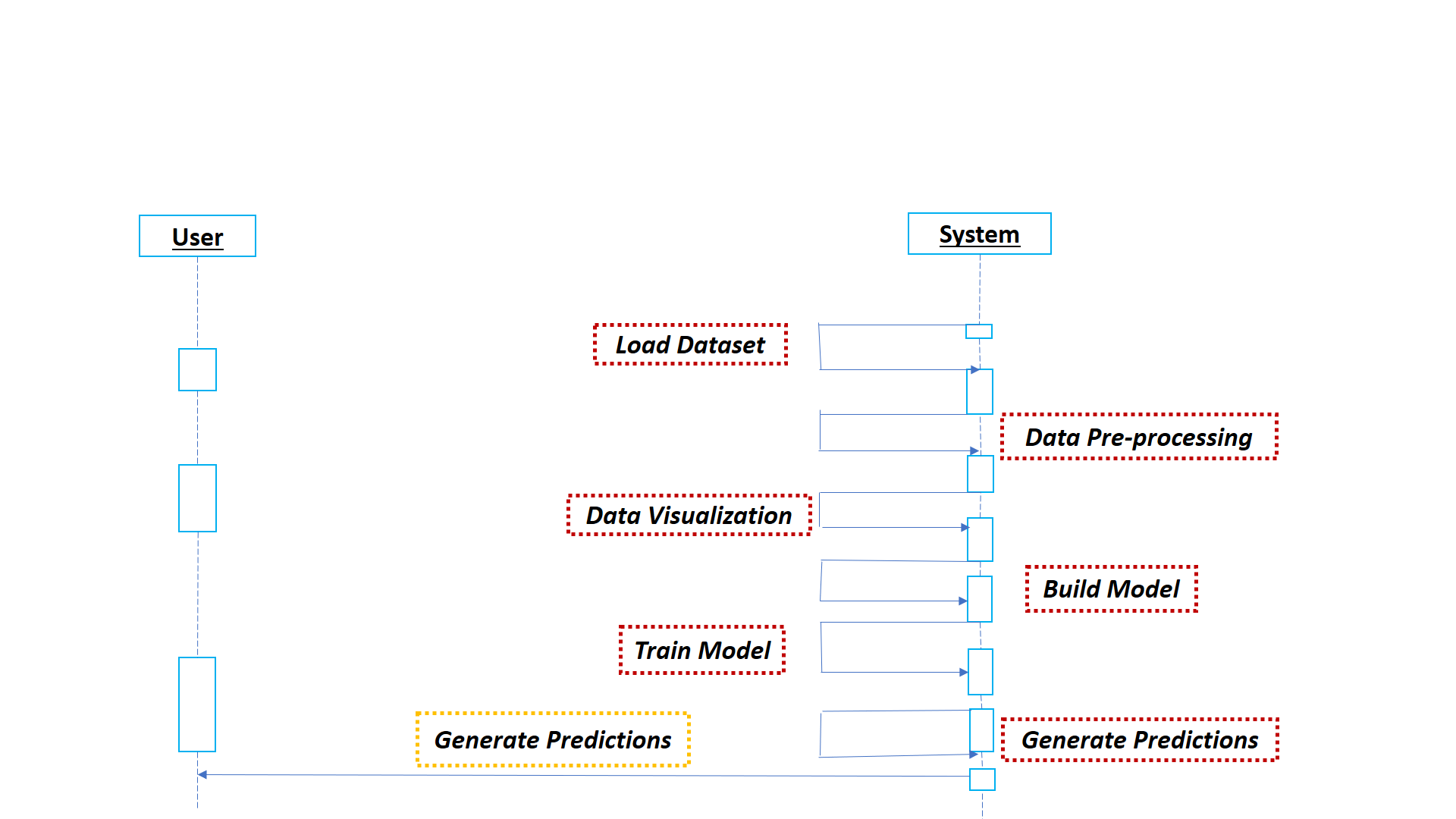


**Fig.2.ActivityDiagram**

#### .4 SEQUENCE DIAGRAM

A sequence diagram in Unified Modelling Language is a kind of interaction diagram that shows the order of operation of the processes with one another. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

Following diagram shows the sequential flow of the system by all the users clearly by means of a sequence diagram.



**Fig.3.Sequence Diagram**

**CHAPTER-IV**

## SYSTEM IMPLEMENTATION

### 4.1 Language Selection

The requirement for this python project is a webcam through which we will capture images. You need to have python(3.6 version recommended) installed on your system, then using pip, you can install the necessary packages.

**4.1.1 OpenCV- Pip install OpenCV-python(face and eye detection).**

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human.

**4.1.2.Tensorflow - Pip install tensorFlow (keras uses tensorFlow as backend)**

* Image Recognition. It is most one of the most popular Uses of TensorFlow.
* Voice Recognition. TensorFlow has significant use in voice recognition systems like Telecom, Mobile companies, security systems, search engines., etc.
* Video Detection.
* Text-based applications

**4.1.3.Keras – Pip install keras(to build our classification models)**

Keras is highly powerful and dynamic framework and comes up with the following advantages-

* Larger community support.
* Easy to test
* Keras neural networks are written in python which makes things simpler
* Keras supports both convolution and recurrent networks.
* Deep earning models are discrete components, so that, you can combine into many ways.

**4.1.4. Pygame – pip install pygame (to play alarm sound)**

* Pygame is a cross-platform set of Python modules which is used to create video games.
* It consists of computer graphics and sound libraries designed to be used with the Python programming language.
* Pygame was officially written by Pete Shinners to replace PySDL
* Pygame is suitable to create client-slide applications that can be potentially wrapped in a standalone executable.

### 4.2. Algorithm

#### Step: -1 Take image as input from a camera

With a webcam, we will take images as input. So to access the webcam, we made an infinite loop that will capture each frame. We use the method provided by

OpenCV, **cv2.VideoCapture(0)** to access the camera and set the capture object

(cap). **cap.read()** will read each frame and we store the image in a frame variable.

A threshold is defined for example if score becomes greater than 15 that means the persons eyes are closed for a long period of time . This is when we beep the alarm using sound.play() **Step: - 2 – Detect Face in the Image and Create a Region of Interest (ROI)**

To detect the face in the image, we need to first convert the image into grayscale as the

OpenCV algorithm for object detection takes gray images in the input. We don’t need color information to detect the objects. We will be using haar cascade classifier to detect faces.

This line is used to set our classifier **face = cv2.CascadeClassifier(‘ path to our haar cascade xml file’)**. Then we perform the detection using **faces =**

**face.detectMultiScale(gray)**. It returns an array of detections with x,y coordinates, and height, the width of the boundary box of the object. Now we can iterate over the faces and draw boundary boxes for each face. for (x,y,w,h) in faces:

cv2.rectangle(frame, (x,y), (x+w, y+h), (100,100,100), 1 )

#### Step: -3 – Detect the eyes from ROI and feed it to the classifier

The same procedure to detect faces is used to detect eyes. First, we set the cascade classifier for eyes in leye and reye respectively then detect the eyes using left\_eye = leye.detectMultiScale(gray). Now we need to extract only the eyes data from the full image. This can be achieved by extracting the boundary box of the eye and then we can pull out the eye image from the frame with this code.

l\_eye = frame[ y : y+h, x : x+w ] **l\_eye** only contains the image data of the eye. This will be fed into our CNN classifier which will predict if eyes are open or closed. Similarly, we will be extracting the right eye into **r\_eye**.

#### Step: -4 – Classifier will Categorize whether Eyes are Open or Closed

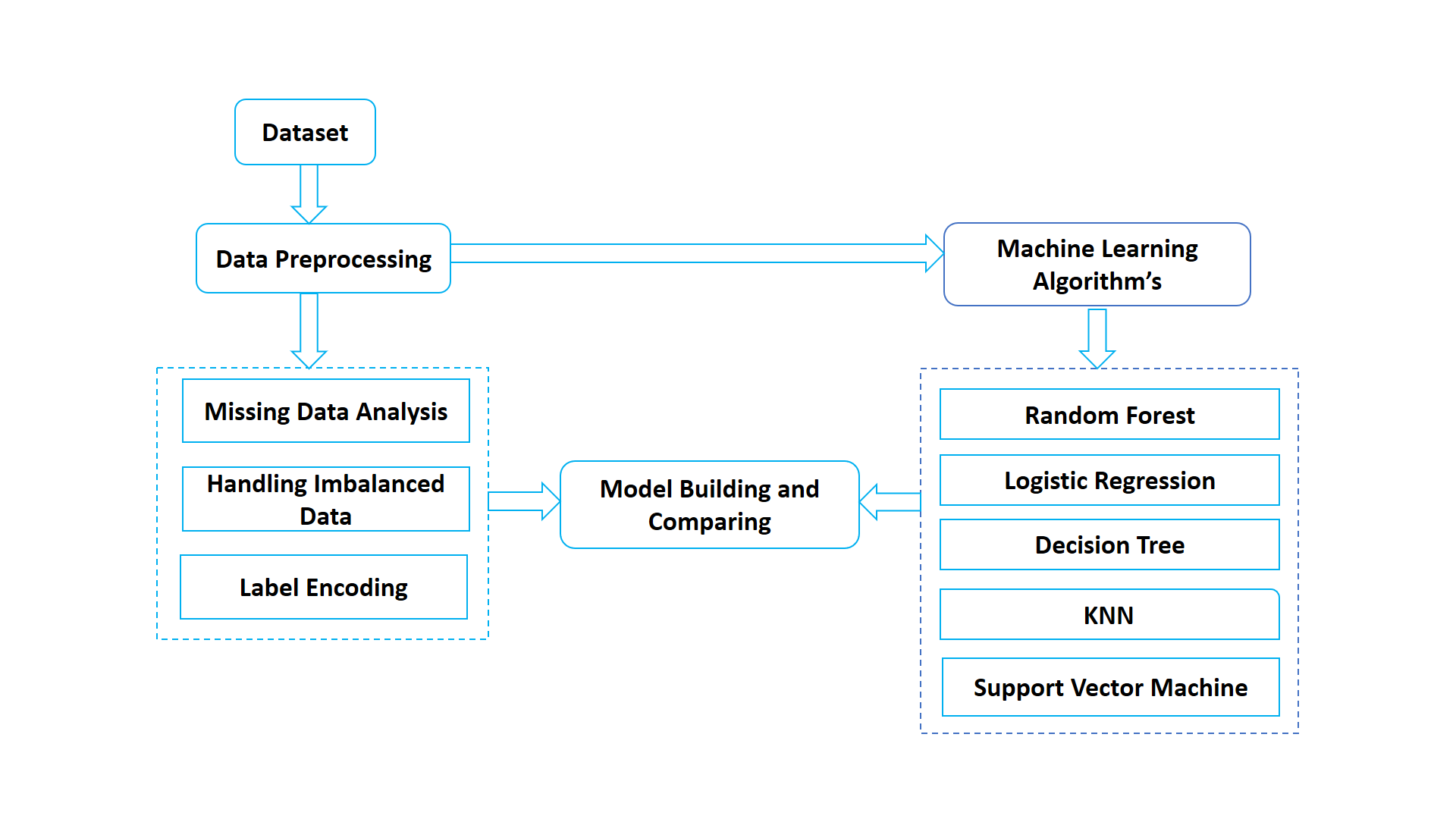
We are using [CNN](https://en.wikipedia.org/wiki/Convolutional_neural_network) classifier for predicting the eye status. To feed our image into the model, we need to perform certain operations because the model needs the correct dimensions to start with. First, we convert the color image into grayscale using **r\_eye = cv2.cvtColor(r\_eye, cv2.COLOR\_BGR2GRAY)**. Then, we resize the image to 24\*24 pixels as our model was trained on 24\*24 pixel images **cv2.resize(r\_eye, (24,24))**.

We normalize our data for better convergence **r\_eye = r\_eye/255** (All values will be between 0-1). Expand the dimensions to feed into our classifier. We loaded our model using **model = load\_model(‘models/cnnCat2.h5’)**.Now we predict each eye with our model. **lpred = model.predict\_classes(l\_eye)**. If the value of lpred[0] = 1, it states that eyes are open, if value of lpred[0] = 0 then, it states that eyes are closed.

#### Step: - 5 – Calculate Score to Check whether Person is Drowsy

The score is basically a value we will use to determine how long the person has closed his eyes. So if both eyes are closed, we will keep on increasing score and when eyes are open, we decrease the score. We are drawing the result on the screen using cv2.putText() function which will display real time status of the person.

##### 4.3. FLOW CHART



**Fig:4 Flow Chart**

### 4.4 Screen Shots

## Figure when a person fully opened his eyes



## Figure when a person half closed his eyes



## Figure when a person is in drowsy state

### 4.5. Sample Code

Sample code has been given to show how the system has been implemented. The methodology used to create the screens and the sample python code are given here.

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

plt.rcParams['figure.figsize']=(5,5)

np.set\_printoptions(precision=4)

from sklearn.metrics import accuracy\_score,confusion\_matrix

from sklearn.inspection import permutation\_importance

import warnings

warnings.filterwarnings('ignore')

df=pd.read\_csv('/content/healthcare-dataset-stroke-data.csv')

df.head(6)

df[['gender','id']].groupby(['gender']).count()

df=df[df['gender']!='Other']

df.head(6)

df.isnull().sum()

mean\_bmi\_has\_stroke=df[df['stroke']==1]['bmi'].mean()

mean\_bmi\_has\_stroke

df.isnull().sum()

df[['stroke','id']].groupby(['stroke']).count()

df\_all=df.copy()

df\_stroke=df[df['stroke']==1]

df\_non=df[df['stroke']==0]

def plot\_pie(column,title="All"):

  fig,axis=plt.subplots(1,1)

  data=df\_all[column].value\_counts()

  plt.pie(data,autopct="%1.2f%%",labels=data.index)

  plt.title(title)

  plt.show()

def plot\_hist(column,title="All"):

  plt.hist(df\_all[column],density=True)

  plt.title(title)

  plt.show()

def plot\_bar\_compare(column,sort=False):

  if sort:

    data\_chruned=df\_stroke[column].value\_counts().sort\_index()

    data\_nonchruned=df\_stroke[column].value\_counts().sort\_index()

  else:

    data\_chruned=df\_stroke[column].value\_counts()

    data\_nonchruned=df\_stroke[column].value\_counts()

  fig,axs=plt.subplots(2,1)

  plt.subplots\_adjust(left=0,bottom=0,right=1,top=2,wspace=0,hspace=0.2)

  axs[0].bar(data\_nonchruned.index,data\_nonchruned)

  axs[0].title.set\_text('No stroke')

  axs[1].bar(data\_chruned.index,data\_chruned)

  axs[1].title.set\_text('Has stroke')

  plt.show()

def plot\_hist\_compare(colums,bins=6):

  plt.hist([df\_non[column], df\_stroke[column]],color=['c','r'])

  plt.legend(('No stroke','Has Stroke'))

  plt.show()

def plot\_bar\_compare(column):

  data\_chruned=df\_stroke[column].value\_counts()

  data\_nonchruned=df\_stroke[column].value\_counts()

  fig,axs=plt.subplots(2,1)

  plt.subplots\_adjust(left=0,bottom=0,right=1,top=2,wspace=0,hspace=0.2)

  axs[0].bar(data\_nonchruned.index,autopct="%1.2f%%",labels=data\_nonchruned.index)

  axs[0].title.set\_text('No stroke')

  axs[1].bar(data\_chruned.index,autopct="%1.2f%%",labels=data\_chruned.index)

  axs[1].title.set\_text('Has stroke')

  plt.show()

def plot\_boxplot(column,title=""):

  sns.boxplot(x="stroke",y=column,palette=["c","r"],hue="stroke",data=df\_all).set.title(title,fontsize=15)

def check\_median(column):

  data\_chruned=df\_stroke[column].describe()

  data\_nonchruned=df\_non[column].describe()

  print("no stroke:{}".format(data\_nonchruned['50%']))

  print("has stroke:{}".format(data\_chruned['50%']))

def check\_most(column):

  data\_chruned=df\_stroke[column].value\_counts()

  data\_nonchruned=df\_non[column].value\_counts()

  print("no stroke: {}".format(data\_nonchruned.index[0]))

  print("has stroke: {}".format(data\_chruned.index[0]))

plot\_pie('stroke')

plot\_pie('hypertension')

plot\_pie('gender')

plot\_pie('heart\_disease')

plot\_pie('ever\_married')

plot\_pie('Residence\_type')

plot\_pie('smoking\_status')

plot\_pie('work\_type')

plot\_hist('age')

plot\_hist('bmi')

plot\_hist('avg\_glucose\_level')

check\_median('age')

check\_most('ever\_married')

from sklearn.preprocessing import LabelEncoder

enc=LabelEncoder()

gender=enc.fit\_transform(df['gender'])

gender=enc.fit\_transform(df['gender'])

smoking\_status=enc.fit\_transform(df['smoking\_status'])

work\_type=enc.fit\_transform(df['work\_type'])

Residence\_type=enc.fit\_transform(df['Residence\_type'])

ever\_married=enc.fit\_transform(df['ever\_married'])

df['work\_type']=work\_type

df['gender']=gender

df['ever\_married']=ever\_married

df['Residence\_type']=Residence\_type

df['smoking\_status']=smoking\_status

df

x=df.drop('stroke',axis=1)

x.head()

from sklearn.tree import DecisionTreeClassifier

from sklearn.neighbors import KNeighborsClassifier

from sklearn.svm import SVC

from sklearn.ensemble import RandomForestClassifier

from sklearn.linear\_model import LogisticRegression

classifiers=[[DecisionTreeClassifier(),'DecisionTreeClassifier'],[KNeighborsClassifier(),'KNeighborsClassifier'],[RandomForestClassifier(),'RandomForestClassifier'],

             [SVC(),'SVC'],[LogisticRegression(),'LogisticRegression']]

for cls in classifiers:

  model=cls[0]

  model.fit(x\_train,y\_train)

  y\_pred=model.predict(x\_test)

  print(cls[1])

  print("Confusion Matrix:\n",confusion\_matrix(y\_test,y\_pred))

  print("Accuracy:",accuracy\_score(y\_test,y\_pred)\*100)

models=['DecisionTree','KNeighbors','RandomForest','SVM','LogisticRegression']

Accuracy=[91.58512720156556,95.00978473581213,95.00978473581213,95.00978473581213,94.81409001956948]

colors=['green','pink','blue','yellow','orange']

plt.title('Models Accuracy')

plt.xlabel('models')

plt.ylabel('Accuracy')

plt.bar(models,Accuracy,color=colors)

plt.show()

**CHAPTER -V**

## SYSTEM TESTING

### 5.1 Testing Description

Testing is the process of detecting errors. Testing perform a very critical role for quality assurance and for ensuring the reliability of software. The results of testing are used later on during maintenance also.

The main aim of testing is often to demonstrate that a program works by showing that it has to errors. The basic purpose of testing phase is to detect the errors then lot of technique to test the code or the project in the form testing fundamentals be present in the program. Testing is the process of executing a program with the intent of finding errors.

### 5.2 Testing Strategy

#### 5.2.1 Unit Testing

The software is tested using the unit test method. Unit testing focuses validation effort in the smallest unit of software design module description of the project. Using the procedural design description as a staff, import control parts are tested to uncover errors with in the secondary of the module.

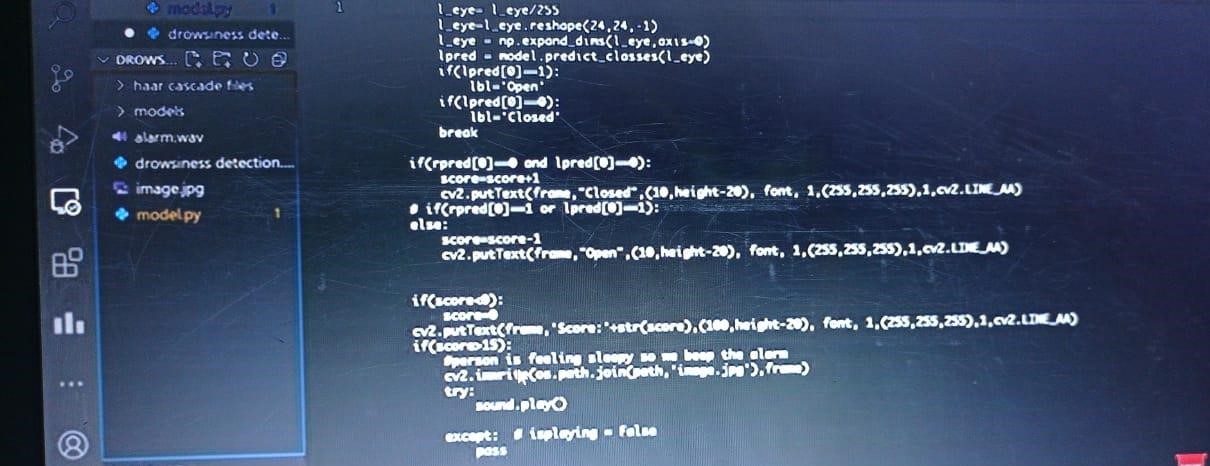
### 5.2.2Integration Testing

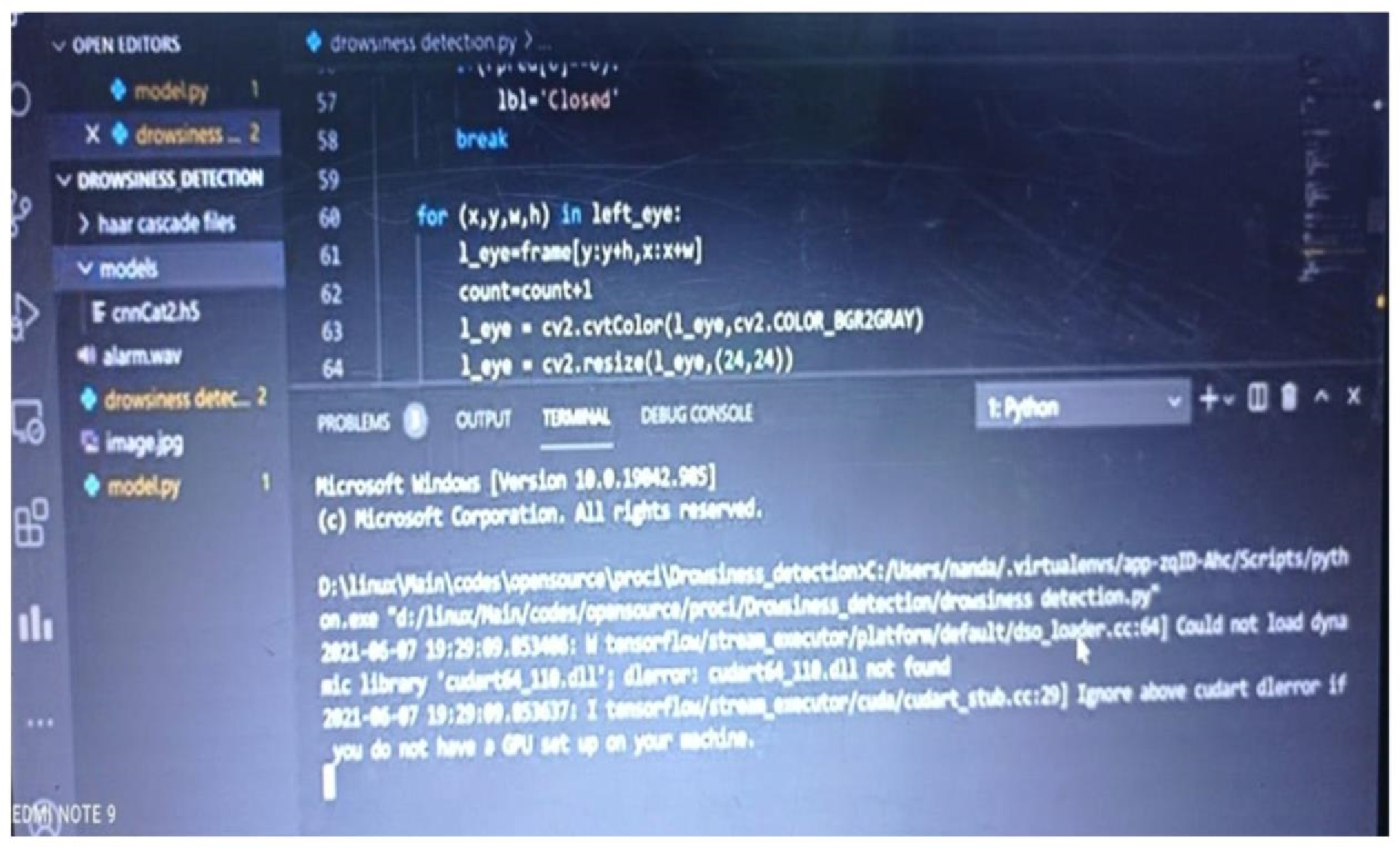
The testing is the systematic technique for constructing the program structure by informing the test in each module and later combining the entire individual module to form a large program.

### 5.2.3Validation Testing

Validation testing is the process of testing the input. Whether the given inputs are valid or not. When working with databases it is important to validate user’s entries, which can be done by using scripting code.The terms verification and validation are used interchangeably we will describe both these methods.

## Screen Shots





**CHAPTER –VI**

## CONCLUSION

* In this project, we have reviewed the various methods available to determine the drowsiness state of a driver. Although there is no universally accepted definition for drowsiness, the various definitions and the reasons behind them were discussed. This paper also discusses the various ways in which drowsiness can be manipulated in a simulated environment.
* The various measures used to detect drowsiness include subjective, vehicle-based, physiological and behavioral measures; these were also discussed in detail and the advantages and disadvantages of each measure were described. Although the accuracy rate of using physiological measures to detect drowsiness is high, these are highly intrusive. However, this intrusive nature can be resolved by using contactless electrode placement. Hence, it would be worth fusing physiological measures in the development of an efficient drowsiness detection system.

**CHAPTER -VII**

## REFERENCES

1. Makowiec-Dąbrowska T. et al.: Zmęczenie pracą u kierowców autobusów miejskich (Workfatigue in Urban bus driver”, article of the Institute of

Occupational Medicine im. Professor J. Nofer, Łódź 2015 (in Polish).

1. Statistics related to fatique due to drivers {Available –2017.06.01:http://www. orlenbezpiecznedrogi.pl/downloads/material-backgroundowy.doc} (in Polish).
2. The fatique detection system implemented in cars Ford {Available – 2017.06.01:http://www.technowinki.onet.pl/motoryzacja/system-monitorowaniakoncentracji-kierowcy-w-samochodzie-ford-focus/6f4kdg} (in Polish).
3. The fatique detection system used in the car Skoda Octavia{Available – 2017.06.01:

http://www.mojafirma.infor.pl/moto/eksploatacja-auta/bezpieczenstwo/315625,2, Skoda-Octavia-systemy-bezpieczenstwa.html} (in Polish). .

1. Skoda brand fatique detection system {Available – 2017.06.01:http://www.sklep

.skoda.pl/upload/files/Fabia\_3\_akcesoria\_2015-01.pdf} (inPolish).

1. Bosch Drowsiness System { Available – 2017.06.01:http://www.boschprasa.pl/informacja.php?idinformacji=1356}.
2. Pawelec J., Krawczyk Z.: „Bezpieczeństwo w ruchudrogowym. Możliwościi przykładyzastosowańnowychtechnologii”,Transcomp –XIV International Conference Computer Systems Aided Science, Industry and Transport.
3. View at driver fatigue information (Ford Driver Assistant){Available2017.06.01:https://cdsdetroit.files.wordpress.com/2011/10/ford-arning.jpg?w=601}.
4. Information about the vision system for fatique detection {Available –2017.06.01:

http://www.neuroblog.pl/video/kamera-monitorujace-emocje-kierownica/}(in Polish).