**“DIVYANG HELPER APPLICATION”**

A Report

Submitted in partial fulfilment for the Award of the degree of

**BACHELOR OF TECHNOLOGY**

In

**COMPUTER SCIENCE ENGINEERING**

Submitted by

**SARVAGYA SRIRAM MAMGAIN (1013528)**

**Under the Guidance of Prof :– Mrs. Rishika Yadav**



**DEPARTMENT OF COMPUTER SCIENCE GRAPHIC ERA HILL UNIVERSITY, UTTARAKHAND**

**2020-2021**

# DECLARATION

We hereby declare that this submission is our own work to the best of our knowledge and belief. It contains no material previously or written by any other persons nor material which to a substantial extent has been accepted for the award of any degree or diploma of the university or other institute of higher learning, except the acknowledgement and references has been made in the text.

|  |  |  |
| --- | --- | --- |
| **NAME** | **ROLL NUMBER** | **SIGNATURE** |
| UDAY SURAJ YADAV | 1018379 | UDAY |
| SARVAGYA SRIRAM MAMGAIN  RITWIK BAHUKHANDI | 1013528  1018332 | SARVAGYA  RITWIK |

**CERTIFICATE**

This is to certify that project report entitled “DIVYANG HELPER” which is being submitted by UDAY SURAJ YADAV (1018379), SARVAGYA MAMGAIN (1013528), RITWIK BAHUKHANDI (1018332) in partial fulfilment for the requirement for the award of the degree of Bachelor of Technology, VIII Semester in Department of Computer Science Engineering of Graphic Era Hill University, Dehradun, Uttarakhand in the year 2021. We have worked under the guidance of Mrs. Rashika Yadav (Professor of CSE, GEHU) and have fulfilled the requirement for the submission of the project.

The matter presented in this dissertation has not been submitted by me anywhere for the award of any other degree of this or any other Institute (2021).

UDAY SURAJ YADAV SARVAGYA MAMGAIN RITWIK BAHUKHANDI

This to certify that the above statement made by the candidate is correct and true to the best of our knowledge.

Mentor/Project Guide :- Mrs. Rishika Yadav(Prof. of CSE, GEHU)

Date: 27/5/2021

External Examiner Signature

Internal Examiner Signature

**ACKNOWLEDGEMENT**

First of all, we would like to thank the almighty God for listening our prayers and giving us strength to complete the dissertation work.

We would like to express the deep sense of gratitude of and thanks profusely to Mr. Amit, Assistant Professor, Department of Computer Science, Graphic Era Hill University, Dehradun (Uttarakhand), our guides and mentors, without the wise counsel and able guidance, it would have been impossible to complete the dissertation in this manner. I would also sincerely like to thank CSE Department for giving us the opportunity to work on enhancing our technical skills while undergoing this project. This project also helped in understanding the various parameters which are involved in the development of a web application and the working and integration of front end along with the back end to create a fully functional web application.

We are also grateful to Digital India Initiative of Indian Government for providing us uninterrupted Internet access with good internet speed quality, because of which we all are able to prepare it at home with the help of various Digital sources and by taking guidance digitally from our guides during these extreme days of pain and anguish.

We would like to thank Professors and HOD CSE Department and whole

of department for their constant support.

We finally pray that Almighty fulfil the aspirations of all the people who have been a part of this journey and those who will be part of future journeys.

|  |  |  |
| --- | --- | --- |
| **NAME** | **ROLL NUMBER** | **SIGNATURE** |
| UDAY SURAJ YADAV | 1018379 | UDAY |
| SARVAGYA MAMGAIN  RITWIK BAHUKHANDI | 1013528  1018332 | SARVAGYA  RITWIK |

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **Title** | **Page No.** |
|  |  |
| Abstract | 6 |
| Scope & Objective of the project | 7 |
| Introduction | 8 |
| Working , Resources , Features & Requirements | 8-9 |
| Code and Snapshots | 10-22 |
| Conclusion & Future Scope | 23 |

**ABSTRACT:-**

A “DIVYANG HELPER” is an software for the disabled ones . It provides a digital platform mainly for persons with disabilities for their use in various fields.

Application would provide an easy-to-use digital platform for persons with different disabilities.

First, Eye Tracking, for those who are limbless.

Second, Voice Recognition, for those who are not able to see objects clearly or partially and also extending this facility to those who are blind.

Over the years, facilities for disables by using innovative technologies have increased and diversified. Multiple facilities provided by different social groups and by government at different places by using different means of modes and schemes .

We know that these people with disabilities have problem while expressing

their thoughts to others . And, seeing this gap between them and the real world, we made this project so as to bridge the gap between them and the world.

For the paralysed people, we know they cannot move, talk, walk, etc. only their eyes can move, so we made a keyboard such that it recognizes the eye movement, and based on that, it can detect which word the user want to press and so on , it goes on to make up sentences, which later the computer says, expressing what the user wanted to actually say.

Now, for those who are blind or partially blind, we can recognize their voice and based on that we do the needful actions.

**SCOPE AND OBJECTIVE :-**

This project has got a great scope in the medical field , as this can act as a tool for the needy ones . It will act as a translator for those who cannot do it for themselves , as using this paralysed and blind people can express what they want, what they mean.

We got the motivation for the project, when we saw a somewhat similar device , but it had lots of equipments and costed lakhs of rupees, but our model is just a piece of code, that can run efficiently free of cost, even we don’t need any graphic card to work this. This is a major upgradation we worked upon.

**INTRODUCTION**

The main objective of this project is to design a dynamic, supportive application with grievance redressal mechanism with full-fledged logic and conducive environment using new and innovative technologies in market with maximum functionalities, efficiency, maximum optimal speed on internet and one key feature it must be informative, supportive.

**WORKING:-**

1)For Paralysed Eye Keyboard ->

* Firstly, we used dlib predictor to find exact eye coordinates
* Next, we extract the pupil from the eye, using XOR operation from the thresholded pupil image .
* Now, we convert it into black and white image, such that, white = 0 and black =1.So, if our eye moves left, more will be 1’s, ie, more black part, and similarly we find the right side movement.
* For blinking we calculate the eulcedean distance between the upper and lower limits of eye, and if it is almost zero, means the eye is closed.
* Then we make a keyboard, and the eye movements give the directions to the cursor on the keyboard and hence we type.

2)For Blind – Speech recognition system ->

* Based on the speech, we convert it into a text format.
* Now we find out the words, which mean something, Eg: “Tell the TIME “ , here TIME is important word, and finding such keywords, we depict what the user want us to do.
* Now, the recognition system does the needful and returns another text, which is reconverted to Speech format.

**DEVELOPMENT TOOLS: -**

* We make our interface with Tkinter library of python.
* Eye keyboard used – opencv library of python
* Speech recognition system – pyttsx3 and speechrecognition library of python
* Language used: python
* We used Jupyter notebook as our editor.
* other libraries: dlib , numpy , playsound, gtts , etc.

**SYSTEM REQUIREMENTS:**

**On system ->**

* Operating system(android)
* Web browser = (Mozilla Firefox, Google chrome, Opera, Safari)
* Langauge = Python
* Other support = Webcam, microphone

**FEATURES: -**

* Inexpensive
* User friendly
* Fully informative
* Easily available and affordable
* Don’t need any high specifications on your system

**SOURCE CODE**

import cv2

import numpy as np import dlib

import playsound import time

from gtts import gTTS

from datetime import datetime import pyttsx3

import datetime

import speech\_recognition as sr import wikipedia

import webbrowser import os

from tkinter import \*

from math import hypot

*# DLIB settings*

detector = dlib.get\_frontal\_face\_detector() predictor =

dlib.shape\_predictor(**"shape\_predictor\_68\_face\_landmarks.dat"**)

*#Speech Engine settings*

engine=pyttsx3.init(**'sapi5'**) *#sapi5 is a prog of microsoft i.e. used to control voices*

voices=engine.getProperty(**'voices'**) engine.setProperty(**'voice'**,voices[1].id)

*# Keyboard settings*

keyboard = np.zeros((800,1500,3),np.uint8) keys\_set\_1 =

{0:**"Q"**,1:**"W"**,2:**"E"**,3:**"R"**,4:**"T"**,5:**"Y"**,6:**"U"**,7:**"I"**,8:**"O"**,9:**"P"**,

10:**"A"**,11:**"S"**,12:**"D"**,13:**"F"**,14:**"G"**,15:**"H"**,16:**"J"**,17:**"K"**,18:**"L"**,

19:**"Z"**,20:**"X"**,21:**"C"**,22:**"V"**,23:**"B"**,24:**"N"**,25:**"M"**,

26:**"| |"**,27:**"SAY"**}

board = np.zeros((500,500),np.uint8) board[:] = 255

*# FUNCTIONS USED =>*

*#Function 1) for designing the keyboard*

def letter(letter\_index,this\_letter,letter\_light):

*# Keys*

if (letter\_index==0): x=0

y=0

elif (letter\_index==1): x=150

y=0

elif (letter\_index==2): x=300

y=0

elif (letter\_index==3): x=450

y=0

elif (letter\_index==4): x=600

y=0

elif (letter\_index==5): x=750

y=0

elif (letter\_index==6): x=900

y=0

elif (letter\_index==7): x=1050

y=0

elif (letter\_index==8): x=1200

y=0

elif (letter\_index==9): x=1350

y=0

elif (letter\_index==10): x=0+70

y=200

elif (letter\_index==11): x=150+70

y=200

elif (letter\_index==12): x=300+70

y=200

elif (letter\_index==13): x=450+70

y=200

elif (letter\_index==14): x=600+70

y=200

elif (letter\_index==15): x=750+70

y=200

elif (letter\_index==16): x=900+70

y=200

elif (letter\_index==17): x=1050+70

y=200

elif (letter\_index==18): x=1200+70

y=200

elif (letter\_index==19): x=150

y=400

elif (letter\_index==20): x=300

y=400

elif (letter\_index==21): x=450

y=400

elif (letter\_index==22): x=600

y=400

elif (letter\_index==23): x=750

y=400

elif (letter\_index==24): x=900

y=400

elif (letter\_index==25): x=1050

y=400

elif (letter\_index==26): x=250

y=600

elif (letter\_index==27): x=850

y=600

height = 200

thickness = 3

if letter\_index == 26: width = 600

elif letter\_index == 27: width = 350

else:

width = 150

*# Text Settings*

font\_letters = cv2.FONT\_HERSHEY\_PLAIN text = this\_letter

font\_scale = 10

font\_thickness = 4

if letter\_light is True: cv2.rectangle(keyboard,(x+thickness,y+thickness),(x+width-

thickness,y+height-thickness),(255,255,255),-1) else:

cv2.rectangle(keyboard,(x+thickness,y+thickness),(x+width- thickness,y+height-thickness),(255,0,0),thickness)

text\_size = cv2.getTextSize(text,font\_letters,font\_scale,font\_thickness)[0] *#has value = ((104,92) , 50) , hence taking first part only,i.e,(104,92)*

width\_text , height\_text = text\_size[0],text\_size[1] *#taking 104 and 92 from (104,92)*

text\_x = int((width - width\_text)/2) +x text\_y = int((height + height\_text)/2)+y

cv2.putText(keyboard,text,(text\_x,text\_y),font\_letters,font\_scale,(25 5,0,0),font\_thickness)

*#Function 2) for finding the middle point*

def midpoint(p1,p2):

return int((p1.x+p2.x)/2),int((p1.y+p2.y)/2)

*#Function 3) calculates the ratio to blink , and if satisfies the condition than blink is shown and letter gets selected.*

def get\_blinking\_ratio(eye\_points,facial\_landmarks): left\_point =

(facial\_landmarks.part(eye\_points[0]).x,facial\_landmarks.part(eye\_poi nts[0]).y)

right\_point = (facial\_landmarks.part(eye\_points[3]).x,facial\_landmarks.part(eye\_poi nts[3]).y)

center\_top = midpoint(facial\_landmarks.part(eye\_points[1]),facial\_landmarks.part(e ye\_points[2]))

center\_bottom = midpoint(facial\_landmarks.part(eye\_points[5]),facial\_landmarks.part(e ye\_points[4]))

*#hor\_line = cv2.line(frame,left\_point,right\_point,(0,255,0),2)*

*#ver\_line = cv2.line(frame,center\_top,center\_bottom,(0,255,0),2)*

*#Ecledian distance to find the dist bw center top and bottom to check eye blinking :-root (x1-x2 square + y1-y2 square)*

hor\_line\_length = hypot((left\_point[0]- right\_point[0]),(left\_point[1]-right\_point[1])) ver\_line\_length = hypot((center\_top[0]-

center\_bottom[0]),(center\_top[1]-center\_bottom[1]))+0.0001 ratio = hor\_line\_length/ver\_line\_length

return ratio

*#Function 4) calculates the gaze(% of eye black) for left and right eyes*

def get\_gaze\_ratio(eye\_points,facial\_landmarks): left\_eye\_region =

np.array([(facial\_landmarks.part(eye\_points[0]).x,facial\_landmarks.pa rt(eye\_points[0]).y),

(facial\_landmarks.part(eye\_points[1]).x,facial\_landmarks.part(eye\_poi nts[1]).y),

(facial\_landmarks.part(eye\_points[2]).x,facial\_landmarks.part(eye\_poi nts[2]).y),

(facial\_landmarks.part(eye\_points[3]).x,facial\_landmarks.part(eye\_poi nts[3]).y),

(facial\_landmarks.part(eye\_points[4]).x,facial\_landmarks.part(eye\_poi nts[4]).y),

(facial\_landmarks.part(eye\_points[5]).x,facial\_landmarks.part(eye\_poi nts[5]).y)],np.int32)

*#print(left\_eye\_region) #cv2.polylines(frame,[left\_eye\_region],True,(0,0,255),2)*

height,width,\_=frame.shape

mask = np.zeros((height,width),np.uint8) cv2.polylines(mask,[left\_eye\_region],True,255,2) *# color*

*should be 3 in rgb but in mask only 0 or 1 , hence only 2 colors black or white*

cv2.fillPoly(mask,[left\_eye\_region],255)

eye = cv2.bitwise\_and(gray , gray , mask=mask)

min\_x = np.min(left\_eye\_region[:,0]) max\_x = np.max(left\_eye\_region[:,0]) min\_y = np.min(left\_eye\_region[:,1]) max\_y = np.max(left\_eye\_region[:,1])

left\_eye\_cropped=eye[min\_y:max\_y,min\_x:max\_x]

\_,threshold\_eye = cv2.threshold(left\_eye\_cropped,70,255,cv2.THRESH\_BINARY)

h,w= threshold\_eye.shape

left\_side\_threshold = threshold\_eye[0:h,0:int(w/2)] left\_side\_white = cv2.countNonZero(left\_side\_threshold)

right\_side\_threshold = threshold\_eye[0:h,int(w/2):w] right\_side\_white = cv2.countNonZero(right\_side\_threshold)

*# if left\_side\_white == 0:*

*# right*

gaze\_ratio = left\_side\_white/((0.0001)+right\_side\_white) return gaze\_ratio

*#Function 5)* speaks the text out

def speak(text):

tts = gTTS(text=text,lang=**"en"**) x = datetime.now()

s=**""**+str(x)

s1 = **"year\_"**+s[:4]

+**"\_"**+s[5:7]+**"\_"**+s[8:10]+**"\_time"**+**"\_"**+s[11:13]+**"\_"**+s[14:16]+**"\_"**+s[17:19

]

filename = s1+**"\_.mp3"** tts.save(filename) playsound.playsound(filename)

*#Function 6) Used to play any audio*

def speak(audio): engine.say(audio) engine.runAndWait()

*#Function 7) default speech that speech rec. system starts with*

def wishMe():

hour=int(datetime.datetime.now().hour) if hour >=0 and hour <12:

speak(**"Good Morning Sir"**)

print(**"Good Morning Sir"**)

elif hour >=12 and hour<18:

speak(**"Good Afternoon Sir"**)

print(**"Good Afternoon Sir"**)

else:

speak(**"Good Evening Sir"**)

print(**"Good Evening Sir"**)

speak(**" I am JARVIS Please tell How may I help you?"**)

print(**" I am JARVIS Please tell How may I help you?"**)

*#Function 8) used to take input from the user*

def takeCommand():

*# it takes microphone input from user and return string output*

r=sr.Recognizer()

with sr.Microphone() as source: print(**' Listening...'**)

r.pause\_threshold=0.5 *# max gap can be 1 sec while giving input else it will take input after 1 sec gap*

audio=r.listen(source)

try:

print(**'Recognising...'**) query=r.recognize\_google(audio,language=**"en-in"**) print(**'User said:-**\n**'**,query)

except Exception as e:

*# print(e)*

print(**'Can you repeat it , PLEASE'**) return **"None"**

return query

*# Opening the webcam*

cap = cv2.VideoCapture(0)

font = cv2.FONT\_HERSHEY\_SIMPLEX

*#Counters*

frames\_cnt = 0

letter\_number = 0

blinking\_frame = 0

text = **" "** *#single blank becoz it if by mistake as start SAY button pressed then also error*

keyboard\_selected = **"left"** last\_keyboard\_selected = **"left"** turn\_cnt\_left = 0

turn\_cnt\_right = 0

*# starting to make interface to take input via tkinter library*

t=Tk() canvas=Canvas(t,width=300,height=160) canvas.pack()

t.geometry(**"400x400"**)

var1 = StringVar()

label = Label( t, width=65, padx=10, pady=20, anchor=CENTER, font=88, textvariable=var1 )

var1.set(**"DIVYANG HELPER**\n**An Aid to the Needy..."**) label.pack()

var2 = StringVar()

label = Label( t, width=20, padx=15, pady=22, anchor=CENTER, font=24, textvariable=var2 )

var2.set(**"Select Your Choice**\n**1:Eye Keyboard**\n**2:Speech Recognition system"**)

label.pack()

*# function for tkinter to to run when button1 is presssed*

def button\_command(): global choices choices = entry1.get() exit()

t.quit() return None

entry1 = Entry(t,width = 40) entry1.pack(padx = 20, pady = 20)

button1 = Button( text=**"Click"**, padx=**"66px"**, pady=**"20px"**, font=**"45"**, bg=**"lightblue"**, activeforeground=**"crimson"**, activebackground=**"red"**, command=button\_command)

button1.pack() t.mainloop()

*#option for choice :- #1-Eye keyboard*

*#2-Speech rec. system*

*#3-Invalid input*

print(**"You chose -"**) print(choices)

if choices != **"1"** and choices!=**"2"**: print(**"Wrong choice"**)

elif choices ==**"1"**:

*# eye keyboard*

while True:

ret,frame = cap.read(0)

frame = cv2.resize(frame,None,fy=0.5,fx=0.5)

keyboard[:] = (0,0,0) *# this is used as without it keyboard key once lightened is always lightened , which is not desired*

frames\_cnt+=1

*#showing directions*

new\_frame = np.zeros((500,500,3),np.uint8) gray = cv2.cvtColor(frame,cv2.COLOR\_BGR2GRAY) active\_letter = keys\_set\_1[letter\_number]

faces=detector(gray) for face in faces:

x,y = face.left(),face.top()

x1,y1 = face.right(),face.bottom()

*#cv2.rectangle(frame,(x,y),(x1,y1),(0,255,0),2)*

landmarks = predictor(gray,face)

*#DETECT BLINKING*

left\_eye\_ratio= get\_blinking\_ratio([36,37,38,39,40,41],landmarks)

right\_eye\_ratio = get\_blinking\_ratio([42,43,44,45,46,47],landmarks)

blinking\_ratio = (right\_eye\_ratio+left\_eye\_ratio)/2 if blinking\_ratio >5.5:

cv2.putText(frame,**"Blinking"**,(50,55),font,2,(20,240,90),2) blinking\_frame+=1

frames\_cnt -=1 *# when eyes are closed don’t let frames go ahead else letter would also change*

*#Typing letters*

if blinking\_frame == 7:

if active\_letter == **"| |"**: text+=**" "**

elif active\_letter == **"SAY"**:

*#speak(text)*

print(text)

*#below code to skip SAY button been pressed again(due to permission issue) by mistake in process of speaking*

if last\_keyboard\_selected == **"right"**: letter\_number+=1

elif last\_keyboard\_selected == **"left"**: letter\_number-=1

letter\_number%=28

else:

text+=active\_letter play\_audio(**"pop.mp3"**) time.sleep(1)

else:

blinking\_frame = 0 *# To stop count if we open the*

*eyes*

*#GAZE DETECTION*

gaze\_ratio\_left\_eye = get\_gaze\_ratio([36,37,38,39,40,41],landmarks)

gaze\_ratio\_right\_eye = get\_gaze\_ratio([42,43,44,45,46,47],landmarks)

gaze = (gaze\_ratio\_left\_eye+gaze\_ratio\_right\_eye)/2

*#using only gaze for right eye as it is giving more accurate results than left*

side=**""**

if blinking\_frame > 0: *# just to stop left right movement by mistake when eyes are closed*

continue

elif blinking\_frame == 0: if gaze<0.5:

cv2.putText(frame,str("Right"),(50,100),font,2,(0,0,255),3)

side+=**"right"** new\_frame[:] = (0,0,255)

keyboard\_selected = **"right"**

if turn\_cnt\_left == 0: turn\_cnt\_right+=1

else:

turn\_cnt\_left=0

if turn\_cnt\_right ==35: play\_audio(**"right.mp3"**) last\_keyboard\_selected = **"right"**

elif gaze\_ratio\_left\_eye>1.2:

cv2.putText(frame,str("Left"),(50,100),font,2,(0,0,255),3)

side+=**"left"**

new\_frame[:] = (255,0,0) keyboard\_selected = **"left"**

if turn\_cnt\_right == 0:

turn\_cnt\_left+=1 else:

turn\_cnt\_right=0

if turn\_cnt\_left == 40: play\_audio(**"left.mp3"**) last\_keyboard\_selected = **"left"**

else:

side+=**"center"**

cv2.putText(frame,str(gaze),(50,100),font,2,(0,0,255),3)

*# Drawing letters*

if frames\_cnt == 30: *# Changing each letter after these many frames*

if last\_keyboard\_selected == **"right"**: letter\_number+=1

elif last\_keyboard\_selected == **"left"**: letter\_number-=1

letter\_number%=28 *# To Continue from start again even after the end of 15 letters*

frames\_cnt=0

for i in range(0,28): if i==letter\_number:

light = True else:

light = False letter(i,keys\_set\_1[i],light)

cv2.putText(board,text,(50,100),font,2,0,3)

*# Showing various frames* cv2.imshow(**"Frame = "**,frame) cv2.imshow(**"New Frame = "**,new\_frame) cv2.imshow(**"Key board = "**,keyboard) cv2.imshow(**"Board = "**,board) key=cv2.waitKey(1)

if key==27:

break

cap.release() cv2.destroyAllWindows()

elif choices == **"2"**: *#print("Yo")* wishMe()

if 1:

query= takeCommand().lower()

*#logic for executing tasks based on query*

if **'wikipedia'** in query: speak(**'Searching wikipedia...'**) query=query.replace(**'wikipedia'**,**''**)

results=wikipedia.summary(query,sentences=2) speak(**'According to wikipedia '**) print(results)

speak(results)

elif **'open youtube'** in query: webbrowser.open(**'youtube.com'**)

elif **'open brainly'** in query: webbrowser.open(**'brainly.com'**)

elif **'open google'** in query: webbrowser.open(**'google.com'**)

elif **'open stackoverflow'** in query: webbrowser.open(**'stakoverflow.com'**)

elif **'play music'** in query: music\_dir=**'C:**\\**music\_dir'** songs=os.listdir(music\_dir) os.startfile(os.path.join(music\_dir,songs[0]))

elif **'the time'** in query:

strTime =datetime.datetime.now().strftime(**"%H:%M:%S"**) speak(**f"sir , the time is** {strTime}**"**)

elif **'open code'** in query:

codepath=**"C:**\\**Users**\\**HP**\\**AppData**\\**Local**\\**Programs**\\**Microsoft VS Code**\\**Code.exe"**

os.startfile(codepath)

elif **'email to mom'** in query: try:

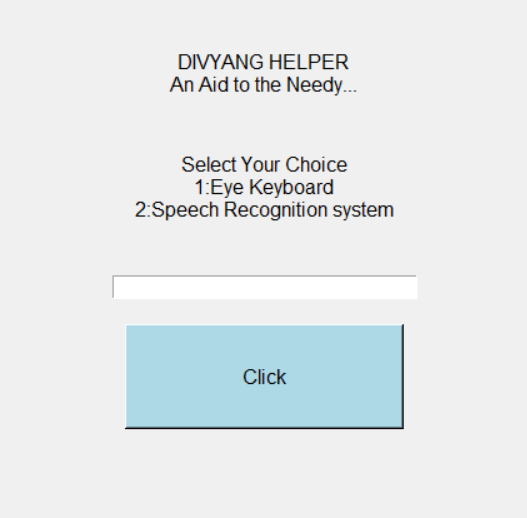
speak(**'What should i send sir '**) content=takeCommand() to=**'sramlax72@gmail.com'** sendemail(to, content) speak(**'sir, email has been sent'**)

except Exception as e: print(e)

speak(**"Sorry but the email was not sent"**)

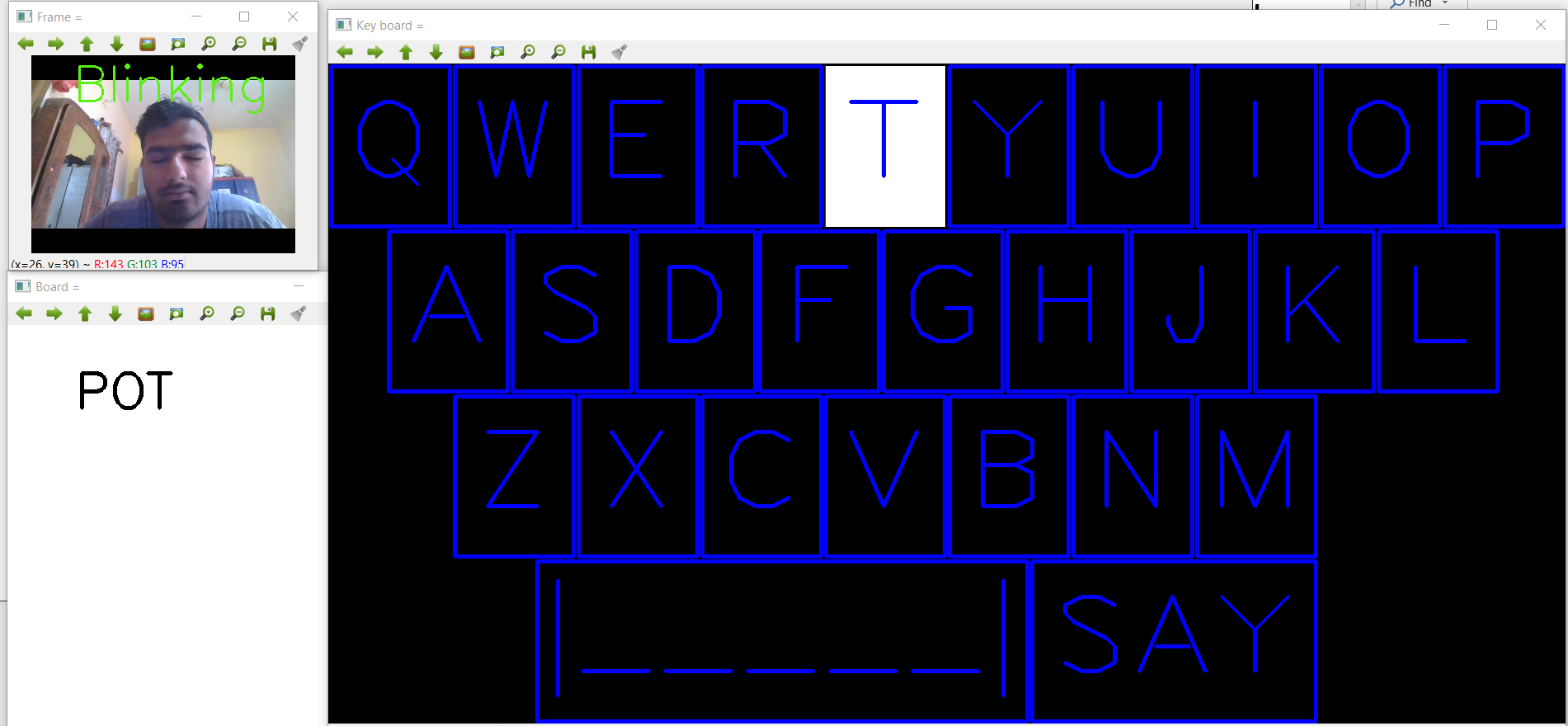
**SNAPSHOTS :-**

1)Staring interface :-

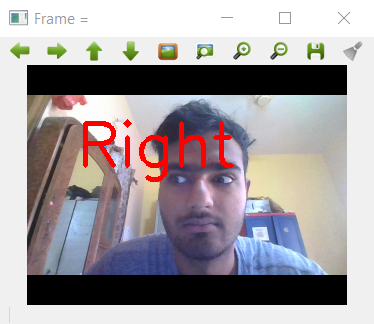


2)Eye Keyboard :-

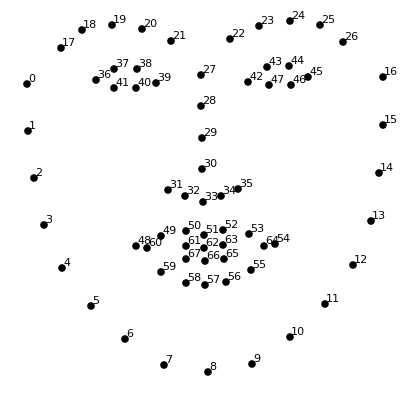
[2.1] = selecting any alphabet



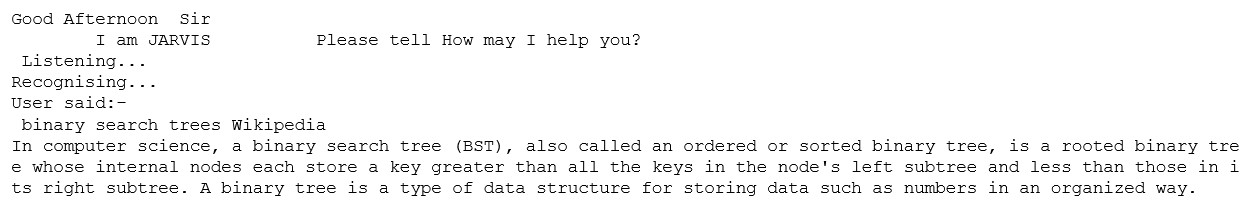
[2.2] = looking right side(remember that if you look right, then your webcam image will look to be seeing left, but actually it is right only, as shown below)

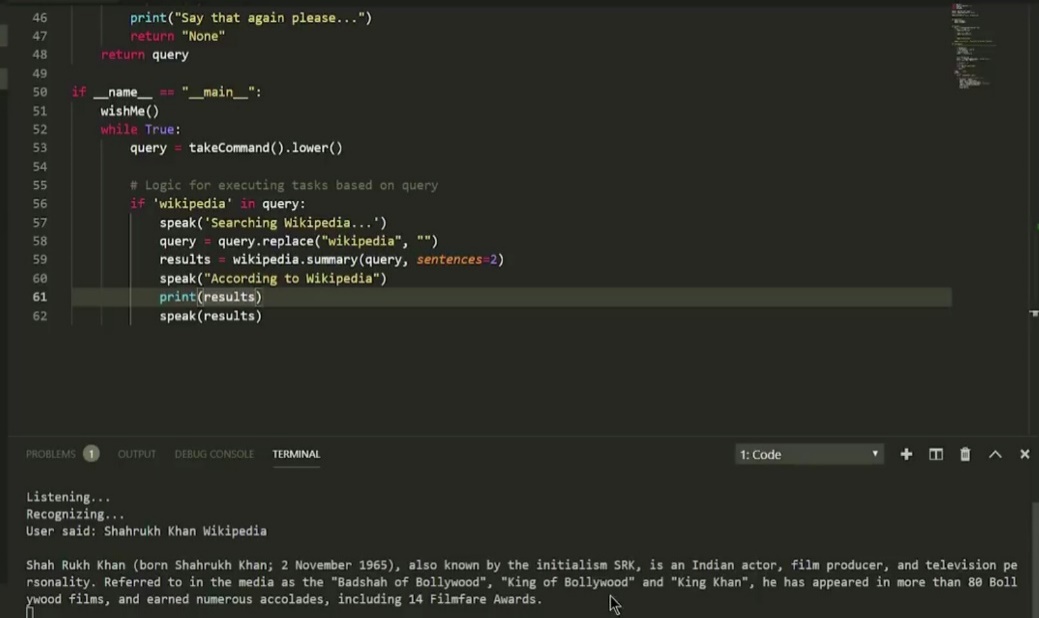


[2.3] = Facial points that we used to extract eye.

****

3)Speech Recognition System:-



****

**CONCLUSION & FUTURE SCOPE:-**

So, in nut shell, we would tell that this model is a successful model in the objective to provide help to the needy disabled people by predicting input correctly and giving correct outputs.

Although this model is self-efficient and sufficient in its operations, but still it can be improved in some fields, such as making this available to all mobile users via android and MAC-OS, also we can increase its limit over more type of diseases , etc. These changes can be brought up in the future versions of the model.