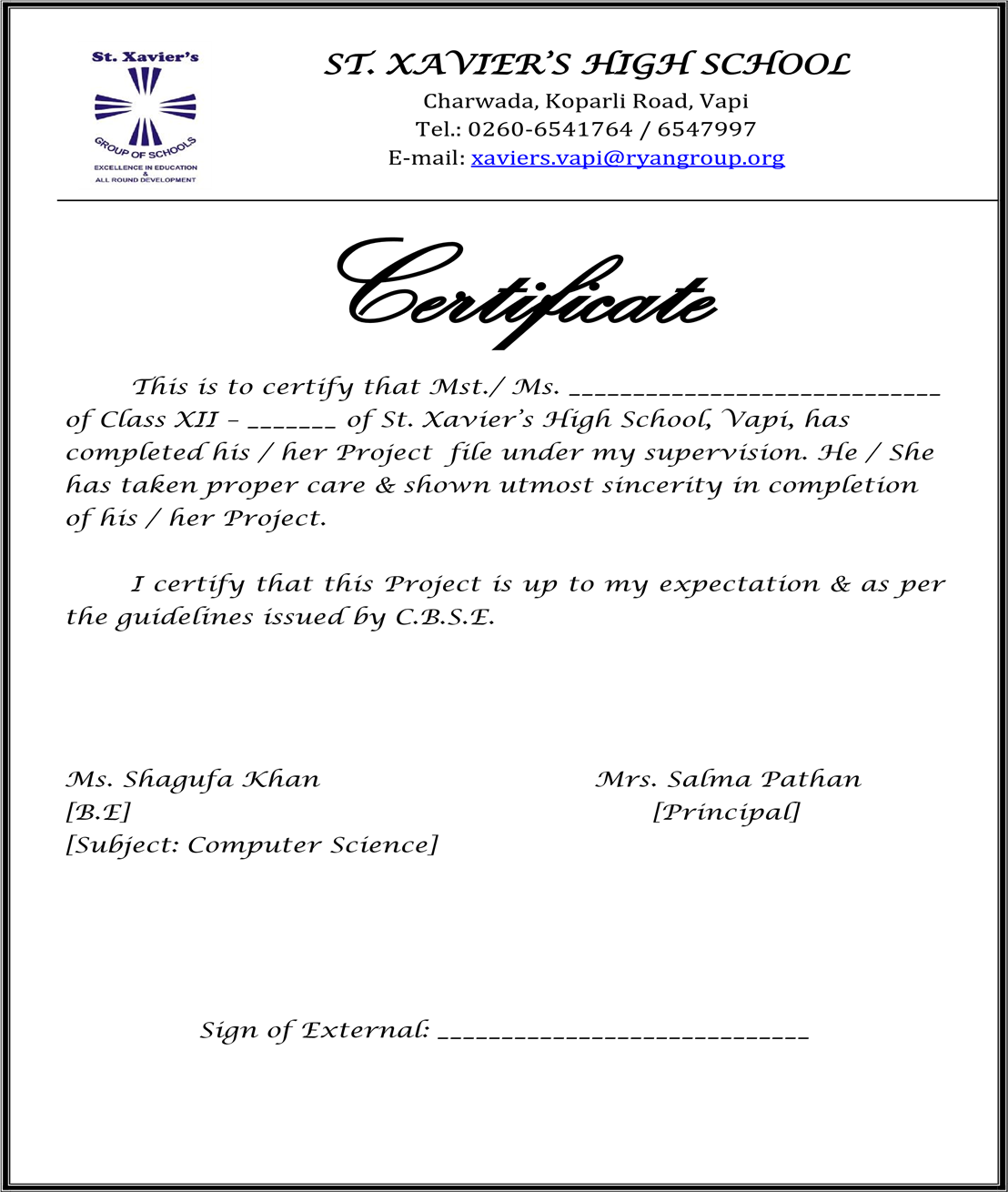
**KARA VIRTUAL ASSISTANT**



****

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr no** | **Index** | **Page no** |
| 1 | Declaration | 4 |
| 2 | Acknowledgement | 5 |
| 3 | Introduction | 6 |
| 4 | Working description | 7-8 |
| 5 | Source code | 9-23 |
| 6 | Output screen | 24-25 |
| 7 | Bibliography | 25 |

DECLARATION

I hereby declare that project work entitled,

**“KARA VIRTUAL ASSISTANT”**

Submitted to department of Computer Science, St. Xavier’s High School, Vapi is prepared by Priyanshu Yadav, Aryan Parmar, Aayan Mundhekar and Atyushraj Yadav. All the coding are result of our personal efforts.

|  |  |
| --- | --- |
| Name | Roll.no. |
| Priyanshu Yadav XII-B |  |
| Aryan Parmar XII-A |  |
| Aayan Mundhekar XII-B |  |
| Atyushraj Yadav XII-B |  |

**ACKNOWLEDGEMENT**

I PRIYANSHU YADAV OF CLASS XII-B WOULD LIKE TO EXPRESS MY SINCERE GRATITUDE TO MY COMPUTER SCIENCE TEACHER MS. SHAGUFA KHAN, PGT COMPUTER SCIENCE AND PRINCIPAL MRS. SALMA PATHAN , FOR HER VITAL SUPPORT, GUIDANCE AND ENCOURAGEMENT—WITHOUT WHICH THIS PROJECT WOULD NOT HAVE COME FORTH.

I WOULD ALSO LIKE TO EXPRESS MY GRATITUDE TO MY SCHOOL ST.XAVIER’S HIGH SCHOOL FOR LETTING ME USE THE SCHOOL’S LABORATORY .

# INTRODUCTION

Virtual Assistants are known for being great at many things and can make a

positive contribution to the success of your SME. They can also assist with your organization skills and productivity by freeing up your time for you to focus on other important activities.

**Working description**

Voice assistants are programs on digital devices that listen and respond to verbal commands. Kara does the same, it an apt voice assistant, it can do tasks like presenting myriad google based information, basic calculations, general information on the computer and can even chat to rid our boredom.

To use Kara one needs to log in first with their name and password. Kara has a very friendly user interface . The programming language used for this project is python and MySQL. The whole program is on python whereas the database is stored on MySQL server database. Kara is the best voice assistant in its way.

**What can Kara assistant do for YOU?**

1) you can do ask simple mathematical questions

2)You can see your command history :**"What Was My Recent Task”, “Recent**"

3)you can delete your command History: "Delete”, “Delete History"

4)You can see live covid-19 cases in India **:"Covid”, Corona"**

5)you can ask for Wikipedia searches : **"Just Say The Name Of The Topic" For Eg : ‘Who Is SRK’**

6)you can ask for the status of battery percentage : “**What Is Battery Percentage**”

7)you can ask for present date/time : **"What Date It Is"/ “What Time ”**

8)you can take screenshot : **“Take Screenshot”**

9)you can take selfie of yours : **“Take Picture” Or “Take Pic” Or “Take Selfie”**

10) you can see your YouTube subscriptions :"**My Subscription**", Watch Later List :"**Watch Later**"

11)you can ask her for opening any application install in your device : **Open** ‘**Name Of Application’**

**Some funny stuff you can do with it**

**by asking :**

* + who made you?
  + where do you live?
  + what is your name ?
  + what is my name?
  + what can you do?
  + do you love me ?
  + can I call you "name you like”?
  + how are you?

**Source Code in Python**

Logic.py:

# import useful module

import os

import datetime

import webbrowser

import random

import time

from tkinter import \*

from plyer import battery

import pendulum

import pyautogui

import cv2

import pyjokes

import mysql.connector as mysql

import wikipedia

import pyttsx3 as p

import speech\_recognition as sr

from PIL import Image, ImageTk, ImageSequence

import Calculation

import historyPdf

import covidNotifier

engine = p.init('sapi5')  # initiating speak engine

voices = engine.getProperty('voices')

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

engine.setProperty('rate', 150)

engine.setProperty('volume', 2.0)

new = 2

app = ("zoom", "google", "google chrome", "spotify", "powerpoint", "power point", "paint",

        "mysql", "my sql")

i = 1

# defining part starts from here

dt = pendulum.now('Asia/Calcutta')

a = random.randint(1, 2)

admin = 'Kalios'

db = mysql.connect(

    host="localhost",

    user="root",

    passwd="1234"

)

data = db.cursor()

try:

    data.execute("USE karadb")

except:

    data.execute("CREATE DATABASE karadb")

    db.commit()

def makeDatabase1():

    pas = pass\_ent.get()

    nam = nam\_ent.get()

    data.execute("CREATE TABLE karadb(id INT(5), name VARCHAR(20), password VARCHAR(20), recent VARCHAR(50))")

    data.execute("INSERT INTO karadb(id, name, password) VALUES(1, %s, %s)", (nam, pas))

    db.commit()

    userdata.destroy()

try:

    data.execute("USE karadb")

    data.execute("SELECT id FROM karadb")

    no\_use\_data = data.fetchall()

except:

    data.execute("USE karadb")

    userdata = Tk()

    userdata.geometry('500x270')

    userdata.wm\_iconbitmap('resources\\winlogo.ico')

    userdata.title("Hello")

    userdata.configure(background="black")

    bgImg = PhotoImage(file="resources\\signup.png", master=userdata)

    frame = Label(userdata,image=bgImg)

    frame.image = bgImg

    frame.place(x=0, y=0, relwidth = 1, relheight=1)

    nam\_ent = Entry(userdata, width=30)

    nam\_ent.place(relx=.7, rely=.55, anchor="center")

    pass\_ent = Entry(userdata, width=30)

    pass\_ent.place(relx=.7, rely=.68, anchor="center")

    sign\_img = PhotoImage(file="resources\sign\_button.png", master=userdata)

    but1 = Button(userdata,image=sign\_img,bg="#121212", activebackground="#FF5733", bd=0,command=makeDatabase1)

    but1.image = sign\_img

    but1.place(relx=.5, rely=.9, anchor="center")

    userdata.mainloop()

try:

    data.execute("USE karadb")

    data.execute("SELECT user FROM recent\_data")

    no\_use\_data = data.fetchall()

except:

    data.execute("USE karadb")

    data.execute("CREATE TABLE recent\_data(time VARCHAR(50),user VARCHAR(20), query VARCHAR(50), date VARCHAR(50))")

    db.commit()

def convertuple(tup):

    stri = ''.join(tup)

    return stri

def username():

    data.execute("USE karadb")

    data.execute("SELECT name FROM karadb")

    databases = data.fetchall()

    us = databases[0]

    usern = convertuple(us)

    return usern

user = username()

def change\_name(nm):

    data.execute("USE karadb")

    query = "UPDATE karadb SET name =\'" + nm + "\'WHERE id = '1'"

    data.execute(query)

    db.commit()

def recent(strre):

    data.execute("USE karadb")

    query = "UPDATE karadb SET recent =\'" + strre + "\'WHERE id = '1'"

    data.execute(query)

    db.commit()

def getrecent():

    data.execute("USE karadb")

    data.execute("SELECT recent FROM karadb")

    databases = data.fetchall()

    data = databases[0]

    rec = dataa

    return rec

def getpasswrd():

    data.execute("USE karadb")

    data.execute("SELECT password FROM karadb")

    databases = data.fetchall()

    ps = databases[0]

    passn = ps

    return passn

def history(user, recent):

    data.execute("USE karadb")

    data.execute(

        "INSERT INTO recent\_data VALUES ('" + dt.format("LT") + "','" + user + "','" + recent + "','" + dt.format(

            "LL") + "')")

    db.commit()

def get\_history():

    data.execute("use karadb")

    data.execute("SELECT\* from recent\_data")

    datarec = data.fetchall()

    return datarec

def del\_history():

    data.execute("use karadb")

    data.execute("DELETE FROM recent\_data")

    db.commit()

def speak(audio):  # speaks the output

    engine.say(audio)

    engine.runAndWait()

def battery\_level():

    c = dict(battery.get\_state())

    m = c["isCharging"]

    y = c["percentage"]

    if m != 'charging':

        if 1 < y < 20:

            speak("battery is low connect charger")

        else:

            speak("battery is sufficient")

def wish():  # just wishes u

    hour = datetime.datetime.now().hour

    if 0 < hour < 12:

        speak(f"good morning {user}")

        speak("how can i help you")

    elif 12 <= hour <= 18:

        speak(f"good afternoon {user}")

        speak("how can i help you")

    else:

        speak(f"good evening {user}")

        speak("how can i help you")

def bat\_stat():

    if a == 1:

        y = f"{x} percentage of battery is left"

    else:

        y = f"percentage of battery is {x}"

    return y

def pic\_stat():

    if a == 1:

        w = "say cheese"

    else:

        w = "smile please"

    return w

def take\_pic(image\_counter=0):

    face\_cascade = cv2.CascadeClassifier(

        r'resources\haarcascade\_frontalface\_alt.xml')

    video = cv2.VideoCapture(0)

    video.set(3, 852)

    video.set(4, 480)

    while True:

        check, frame = video.read()

        gray\_f = cv2.flip(frame, 1)

        gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

        faces = face\_cascade.detectMultiScale(gray, 1.3, 5)

        for (x, y, w, h) in faces:

            cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 255, 255), 2)

        gray\_flip = cv2.flip(frame, 1)

        cv2.imshow("kara", gray\_flip)

        key = cv2.waitKey(1)

        if key == ord('q'):

            break

        elif key == ord('s'):

            # SPACE pressed

            img\_name = "kara\_capture{}.png".format(image\_counter)

            cv2.imwrite(img\_name, gray\_f)

            print("{} captured!".format(img\_name))

            image\_counter += 1

    video.release()

    cv2.destroyAllWindows()

# welcoming the user

def take\_screenshot(ss=1):

    time.sleep(5)

    pyautogui.screenshot().save(f"screenshot{ss}.jpg")

# welcoming the user

def changed\_name():

    a = name.get()

    print(a)

    change\_name(a)

    speak("done")

    username()

    print("username will change when you restart the program")

def output(q):

    global x

    global name

    if 'open' in q:

        if q[5:] in app:

if q[5:] == 'zoom':

                speak(f"opening {q[5:]}")

                try:

                    os.startfile(r"C:\Users\Admin\AppData\Roaming\Zoom\bin\Zoom.exe")

                except:

                    query\_string = q[5:]

                    webbrowser.open("http://www." + query\_string + ".com", new=new)

            elif q[5:] == 'command prompt' or q[5:] == 'cmd':

                speak(f"opening {q[5:]}")

                os.startfile(r"C:\Windows\system32\cmd.exe")

            elif q[5:] == 'mysql' or q[5:] == 'my sql':

                speak(f"opening {q[5:]}")

                os.startfile(r"C:\Program Files\MySQL\MySQL Server 5.5\bin\mysql.exe")

            elif q[5:] == 'spotify':

                speak(f"opening {q[5:]}")

                try:

                    os.startfile(r"C:\Users\Admin\AppData\Roaming\Spotify\Spotify.exe")

                except:

                    query\_string = q[5:]

                    webbrowser.open("http://www." + query\_string + ".com", new=new)

            elif q[5:] == 'illustrator' or q[5:] == 'adobe illustrator':

                speak(f"opening {q[5:]}")

                try:

                    os.startfile(

                        r"C:\Program Files\Adobe\Adobe Illustrator CS6 (64 Bit)\Support Files\Contents\Windows\

                        Illustrator.exe")

                except:

                    query\_string = q[5:]

                    webbrowser.open("http://www." + query\_string + ".com", new=new)

        else:

            speak(f"opening {q[5:]}")

            query\_string = q[5:]

            webbrowser.open("http://www." + query\_string + ".com", new=new)

    elif "who made you" in q or "who is your maker" in q:

        speak(f'i am made by a wonderfull person named {admin}')

    elif "what was my recent task" in q or 'recent' in q:

        speak(f"your recent query was {convertuple(recdata)}")

        print(f"your recent query: \"{convertuple(recdata)}\"")

    elif "where do you live" in q:

        speak("help!, i am stuck inside a device")

    elif "you" in q and "look" in q:

        print("(^\_^)")

    elif "delete" in q or "delete history" in q:

        del\_history()

    elif "history" in q:

        for y in get\_history():

            print(y)

        historyPdf.createPdf(get\_history())

        os.startfile(r"resources\history.pdf")

    elif "my subscriptions" in q or "my subscription" in q:

        speak("showing active subscriptions on youtube")

        webbrowser.open("www.youtube.com/subscription\_manager", new=new)

    elif "watchlater" in q or "watch later" in q:

        speak("taking you to wach later list")

        webbrowser.open("https://www.youtube.com/playlist?list=WL", new=new)

    elif "covid" in q or "corona" in q:

        covidNotifier.getData()

    elif "what is your name" in q or "whats your name" in q or "your name" in q:

        speak("i think i should introduce myself")

        speak("hey there i am kara, your virtual assistant")

    elif "take picture" in q or "photo" in q or "picture" in q or "pic" in q or "take selfie" in q:

        print("click s for capturing image and q for closing camera")

        speak(pic\_stat())

        take\_pic()

    elif q == "thank you kara" or q == "thanks" or q == "thank you" or q == "thanks kara":

        speak("always there to help you")

        speak("come back soon")

    elif "what is my name" in q:

        speak(f"your name is {user}")

    elif "change my name" in q or "change my username" in q or 'change' in q or "username" in q:

        speak("tell your new name")

        new\_name = Tk()

        new\_name.geometry('400x300')

        new\_name.wm\_iconbitmap('resources\\winlogo.ico')

        new\_name.configure(background="black")

        new\_name.title("Change Name")

        name = StringVar()

        name\_lab = Label(new\_name,text="New name :-", fg="white", bg="black")

        name\_lab.place(relx=.4, rely=.3, anchor="center")

        new\_name\_entry = Entry(new\_name, textvariable=name).place(relx=.7, rely=.3, anchor="center")

        but2 = Button(new\_name, text="submit", width=20, bg="#15E546", command=lambda: changed\_name())

        but2.place(relx=.5, rely=.7, anchor="center")

    elif "take" in q or "screenshot" in q:

        speak("taking screenshot in 5 seconds")

        take\_screenshot()

        speak("done")

    elif "what can i call" in q:

        speak("im kara your virtual assistant")

    elif "joke" in q:

        speak(pyjokes.get\_joke('en', "all"))

    elif 'what time' in q or 'the time' in q or 'what\'s the time' in q:

        speak(dt.format("LT"))

    elif "can" in q and "i" in q and "call you" in q:

        speak("you can but i love my original name given by my maker")

    elif (("i" in q and "am" in q) or "im" in q) and ("bored" in q or "bore" in q):

        speak("you can ask me to play some songs or i can tell you a joke")

    elif "what can you do" in q:

        speak("i can do anything you want")

    elif "what is todays date" in q or "what is todays date" in q or "what date it is" in q:

        speak(dt.format('Do MMMM '))

    elif "what day it is today" in q or ("day" in q and "today" in q):

        speak(dt.format('dddd'))

    elif "do you love me" in q or "love" in q:

        speak('i love you but as a friend')

    elif "what is battery percentage" in q or "battery" in q or "left" in q or "battery percenntage" in q:

        a = dict(battery.get\_state())

        for i in a:

            x = (a[i])

        speak(bat\_stat())

        battery\_level()

    elif "what is" in q:

        try:

            speak(wikipedia.summary(q[8:], sentences=2))

        except:

            query\_string = '''https://www.google.com/search?rlz=1C1CHBF\_enIN861IN861&sxsrf

                                   ALeKk000cOJ790\_t5d8jkFcT0U0f0dgvow%3A1592048167474&ei=J7rkXv6-HOPE4-EPgNCM-Aw&q='''

            webbrowser.open(query\_string + q, new=new)

    elif "who is" in q:

        try:

            speak(wikipedia.summary(q[6:], sentences=2))

        except:

            query\_string = '''https://www.google.com/search?rlz=1C1CHBF\_enIN861IN861&sxsrf

                                   ALeKk000cOJ790\_t5d8jkFcT0U0f0dgvow%3A1592048167474&ei=J7rkXv6-HOPE4-EPgNCM-Aw&q='''

            webbrowser.open(query\_string + q, new=new)

    elif "say" in q:

        speak(q[4:])

    elif "repeat" in q:

        speak(q[6:])

    elif "hey" in q or "hi" in q or "hello" in q:

        wish()

    elif "how are you" in q:

        speak("i am doing well")

    elif q == "stop" or q == "stop kara" or q == "bye" or q == "bye kara":  # stop the code

        speak("bye boss its my pleasure to help you")

        system\_running = False

    elif "divide" in q or "multiply" in q or "plus" in q or "minus" in q or "-" in q or "+" in q or "/" in q or "power" in q or "\*" in q or "into" in q or "divided" in q:

        try:

            a = Calculation.calc(q)

            print(a)

            speak(a)

        except:

            print("hmm, try again")

    else:

        try:

            speak(eval(q))

        except:

            query\_string = '''https://www.google.com/search?rlz=1C1CHBF\_enIN861IN861&sxsrf

                               ALeKk000cOJ790\_t5d8jkFcT0U0f0dgvow%3A1592048167474&ei=J7rkXv6-HOPE4-EPgNCM-Aw&q='''

            webbrowser.open(query\_string + q, new=new)

recdata = getrecent()

r = sr.Recognizer()

mic\_list = sr.Microphone.list\_microphone\_names()

def input():

    required = 0

    for index, name in enumerate(sr.Microphone.list\_microphone\_names()):

        if "pulse" in name:

            required = index

    r = sr.Recognizer()

    with sr.Microphone(device\_index=required) as source:

        r.adjust\_for\_ambient\_noise(source)

        r.dynamic\_energy\_threshold = 4000

        print("Say something!")

        audio = r.listen(source, phrase\_time\_limit=5)

    try:

        q = r.recognize\_google(audio, show\_all=False, language="en-IN")

        print(q)

        output(q.lower())

        try:

            if q != "history" or q != "delete":

                history(username(), q)

                recent(q)

            else:

                pass

        except:

            pass

    except sr.UnknownValueError:

        pass

    except sr.RequestError as e:

        print("seem's like you are offline")

**Calculation:**

def tokened(inp):

    inp = inp.lower()

    inpToken = inp.split()

    return inpToken

def operatorChange(inp):

    index = 0

    toke = tokened(inp)

    for i in toke:

        if i == "plus":

            toke[index] = '+'

            index += 1

        elif i == "power":

            toke[index] = '\*\*'

            index += 1

        elif i == "multiply":

            toke[index] = '\*'

            index += 1

        elif i == "divide":

            toke[index] = '/'

            index += 1

        elif i == "divided":

            toke[index] = '/'

            index += 1

        elif i == "minus":

            toke[index] = '-'

            index += 1

        elif i == "into":

            toke[index] = '\*'

            index += 1

        else:

            try:

                x = int(i)

                index += 1

                pass

            except:

                toke[index] = ''

                index += 1

    return toke

def detoken(inp):

    str1 = " "

    s = operatorChange(inp)

    return eval(str1.join(s))

def calc(inp):

    try:

        out = eval(inp)

        return out

    except:

        return detoken(inp)

**Main.py:**

from logic import \*

class SplashScreen:

    def \_\_init\_\_(self, parent):

        self.image = Image.open(r'resources\\logo.png')

        self.parent = parent

        self.imgSplash = ImageTk.PhotoImage(self.image)

        self.splashWindow()

    def splashWindow(self):

        imagew, imageh = self.image.size

        setscreenw = (self.parent.winfo\_screenwidth() - imagew) // 2

        setscreenh = (self.parent.winfo\_screenheight() - imageh) // 2

        self.parent.geometry("%ix%i+%i+%i" % (imagew, imageh, setscreenw, setscreenh))

        self.logo = Label(self.parent, image=self.imgSplash)

        self.logo.pack()

        self.parent.after(4050, lambda: self.parent.destroy())

class pass\_check:

    def \_\_init\_\_(self):

        self.pas\_check = Tk()

        self.pas\_check.geometry('500x270')

        self.pas\_check.wm\_iconbitmap('resources\\winlogo.ico')

        self.pas\_check.title("Hello")

        self.pas\_check.configure(background="black")

        bgImg = PhotoImage(file="resources\\signup.png", master=self.pas\_check)

        frame = Label(self.pas\_check,image=bgImg)

        frame.image = bgImg

        frame.place(x=0, y=0, relwidth = 1, relheight=1)

        self.nam\_ent = Entry(self.pas\_check, width=30)

        self.nam\_ent.place(relx=.7, rely=.55, anchor="center")

        self.pass\_ent = Entry(self.pas\_check, width=30, show="\*")

        self.pass\_ent.place(relx=.7, rely=.68, anchor="center")

        sign\_img = PhotoImage(file="resources\\sign\_button.png", master=self.pas\_check)

        but1 = Button(self.pas\_check,image=sign\_img,bg="#121212", activebackground="#FF5733", bd=0,command=self.check)

        but1.image = sign\_img

        but1.place(relx=.5, rely=.9, anchor="center")

        self.pas\_check.mainloop()

    def check(self):

        a = self.nam\_ent.get()

        b = self.pass\_ent.get()

        d = username()

        c = getpasswrd()

        print(d, c[0])

        if a == d and b == c[0]:

            mainroot()

            self.pas\_check.destroy()

        else:

            pass\_ent = Label(self.pas\_check, text="Invalid Username or Password, Try again",bg="#121212", fg="red")

            pass\_ent.place(relx=.5, rely=.78, anchor="center")

class mainroot:

    def \_\_init\_\_(self):

        self.sroot = Tk()

        self.sroot.geometry('1280x800')

        self.sroot.wm\_iconbitmap(r'resources/winlogo.ico')

        self.sroot.minsize(1280, 760)

        self.sroot.maxsize(1280, 760)

        self.sroot.configure(background="black")

        self.sroot.title("KARA")

        bgImg = PhotoImage(file="resources\\bg.png", master=self.sroot)

        frame = Label(self.sroot,image=bgImg)

        frame.image = bgImg

        frame.place(x=0, y=0, relwidth = 1, relheight=1)

        mic1 = PhotoImage(file="resources\\micr.png", master=self.sroot)

        self.mic = Button(self.sroot, image=mic1, command=self.startMic, bg="white", activebackground="#FF5733",bd=0)

        self.mic.place(relx=.5, rely=.6, anchor="center")

        self.mic.image = mic1

        his\_img = PhotoImage(file="resources\\history.png", master=self.sroot)

        self.his = Button(self.sroot, image=his\_img, command=lambda: output("history"),bg="white", activebackground="#FF5733", bd=0)

        self.his.image = his\_img

        self.his.place(relx=.9, rely=.05, anchor="center")

        his\_del\_img = PhotoImage(file="resources\\del.png", master=self.sroot)

        self.his\_del = Button(self.sroot, image=his\_del\_img, command=lambda: del\_history(),bg="white", activebackground="#FF5733", bd=0)

        self.his\_del.place(relx=.97, rely=.05, anchor="center")

        self.his\_del.image = his\_del\_img

        self.covid\_button = Button(self.sroot, text="get info about corona cases in india, and take precautions", bg="black", fg="white",activebackground="#FF5733", borderwidth=1, highlightthickness=0, padx=9, height=1,font=("arial", 10),command=lambda : webbrowser.open("https://www.covid19india.org/"))

        self.covid\_button.place(relx=.5, rely=0.95, anchor="center")

        name = username()

        self.name\_lab = Label(self.sroot,text=name.title(), font=("open serif", 60, "bold"), bg="white", bd=0)

        self.name\_lab.place(relx=.6, rely=0.29, anchor="center")

        hide\_lab = Label(self.sroot, bg="white", height=10, width=50)

        hide\_lab.place(relx=.3, rely=0.75, anchor="center")

    def startMic(self):

        input()

if \_\_name\_\_ == '\_\_main\_\_':

    root = Tk()

    root.overrideredirect(True)

    app = SplashScreen(root)

    try:

        f = open("resources\\firstTimeusr.dat", "rt")

        root.after(4010, pass\_check)

    except:

        f = open("resources\\firstTimeusr.dat", "w+")

        root.after(4010, mainroot)

    root.mainloop()

**HISTORYPDF.PY:**

from fpdf import FPDF

pdf = FPDF()

pdf.add\_page()

pdf.set\_font("Arial", size=15)

def convertTuple(tup):

    str1 = '  |  '.join(tup)

    return str1

def createPdf(inp):

    lineNo = 4

    pdf.cell(200, 10, txt='HISTORY',

             ln=1, align='C')

    pdf.cell(200, 10, txt='Time  |  user  |  query  |  date',

             ln=3, align='C')

    for i in inp:

        a = convertTuple(i)

        pdf.cell(200, 10, txt=a,

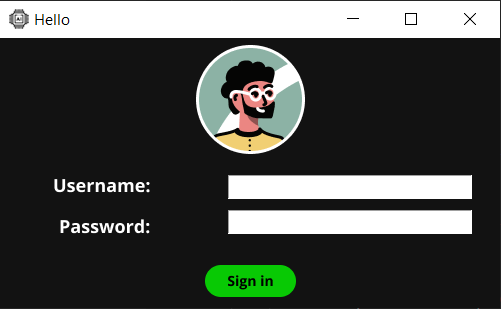
                 ln=lineNo, align='C')

        lineNo += 1

    pdf.output("resources\\history.pdf")

OUTPUT

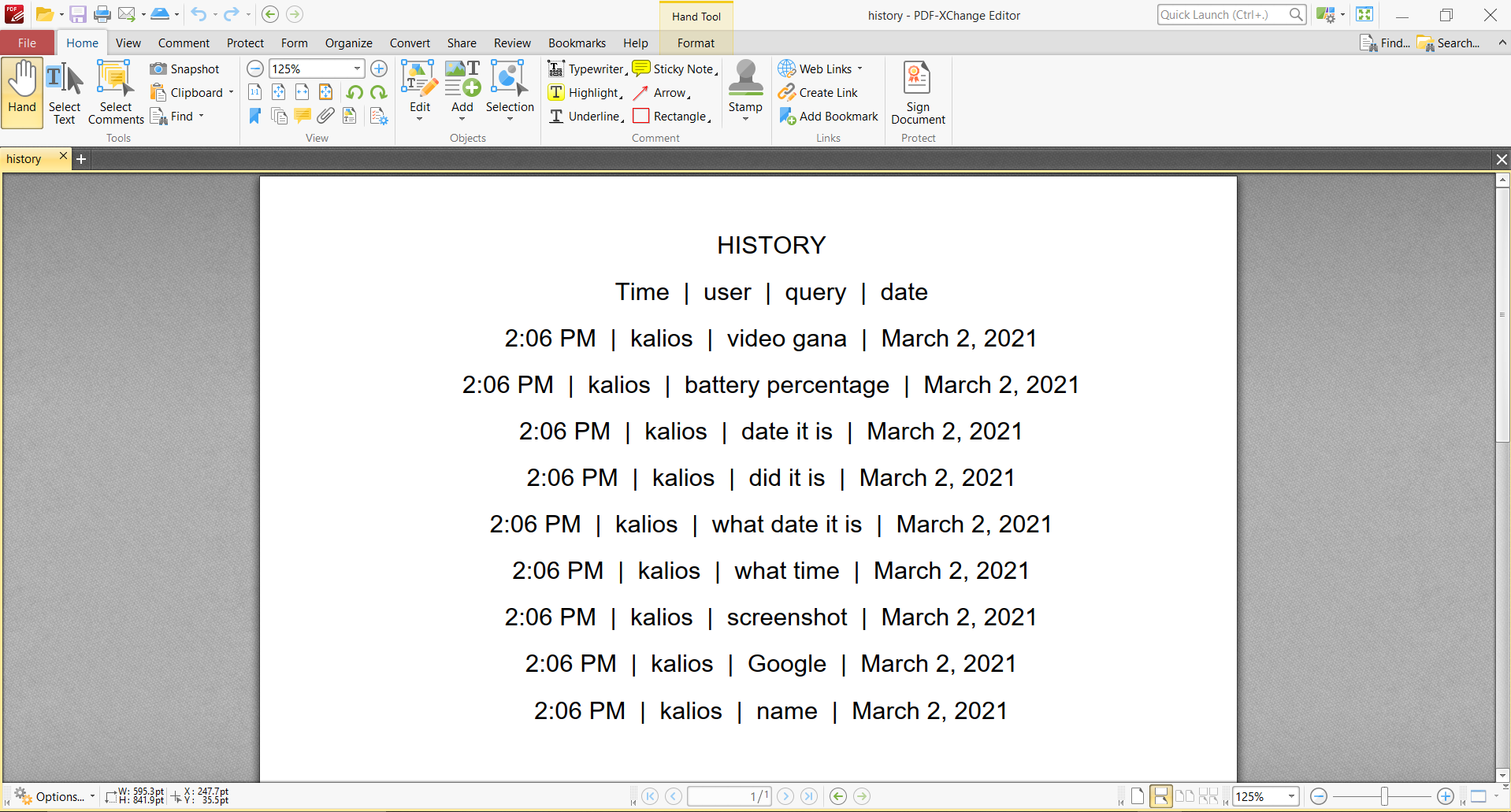
Login screen:



Main screen:



History Button :



BIBLIOGRAPHY:

* https://www.figma.com/
* https://www.w3schools.com/
* <https://www.geeksforgeeks.org/>
* <https://stackoverflow.com/>