**PROJECT REPORT**

**On**

**Secure File Transfer using Sockets**

**Submitted by**

Harshpreet Singh Sandhu (Enroll No. R134218065)

Neelesh Kumar (Enroll. No. R134218102)

Naman Sharma(Enroll. No. R134218099)

Aarohi Mangal(Enroll. No. R134218001)

**Under the guidance of**

Mr. Keshav Kaushik

Department of Systemics



**SCHOOL OF COMPUTER SCIENCE**

UNIVERSITY OF PETROLEUM & ENERGY STUDIES

Bidholi Campus, Energy Acres, Dehradun – 248007.

**Jan May-2021**

****

**CANDIDATE’S DECLARATION**

We hereby certify that the project work entitled **“Secure File Transfer using Sockets”**in partial fulfillment of the requirements for the award ofthe Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in **CYBER SECURITY AND FORENSICS** and submitted to the School of Computer Science, Department of Systemics, University of Petroleum & Energy Studies, Dehradun, is an authentic record of my/ our work carried out during a period from **January**-**2021** to **May-2021** under the supervision of **Mr. Keshav Kaushik , Dept. of Systemics.**

The matter presented in this project has not been submitted by us for the award of any other degree of this or any other University.

**Aarohi Mangal** R134218001

|  |  |
| --- | --- |
| This is to certify that the above statement made by the | candidate is correct to the best of |
| my knowledge. |  |
| Date: \_\_\_\_\_\_\_\_\_\_\_\_\_2020 | **Mr. Keshav Kaushik** |
|  | Project Guide |
| **Dr. Neelu Jyoti Ahuja** |  |
| HOD – Systemics |  |
| School of Computer Science |  |
| University of Petroleum & Energy Studies |  |
| Dehradun – 248 001 (Uttarakhand) |  |
|  |  |

**Harshpreet Singh Sandhu** R134218065

**Naman Sharma** R134218099

**Neelesh Kumar Gupta**

R134218102

**ACKNOWLEDGEMENT**

We wish to express our deep gratitude to our guide **Mr. Keshav Kaushik**, for all advice, encouragement and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thank to our respected Program Head of the Department, **Dr. Manoj Kumar**, for her great support in doing our project in **Dept. of Systemics** at **SoCS**.

We are also grateful to **Dr. Priyadarshan Patra,** Dean- SoCS, UPES for giving us the necessary facilities to carry out our project work successfully.

We would like to thank all our **friends** for their help and constructive criticism during our project work. Finally we have no words to express our sincere gratitude to our **parents** who have shown us this world and for every support they have given us

**Name:** Aarohi Mangal Neelesh Kumar Harshpreet Sandhu Naman Sharma

**Roll No:** R134218001 R134218102 R134218065 R134218099

 **School of Computer Science**

**University of Petroleum & Energy Studies, Dehradun**

**PROJECT TITLE:**  Secure File Transfer using Sockets

**ABSTRACT**

Security plays a most important role in day-to-day life. Security is the only thing with which nobody wants any compromise. That’s why it costs high. And when it comes to data it becomes more crucial. This project is on a GUI-based file transfer program by the use of sockets. As sockets provide security to the data by encrypting the link of communication between a web server and browser so that data will be free from man-in-the-middle attacks. In addition to this, an authentication mechanism is also implemented to substantially reduce the likelihood of data being compromised.

**KEYWORDS:** GUI, sockets, server, authentication, encryption.

**INTRODUCTION**

**Confidentiality of the files transferred over the network**: Confidentiality is the protection of personal information. Confidentiality means keeping a client's information between you and the client. To perform confidentiality, we are using sockets. A socket serves as an endpoint of a two-way communication link between two programs for sending and receiving data across the network. It provides security to the data by encrypting the link of communication between a web server and browser. In this way, the privacy of the users will not be hampered.

**Authentication**: Users will only be allowed to download the files after proper authentication and authorization. Additionally, 2 factor authentication is implemented to make the transfer of files more secure. A particular way by which somebody might be authenticated fall into three classifications, in view of what are known as the components of authentication: something the client knows, something the client has, and something the client is. Every authentication factor covers a scope of components used to confirm or check an individual's personality preceding being allowed access, affirming an exchange demand, marking a report or other work item, giving position to other people, and building up a chain of power.

**Authorisation**: Only authorised users will be allowed to access the files to preserve the integrity of the files. The cycle of authorization is particular from that of authentication. While authentication is the way toward confirming that "you are who you say you are", and checking that "you are allowed to do what you are attempting to do". While authorization regularly happens following authentication (e.g., when signing into a PC framework), this doesn't mean authorization surmises authentication: a mysterious specialist could be approved to a restricted activity set.

**Malicious file analysis**: Report of the files can be generated before downloading to ensure that the file received is not affected by any malware. Malware analysis is the way toward understanding the conduct and motivation behind a suspicious file or URL. The yield of the analysis helps in the location and prevention of the potential threat. The key advantage of malware analysis is that it helps episode responders and security examiners:

Practically emergency occurrences by level of seriousness

Improve the adequacy of security

Improve setting when danger chasing

**Log management**: In the cyber world, a log is a file that is created naturally every time certain occasions happen in your framework. Log files are typically time-stepped, and may record for all intents and purposes whatever is occurring in the background in working frameworks or programming applications - basically, they record all that the worker do, organization, OS, or application believes is essential to monitor. Logs can record all sort of occasions going from messages and exchanges happening between various clients, what occurred during a reinforcement, the blunders that prevented an application from running, or the files that have been mentioned by clients from a site. All the activity history of the user can be tracked for the detection of any suspicious or malicious behaviour.

**PROBLEM STATEMENT**

Creating a GUI-based Secure File Transfer System using Sockets in order to protect users’ privacy & generating malware analysis report of the file.

**LITERATURE REVIEW**

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **Link** | **Author** | **Remarks** |
| File transfer using TCP/IP | https://cutt.ly/bxuS6S1 | Sang Oh Spencer Kam Atsuya Takagi | It explains various protocols. ports and functionality of the TCP/IP model |
| Authentication | https://auth0.com/docs/architecture-scenarios/implementation/b2b/b2b-authentication | Auth0.com | It demonstrates ample of examples and techniques to ensure authentication |
| Authorisation | https://swoopnow.com/security-authentication-vs-authorization/ | swoopnow.com | This post defines authorisation, it’s uses, difference between authentication and authorisation and how to boast it |
| Sockets | https://realpython.com/python-sockets/ | Realpython.com | It explains about pre-defined functions in python, ports used and their functionality |

**OBJECTIVES**

To create a Gui-based Secure File Transfer program using Sockets.

Sub Objectives

* Display Welcome window including options like login/signup, Send/Receive file, generate malware report and track activity.
* Forget password provision is also provided using real time OTP verification.
* After Authenticating and Authorizing user, give privilege to download files.
* Generate Malware Analysis report of the file
* Record all the activities of the users for feasible database collection.

**METHODOLOGY**

In this scenario, we have to transfer the file using sockets. There can be two cases either the user is registered or not.

If the user is not register then the user have to register by providing his details.

If the user is registered then he will directly login using username and password.

There will two sides: **client and server**. Either they can receive or send files using sockets.

**File Type:** User can send different type of file types, therefore user can upload files for that file types. The file types normally

Word File

PDF file

Text File

Images

Videos

**Tracking user activity:** User activity will be traced by administrator. For tracking the user details administrator must provide the username to track the user details.

**Users:** User is only an enlisted user. An enrolled user straightforwardly transfers data into the framework. Prior to transferring he needs to indicate his credentials for authentication.

File sending and receiving: All registered user have an authority to upload file into the system, admin can track all the uploaded files. For file sending and receiving the system provides a good user interface for easy to use. User can save the downloading file.

**Registration:** The framework has an interaction of registration. Each user need to present his total details as registration. At whatever point a user registration finished consequently user can get a user id and password. By utilizing that username and password user can sign into the framework.

**Reports:** Different kind of reports is generated by the system.

• Virus total report of the file.

• Sending and receiving status.

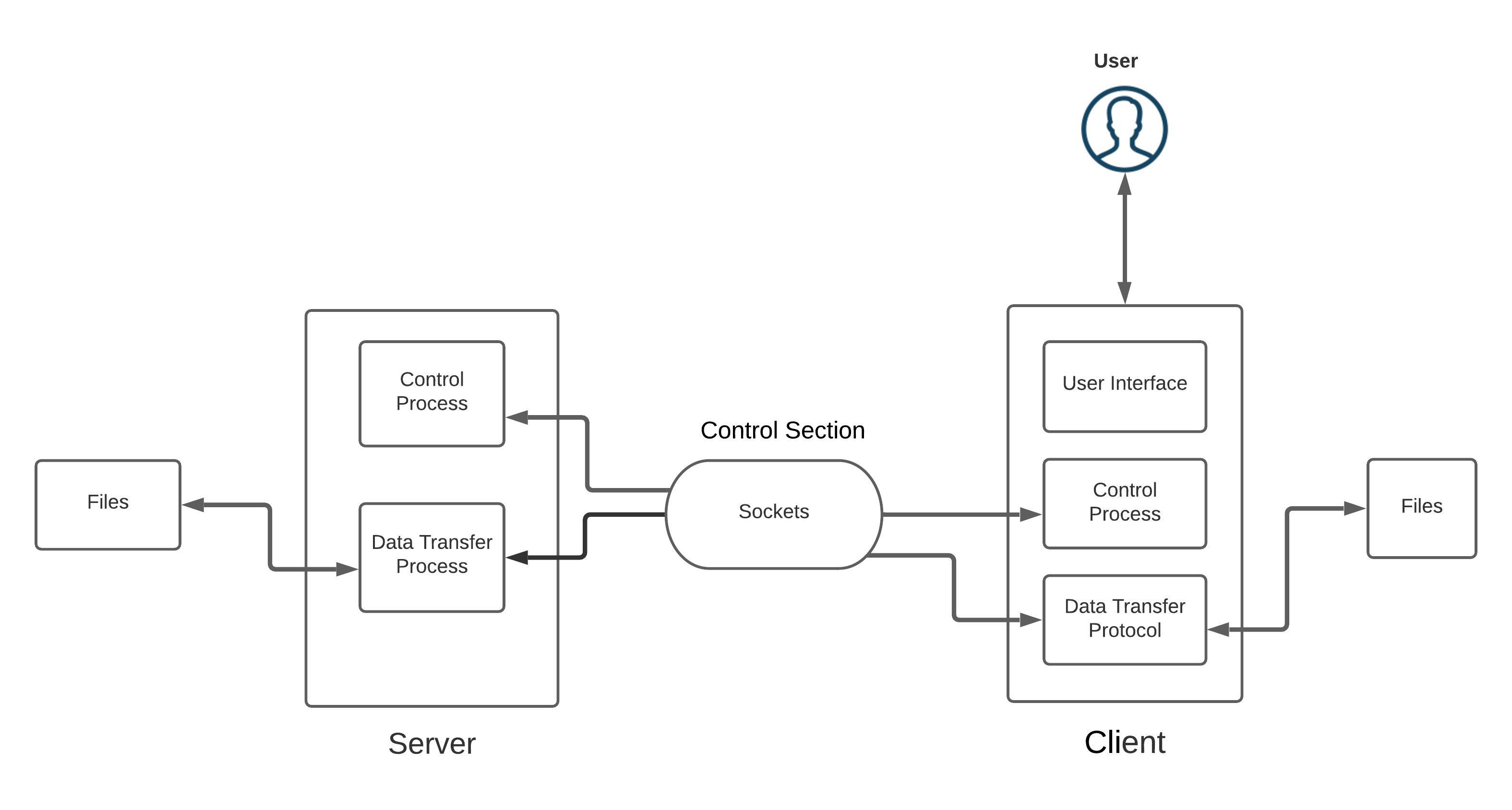
**Authentication:** Authentication is only providing security to the framework. Here each should go into the framework through login page. A user should provide his credentials like username and password for sign into the framework. For that the framework keeps up information for all users. At whatever point a user enters his username and password, it checks in the database for user presence. On the off chance that the user is exists he can be treated as a valid user. In any case the solicitation will toss back.

**Authorization:** Authorization is done when the user forgets the password and want to renew the password for that we are providing a service of OTP by which we send OTP on the registered no in a real time and take the OTP for authorization if OTP matches then he/she can renew its passwords.

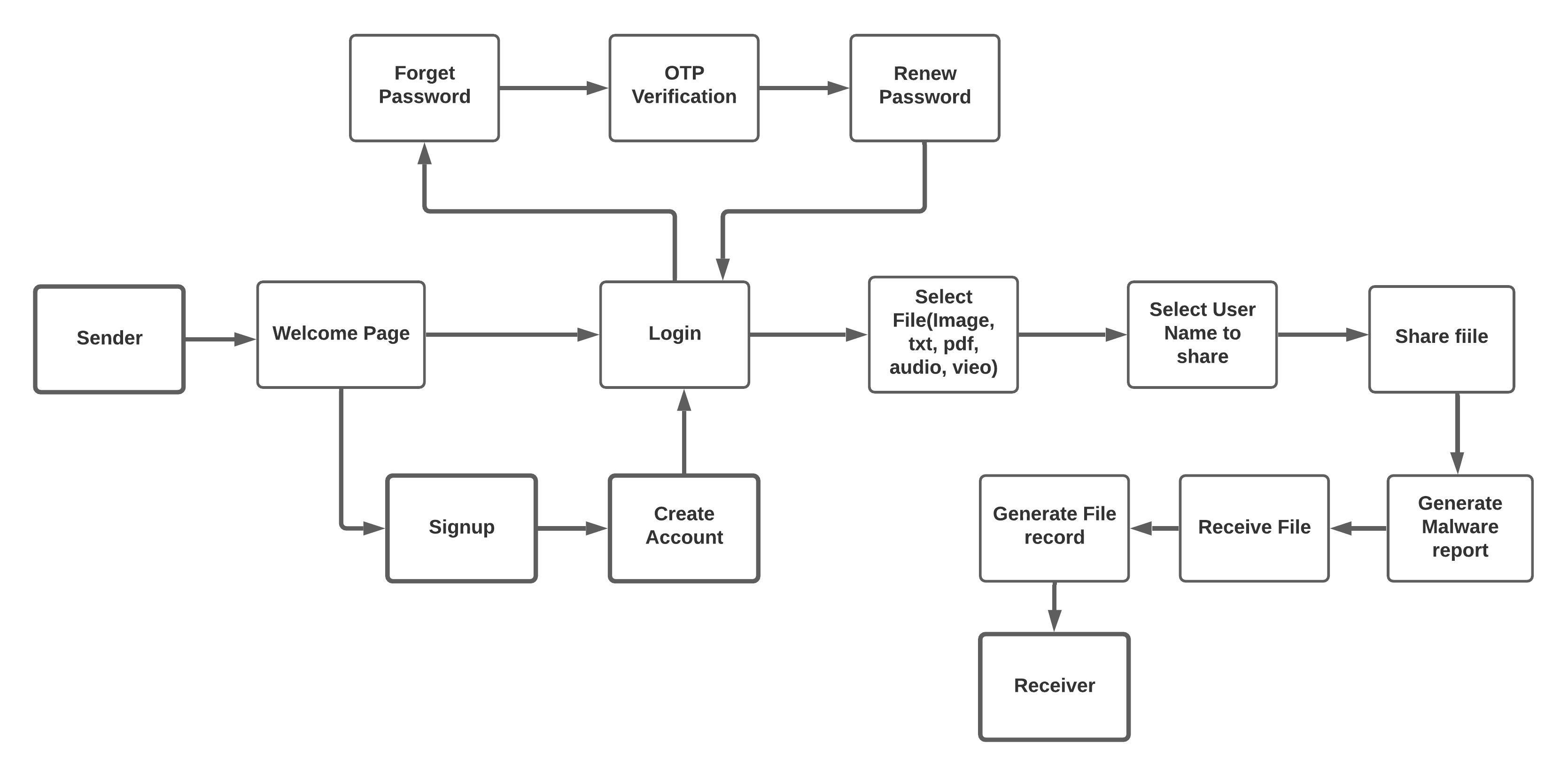
**User Name:** The user identification is what is needed by the worker for admittance to its document framework. This order will typically be the main order sent by the user after the control associations are made (a few workers may require this).

**Password:** This order should be quickly gone before by the user name order, and, for certain locales, finishes the user's identification for access control. Since password data is very touchy, it is attractive overall to "cover" it or stifle type out.

**Data flow Diagram**



**Workflow**



**SYSTEM REQUIREMENTS**

Hardware:

* RAM: 4GB
* Disk Space: 4GB

Software:

* Pycharm( or any other Python IDE)

Operating System:

* Windows.

**REFERENCES**

[1] <https://docs.python.org/3/library/socket.html>

# [2] <https://realpython.com/>

# [3] <https://www.geeksforgeeks.org/>

# [4]<https://www.tutorialspoint.com/python3/python_networking.htm#:~:text=Sockets%20are%20the%20endpoints%20of,%2C%20UDP%2C%20and%20so%20on>.

[5] <https://www.geeksforgeeks.org/python-program-to-generate-one-time-password-otp/>

[6]<https://www.twilio.com/login?g=%2Fconsole%3F&t=2b1c98334b25c1a785ef15b6556396290e3c704a9b57fc40687cbccd79c46a8c>

**Synopsis Draft verified by**

**Project Guide HOD**

Mr. Keshav Kaushik

**(Name & Sign) (Dept. of Systemics)**

Keshav Kaushik

Keshav

**Code –**

# Secure File Transfer using Sockets  
from tkinter import \*  
from tkinter import filedialog  
from tkinter import messagebox  
from textwrap import wrap  
import socket  
import tqdm  
import os  
import random as rnd  
from twilio.rest import Client  
from virustotal\_python import Virustotal  
from pprint import pprint  
import hashlib  
  
def Signup():  
  
 global top\_s  
 top\_s=Toplevel()  
 top\_s.title("Signup Window")  
 top\_s.geometry("1199x600+100+50")  
  
 # Background Image  
 #bg = PhotoImage(file="image/login.png")  
 #bg\_image = Label(root, image=bg).place(x=0, y=0, relwidth=1, relheight=1)  
  
 canvas\_s = Canvas(top\_s, height=500, width=400)  
 canvas\_s.pack(fill="both", expand=True)  
 canvas\_s.create\_image(0, 0, image=bg\_signup, anchor="nw")  
  
 # Login widget  
 Frame\_signup = Frame(top\_s, bg="white")  
 Frame\_signup.place(x=250, y=50, height=420, width=700)  
  
 title = Label(Frame\_signup, text="Signup Here", font=("Segoe Script", 32, "bold"), fg="#5eabf7",bg="white").place(x=200, y=10)  
  
 # Name  
 signup\_name = Label(Frame\_signup, text="Name", font=("PMingLiU-ExtB", 15, "bold"), fg="#4d647a", bg="white").place(x=60,y=80)  
 global txt\_name  
 txt\_name = Entry(Frame\_signup, font=("Times new roman", 12), bg="lightgray")  
 txt\_name.place(x=60, y=110, width=250, height=30)  
  
 # Username  
 signup\_user = Label(Frame\_signup, text="Username", font=("PMingLiU-ExtB", 15, "bold"), fg="#4d647a", bg="white").place(x=400, y=80)  
 global txt\_user  
 txt\_user= Entry(Frame\_signup, font=("Times new roman", 12), bg="lightgray")  
 txt\_user.place(x=400, y=110, width=250, height=30)  
  
 # Email  
 signup\_email = Label(Frame\_signup, text="Email address", font=("PMingLiU-ExtB", 15, "bold"), fg="#4d647a",bg="white").place(x=60, y=150)  
 global txt\_email  
 txt\_email= Entry(Frame\_signup, font=("Times new roman", 12), bg="lightgray")  
 txt\_email.place(x=60, y=180, width=250, height=30)  
  
 # Phoneno  
 signup\_phoneno = Label(Frame\_signup, text="Phone number", font=("PMingLiU-ExtB", 15, "bold"), fg="#4d647a",bg="white").place(x=400, y=150)  
 global txt\_phoneno  
 txt\_phoneno= Entry(Frame\_signup, font=("Times new roman", 12), bg="lightgray")  
 txt\_phoneno.place(x=400, y=180, width=250, height=30)  
  
 # Password  
 signup\_pass = Label(Frame\_signup, text="Password", font=("PMingLiU-ExtB", 15, "bold"), fg="#4d647a", bg="white").place(x=60, y=220)  
 global txt\_pass  
 txt\_pass= Entry(Frame\_signup, font=("Times new roman", 12), bg="lightgray",show='\*')  
 txt\_pass.place(x=60, y=250, width=250, height=30)  
  
 # Confirm Password  
 signup\_confirmpass = Label(Frame\_signup, text="Confirm Password", font=("PMingLiU-ExtB", 15, "bold"), fg="#4d647a",bg="white").place(x=400, y=220)  
 global txt\_confirmpass  
 txt\_confirmpass= Entry(Frame\_signup, font=("Times new roman", 12), bg="lightgray",show='\*')  
 txt\_confirmpass.place(x=400, y=250, width=250, height=30)  
  
 # Question  
 signup\_question = Label(Frame\_signup, text="Select security question", font=("PMingLiU-ExtB", 15, "bold"), fg="#4d647a",bg="white").place(x=60, y=290)  
 # self.txt\_question = Entry(Frame\_signup, font=("Times new roman", 12), bg="lightgray")  
 # self.txt\_question.place(x=60, y=320, width=250, height=30)  
  
 # Drop down menu  
 options = [  
 "What is your place of birth?",  
 "What is your favourite colour?",  
 "Who was your first teacher?",  
 "What is your favourite food?"  
 ]  
  
 clicked = StringVar()  
  
 def show():  
 global txt\_question  
 txt\_question=clicked.get()  
  
 clicked.set("Choose any one question")  
 signup\_drop = OptionMenu(Frame\_signup, clicked, \*options)  
 signup\_drop.config(font=("Arial", 10, "bold"), fg="#3b3d40", bg="light grey", cursor="hand2")  
 signup\_drop.place(x=60, y=320, width=210, height=30)  
 select\_button = Button(Frame\_signup, text="Select", command=show, fg="black", bg="light grey").place(x=270,y=320,width=40,height=28)  
  
 # Answer  
 signup\_answer = Label(Frame\_signup, text="Answer", font=("PMingLiU-ExtB", 15, "bold"), fg="#4d647a", bg="white").place(x=400, y=290)  
 global txt\_answer  
 txt\_answer = Entry(Frame\_signup, font=("Times new roman", 12), bg="lightgray")  
 txt\_answer.place(x=400, y=320, width=250, height=30)  
  
 # Signup button  
 Signup\_btn = Button(top\_s, command=signup\_function, cursor="hand2", text="Signup", fg="white",bg="#5eabf7", font=("Book Antiqua", 20)).place(x=510, y=450, width=180, height=40)  
  
 # Destroy button  
 d\_button = Button(top\_s, text="<- Go Back", command=top\_s.destroy, cursor="hand2", fg="white", bg="#5eabf7",font=("Book Antiqua", 10)).place(x=20, y=20, width=80, height=30)  
  
  
def signup\_function():  
 if txt\_pass.get() == "" or txt\_user.get() == "" or txt\_name.get() == "" or txt\_email.get() == "" or txt\_phoneno.get() == "" or txt\_confirmpass.get() == "" or txt\_answer.get() == "":  
 messagebox.showerror("Error", "All fields are required", parent=top\_s)  
 elif txt\_pass.get() != txt\_confirmpass.get():  
 messagebox.showerror("Error", "Your Passwords does not match", parent=top\_s)  
 else:  
 messagebox.showinfo("Welcome", "You have successfully signed up", parent=top\_s)  
  
  
 # Saving data in text file  
 text\_file = open("data.txt", 'a')  
 text\_file.write(txt\_name.get().ljust(30))  
 text\_file.write(txt\_user.get().ljust(30))  
 text\_file.write(txt\_email.get().ljust(30))  
 text\_file.write(txt\_phoneno.get().ljust(30))  
 text\_file.write(txt\_pass.get().ljust(30))  
 text\_file.write(txt\_question.ljust(35))  
 text\_file.write(txt\_answer.get().ljust(30) + "\n")  
 text\_file.close()  
  
 top\_s.destroy()  
 Login()  
  
  
def Login():  
  
 global top  
 top=Toplevel()  
 top.title("Login Window")  
 top.geometry("1199x600+100+50")  
  
 # Background Image  
  
 #bg = PhotoImage(file="image/login.png")  
 #bg\_image = Label(top, image=bg).place(x=0, y=0, relwidth=1, relheight=1)  
  
 canvas = Canvas(top, height=500, width=400)  
 canvas.pack(fill="both",expand=True)  
 canvas.create\_image(0,0,image=bg\_login,anchor="nw")  
  
 #my\_label=Label(top,image=photo).pack()  
 #bg\_button=Button(top,image=bg\_login)  
  
 # Login widget  
 Frame\_login = Frame(top, bg="white")  
 Frame\_login.place(x=150, y=150, height=340, width=430)  
  
 title = Label(Frame\_login, text="Login Here", font=("Segoe Script", 35, "bold"), fg="#53a2bd",bg="white").place(x=80, y=30)  
  
 # Username  
 login\_user = Label(Frame\_login, text="Username", font=("PMingLiU-ExtB", 15, "bold"), fg="grey", bg="white").place(x=40,y=120)  
 global txt\_user  
 txt\_user = Entry(Frame\_login, font=("Times new roman", 12), bg="lightgray")  
 txt\_user.place(x=40, y=150, width=350, height=30)  
  
 # Password  
 login\_pass = Label(Frame\_login, text="Password", font=("PMingLiU-ExtB", 15, "bold"), fg="grey", bg="white").place(x=40,y=190)  
 global txt\_pass  
 txt\_pass = Entry(Frame\_login, font=("Times new roman", 12), bg="lightgray",show='\*')  
 txt\_pass.place(x=40, y=220, width=350, height=30)  
  
 # Forget  
 forget\_btn = Button(Frame\_login, text="Forget Password?", command=otp\_page,cursor="hand2", bg="white", fg="#53a2bd", bd=0,font=("times new roman", 12)).place(x=40, y=260)  
  
 # Login button  
 Login\_btn = Button(top, command=login\_function, cursor="hand2", text="Login", fg="white",bg="#53a2bd", font=("Book Antiqua", 18)).place(x=300, y=470,width=140,height=40)  
  
 #Destroy button  
 d\_button = Button(top, text="<- Go Back", command=top.destroy,cursor="hand2",fg="white",bg="#53a2bd", font=("Book Antiqua", 10)).place(x=20, y=20, width=80, height=30)  
  
def login\_function():  
  
 # Reading data in text file  
 get\_user=[]  
 get\_pass=[]  
 text\_file = open("data.txt", 'r')  
 l=text\_file.readlines()[1:]  
 for i in l:  
 user\_details\_line=i.rstrip("\n")  
 each\_field=wrap(user\_details\_line,30)  
 get\_user.append(each\_field[1])  
 get\_pass.append(each\_field[4])  
 cred = {get\_user[i]: get\_pass[i] for i in range(len(get\_user))}  
 text\_file.close()  
  
  
 if txt\_pass.get() == "" or txt\_user.get() == "":  
 messagebox.showerror("Error", "All fields are required", parent=top)  
  
 #elif self.txt\_pass.get() != or self.txt\_user.get() != "Harsh":  
 elif txt\_pass.get() in cred.values() or txt\_user.get() in cred.keys():  
 if cred[txt\_user.get()]==txt\_pass.get():  
 #messagebox.showinfo("Welcome", "You have successfully logged in", parent=top)  
 Transfer()  
 #Continue  
 #Continue\_btn = Button(top, command=Transfer, cursor="hand2", text="Continue", fg="white", bg="#53a2bd",font=("Book Antiqua", 18)).place(x=300, y=470, width=140, height=40)  
 else:  
 messagebox.showerror("Error", "Invalid Username/Password", parent=top)  
  
 else:  
 #self.txt\_pass.get() not in cred.values() or self.txt\_user.get() not in cred.keys():  
 messagebox.showerror("Error", "Invalid Username/Password", parent=top)  
  
def otp\_page():  
  
 global top\_o  
 top\_o = Toplevel()  
 top\_o.title("Verification Window")  
 top\_o.geometry("1199x600+100+50")  
  
 # Background Image  
 canvas = Canvas(top\_o, height=500, width=400)  
 canvas.pack(fill="both", expand=True)  
 canvas.create\_image(0, 0, image=bg\_login, anchor="nw")  
  
  
 # OTP widget  
 Frame\_otp = Frame(top\_o, bg="white")  
 Frame\_otp.place(x=150, y=100, height=440, width=430)  
  
 title = Label(Frame\_otp, text="Verify OTP", font=("Segoe Script", 35, "bold"), fg="#53a2bd", bg="white").place(x=80, y=30)  
  
 # Username  
 otp\_user = Label(Frame\_otp, text="Username", font=("PMingLiU-ExtB", 15, "bold"), fg="grey", bg="white").place(x=40, y=120)  
 global otp\_getuser  
 otp\_getuser = Entry(Frame\_otp, font=("Times new roman", 12), bg="lightgray")  
 otp\_getuser.place(x=40, y=150, width=350, height=30)  
  
 # Phone number  
 otp\_phoneno = Label(Frame\_otp, text="Phone number", font=("PMingLiU-ExtB", 15, "bold"), fg="grey", bg="white").place(x=40, y=190)  
 global otp\_getphoneno  
 otp\_getphoneno = Entry(Frame\_otp, font=("Times new roman", 12), bg="lightgray")  
 otp\_getphoneno.place(x=40, y=220, width=220, height=30)  
  
 # OTP  
 otp = Label(Frame\_otp, text="OTP", font=("PMingLiU-ExtB", 15, "bold"), fg="grey", bg="white").place(x=40,y=260)  
 global otp\_get  
 otp\_get = Entry(Frame\_otp, font=("Times new roman", 12), bg="lightgray")  
 otp\_get.place(x=40, y=290, width=350, height=30)  
  
 # Send button  
 Send\_btn = Button(top\_o, command=otp\_generated, cursor="hand2", text="Send OTP", fg="white", bg="#53a2bd",font=("Book Antiqua", 12)).place(x=440, y=315, width=100, height=40)  
  
 # Confirm button  
 Confirm\_btn = Button(top\_o, command=otp\_function,cursor="hand2", text="Confirm", fg="white", bg="#53a2bd",font=("Book Antiqua", 18)).place(x=300, y=470, width=140, height=40)  
  
 # Destroy button  
 d\_button = Button(top\_o, text="<- Go Back", command=top\_o.destroy, cursor="hand2", fg="white", bg="#53a2bd",font=("Book Antiqua", 10)).place(x=20, y=20, width=80, height=30)  
  
def otp\_generated():  
 global otp  
 otp = rnd.randint(100000, 999999)  
 account\_sid = 'ACd351592ec2c7ea3472603bd81618f1d2'  
 auth\_token = '84572bee501bcbe8f08b66d1a4f2c729'  
 client = Client(account\_sid, auth\_token)  
  
 message = client.messages \  
 .create(  
 body='Otp generated for changing password - ' + str(otp),  
 from\_='+13527024673',  
 to='+918696268455'  
 )  
  
def otp\_function():  
 if otp\_getuser.get()=='' or otp\_getphoneno.get()=='':  
 messagebox.showerror("Error", "All fields are required", parent=top\_o)  
 elif otp==int(otp\_get.get()):  
 ChangePass()  
 '''get\_user = []  
 get\_pass = []  
 text\_file = open("data.txt", 'r')  
 l = text\_file.readlines()[1:]  
 for i in l:  
 user\_details\_line = i.rstrip("\n")  
 each\_field = wrap(user\_details\_line, 30)  
 get\_user.append(each\_field[1])  
 get\_pass.append(each\_field[4])  
 cred = {get\_user[i]: get\_pass[i] for i in range(len(get\_user))}  
 text\_file.close()  
 temp\_pass=cred[str(otp\_getuser.get())]  
 messagebox.showinfo("Password", "Your password is - " + temp\_pass, parent=top\_o)'''  
 else:  
 messagebox.showinfo("Error", "Invalid OTP", parent=top\_o)  
  
def ChangePass():  
  
 global top\_c  
 top\_c = Toplevel()  
 top\_c.title("New Password Window")  
 top\_c.geometry("1199x600+100+50")  
  
 # Background Image  
 canvas = Canvas(top\_c, height=500, width=400)  
 canvas.pack(fill="both", expand=True)  
 canvas.create\_image(0, 0, image=bg\_login, anchor="nw")  
  
  
 # OTP widget  
 Frame\_pass = Frame(top\_c, bg="white")  
 Frame\_pass.place(x=150, y=100, height=340, width=430)  
  
 title = Label(Frame\_pass, text="Change Password", font=("Segoe Script", 25, "bold"), fg="#53a2bd", bg="white").place(x=60, y=30)  
  
 # New password  
 new\_pass = Label(Frame\_pass, text="New Password", font=("PMingLiU-ExtB", 15, "bold"), fg="grey", bg="white").place(x=40, y=120)  
 global new\_getpass  
 new\_getpass = Entry(Frame\_pass, font=("Times new roman", 12), bg="lightgray")  
 new\_getpass.place(x=40, y=150, width=350, height=30)  
  
 # Confirm password  
 new\_confirmpass = Label(Frame\_pass, text="Confirm Password", font=("PMingLiU-ExtB", 15, "bold"), fg="grey", bg="white").place(x=40, y=190)  
 new\_getconfirmpass = Entry(Frame\_pass, font=("Times new roman", 12), bg="lightgray")  
 new\_getconfirmpass.place(x=40, y=220, width=350, height=30)  
  
 # Send button  
 Send\_btn = Button(top\_c, command=NewPass\_function, cursor="hand2", text="Change Password", fg="white", bg="#53a2bd",font=("Book Antiqua", 12)).place(x=295, y=380, width=150, height=40)  
  
 # Destroy button  
 d\_button = Button(top\_c, text="<- Go Back", command=top\_c.destroy, cursor="hand2", fg="white", bg="#53a2bd",font=("Book Antiqua", 10)).place(x=20, y=20, width=80, height=30)  
  
def NewPass\_function():  
 # Reading data in text file  
 get\_user = []  
 get\_pass = []  
 text\_file = open("data.txt", 'r')  
 l = text\_file.readlines()[1:]  
 for i in l:  
 user\_details\_line = i.rstrip("\n")  
 each\_field = wrap(user\_details\_line, 30)  
 get\_user.append(each\_field[1])  
 get\_pass.append(each\_field[4])  
 '''for i in get\_user:  
 if otp\_getuser.get() == i:  
 print(i)  
 text\_file.close()'''  
 cred = {get\_user[i]: get\_pass[i] for i in range(len(get\_user))}  
 print(cred[otp\_getuser.get()])  
 print(new\_getpass.get())  
 text\_file.close()  
  
 new\_file\_content = ""  
 text\_file1 = open("data.txt", 'r')  
 m = text\_file1.readlines()[0:]  
 for i in m:  
 user\_details\_line = i.rstrip("\n")  
 new\_line = user\_details\_line.replace(cred[otp\_getuser.get()], new\_getpass.get()).ljust(30)  
 new\_file\_content += new\_line + "\n"  
 text\_file1.close()  
  
 writing\_file = open("data.txt", "w")  
 writing\_file.write(new\_file\_content)  
 writing\_file.close()  
  
  
def Transfer():  
  
 global top\_t  
 top\_t=Toplevel()  
 top\_t.title("File Transfer Window")  
 top\_t.geometry("1199x600+100+50")  
  
 # Background Image  
 canvas = Canvas(top\_t, height=500, width=400)  
 canvas.pack(fill="both", expand=True)  
 canvas.create\_image(0, 0, image=bg\_login, anchor="nw")  
  
 # Login widget  
 Frame\_transfer = Frame(top\_t, bg="white")  
 Frame\_transfer.place(x=50, y=150, height=400, width=530)  
  
 def file\_opener():  
 global file\_input  
 file\_input = filedialog.askopenfilename(initialdir="/")  
 file\_label = Label(top\_t, text=file\_input, font=("PMingLiU-ExtB", 12, "bold"), fg="grey", bg="white").place(x=70, y=210)  
  
  
 # Select file button  
 Select\_btn = Button(top\_t, command=file\_opener, cursor="hand2", text="Select file", fg="white", bg="#53a2bd",font=("Book Antiqua", 18)).place(x=370, y=200, width=140, height=40)  
  
 # Send file button  
 Send\_btn = Button(top\_t, command=client,cursor="hand2", text="Send file", fg="white", bg="#53a2bd",font=("Book Antiqua", 18)).place(x=250, y=290, width=140, height=40)  
  
 title = Label(top\_t, text="------------or------------", font=("Segoe Script", 20, "bold"), fg="#53a2bd",bg="white").place(x=100, y=370)  
  
 # Recieve file button  
 Recieve\_btn = Button(top\_t, command=server,cursor="hand2", text="Receive file", fg="white", bg="#53a2bd",font=("Book Antiqua", 18)).place(x=250, y=450, width=140, height=40)  
  
 # Destroy button  
 d\_button = Button(top\_t, text="Logout", command=top\_t.destroy, cursor="hand2", fg="white", bg="#53a2bd",font=("Book Antiqua", 10)).place(x=20, y=20, width=80, height=30)  
  
 # Report button  
 r\_button = Button(top\_t, text="Generate Report", command=VirusTotal, cursor="hand2", fg="white", bg="#53a2bd",font=("Book Antiqua", 10)).place(x=1000, y=20, width=150, height=30)  
  
def server():  
 global SERVER\_HOST  
 SERVER\_HOST = "0.0.0.0"  
 SERVER\_PORT = 5001  
 # receive 4096 bytes each time  
 BUFFER\_SIZE = 4096  
 SEPARATOR = "<SEPARATOR>"  
 # create the server socket  
 # TCP socket  
 s = socket.socket()  
 # bind the socket to our local address  
 s.bind((SERVER\_HOST, SERVER\_PORT))  
 # enabling our server to accept connections  
 # 5 here is the number of unaccepted connections that  
 # the system will allow before refusing new connections  
 s.listen(5)  
 print(f"[\*] Listening as {SERVER\_HOST}:{SERVER\_PORT}")  
 # accept connection if there is any  
 client\_socket, address = s.accept()  
 # if below code is executed, that means the sender is connected  
 print(f"[+] {address} is connected.")  
 # receive the file infos  
 # receive using client socket, not server socket  
 received = client\_socket.recv(BUFFER\_SIZE).decode()  
 filename, filesize = received.split(SEPARATOR)  
 # remove absolute path if there is  
 filename = os.path.basename(filename)  
 # convert to integer  
 filesize = int(filesize)  
 # start receiving the file from the socket  
 # and writing to the file stream  
 progress = tqdm.tqdm(range(filesize), f"Receiving {filename}", unit="B", unit\_scale=True, unit\_divisor=1024)  
 with open(filename, "wb") as f:  
 while True:  
 # read 1024 bytes from the socket (receive)  
 bytes\_read = client\_socket.recv(BUFFER\_SIZE)  
 if not bytes\_read:  
 # nothing is received  
 # file transmitting is done  
 break  
 # write to the file the bytes we just received  
 f.write(bytes\_read)  
 # update the progress bar  
 progress.update(len(bytes\_read))  
  
 # close the client socket  
 client\_socket.close()  
 # close the server socket  
 s.close()  
  
  
  
def client():  
 SEPARATOR = "<SEPARATOR>"  
 BUFFER\_SIZE = 4096 # send 4096 bytes each time step  
 # the ip address or hostname of the server, the receiver  
 global host  
 host = "192.168.56.1"  
 # the port, let's use 5001  
 port = 5001  
 # the name of file we want to send, make sure it exists  
 filename = file\_input  
 # get the file size  
 filesize = os.path.getsize(filename)  
 # create the client socket  
 s = socket.socket()  
 print(f"[+] Connecting to {host}:{port}")  
 s.connect((host, port))  
 print("[+] Connected.")  
 # send the filename and filesize  
 s.send(f"{filename}{SEPARATOR}{filesize}".encode())  
 # start sending the file  
 progress = tqdm.tqdm(range(filesize), f"Sending {filename}", unit="B", unit\_scale=True, unit\_divisor=1024)  
 with open(filename, "rb") as f:  
 while True:  
 # read the bytes from the file  
 bytes\_read = f.read(BUFFER\_SIZE)  
 if not bytes\_read:  
 # file transmitting is done  
 break  
 # we use sendall to assure transimission in  
 # busy networks  
 s.sendall(bytes\_read)  
 # update the progress bar  
 progress.update(len(bytes\_read))  
 # close the socket  
 s.close()  
  
 # Storing Data in file  
 SERVER\_HOST = "0.0.0.0"  
 text\_file = open("iphistory.txt", 'a')  
 text\_file.write(txt\_user.get().ljust(30))  
 text\_file.write(SERVER\_HOST.ljust(30))  
 text\_file.write(host.ljust(30))  
 text\_file.write(file\_input.ljust(30) + "\n")  
 text\_file.close()  
  
def VirusTotal():  
 vtotal = Virustotal(API\_KEY="ab8230be5cf403599b78f8ca37f76e7cc1074f8fe23104e1b5914e7994921530", API\_VERSION="v3")  
 #vtotal = Virustotal(API\_VERSION="v3")  
 #with Virustotal(API\_KEY="ab8230be5cf403599b78f8ca37f76e7cc1074f8fe23104e1b5914e7994921530", API\_VERSION="v3") as vtotal:  
 #Declare PATH to file  
 FILE\_PATH = file\_input  
 #Create dictionary containing the file to send for multipart encoding upload  
 files = {"file": (os.path.basename(FILE\_PATH), open(os.path.abspath(FILE\_PATH), "rb"))}  
 resp = vtotal.request("files", files=files, method="POST")  
 # The v3 API returns the JSON response inside the 'data' key  
 # https://developers.virustotal.com/v3.0/reference#api-responses  
 # This property retrieves the structure inside 'data' from the JSON response  
 pprint(resp.data)  
 #pprint(resp["json\_resp"])  
  
 filename = file\_input  
 sha256\_hash = hashlib.sha256()  
 with open(filename, "rb") as f:  
 # Read and update hash string value in blocks of 4K  
 for byte\_block in iter(lambda: f.read(4096), b""):  
 sha256\_hash.update(byte\_block)  
 print(sha256\_hash.hexdigest())  
  
 FILE\_ID = sha256\_hash.hexdigest()  
 resp1 = vtotal.request(f"files/{FILE\_ID}")  
 pprint(resp1.data)  
  
root=Tk()  
# Background Image  
bg = PhotoImage(file="image/wel.png")  
bg\_login = PhotoImage(file="image/login.png")  
bg\_signup = PhotoImage(file="image/signup.png")  
bg\_image = Label(root, image=bg).place(x=0, y=0, relwidth=1, relheight=1)  
  
# Welcome widget  
  
#title = Label(root, text="Welcome!!!", font=("Copperplate Gothic Bold", 32, "bold"), fg="#0b0c17",bg="light yellow").place(x=450, y=50)  
  
# Signup button  
Signup\_btn = Button(root,cursor="hand2",command=Signup,text="Signup", fg="white", bg="#8fc7b5",font=("Book Antiqua", 18)).place(x=125, y=275, width=140, height=40)  
  
# Login button  
Login\_btn = Button(root, cursor="hand2", command=Login, text="Login", fg="white", bg="#8fc7b5",font=("Book Antiqua", 18)).place(x=125, y=200, width=140, height=40)  
  
  
root.title("Welcome Window")  
root.geometry("1199x600+100+50")  
root.resizable(False, False)  
root.mainloop()