TUGAS KELOMPOK CHAT APPLICATION
PEMROGRAMAN JARINGAN (D)

Kelompok 2

05111840000057 Maisie Chiara Salsabila

05111840000130 I Gusti Agung Chintya

05111840000163 Putu Putri Natih

Dosen Pengampu:

Royyana Muslim Ijtihadie, S.Kom., M.Kom., Ph.D.

Departemen Teknik Informatika

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember (ITS)

Surabaya

2021

SOAL:

1. *Graphical User Interface* untuk *chat client*
2. Pengiriman dan penerimaan *file* antar *user*
3. *Group Chat*

KONTRIBUSI:

4. 05111840000057 Maisie Chiara Salsabila
    a. *Group Chat*
5. 05111840000130 I Gusti Agung Chintya Prema Dewi
    a. Pengiriman dan penerimaan *file* antar *user*
6. 05111840000163 Putu Putri Natih
    a. *Graphical User Interface* untuk *chat client*

Link: Github

1. CHAT GUI

1.1. Protokol

Pada *file* ***clientgui.py***

1.2. Tujuan

File ini berfungsi untuk membentuk tampilan aplikasi *chat* serta mengirimkan data ke server.

1.3. Aturan Protokol

1.3.1. Server telah dijalankan

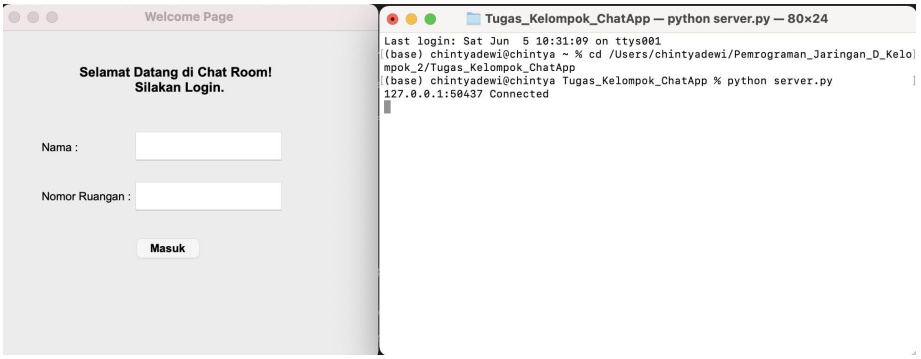
1.3.2. String diinput pada entri yang tersedia

1.4. GUI (*Graphical User Interface*)

Dapat dilihat pada link.

1.4.1. *__init__*

Fungsi ini merupakan inisialisasi dari aplikasi untuk menghubungkan client ke server, hasil tampilannya berupa window GUI. Halaman ini merupakan welcome page dimana pengguna harus mengisikan nama dan room_id untuk menggunakan aplikasi.




```
def __init__(self, ip_address, port):
    self.server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    self.server.connect((ip_address, port))

    self.Window = tk.Tk()
    self.Window.withdraw()

    self.login = tk.Toplevel()

    self.login.title("Welcome Page")
    self.login.resizable(width=False, height=False)
    self.login.configure(width=400, height=350)

    self.pls = tk.Label(self.login,
                        text="Selamat Datang di Chat Room! \n Silakan Login.",
                        justify=tk.CENTER,
                        font="Arial 14 bold")
```

```
self.pls.place(relheight=0.15, relx=0.2, rely=0.07)

self.userLabel = tk.Label(self.login, text="Nama : ", font="Arial 12")
self.userLabel.place(relheight=0.2, relx=0.1, rely=0.25)

self.userEntry = tk.Entry(self.login, font="Arial 12")
self.userEntry.place(relwidth=0.4, relheight=0.1, relx=0.35, rely=0.30)
self.userEntry.focus()

self.roomLabel = tk.Label(self.login, text="Nomor Ruangan : ", font="Arial 12")
self.roomLabel.place(relheight=0.2, relx=0.1, rely=0.40)

self.roomEntry = tk.Entry(self.login, font="Arial 11", show="*")
self.roomEntry.place(relwidth=0.4, relheight=0.1, relx=0.35, rely=0.45)

self.go = tk.Button(self.login,
                    text="Masuk",
                    font="Arial 12 bold",
                    command=lambda: self.goAhead(self.userEntry.get(), self.roomEntry.get()))

self.go.place(relx=0.35, rely=0.62)

self.Window.mainloop()
```

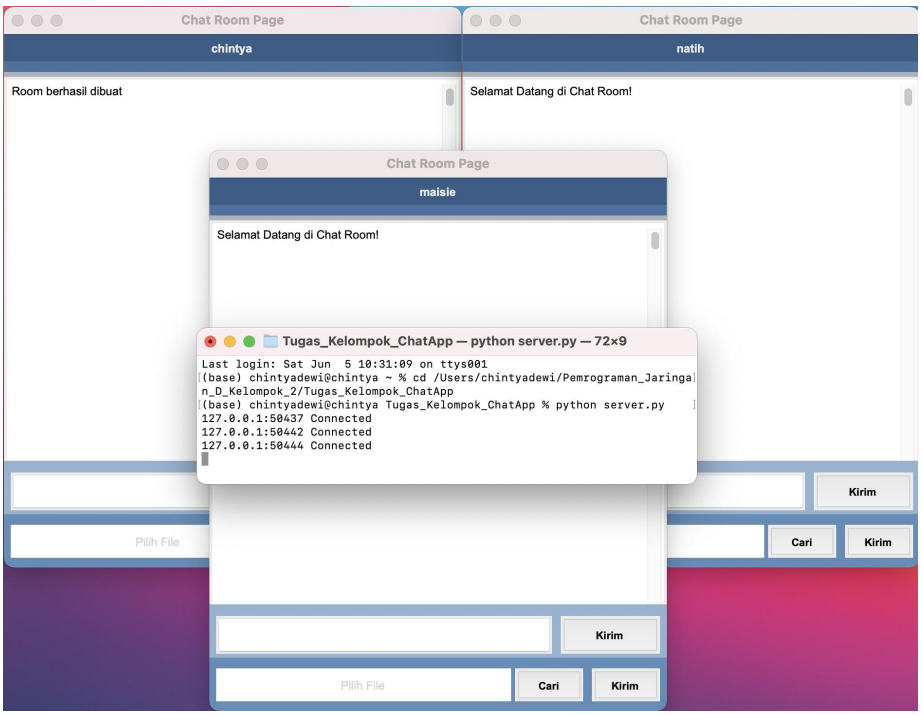
1.4.2. *goAhead*

Fungsi ini akan mengirimkan nama dan *room_id* ke server kemudian menampilkan halaman *chat room* sebagai penutup halaman *login*.

```
def goAhead(self, nama, room_id=0):
    self.name = nama
    self.server.send(str.encode(nama))
    time.sleep(0.1)
    self.server.send(str.encode(room_id))

    self.login.destroy()
    self.layout()

    rcv = threading.Thread(target=self.receive)
    rcv.start()
```

1.4.3. *Layout*

Fungsi ini akan menampilkan halaman secara menyeluruh dari *chat room*.



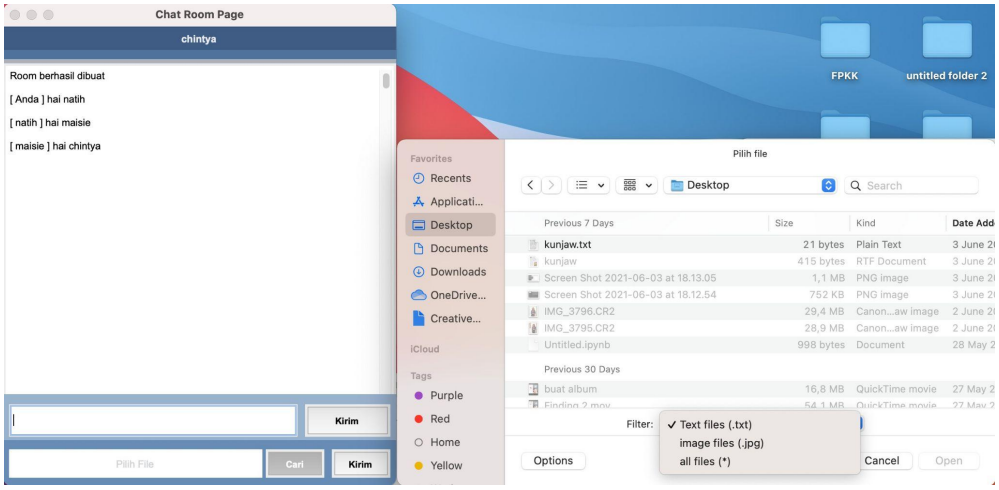

```
def layout(self):
    self.Window.deiconify()
    self.Window.title("Chat Room Page")
    self.Window.resizable(width=False, height=False)
    self.Window.configure(width=470, height=550, bg="#4472a0")
    self.chatBoxHead = tk.Label(self.Window,
                                bg="#365c87",
                                fg="#EAECEE",
                                text=self.name,
                                font="Arial 14 bold",
                                pady=5)

    self.chatBoxHead.place(relwidth=1)

    self.line = tk.Label(self.Window, width=450, bg="#ABB2B9")

    self.line.place(relwidth=1, rely=0.07, relheight=0.012)

    self.textCons = tk.Text(self.Window,
                            width=20,
                            height=2,
                            bg="#ffffff",
                            fg="#000000",
                            font="Arial 12",
                            padx=5,
                            pady=5)

    self.textCons.place(relheight=0.745, relwidth=1, rely=0.08)

    self.labelBottom = tk.Label(self.Window, bg="#93b2d1", height=80)

    self.labelBottom.place(relwidth=1,
                           rely=0.8)

    self.entryMsg = tk.Entry(self.labelBottom,
                             bg="#ffffff",
                             fg="#000000",
                             font="Arial 12")
```

```
        self.entryMsg.place(relwidth=0.74,
                            relheight=0.03,
                            rely=0.008,
                            relx=0.011)
        self.entryMsg.focus()

        self.buttonMsg = tk.Button(self.labelBottom,
                                   text="Kirim",
                                   font="Arial 11 bold",
                                   width=20,
                                   bg="#ABB2B9",
                                   command=lambda: self.sendButton(self.entryMsg.get()))

        self.buttonMsg.place(relx=0.77,
                             rely=0.008,
                             relheight=0.03,
                             relwidth=0.22)

        self.labelFile = tk.Label(self.Window, bg="#5d8cba", height=70)

        self.labelFile.place(relwidth=1,
                             rely=0.9)

        self.fileLocation = tk.Label(self.labelFile,
                                     text="Pilih File",
                                     bg="#FFFFFF",
                                     fg="#d2d4d6",
                                     font="Arial 12")
        self.fileLocation.place(relwidth=0.65,
                                relheight=0.03,
                                rely=0.008,
                                relx=0.011)

        self.browse = tk.Button(self.labelFile,
                                text="Cari",
                                font="Arial 11 bold",
                                width=13,
                                bg="#b9abab",
                                command=self.browseFile)
        self.browse.place(relx=0.67,
                          rely=0.008,
                          relheight=0.03,
                          relwidth=0.15)

        self.kirimfileBtn = tk.Button(self.labelFile,
                                      text="Kirim",
                                      font="Arial 11 bold",
                                      width=13,
                                      bg="#b9abab",
                                      command=self.sendFile)

        self.kirimfileBtn.place(relx=0.84,
                                rely=0.008,
                                relheight=0.03,
                                relwidth=0.15)

        self.textCons.config(cursor="arrow")
        scrollbar = tk.Scrollbar(self.textCons)
        scrollbar.place(relheight=1,
                        relx=0.974)

        scrollbar.config(command=self.textCons.yview)
        self.textCons.config(state=tk.DISABLED)
```

2. *FILE BROADCASTING*

2.1. *BROWSEFILE*

Fungsi ini bertujuan untuk mencari *file* yang dikirimkan melalui *chat room*. Hasil dari fungsi ini adalah lokasi *file* tertentu akan terbuka untuk memilih *file* yang akan dikirim.

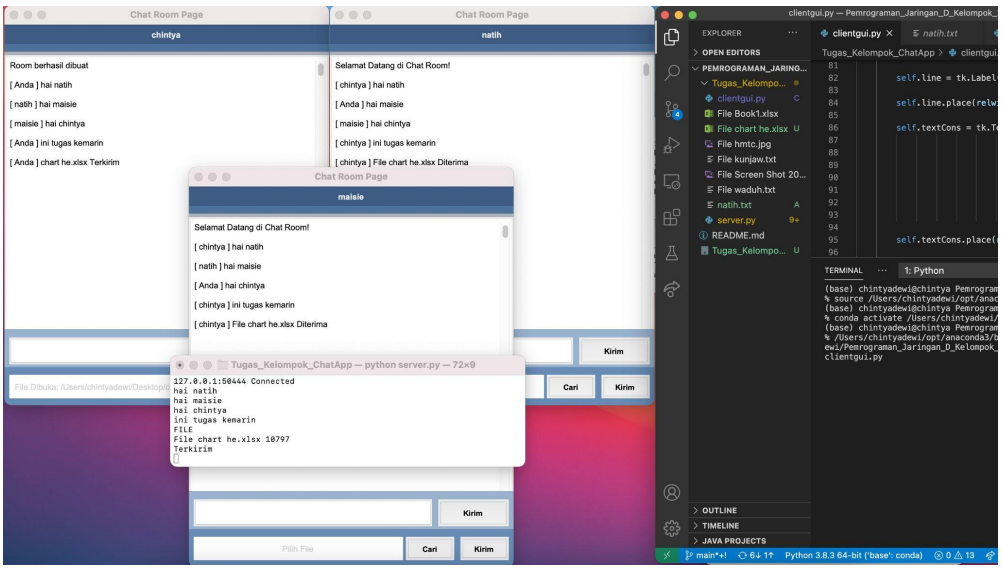



```
def browseFile(self):
    self.filename = filedialog.askopenfilename(initialdir="/",
                                               title="Pilih file",
                                               filetypes=(("Text files",
                                                           "*.txt"),
                                                          ("image files",
                                                           "*.jpg"),
                                                          ("all files",
                                                           "*.*")))
    self.fileLocation.configure(text="File Dibuka: " + self.filename)
```

2.2. SEND*FILE*

Fungsi ini bertujuan untuk mengirim *file* ke server kemudian terkirim melalui *chat room*. Hasil dari fungsi ini adalah jika berhasil maka *file* akan terkirim dengan nama *file* yang sama dengan data *file* tersebut. Sebaliknya jika gagal maka akan muncul pesan kesalahan.

```
def sendFile(self):
    self.server.send("FILE".encode())
    time.sleep(0.1)
    self.server.send(str("File " + os.path.basename(self.filename)).encode())
    time.sleep(0.1)
    self.server.send(str(os.path.getsize(self.filename)).encode())
    time.sleep(0.1)

    file = open(self.filename, "rb")
    data = file.read(1024)
    while data:
        self.server.send(data)
        data = file.read(1024)
    self.textCons.config(state=tk.DISABLED)
    self.textCons.config(state=tk.NORMAL)
    self.textCons.insert(tk.END, "[ Anda ] "
                         + str(os.path.basename(self.filename))
                         + " Terkirim\n\n")
    self.textCons.config(state=tk.DISABLED)
    self.textCons.see(tk.END)
```



3. KIRIM

3.1. TOMBOL KIRIM (sendButton)

Fungsi ini berfungsi untuk menghapus text yang ada pada box input dan memanggil fungsi kirim message.

```
def sendButton(self, msg):
    self.textCons.config(state=tk.DISABLED)
    self.msg = msg
    self.entryMsg.delete(0, tk.END)
    snd = threading.Thread(target=self.sendMessage)
    snd.start()
```

3.2. KIRIM MESSAGE (sendMessage)

Fungsi ini bertujuan untuk mengirim pesan ke server yang nantinya akan diteruskan ke *client* pada *chat room* yang sama, serta untuk menampilkan pesan yang dikirim oleh *client* pengirim. Hasil dari fungsi ini adalah pesan akan terkirim pada server.

```
def sendMessage(self):
    self.textCons.config(state=tk.DISABLED)
    while True:
        self.server.send(self.msg.encode())
        self.textCons.config(state=tk.NORMAL)
        self.textCons.insert(tk.END,
                             "[ Anda ] " + self.msg + "\n\n")

        self.textCons.config(state=tk.DISABLED)
        self.textCons.see(tk.END)
        break
```

4. TERIMA

Fungsi ini bertujuan untuk menerima *file* dari server, menulis *file* pada *client,* dan sebagai pemberitahuan saat *file* diterima. Hasil dari fungsi ini adalah diterimanya *file*. Jika terjadi error maka server akan terputus.

```
def receive(self):
    while True:
        try:
            message = self.server.recv(1024).decode()

            if str(message) == "FILE":
                file_name = self.server.recv(1024).decode()
                lenOfFile = self.server.recv(1024).decode()
                send_user = self.server.recv(1024).decode()

                if os.path.exists(file_name):
                    os.remove(file_name)

                total = 0
                with open(file_name, 'wb') as file:
                    while str(total) != lenOfFile:
                        data = self.server.recv(1024)
                        total = total + len(data)
                        file.write(data)

                self.textCons.config(state=tk.DISABLED)
                self.textCons.config(state=tk.NORMAL)
                self.textCons.insert(tk.END, "[ " + str(send_user) + " ] " + file_name + " Diterima\n\n")
                self.textCons.config(state=tk.DISABLED)
                self.textCons.see(tk.END)

            else:
                self.textCons.config(state=tk.DISABLED)
                self.textCons.config(state=tk.NORMAL)
                self.textCons.insert(tk.END,
                                     message + "\n\n")

                self.textCons.config(state=tk.DISABLED)
                self.textCons.see(tk.END)

        except:
            print("Terdapat error!")
            self.server.close()
            break
```

5. SERVER

File ***server.py*** ini berfungsi untuk melayani *client* dalam *request file server*. Aturan protokolnya adalah *client* harus mengirimkan *request* dalam bentuk string.

5.1. MENERIMA KONEKSI (accept_connections)

Fungsi ini bertujuan untuk menginisiasi koneksi antara server dan klien. Hasil dari fungsi ini adalah jika berhasil koneksi akan terbuat lalu terbuatnya thread. Jika gagal, maka socket akan terputus.

```
def accept_connections(self, ip_address, port):
    self.ip_address = ip_address
    self.port = port
    self.server.bind((self.ip_address, int(self.port)))
    self.server.listen(100)

    while True:
        connection, address = self.server.accept()
        print(str(address[0]) + ":" + str(address[1]) + " Connected")

        start_new_thread(self.clientThread, (connection,))

    self.server.close()
```

5.2. CLIENT THREAD

Fungsi ini bertujuan untuk memeriksa 'apakah *Group Chat* sudah pernah dibuat?' dan 'jenis pesan yang dikirimkan berupa file atau teks?'. Dibutuhkan ***connection*** sebagai parameter. Hasil dari fungsi ini adalah jika berhasil dan *Group Chat* belum ada, maka akan muncul pesan 'Room berhasil dibuat', jika berhasil dan *Group Chat* sudah ada maka akan 'Selamat datang di Chat Room', jika pesan yang dibaca oleh *client* 1 berupa file, maka akan dikirimkan file ke *client* lainnya menggunakan fungsi **broadcastFile**, jika pesan yang dibaca oleh *client* 1 berupa teks, maka akan dikirimkan file ke *client* lainnya menggunakan fungsi **broadcast**. Sebaliknya, jika gagal maka koneksi akan terputus.

```
def clientThread(self, connection):
    user_id = connection.recv(1024).decode().replace("User ", "")
    room_id = connection.recv(1024).decode().replace("Join ", "")

    if room_id not in self.rooms:
        connection.send("Room berhasil dibuat".encode())
    else:
        connection.send("Selamat Datang di Chat Room!!".encode())

    self.rooms[room_id].append(connection)
```

```
while True:
    try:
        message = connection.recv(1024)
        print(str(message.decode()))
        if message:
            if str(message.decode()) == "FILE":
                self.broadcastFile(connection, room_id, user_id)

            else:
                message_to_send = "[ " + str(user_id) + " ] " + message.decode()
                self.broadcast(message_to_send, connection, room_id)

        else:
            self.remove(connection, room_id)
    except Exception as e:
        print(repr(e))
        print("Client disconnected earlier")
        break
```

5.3. BROADCAST FILE

Fungsi **broadcastFile** berfungsi untuk mengirim file antar *client*. Dibutuhkan **connection, room_id,** dan **user_id** sebagai parameter dari fungsi ini. Hasil dari fungsi ini adalah jika berhasil maka file yang telah berhasil dibaca akan terkirim ke *client* lainnya dengan tambahan “Terkirim”, sedangkan jika gagal server *client* akan otomatis terputus (**remove).**

```
def broadcastFile(self, connection, room_id, user_id):
    file_name = connection.recv(1024).decode()
    lenOfFile = connection.recv(1024).decode()
    for client in self.rooms[room_id]:
        if client != connection:
            try:
                client.send("FILE".encode())
                time.sleep(0.1)
                client.send(file_name.encode())
                time.sleep(0.1)
                client.send(lenOfFile.encode())
                time.sleep(0.1)
                client.send(user_id.encode())
            except:
                client.close()
                self.remove(client, room_id)
```

```
    total = 0
    print(file_name, lenOfFile)
    while str(total) != lenOfFile:
        data = connection.recv(1024)
        total = total + len(data)
        for client in self.rooms[room_id]:
            if client != connection:
                try:
                    client.send(data)
                    # time.sleep(0.1)
                except:
                    client.close()
                    self.remove(client, room_id)
    print("Terkirim")
```

5.4. BROADCAST

Fungsi **broadcast** bertujuan untuk mengirim pesan antar *client*. Dibutuhkan **message_to_send** yang merupakan pesan yang akan dikirim, **connection,** dan **room_id** sebagai parameter dari fungsi ini. Hasilnya adalah jika berhasil pesan yang dibaca akan terkirim ke *client* lainnya, sedangkan jika gagal maka koneksi *client* akan otomatis tertutup (**remove**).

```
def broadcast(self, message_to_send, connection, room_id):
    for client in self.rooms[room_id]:
        if client != connection:
            try:
                client.send(message_to_send.encode())
            except:
                client.close()
                self.remove(client, room_id)
```

5.5. REMOVE

Fungsi **remove** bertujuan untuk memutuskan koneksi *client* pada server. Dalam fungsi ini dibutuhkan **room_id** sebagai parameternya.

```
def remove(self, connection, room_id):
    if connection in self.rooms[room_id]:
        self.rooms[room_id].remove(connection)
```