

**TUGAS 5**  
**DOKUMENTASI**  
**LAYANAN *CHAT MULTITHREADED SERVER***



**PEMROGRAMAN JARINGAN B**  
**05111740007002 Muh. Shafwatallah Wahid**

**Dosen:**  
**Royyana Muslim Ijtihadie, S.Kom., Ph.D**

**Departemen Teknik Informatika**  
**Fakultas Teknologi Elektro dan Informatika Cerdas**  
**Institut Teknologi Sepuluh Nopember Surabaya**

## NAMA LAYANAN

Layanan Chat dengan Multithreaded Server

## DESKRIPSI

Layanan Chat Multithreaded Server merupakan layanan untuk bertukar pesan antar client dengan menggunakan socket dengan multithreaded server. Cara kerja layanan ini dimulai dengan server membuka socket dan menunggu adanya koneksi dari client. Setelah itu, client akan mengirimkan request ke server dan akan direspon kembali oleh server menuju client sesuai dengan request yang telah dikirimkan.

Penggunaan multithreaded server dalam layanan ini memungkinkan satu server dapat menangani lebih dari satu client secara bersamaan dalam satu jaringan yang sama.

## FORMAT DATA

Format data dari tiap fitur dalam layanan ini adalah sebagai berikut:

1. **Fitur Login**  
Input command: **login** (*username*) (*password*)
2. **Fitur chat user lain**  
Input command: **chat** (*user\_tujuan*) (*pesan*)
3. **Fitur kotak masuk**  
Input command: **kotakmasuk**
4. **Fitur melihat user aktif**  
Input command: **user\_on**
5. **Fitur Logout**  
Input command: **logout**

## DETAIL FITUR

### 1. LOGIN

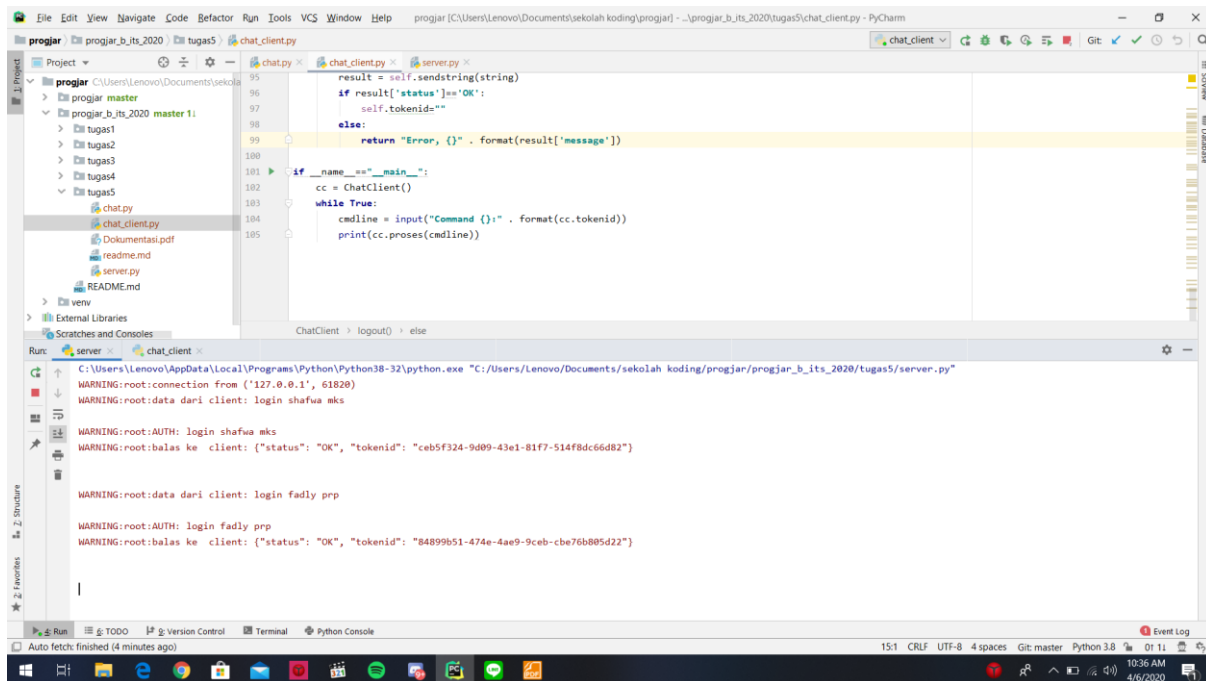
FUNGSI : Melakukan autentikasi user agar dapat mengakses fitur-fitur yang lain.

PARAMETER: (*username*) (*password*)

SYARAT : Username dan password yang dimasukkan sudah dalam daftar users atau diinputkan secara manual pada program.

OUTPUT : a. Jika respon berhasil (OK), outputnya adalah username [user] loggen in, token: [tokenid]  
b. Jika respon gagal (ERROR), outputnya adalah message [message\_error]

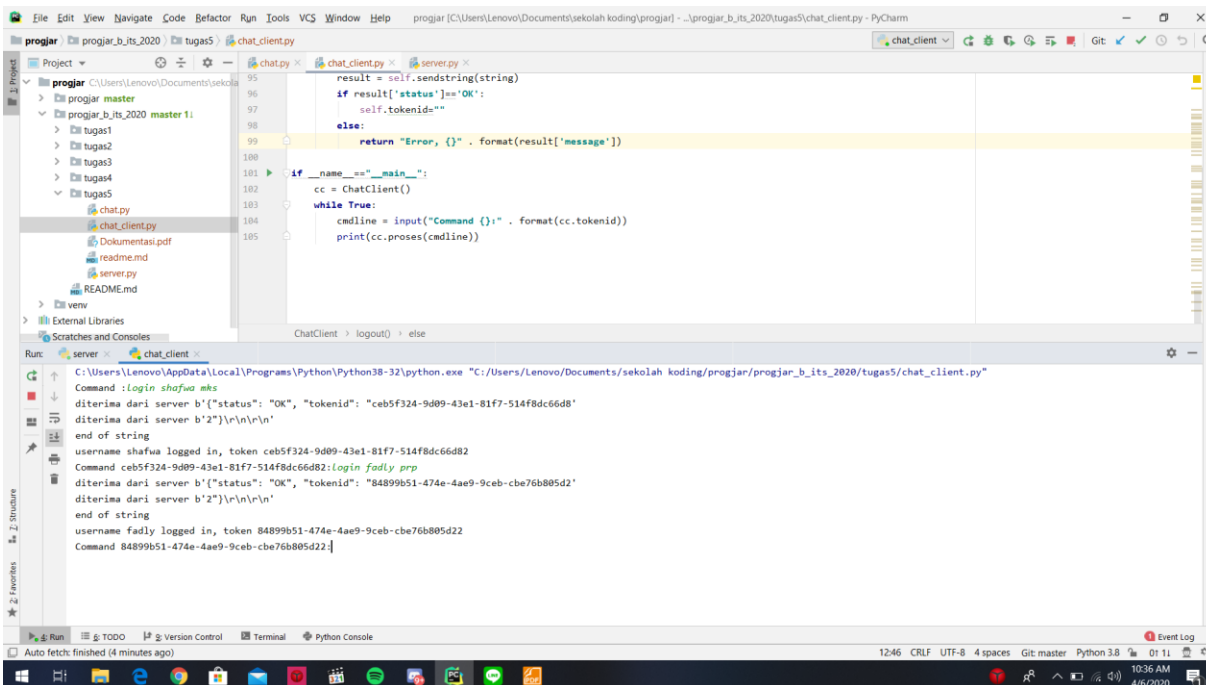
## DOKUMENTASI LOGIN: server.py



The screenshot shows the PyCharm IDE with the `server.py` file open. The code defines a `ChatClient` class and a `main` function. The `main` function creates a `ChatClient` instance and enters a loop where it prompts the user for a command. The output window shows the following logs:

```
WARNING:root:connection from ('127.0.0.1', 61820)
WARNING:root:data dari client: login shafwa mks
WARNING:root:AUTH: login shafwa mks
WARNING:root:balas ke client: {"status": "OK", "tokenId": "ceb5f324-9d09-43e1-81f7-514f8dc66d82"}
WARNING:root:data dari client: login fadly prp
WARNING:root:AUTH: login fadly prp
WARNING:root:balas ke client: {"status": "OK", "tokenId": "84899b51-474e-4ae9-9ceb-cbe76b805d22"}
```

## chat\_client.py



The screenshot shows the PyCharm IDE with the `chat_client.py` file open. The code defines a `ChatClient` class and a `main` function. The `main` function creates a `ChatClient` instance and enters a loop where it prompts the user for a command. The output window shows the following logs:

```
Command : login shafwa mks
diterima dari server b'{"status": "OK", "tokenId": "ceb5f324-9d09-43e1-81f7-514f8dc66d82"}'
diterima dari server b'{"status": "OK", "tokenId": "84899b51-474e-4ae9-9ceb-cbe76b805d22"}'
end of string
username shafwa logged in, token ceb5f324-9d09-43e1-81f7-514f8dc66d82
Command : login fadly prp
diterima dari server b'{"status": "OK", "tokenId": "84899b51-474e-4ae9-9ceb-cbe76b805d22"}'
diterima dari server b'{"status": "OK", "tokenId": "84899b51-474e-4ae9-9ceb-cbe76b805d22"}'
end of string
username fadly logged in, token 84899b51-474e-4ae9-9ceb-cbe76b805d22
Command : 84899b51-474e-4ae9-9ceb-cbe76b805d22
```

## 2. CHAT

FUNGSI : Mengirim pesan antar client.

PARAMETER: (*user\_tujuan*) (*pesan*)

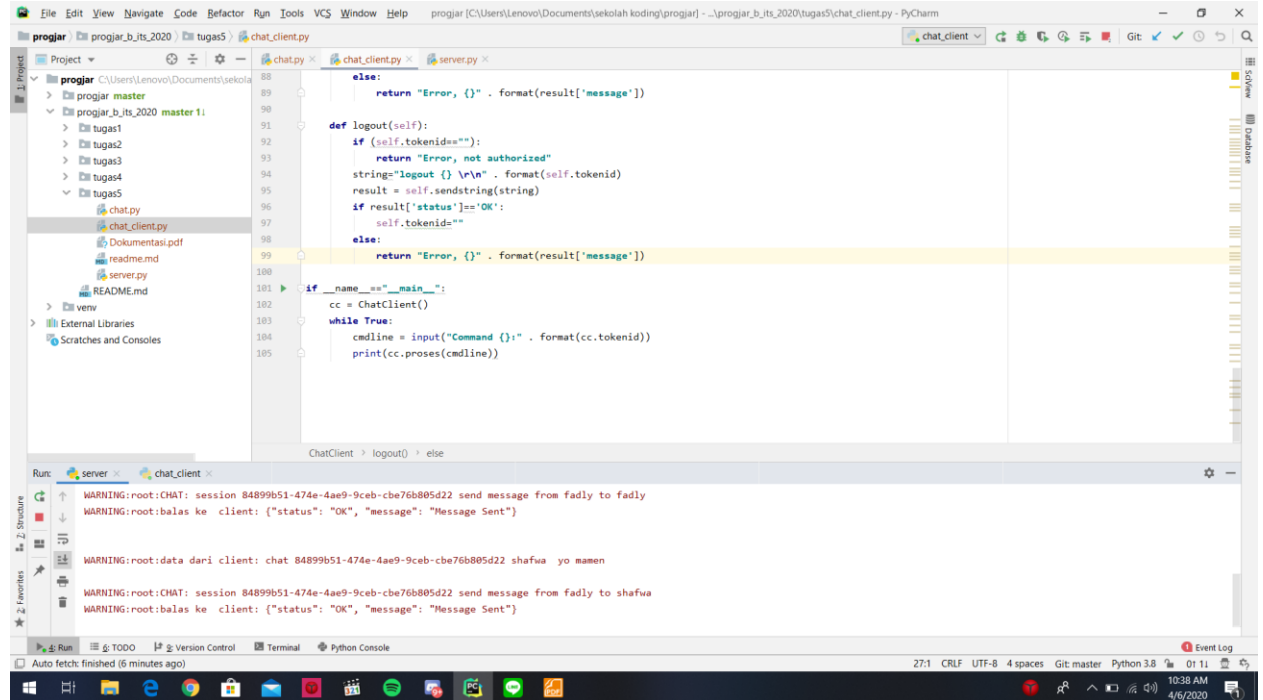
SYARAT : Sender harus melakukan login terlebih dahulu

OUTPUT : a. Jika respon berhasil (OK), outputnya adalah send to [username to]

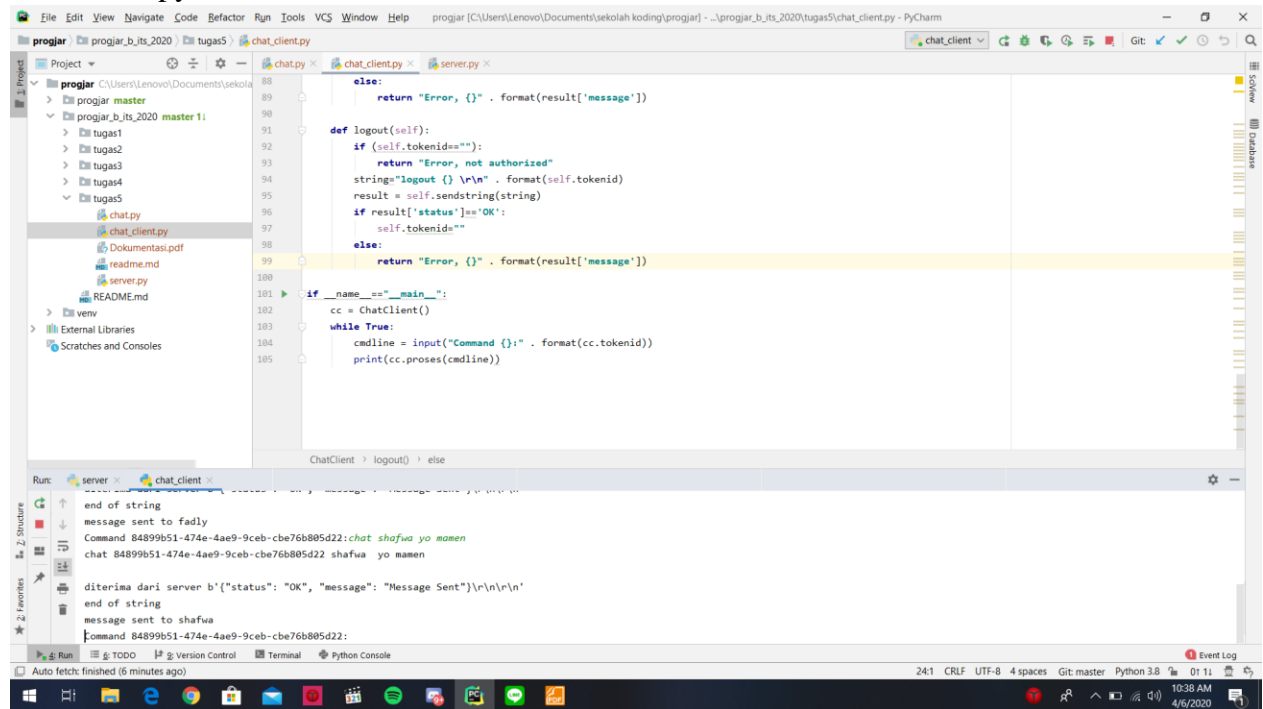
b. Jika respon gagal (ERROR), outputnya adalah message [message\_error]

## DOKUMENTASI CHAT

### server.py



### chat\_client.py



## 3. KOTAKMASUK

FUNGSI : Melihat pesan masuk yang dikirimkan antar user.

PARAMETER: -

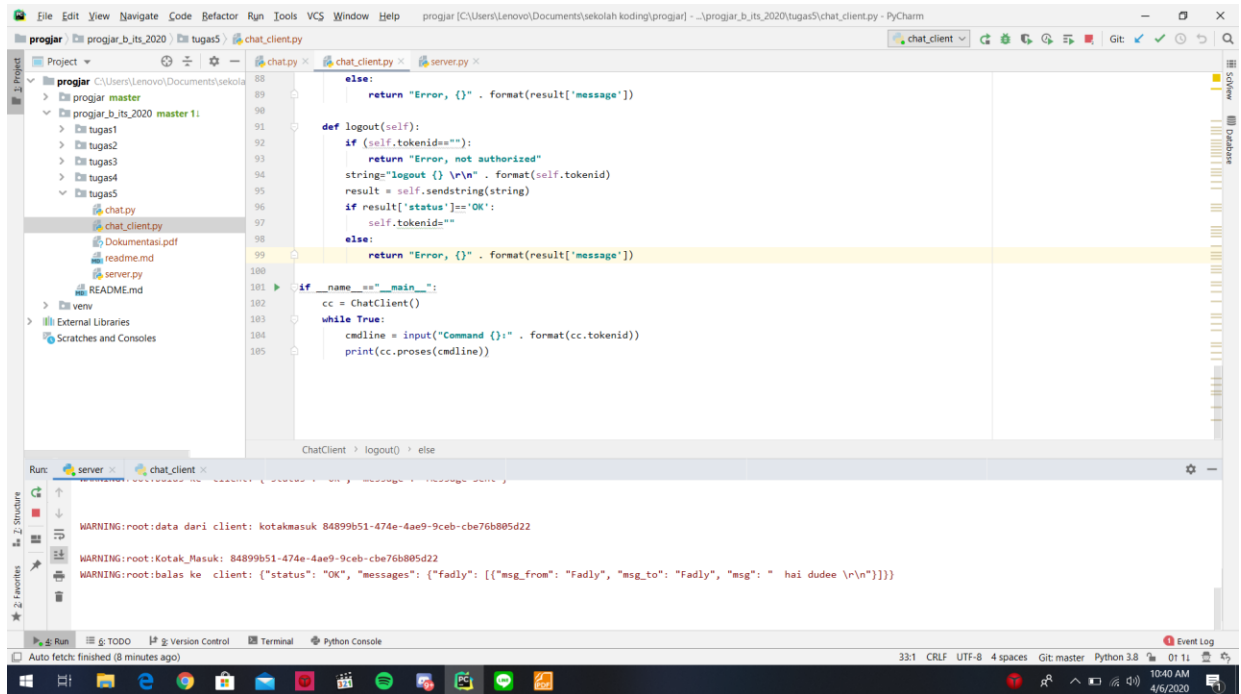
SYARAT : User harus melakukan login terlebih dahulu

OUTPUT : a. Jika respon berhasil (OK), outputnya adalah message  
[usernamefrom] [username to] [message]

b. Jika respon gagal (ERROR), outputnya adalah message  
[message\_error]

## DOKUMENTASI KOTAKMASUK

server.py



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help
progiar [C:\Users\Lenovo\Documents\sekolah koding\progiar] - \progiar_b_its_2020\tugas5\chat_client.py - PyCharm
Project: progiar_b_its_2020 master 1!
  - chat_client.py
  - Dokumentas.pdf
  - readme.md
  - server.py
  - README.md
  - External Libraries
  - Scratches and Consoles
chat_client.py
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
def login(self):
    if (self.tokenid==""):
        return "Error, not authorized"
    string="login {} \n\n".format(self.tokenid)
    result = self.sendstring(string)
    if result['status']=='OK':
        self.tokenid=""
    else:
        return "Error, {}" . format(result['message'])

def logout(self):
    if (self.tokenid!=""):
        return "Error, not authorized"
    string="logout {} \n\n".format(self.tokenid)
    result = self.sendstring(string)
    if result['status']=='OK':
        self.tokenid=""
    else:
        return "Error, {}" . format(result['message'])

if __name__=="__main__":
    cc = ChatClient()
    while True:
        cmdline = input("Command {}:" . format(cc.tokenid))
        print(cc.proses(cmdline))
```

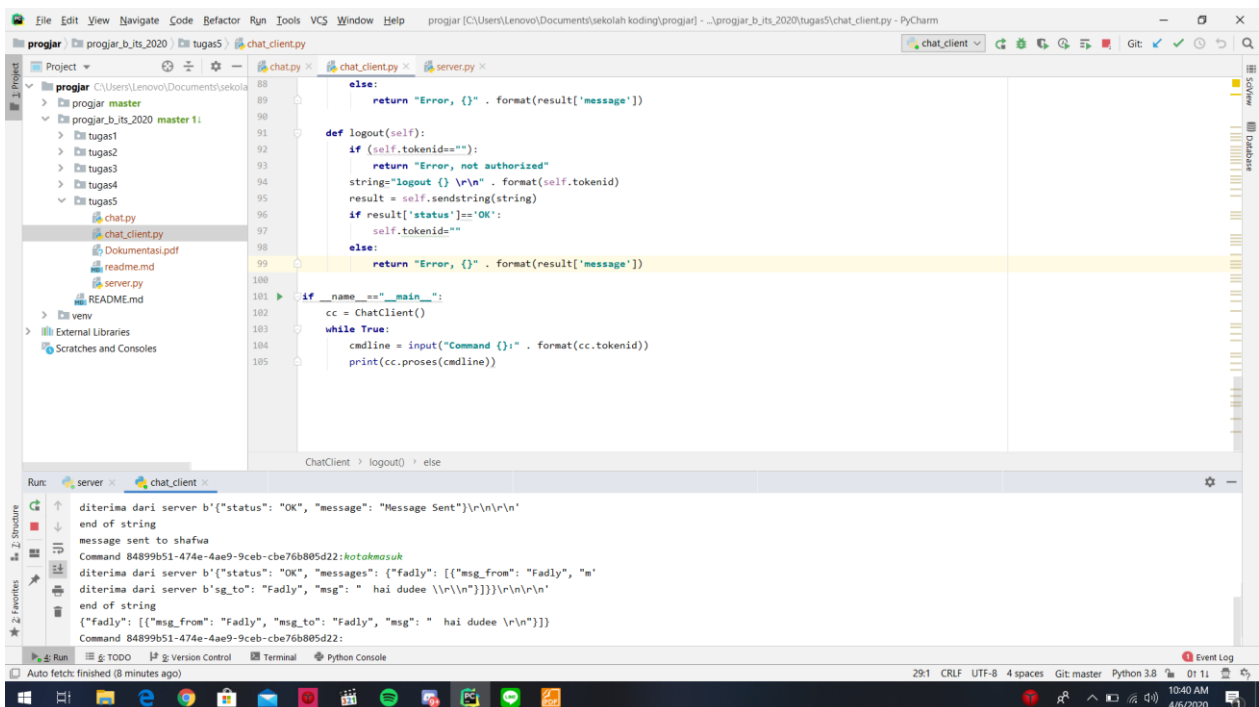
Run: server chat\_client

WARNING:root:data dari client: kotakmasuk 84899b51-474e-4ae9-9ceb-cbe76b805d22

WARNING:root:Kotak\_Masuk: 84899b51-474e-4ae9-9ceb-cbe76b805d22

WARNING:root:balas ke client: ("status": "OK", "messages": [{"msg\_from": "Fadly", "msg\_to": "Fadly", "msg": " hai dudee \r\n\r\n"}])

chat\_client.py



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help
progiar [C:\Users\Lenovo\Documents\sekolah koding\progiar] - \progiar_b_its_2020\tugas5\chat_client.py - PyCharm
Project: progiar_b_its_2020 master 1!
  - chat_client.py
  - Dokumentas.pdf
  - readme.md
  - server.py
  - README.md
  - External Libraries
  - Scratches and Consoles
chat_client.py
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
def login(self):
    if (self.tokenid==""):
        return "Error, not authorized"
    string="login {} \n\n".format(self.tokenid)
    result = self.sendstring(string)
    if result['status']=='OK':
        self.tokenid=""
    else:
        return "Error, {}" . format(result['message'])

def logout(self):
    if (self.tokenid!=""):
        return "Error, not authorized"
    string="logout {} \n\n".format(self.tokenid)
    result = self.sendstring(string)
    if result['status']=='OK':
        self.tokenid=""
    else:
        return "Error, {}" . format(result['message'])

if __name__=="__main__":
    cc = ChatClient()
    while True:
        cmdline = input("Command {}:" . format(cc.tokenid))
        print(cc.proses(cmdline))
```

Run: server chat\_client

diterima dari server b'("status": "OK", "message": "Message Sent")\r\n\r\n'

end of string

message sent to shafwa

Command 84899b51-474e-4ae9-9ceb-cbe76b805d22:kotakmasuk

diterima dari server b'("status": "OK", "messages": [{"msg\_from": "Fadly", "m'}

diterima dari server b'gsto: "Fadly", "msg": " hai dudee \r\n\r\n"}])\r\n\r\n'

end of string

{"msg\_from": "Fadly", "msg\_to": "Fadly", "msg": " hai dudee \r\n\r\n"}]

Command 84899b51-474e-4ae9-9ceb-cbe76b805d22:

#### 4. USER\_ON

**FUNGSI** : Melihat user yang sedang aktif/login

PARAMETER: -

**SYARAT** : User harus melakukan login terlebih dahulu

OUTPUT : a. Jika respon berhasil (OK), outputnya adalah message user aktif:

```
[username_user_aktif]
```

b. Jika respon gagal (ERROR), outputnya adalah message

```
[message_error]
```

# DOKUMENTASI USER\_ON

server.py

[illegible]

## chat\_client.py

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The top toolbar contains icons for saving, running, and other development actions. The left sidebar shows the Project structure with a tree view of the project files and folders. The main editor window displays the code for 'chat\_client.py'. The code includes a 'logout' method and a main loop that handles user input. The bottom console shows the output of the program, including warnings and a successful login message.

```

88         else:
89             return "Error, {}".format(result["message"])
90
91     def logout(self):
92         if (self.tokenid==""):
93             return "Error, not authorized"
94         string="logout {} \n\n".format(self.tokenid)
95         result = self.sendstring(string)
96         if result["status"]=="OK":
97             self.tokenid=""
98         else:
99             return "Error, {}".format(result["message"])
100
101     if __name__ == "__main__":
102         cc = ChatClient()
103         while True:
104             cmdline = input("Command {}: ".format(cc.tokenid))
105             print(cc.proses(cmdline))

```

The bottom console shows the following output:

```

WARNING:root:data dari client: user_on 84899b51-474e-4ae9-9ceb-cbe76b805d22
WARNING:root:LIST USER: 84899b51-474e-4ae9-9ceb-cbe76b805d22
WARNING:root:balas ke client: {"status": "OK", "message": "shafwa, fadly, "}

```

## 5. LOGOUT

FUNGSI : Mengakhiri session pada user

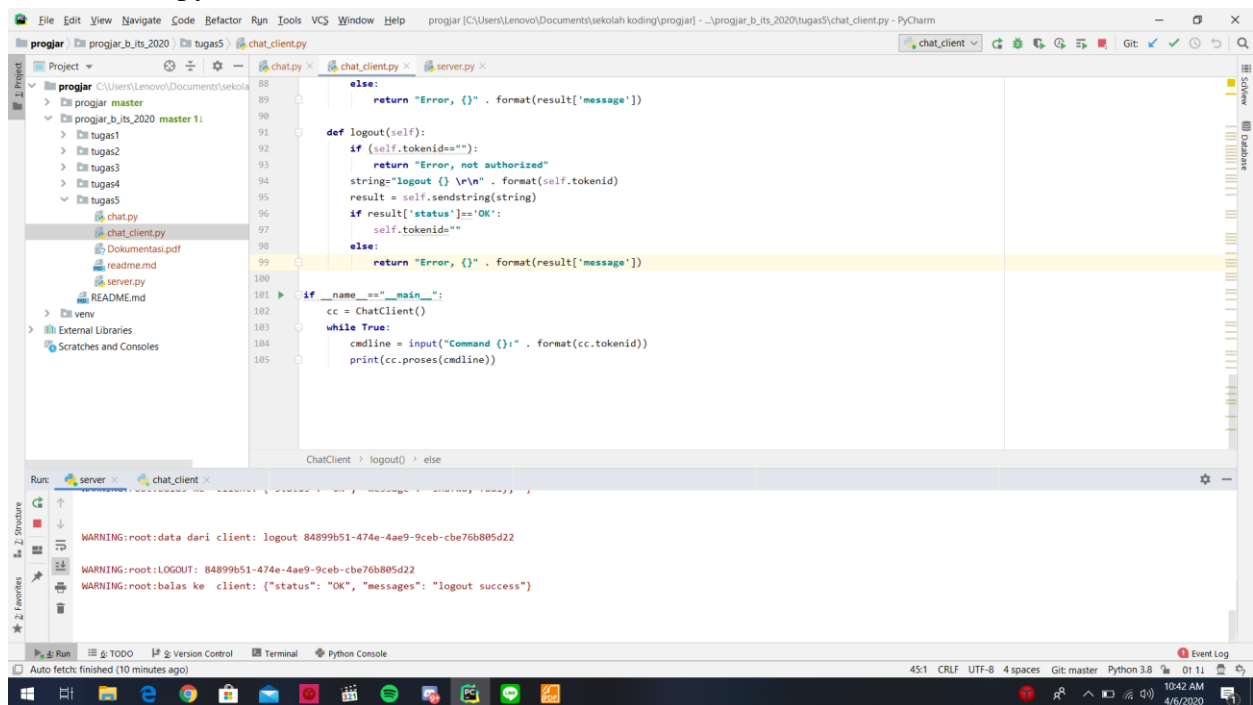
PARAMETER: -

SYARAT : User harus melakukan login terlebih dahulu

OUTPUT : a. Jika respon berhasil (OK), outputnya adalah message logout success  
b. Jika respon gagal (ERROR), outputnya adalah message [message\_error]

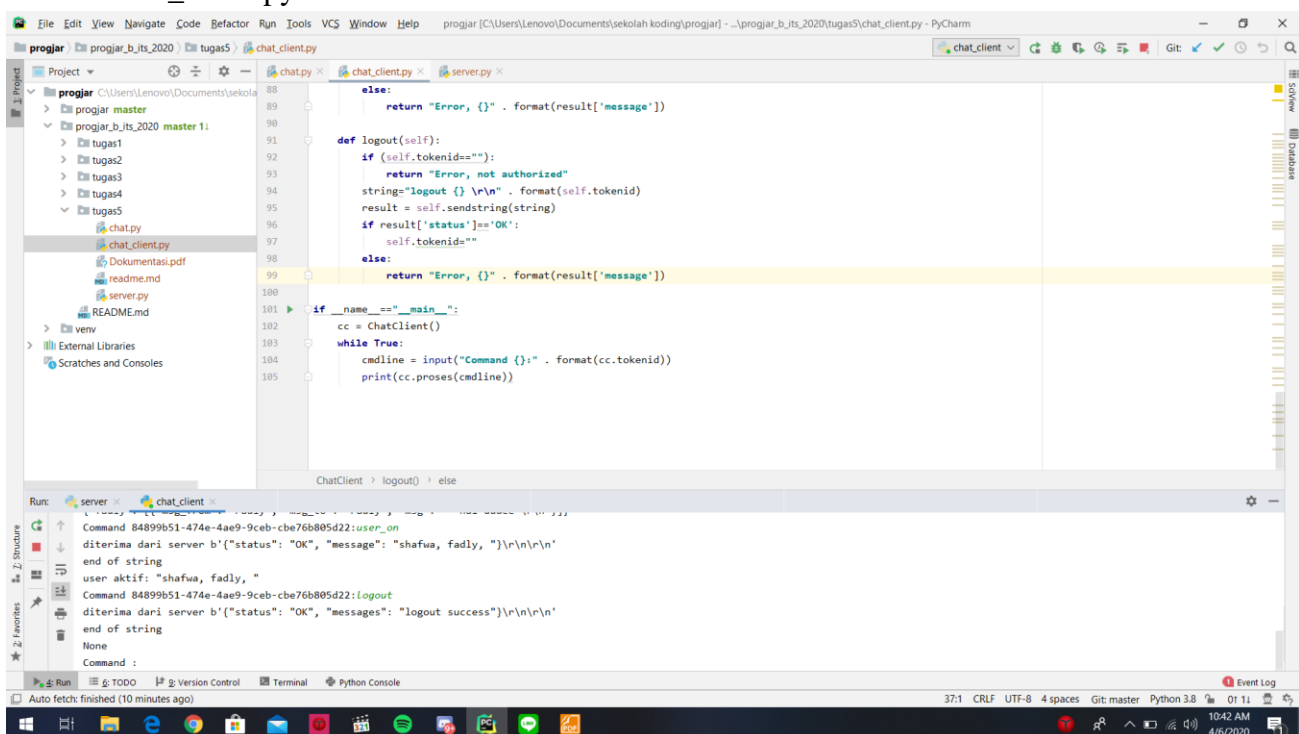
## DOKUMENTASI LOGOUT

server.py



```
88         return "Error, {}".format(result['message'])
89     else:
90         return "Error, {}".format(result['message'])
91
92     def logout(self):
93         if (self.tokenid == ""):
94             return "Error, not authorized"
95         string = "logout {} \n".format(self.tokenid)
96         result = self.sendstring(string)
97         if result['status'] == 'OK':
98             self.tokenid = ""
99             return "logout success"
100         else:
101             return "Error, {}".format(result['message'])
102
103     if __name__ == "__main__":
104         cc = ChatClient()
105         while True:
106             cmdline = input("Command {}: ".format(cc.tokenid))
107             print(cc.proses(cmdline))
```

chat\_client.py



```
88         return "Error, {}".format(result['message'])
89     else:
90         return "Error, {}".format(result['message'])
91
92     def login(self):
93         string = "login {} \n".format(self.tokenid)
94         result = self.sendstring(string)
95         if result['status'] == 'OK':
96             self.tokenid = result['token']
97             return "login success"
98         else:
99             return "Error, {}".format(result['message'])
100
101     if __name__ == "__main__":
102         cc = ChatClient()
103         while True:
104             cmdline = input("Command {}: ".format(cc.tokenid))
105             print(cc.proses(cmdline))
```

