

BÁO CÁO ĐỒ ÁN LẬP TRÌNH SOCKET TẠO RA MỘT WEB SERVER SỬ DỤNG JAVASCRIPT VÀ PYTHON

I.Thành viên tham gia:

STT	Họ và Tên	MSSV
1	Trà Anh Toàn	18120662
2	Võ Thế Minh	18120211
3	Nguyễn Điền Thanh Phong	18120221

II.Bản phân công công việc:

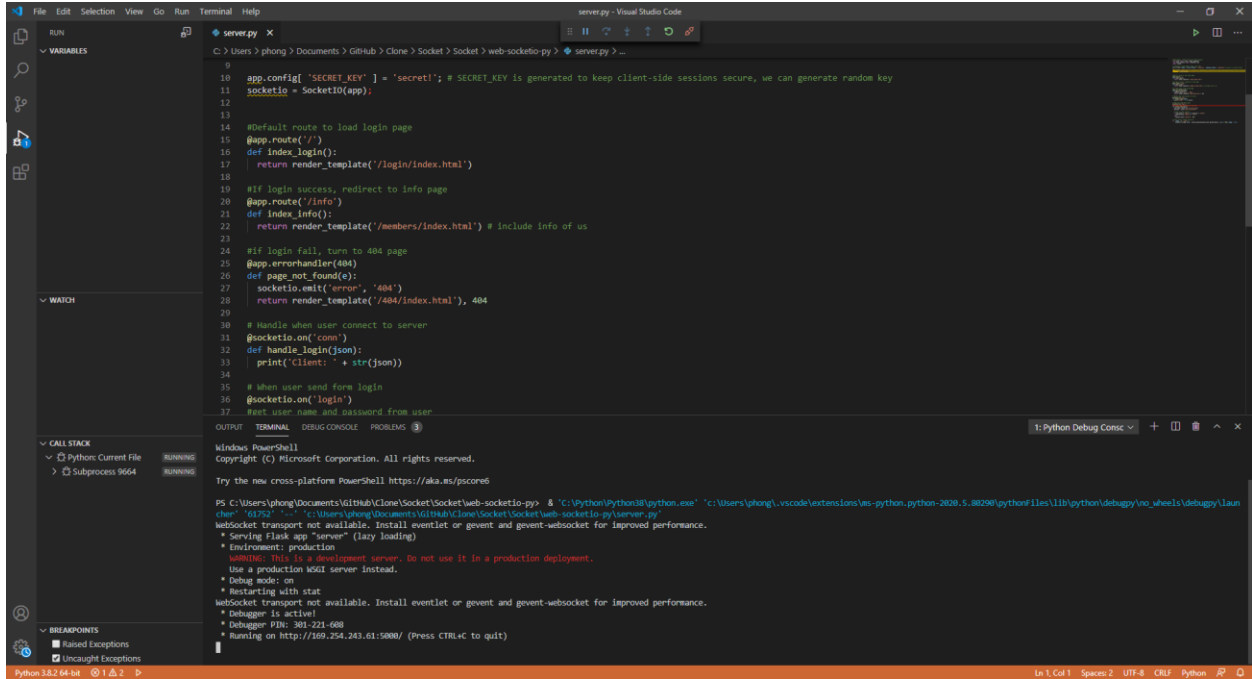
Tên	Công việc	Đánh giá
Trà Anh Toàn	Tìm hiểu thư viện flask, xử lý socket, viết báo cáo	Hoàn thành đúng thời hạn (10/10)
Võ Thế Minh	Xử lý phần đăng nhập, socket ở phía client	Hoàn thành đúng thời hạn (10/10)
Nguyễn Điền Thanh Phong	Thiết kế giao diện các trang login, info, 404, xử lý thiết lập IP server tự động	Hoàn thành đúng thời hạn (10/10)

III. Hướng dẫn thực hiện:

- Ta sử dụng thư viện Flask để render template và thực hiện các xử lý trong chương trình.
- Ta mở kết nối bên phía Server để người dùng có thể truy cập vào, sau đó ta chọn Default Page là Login Page.
- Khi người dùng nhập đúng tài khoản và mật khẩu ta sẽ redirect đến Info Page nếu không ta chuyển đến 404 Page.

IV. Testcase:

- Thiết lập Server thành công



The screenshot shows the Visual Studio Code editor with the file `server.py` open. The code defines a Flask application with routes for login and info, and a WebSocket endpoint for handling login requests. The terminal output shows the server starting successfully on port 5000.

```
10 app.config['SECRET_KEY'] = 'secret!'; # SECRET_KEY is generated to keep client-side sessions secure, we can generate random key
11 socketio = SocketIO(app)
12
13 #Default route to load login page
14 @app.route('/')
15 def index_login():
16     return render_template('/login/index.html')
17
18 #If login success, redirect to info page
19 @app.route('/info')
20 def index_info():
21     return render_template('/members/index.html') # Include info of us
22
23 #If login fail, turn to 404 page
24 @app.errorhandler(404)
25 def page_not_found(e):
26     socketio.emit('error', '404')
27     return render_template('/404/index.html'), 404
28
29
30 # Handle when user connect to server
31 @socketio.on('conn')
32 def handle_login(json):
33     print('Client: ' + str(json))
34
35 # when user send form login
36 @socketio.on('login')
37 def handle_login(data):
38     #Get user name and password from user
39     user_name = data['data']['user_name']
40     password = data['data']['password']
41
42     # Check if login success
43     if user_name == "admin" and password == "admin":
44         socketio.emit('redirect', 'info')
45     else:
46         socketio.emit('redirect', 404)
47
48 # Start server: python server.py
49 if __name__ == '__main__':
50     socketio.run(app, host = socket.gethostname(), port = 5000, debug = True)
```

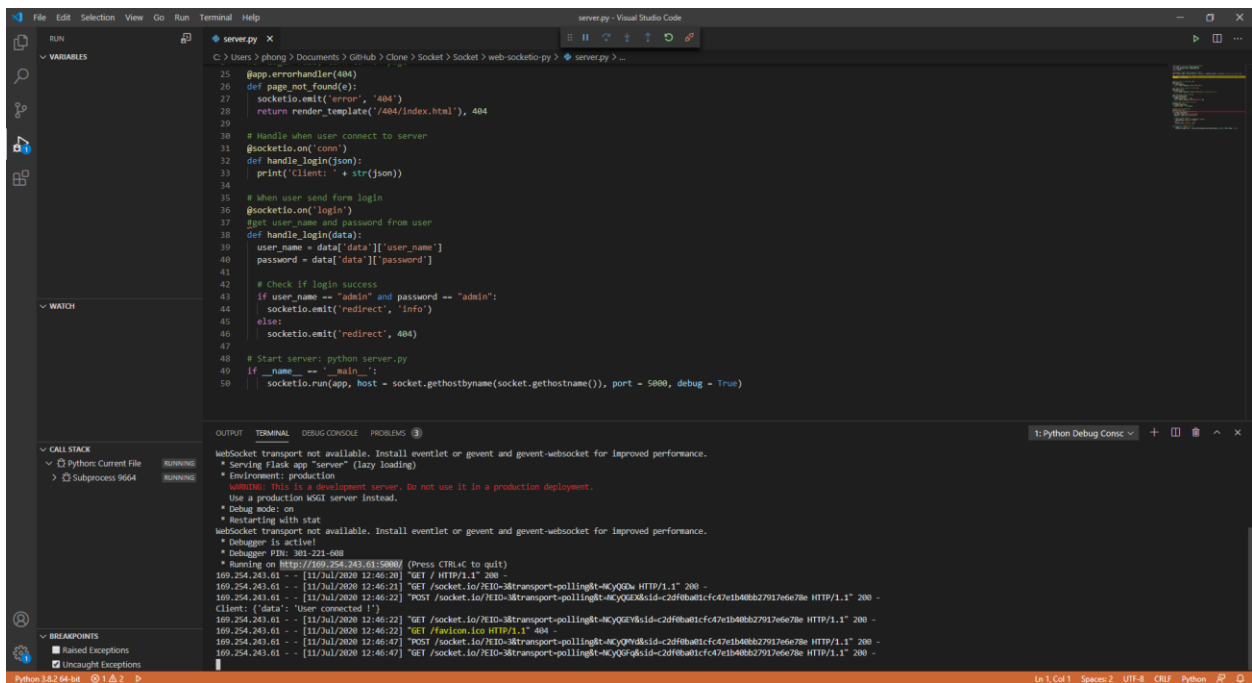
Terminal Output:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/powershell

PS C:\Users\phong\Documents\GitHub\Clone\Socket> socketio web-socketio.py & "C:\Python\Python38\python.exe" "C:\Users\phong\Documents\GitHub\Clone\Socket\socketio.py"
webSocket transport not available. Install eventlet or gevent and gevent-websocket for improved performance.
* Serving Flask app "server" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
webSocket transport not available. Install eventlet or gevent and gevent-websocket for improved performance.
* Debugger is active!
* Debugger PIN: 301-221-088
* Running on http://169.254.243.61:5000/ (Press CTRL+C to quit)
```

- Khi có người dùng truy cập đến Server



The screenshot shows the Visual Studio Code editor with the file `server.py` open. The code defines a Flask application with routes for login and info, and a WebSocket endpoint for handling login requests. The terminal output shows the server starting successfully on port 5000, and a client connection is established.

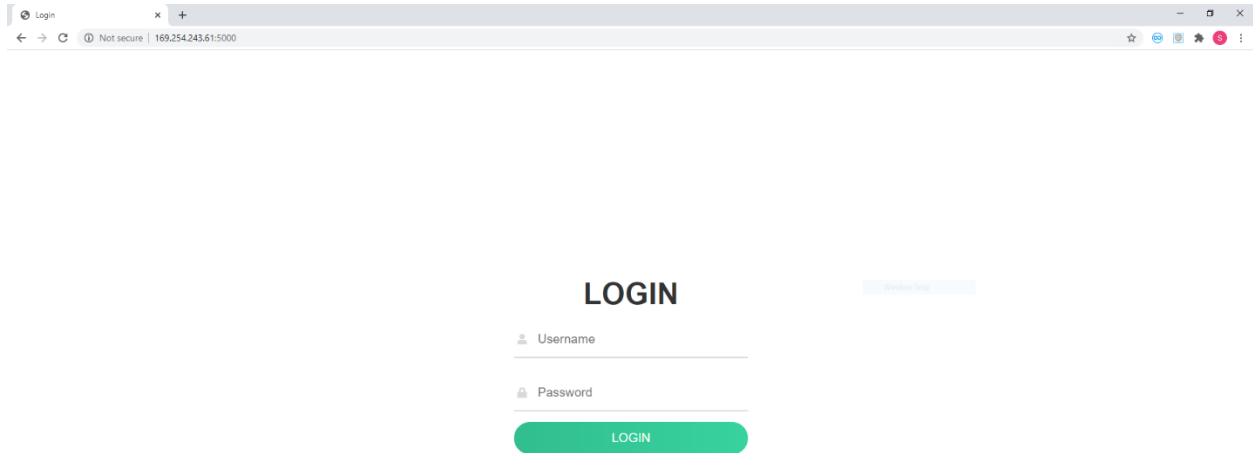
```
25 @app.errorhandler(404)
26 def page_not_found(e):
27     socketio.emit('error', '404')
28     return render_template('/404/index.html'), 404
29
30 # Handle when user connect to server
31 @socketio.on('conn')
32 def handle_login(json):
33     print('Client: ' + str(json))
34
35 # when user send form login
36 @socketio.on('login')
37 def handle_login(data):
38     #Get user name and password from user
39     user_name = data['data']['user_name']
40     password = data['data']['password']
41
42     # Check if login success
43     if user_name == "admin" and password == "admin":
44         socketio.emit('redirect', 'info')
45     else:
46         socketio.emit('redirect', 404)
47
48 # Start server: python server.py
49 if __name__ == '__main__':
50     socketio.run(app, host = socket.gethostname(), port = 5000, debug = True)
```

Terminal Output:

```
webSocket transport not available. Install eventlet or gevent and gevent-websocket for improved performance.
* Serving Flask app "server" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
webSocket transport not available. Install eventlet or gevent and gevent-websocket for improved performance.
* Debugger is active!
* Debugger PIN: 301-221-088
* Running on http://169.254.243.61:5000/ (Press CTRL+C to quit)
169.254.243.61 - [11/Jul/2020 12:46:20] "GET / HTTP/1.1" 200 -
169.254.243.61 - [11/Jul/2020 12:46:21] "GET /socket.io/?EIO=3&transport=polling&st=4c7e1b488a27917e6e78e HTTP/1.1" 200 -
169.254.243.61 - [11/Jul/2020 12:46:22] "POST /socket.io/?EIO=3&transport=polling&st=4c7e1b488a27917e6e78e HTTP/1.1" 200 -
Client: {'data': 'User connected!'}
169.254.243.61 - [11/Jul/2020 12:46:22] "GET /socket.io/?EIO=3&transport=polling&st=4c7e1b488a27917e6e78e HTTP/1.1" 200 -
169.254.243.61 - [11/Jul/2020 12:46:22] "GET /favicon.ico HTTP/1.1" 404 -
169.254.243.61 - [11/Jul/2020 12:46:47] "POST /socket.io/?EIO=3&transport=polling&st=4c7e1b488a27917e6e78e HTTP/1.1" 200 -
169.254.243.61 - [11/Jul/2020 12:46:47] "GET /socket.io/?EIO=3&transport=polling&st=4c7e1b488a27917e6e78e HTTP/1.1" 200 -
```

- Giao diện trang đăng nhập:

- Khi người dùng nhập đúng tài khoản và mật khẩu (**admin - admin**)



LOGIN

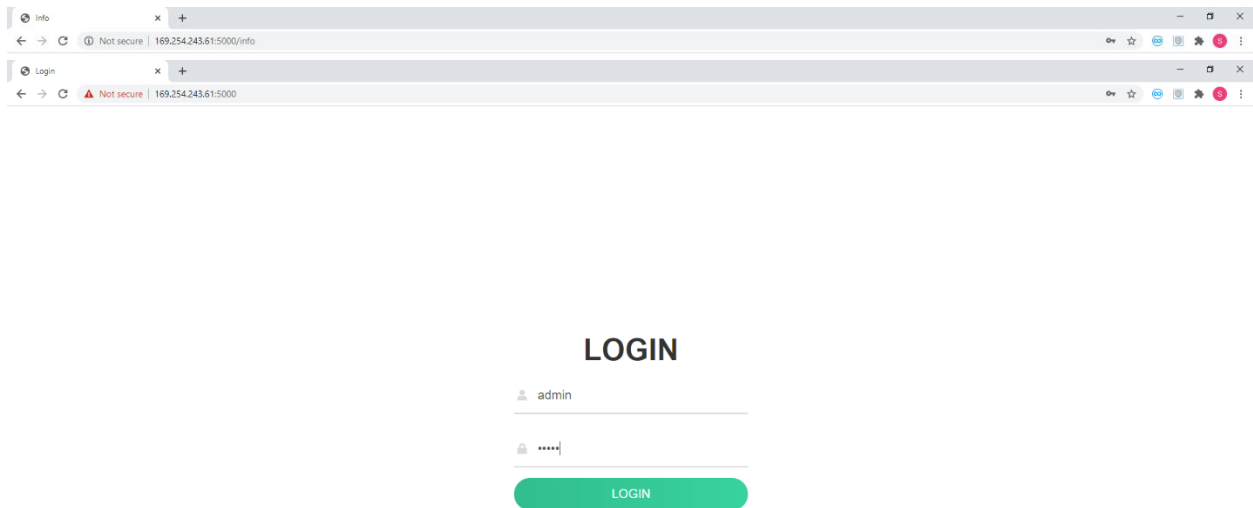
☐ Remember Me

Username

Password

LOGIN

Khi đó người dùng sẽ được chuyển đến trang Thành viên khi đăng nhập thành công



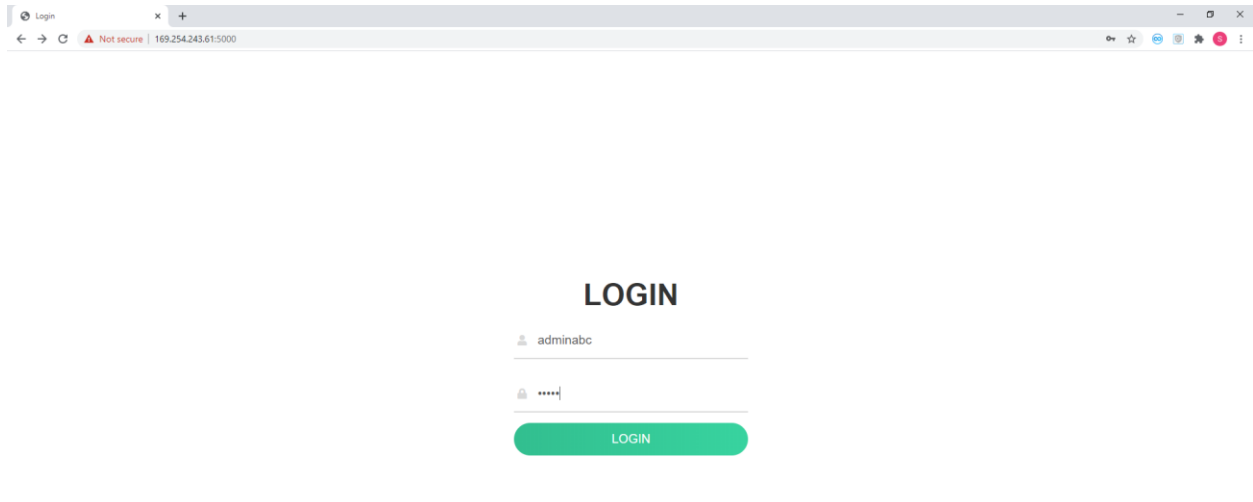
LOGIN

☐ Remember Me

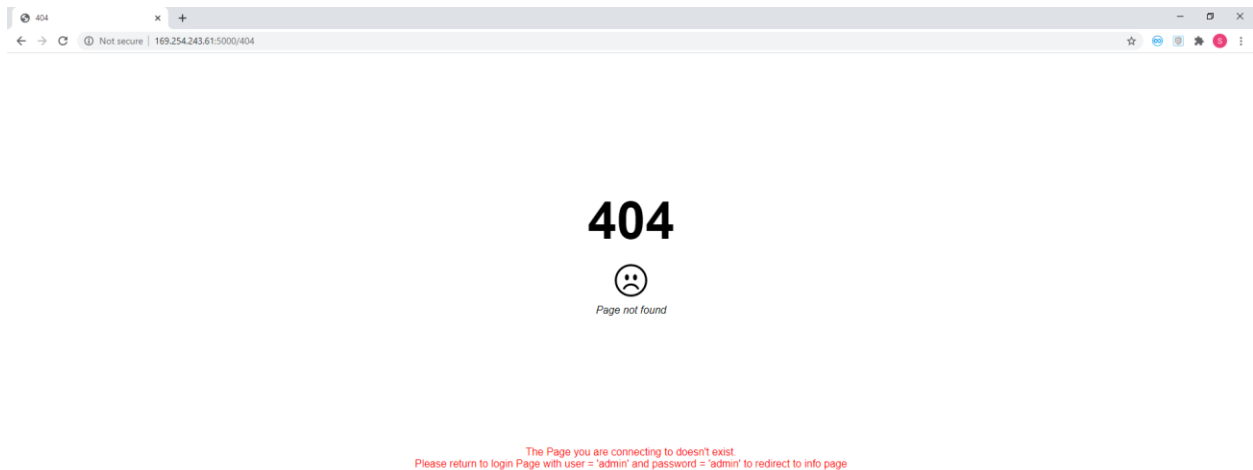
admin

LOGIN

- Khi người dùng nhập sai



Trang báo lỗi sẽ xuất hiện



V. Tài liệu tham khảo:

<https://blog.miguelgrinberg.com/post/easy-websockets-with-flask-and-gevent>

<https://www.w3schools.com/>

<https://flask.palletsprojects.com/en/1.1.x/quickstart/>

<https://topdev.vn/blog/json-la-gi/>

<https://flask.palletsprojects.com/en/0.12.x/tutorial/templates/>