Flask Web Application Server

COP4635 Sys & Net II Project 3
Spring 2024

OVERVIEW

This project aims to showcase effective HTTP method handling, specifically focusing on GET and POST requests, using the Python Flask web framework. The goal is to create a straightforward web calculator application that receives **mathematical expressions** (text) from users and **computes the results** upon clicking the **'Calculate'** button. Additionally, you'll implement a feature allowing users to view the **history** of all calculations performed from the release of your web application (you might have restarted your server several times but the user should be able to see all calculations performed when the server is up and running). The history page should provide an option **to go back to calculator page**(home). The historical data will be stored in a **text file acting as the database**.

As part of the project, you'll develop a Flask web server with the mentioned functionalities. This server will also **handle the HTTP error '404' gracefully**, presenting a customized message and a **link back to the home page**. This process will illustrate how to serve various web pages (HTML pages and routes) to clients, specifically web browsers.

Furthermore, you'll create an **API** (with endpoint '***api/history***') for the calculation history, presenting information in **JSON format** (see attached output screenshot for the structure of json). This API serves both a web browser client and lays the foundation for a future programmatic client. The project provides a practical demonstration of serving web pages, handling errors, and creating APIs using Flask.

WORKING IN TEAMS

Teams of up to two students can work on this project. If you wish to work in teams, you must make sure that you inform the instructor about your team ASAP by responding to the email from the instructor about this, and at least a week before the deadline. Last minute information about any teams to the instructor will not be entertained and such groups will not be allowed.

ENVIRONMENT

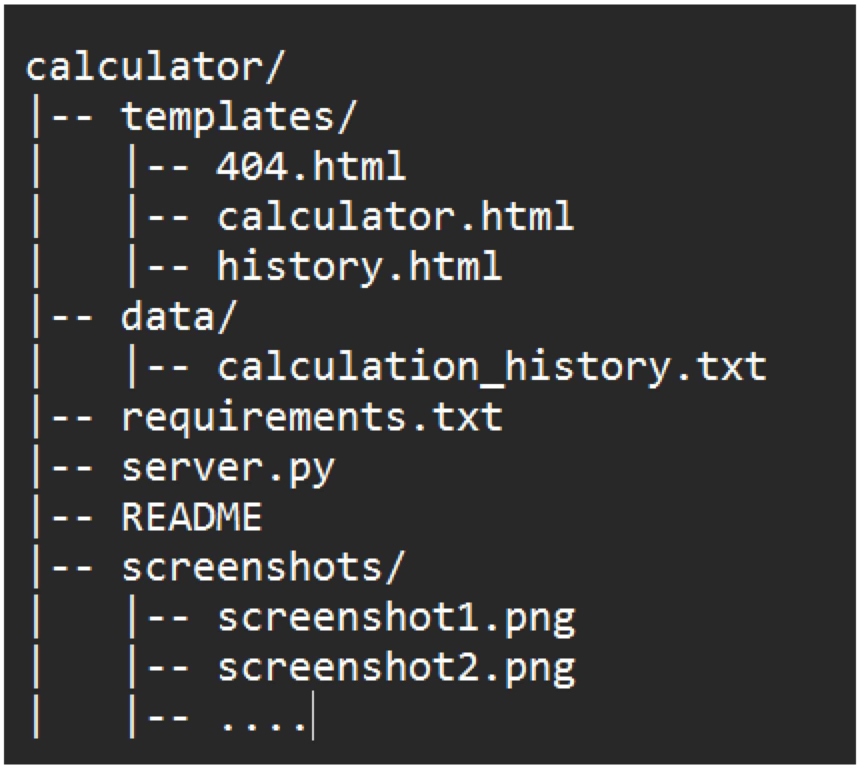
1. Python >= 3.8 and Flask == 3.0.2 (latest as of this project was created) , it is recommended to work on your project by creating python virtual environment.

IMPLEMENTATION

Program the web server in python using flask framework. The server will listen on a valid port. Make sure you use a port number from the valid port number range (60001 – 60099) if you use CS department's SSH server, you may use any available port (ex: 8080, 8000, 5000) if you are running on your machine. When client (browser) request for an html file (route) that does not exists at the server, then your server should respond with small, but descriptive error message and link to home page of your application that should be received by the client and displayed. Your HTTP server should be verbose, and display all activity going on at the server.

Your server should respond to any standard browser program that is chosen to run as the client. If a user makes a request for the HTML page files while the server is running, the server should send the file to the browser if the request is valid. In case of the valid request/s the browser would receive the HTML file and display it on the screen. Make sure that your HTTP server program does not exit unless forced by the tester to stop. The server program should be coded by you so as to let the user perform more calculations in the same way through the browser client, and not exit immediately.

PROJECT FOLDER STRUCTURE AND FILES



```
calculator/
|-- templates/
|    |-- 404.html
|    |-- calculator.html
|    |-- history.html
|-- data/
|    |-- calculation_history.txt
|-- requirements.txt
|-- server.py
|-- README
|-- screenshots/
|    |-- screenshot1.png
|    |-- screenshot2.png
|    |-- ....
```

- Calculator --- to store all the files related to this project
    - Templates --- folder to store all html files (flask uses jinja template format)
        - 404.html --- to customize 'not found error message' and add link to home page
        - calculator.html --- index (or home) page for your application
        - history.html --- this is used to display the list of calculations performed
    - data
        - calculation_history.txt --- to store the calculations performed in a file
    - requirements.txt --- specify the python dependencies for your project (here you should add flask dependency as "**Flask==3.0.2**")
    - server.py --- your server program and main logic goes here
    - README --- contains names of all project members, directions on running your code indicating usage and examples, python and flask versions and any other relevant details like an analysis if project wasn't completed, and what part was accomplished.
    - screenshots --- folder to store all your screenshots of the features you implemented

DELIVERABLES & EVALUATION

Your project submission should follow the instructions below. Any submissions that do not follow the stated requirements will not be graded.

1. Follow the submission requirements of the instructor as published on eLearning under the Content area.
2. You should submit a zip file with all the files mentioned in "**Project folder structure and files**" above.

Your program will be evaluated according to the steps shown below. Notice that the instructor/GA will not fix your syntax errors. However, they will make grading quick!

1. Program runs as expected. If errors occur during program execution, the instructor/GA will not fix your code to get it to run. The project will be given zero/partial points.
2. Program documentation and code structure.
    a. The source code must be properly documented and the code must be structured to enhance readability of the code.
    b. Each source code file must include a header that describes the purpose of the source code file, the name of the programmer(s), the date when the code was written, and the course for which the code was developed.

3. Perform several test-runs with input of the grader's own choosing. At a minimum, the test runs should address the following questions.
    a. Does the server program properly run, and is the communication between the server process and a standard browser process evident?
    b. Can valid and invalid file links be tested with the browser as a client for your server?
    c. Is there ample and evident feedback on what is going on in the server program while they are running, and when they exit? For example, when the server gets a connection request from the client, and when it proceeds to respond (or cannot respond) to a request does the server display what is going on in the server

Keep in mind that documentation of source code is an essential part of computer programming. The better your code is documented, the better it can be maintained and reused. If you do not include comments in your source code, points will be deducted. You should refactor your code to make it more manageable.

DUE DATE

The project is due as indicated by the Dropbox for this project in eLearning. Upload your complete solution to the dropbox. You can use the shared VM for creating and testing your code. Please mention in the comments at the top of your code if it tested perfectly with the shared VM and/or on the UWF-CS SSH server. Else, if you used WSL, please do that in the code as well as in the README file.

TESTING

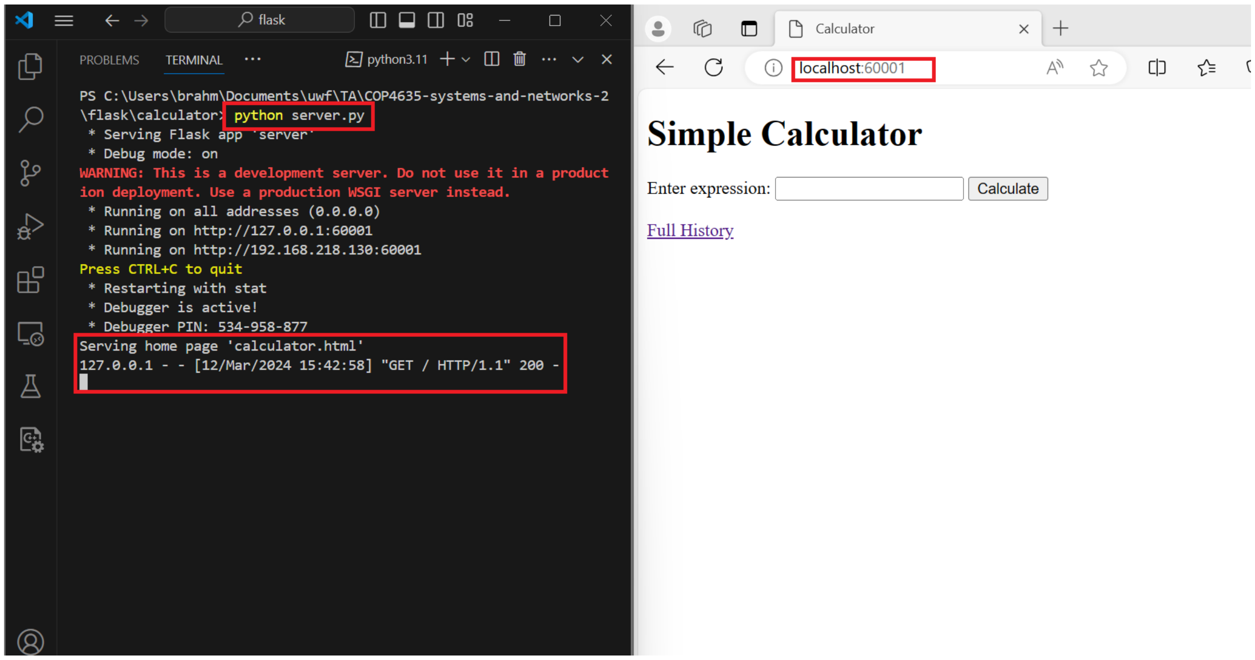
Your solution needs to run on any machine that has the environment mentioned in 'ENVIRONMENT' section above. If you use different versions mention this in your README file. If you plan to use CS department's SSH server, for security reasons, most of the ports on the servers have been blocked from communication. Ports that are open for communication between the public servers' range in values from 60,000 to 60,099. Some students have found it easier to develop utilizing WSL or their favorite code editors (ex: visual studio code) for the project rather than the ssh server and/or the provided VM image. If your program works in WSL or another environment which you used, please mention this in your README file and that is how we will grade them.

GRADING

This project is worth 100 points total. Please remember that there will be deductions if your code has crashes, has endless loops or is otherwise, poorly documented or organized. We will not be able to grant any points for code that does not compile.

EXPECTED OUTPUT SCREENSHOTS (OBSERVE THE HIGHLIGHTED PORTION):

1. Home page



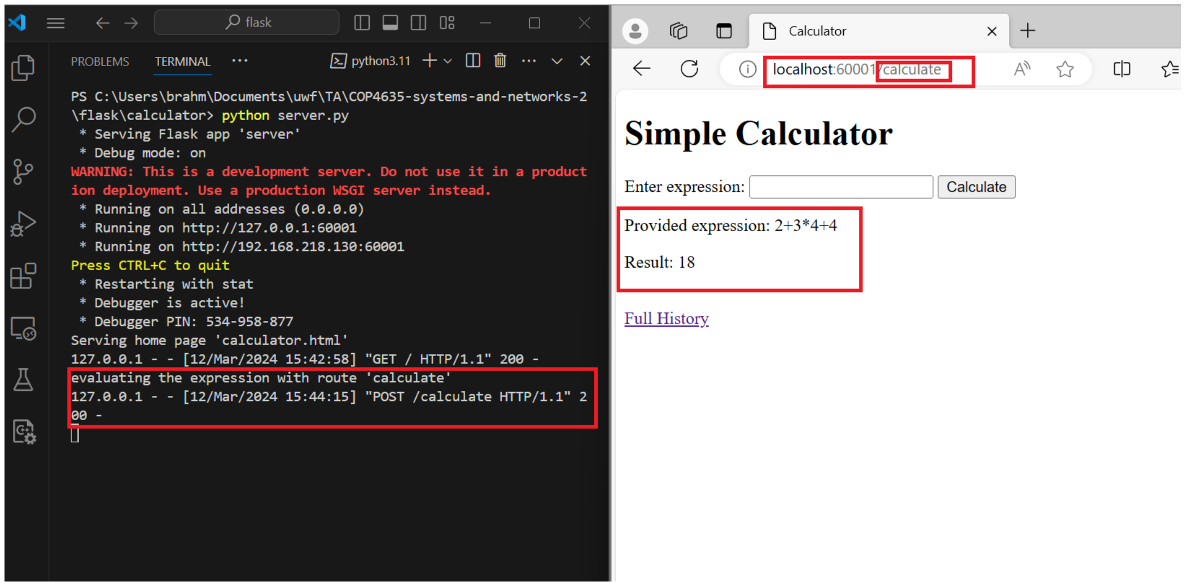
```
PS C:\Users\brahm\Documents\uwf\TA\COP4635-systems-and-networks-2
\flask\calculator> python server.py
 * Serving Flask app 'server'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a product
ion deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:60001
 * Running on http://192.168.218.130:60001
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 534-958-877
Serving home page 'calculator.html'
127.0.0.1 - - [12/Mar/2024 15:42:58] "GET / HTTP/1.1" 200 -
```

Simple Calculator

Enter expression: [ ] Calculate

Full History

2. First calculation



```
PS C:\Users\brahm\Documents\uwf\TA\COP4635-systems-and-networks-2
\flask\calculator> python server.py
 * Serving Flask app 'server'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a product
ion deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:60001
 * Running on http://192.168.218.130:60001
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 534-958-877
Serving home page 'calculator.html'
127.0.0.1 - - [12/Mar/2024 15:42:58] "GET / HTTP/1.1" 200 -
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:44:15] "POST /calculate HTTP/1.1" 2
00 -
```

localhost:60001/calculate

Simple Calculator

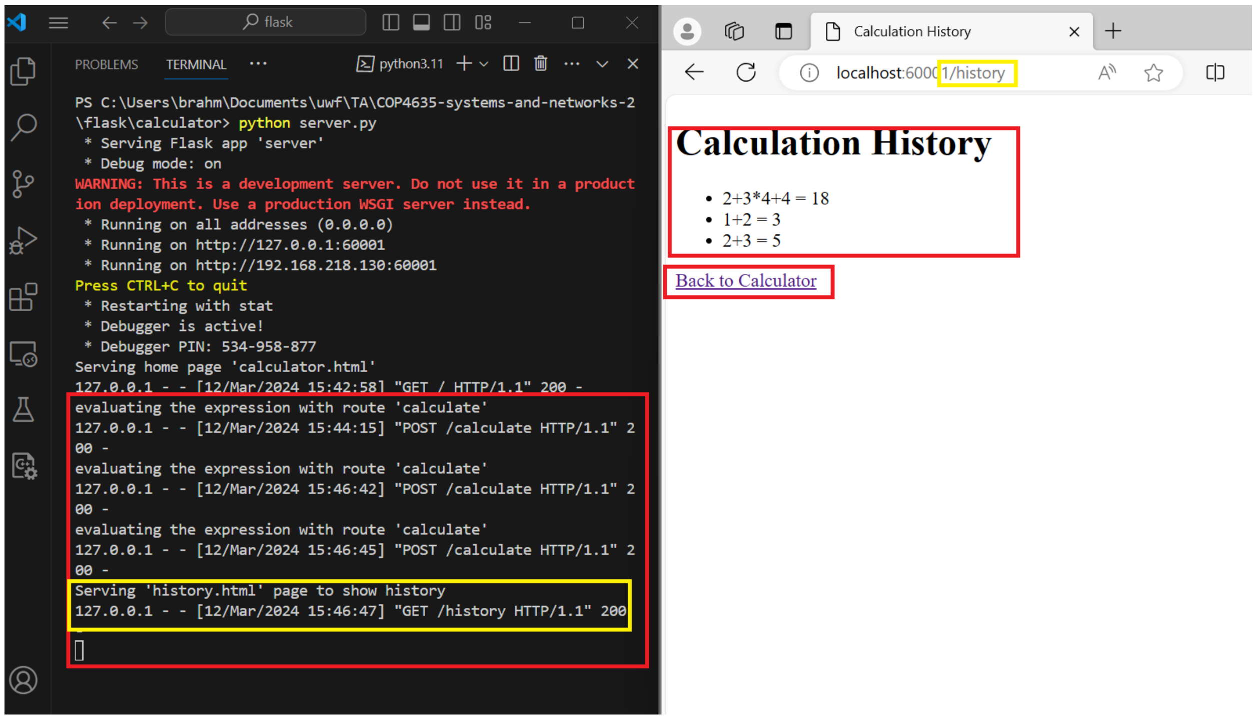
Enter expression: Calculate

Provided expression: 2+3*4+4

Result: 18

Full History

3. After performing several calculations, displaying history by clicking on ‘**Full History**’ link



```
PS C:\Users\brahm\Documents\uwf\TA\COP4635-systems-and-networks-2
\flask\calculator> python server.py
 * Serving Flask app 'server'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a product
ion deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:60001
 * Running on http://192.168.218.130:60001
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 534-958-877
Serving home page 'calculator.html'
127.0.0.1 - - [12/Mar/2024 15:42:58] "GET / HTTP/1.1" 200 -
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:44:15] "POST /calculate HTTP/1.1" 2
00 -
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:46:42] "POST /calculate HTTP/1.1" 2
00 -
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:46:45] "POST /calculate HTTP/1.1" 2
00 -
Serving 'history.html' page to show history
127.0.0.1 - - [12/Mar/2024 15:46:47] "GET /history HTTP/1.1" 200
```

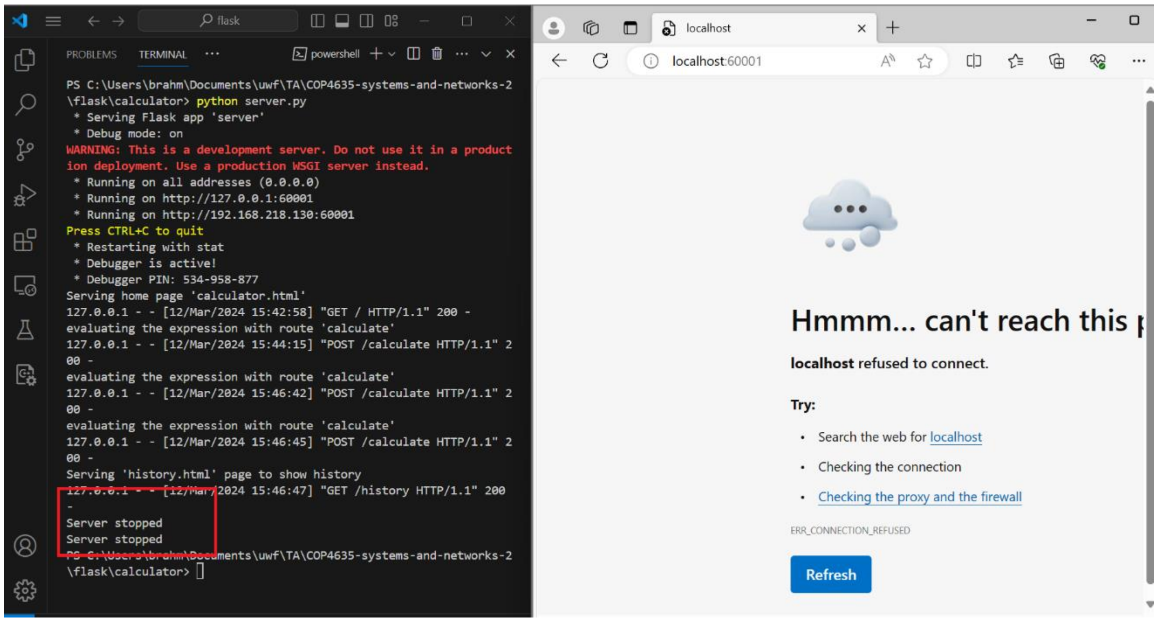
localhost:60001/history

Calculation History

- 2+3*4+4 = 18
- 1+2 = 3
- 2+3 = 5

Back to Calculator

4. Server stopped



```
PS C:\Users\brahm\Documents\uwf\TA\COP4635-systems-and-networks-2
\flask\calculator> python server.py
 * Serving Flask app 'server'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a product
ion deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:60001
 * Running on http://192.168.218.130:60001
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 534-958-877
Serving home page 'calculator.html'
127.0.0.1 - - [12/Mar/2024 15:42:58] "GET / HTTP/1.1" 200 -
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:44:15] "POST /calculate HTTP/1.1" 2
00 -
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:46:42] "POST /calculate HTTP/1.1" 2
00 -
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:46:45] "POST /calculate HTTP/1.1" 2
00 -
Serving 'history.html' page to show history
127.0.0.1 - - [12/Mar/2024 15:46:47] "GET /history HTTP/1.1" 200
-
Server stopped
Server stopped
PS C:\Users\brahm\Documents\uwf\TA\COP4635-systems-and-networks-2
\flask\calculator>
```

localhost:60001

Hmmm... can't reach this

localhost refused to connect.

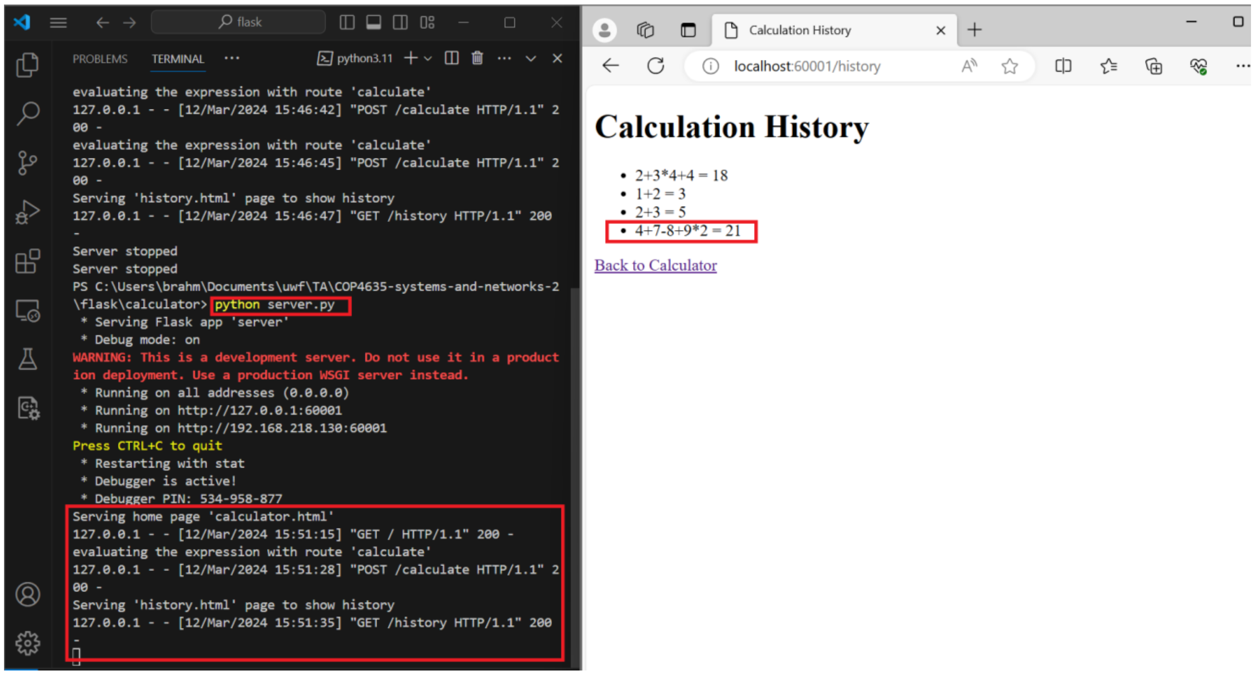
Try:

- Search the web for localhost
- Checking the connection
- Checking the proxy and the firewall

ERR_CONNECTION_REFUSED

Refresh

5. Start the server again and perform couple of calculations and show history (this time it should show previous + calculations performed after the second start of the server)



```
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:46:42] "POST /calculate HTTP/1.1" 2
00 -
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:46:45] "POST /calculate HTTP/1.1" 2
00 -
Serving 'history.html' page to show history
127.0.0.1 - - [12/Mar/2024 15:46:47] "GET /history HTTP/1.1" 200
-
Server stopped
Server stopped
PS C:\Users\brahm\Documents\uwf\TA\COP4635-systems-and-networks-2
\flask\calculator> python server.py
 * Serving Flask app 'server'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a product
ion deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:60001
 * Running on http://192.168.218.130:60001
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 534-958-877
Serving home page 'calculator.html'
127.0.0.1 - - [12/Mar/2024 15:51:15] "GET / HTTP/1.1" 200 -
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:51:28] "POST /calculate HTTP/1.1" 2
00 -
Serving 'history.html' page to show history
127.0.0.1 - - [12/Mar/2024 15:51:35] "GET /history HTTP/1.1" 200
-
```

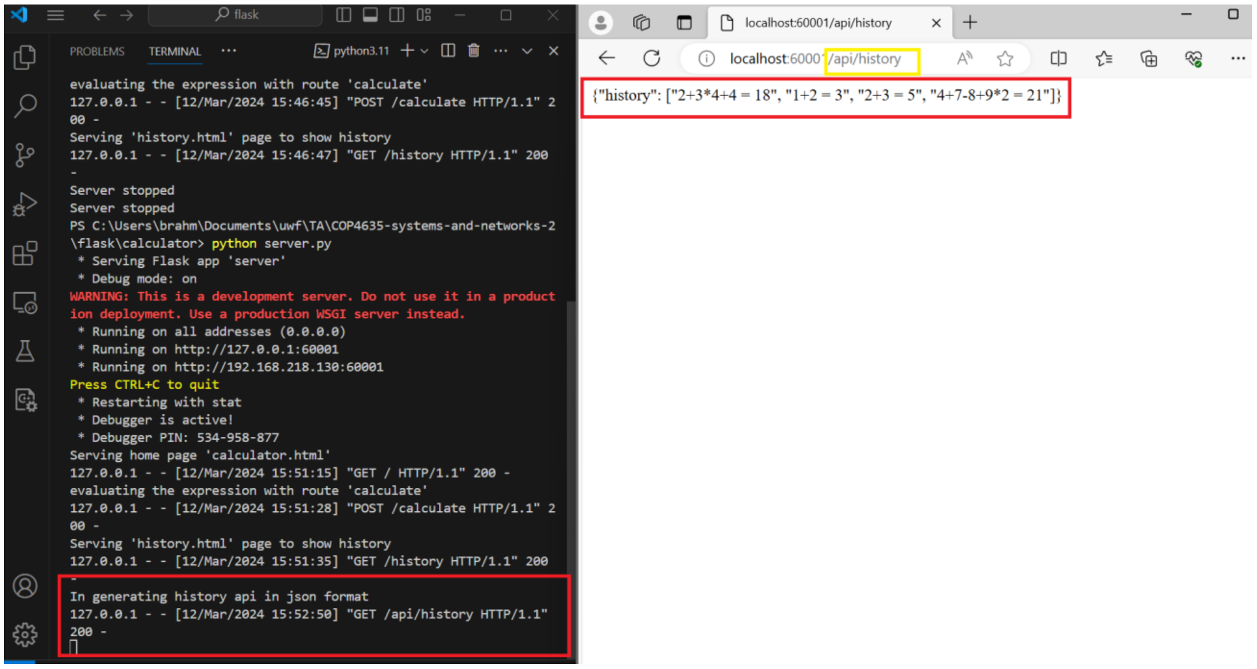
localhost:60001/history

Calculation History

- 2+3*4+4 = 18
- 1+2 = 3
- 2+3 = 5
- 4+7-8+9*2 = 21

Back to Calculator

6. Json history api endpoint

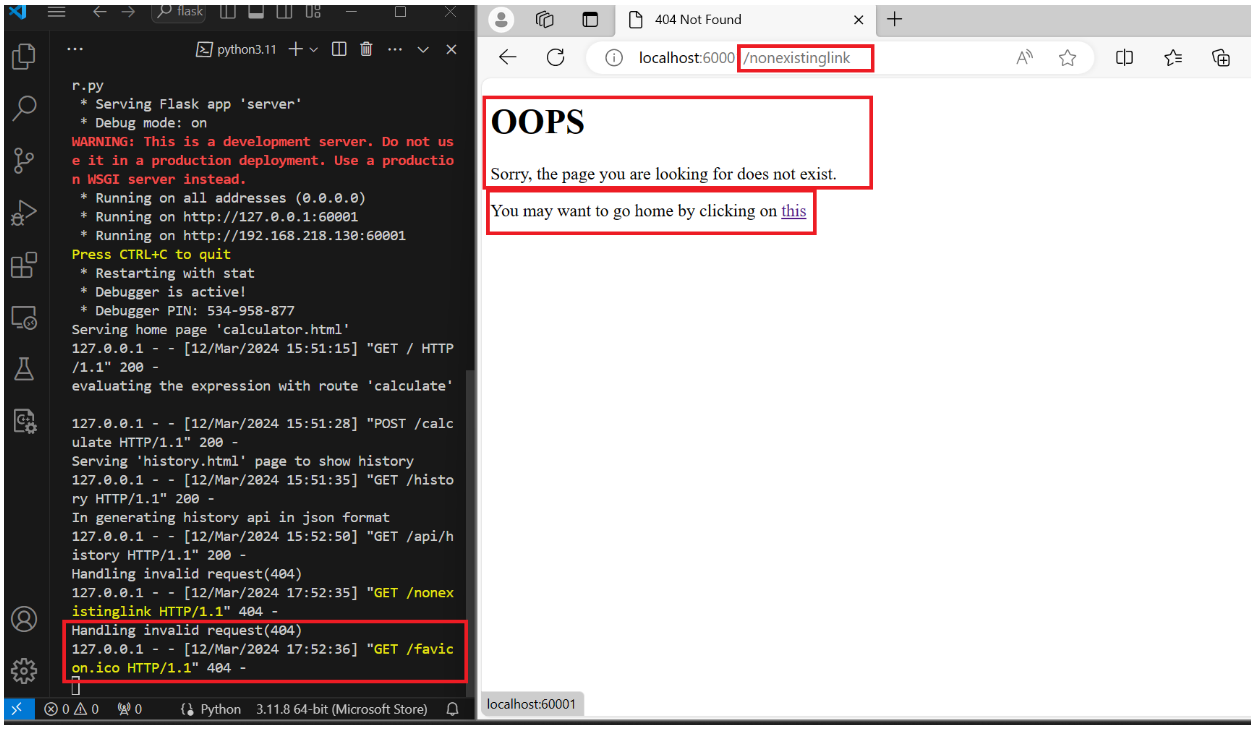


```
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:46:45] "POST /calculate HTTP/1.1" 2
00 -
Serving 'history.html' page to show history
127.0.0.1 - - [12/Mar/2024 15:46:47] "GET /history HTTP/1.1" 200
-
Server stopped
Server stopped
PS C:\Users\brahm\Documents\uwf\TA\COP4635-systems-and-networks-2
\flask\calculator> python server.py
 * Serving Flask app 'server'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a product
ion deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:60001
 * Running on http://192.168.218.130:60001
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 534-958-877
Serving home page 'calculator.html'
127.0.0.1 - - [12/Mar/2024 15:51:15] "GET / HTTP/1.1" 200 -
evaluating the expression with route 'calculate'
127.0.0.1 - - [12/Mar/2024 15:51:28] "POST /calculate HTTP/1.1" 2
00 -
Serving 'history.html' page to show history
127.0.0.1 - - [12/Mar/2024 15:51:35] "GET /history HTTP/1.1" 200
-
In generating history api in json format
127.0.0.1 - - [12/Mar/2024 15:52:50] "GET /api/history HTTP/1.1"
200 -
```

localhost:60001/api/history

```
{"history": ["2+3*4+4 = 18", "1+2 = 3", "2+3 = 5", "4+7-8+9*2 = 21"]}
```

7. 404 not found custom message and link to home page



```
r.py
 * Serving Flask app 'server'
 * Debug mode: on
WARNING: This is a development server. Do not us
e it in a production deployment. Use a productio
n WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:60001
 * Running on http://192.168.218.130:60001
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 534-958-877
Serving home page 'calculator.html'
127.0.0.1 - - [12/Mar/2024 15:51:15] "GET / HTTP
/1.1" 200 -
evaluating the expression with route 'calculate'

127.0.0.1 - - [12/Mar/2024 15:51:28] "POST /calc
ulate HTTP/1.1" 200 -
Serving 'history.html' page to show history
127.0.0.1 - - [12/Mar/2024 15:51:35] "GET /histo
ry HTTP/1.1" 200 -
In generating history api in json format
127.0.0.1 - - [12/Mar/2024 15:52:50] "GET /api/h
istory HTTP/1.1" 200 -
Handling invalid request(404)
127.0.0.1 - - [12/Mar/2024 17:52:35] "GET /nonex
istinglink HTTP/1.1" 404 -
Handling invalid request(404)
127.0.0.1 - - [12/Mar/2024 17:52:36] "GET /favic
on.ico HTTP/1.1" 404 -
```

404 Not Found

localhost:6000/nonexistinglink

OOPS

Sorry, the page you are looking for does not exist.

You may want to go home by clicking on this