Project Description

- This project implemented a web server supported by HTTP 1.1. When multiple simultaneous TCP connections are requested by clients, the server retrieves the requested file or error page to the clients by multi-threaded approach. Once a connection is idle for 60 seconds, the connection is terminated by the server automatically.
- 200, 404, 403, 400 status code is supported, and related headers will show. HTML, TXT, JPG and GIF files are supported. Get method is the only method supported in the server.
- A simple heuristic has been implemented to control the timeout of the idle connection based on the number of existing active connections.
  Number of threads < 100, period of time for idle connections: 5 minutes
  100 <= Number of threads < 200, period of time for idle connections: 1 minutes
  Number of threads >= 200, close idle connections

Uploaded Files

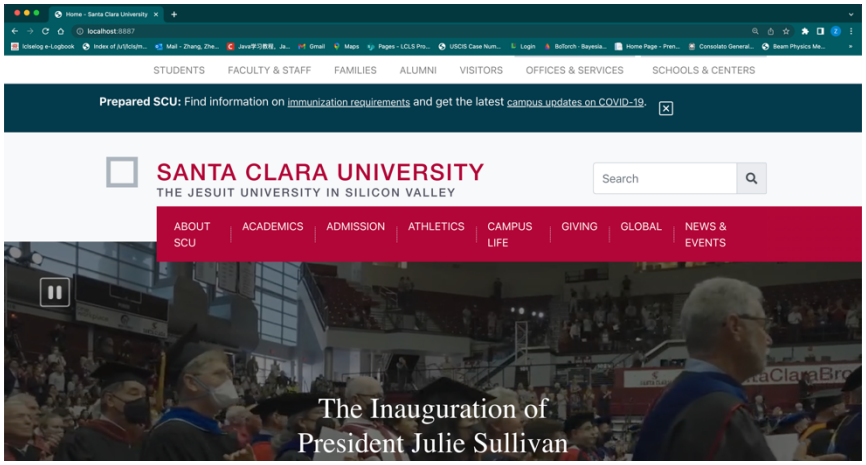
- server_zhen.py
- readme.pdf
- a folder including test files
  - jpgtest.jpg
  - giftest.gif
  - index.html
  - txttest.txt
  - permission.txt

Instructions for Running the Program

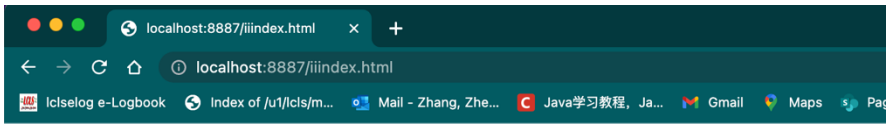
- Please make sure the computer has a Python 3.0 environment.
- Start a terminal and change directory to the server_zhen.py
- Run the server by as
  python server_zhen.py -document_root "/Users/zhen/documents/files" -port 8887
  “-document_root” and “-port” are used to set the supported files and port number.
- Simply, you can run the server with default settings as
  python server_zhen.py
  The default root document is ./files and default port number is 8887

Summaries of running results

1. Normal request 200 for index.html
   Input “localhost:8887/index.html”. SCU web page is displayed.

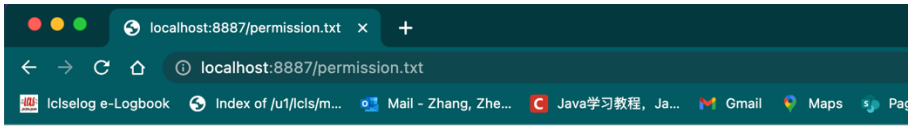


2. Default page when path is not specified
   Input “localhost:8887”. SCU web page is displayed.
3. 404 not found error
   Input ”localhost:8887/iiindex.html”. 404 error page is displayed. Other file names which are not included in the “files” document also apply here.




```
ERROR 404: Page Not Found
The requested URL was not found in this server
```

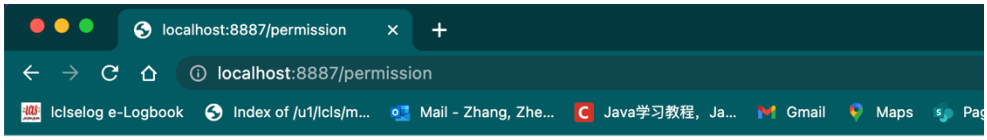
4. 403 forbidden request error
   The file permission.txt has been set as not readable.
   Input ”localhost:8887/ permission.txt”. 404 error page is displayed.




```
ERROR 403: Forbidden Access
Access to this resource on this server is denied.
```

5. 400 bad request error
   Input “localhost:8887/permission”. 400 error page is displayed. This server simulates the scenario where the file without a suffix is defined as a bad request. Other file names

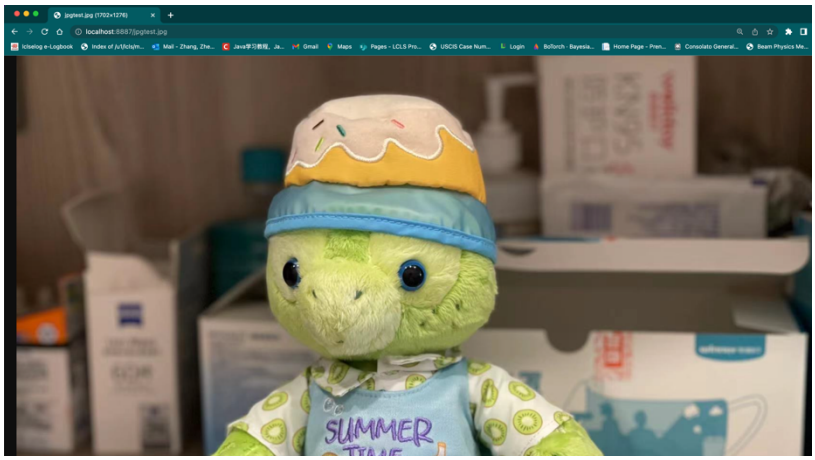
without suffix also apply.



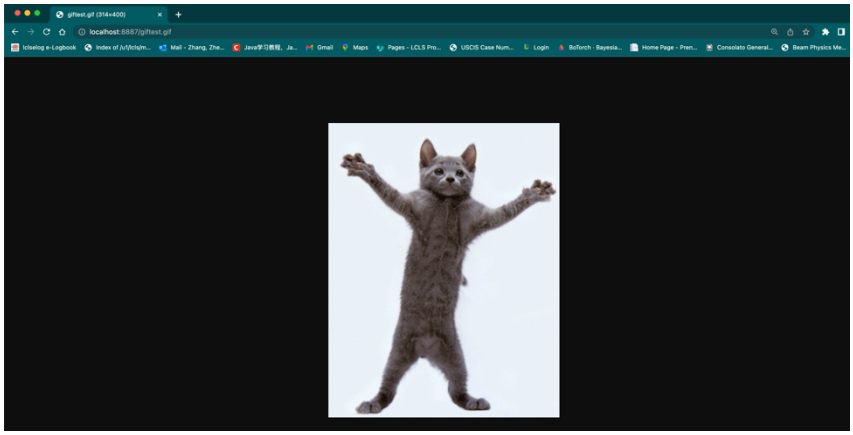

```
ERROR 400: Request Error
Your client has issued a malformed or illegal request.
```

6. Other types of files

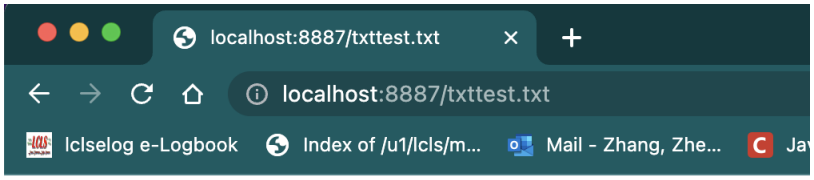
JPG:



GIF:

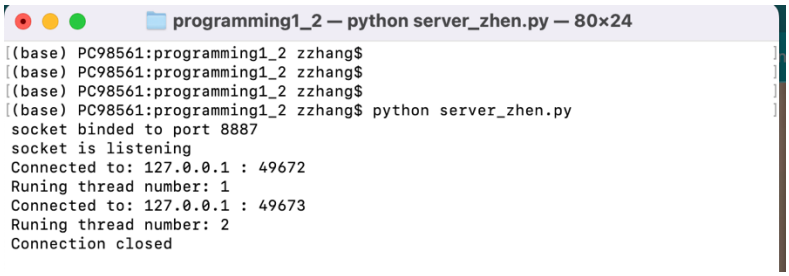


TXT:




```
Have a nice day!
```

7. Terminal display during processing requests
   Display socket start/listen/close status
   Display connection established/closed status
   Display existing number of threads



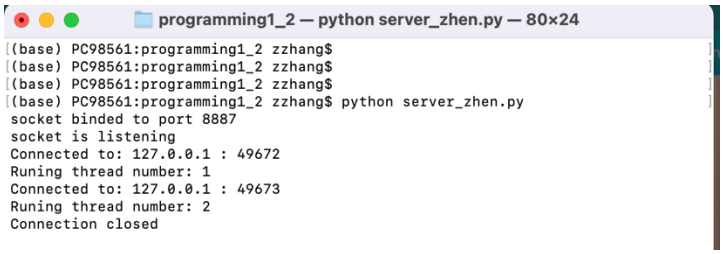
Multi-threaded

- Use multi-thread to spawn new thread for each connection with a Python module named ***threading***

```
# Using multithread to process client request
thread = threading.Thread(target=server_process, args=(connection_socket,numthread))
thread.start()
numthread = threading.activeCount() - 1
print(f"Runing thread number: {numthread}")
```

Close connection after every request

- In the code, we use “break” in while loop to close the connection after every request



Extra credit:

- A simple heuristic has been implemented to control the timeout of the idle connection based on the number of existing active connections.
  Number of threads < 100, period of time for idle connections: 5 minutes
  100 <= Number of threads < 200, period of time for idle connections: 1 minutes
  Number of threads >= 200, close idle connections
- This function is achieved by setting the number of threads as the argument of the processing request function (server_process) to determine the timeout of idle connections.