

SAN JOSE STATE UNIVERSITY

# **CMPE 148 - Computer Network I**

## **LAB 2 REPORT**

### **Web Server Lab**

Instructor: Prof. Andrew Bond

Group: Ladybugs

Do Tran

Quyen Nguyen

Mai Vo Xuan Tran

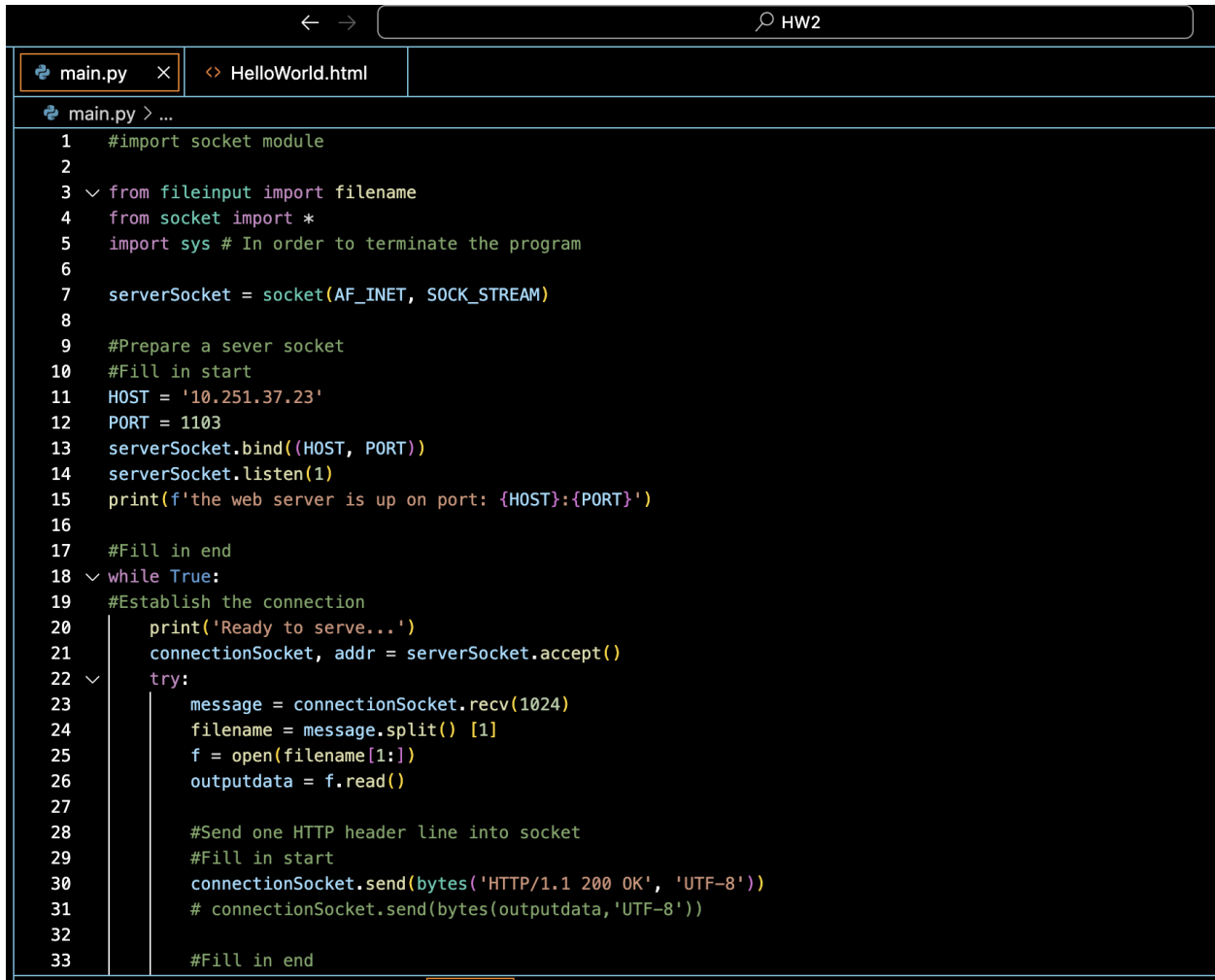
My Nguyen

Due: 10/2023

## Code

Below you will find the skeleton code for the Web server. You are to complete the skeleton code. The places where you need to fill in code are marked with **#Fill in start** and **#Fill in end**. Each place may require one or more lines of code.

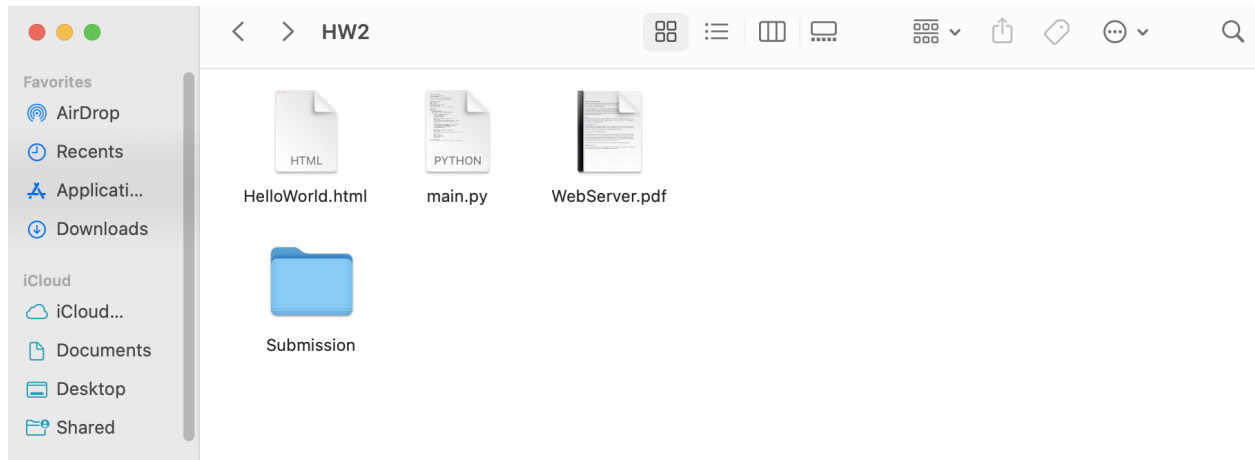
**Link for GitHub:** <https://github.com/nhutdh1103/HW2.git>

A screenshot of a code editor window. The title bar shows navigation arrows and a search icon with the text 'HW2'. The editor has two tabs: 'main.py' (active) and 'HelloWorld.html'. The code in 'main.py' is a Python script for a web server. It starts with imports for 'socket', 'fileinput', and 'sys'. It then sets up a server socket, binds it to '10.251.37.23' on port '1103', and starts listening. A while loop begins with a comment '#Establish the connection' and a print statement 'Ready to serve...'. Inside the loop, it accepts a connection, reads a message, splits it to get a filename, opens the file, and reads its content. It then sends an 'HTTP/1.1 200 OK' response. There are three placeholder comments: '#Fill in start' at line 10, '#Fill in end' at line 17, and another '#Fill in end' at line 33. The code ends with a print statement showing the host and port.

```
1  #import socket module
2
3  from fileinput import filename
4  from socket import *
5  import sys # In order to terminate the program
6
7  serverSocket = socket(AF_INET, SOCK_STREAM)
8
9  #Prepare a sever socket
10 #Fill in start
11 HOST = '10.251.37.23'
12 PORT = 1103
13 serverSocket.bind((HOST, PORT))
14 serverSocket.listen(1)
15 print(f'the web server is up on port: {HOST}:{PORT}')
16
17 #Fill in end
18 while True:
19     #Establish the connection
20     print('Ready to serve...')
21     connectionSocket, addr = serverSocket.accept()
22     try:
23         message = connectionSocket.recv(1024)
24         filename = message.split()[1]
25         f = open(filename[1:])
26         outputdata = f.read()
27
28         #Send one HTTP header line into socket
29         #Fill in start
30         connectionSocket.send(bytes('HTTP/1.1 200 OK', 'UTF-8'))
31         # connectionSocket.send(bytes(outputdata, 'UTF-8'))
32
33         #Fill in end
```

## Running the Server

Put an HTML file (e.g., HelloWorld.html) in the same directory that the server is in. Run the server program. Determine the IP address of the host that is running the server (e.g., 128.238.251.26).



## Running the server

```
nina@DoTran HW2 % /usr/bin/python3 "/Users/nina/Documents/CMPE 148/HW2/main.py"
the web server is up on port: 10.251.37.23:1103
Ready to serve...
```

Server receives the request for “HelloWorld.html” from client

```
message = connectionSocket.recv(1024)
```

And return HelloWorld.html to client

```
filename = message.split() [1]
f = open(filename[1:])
outputdata = f.read()

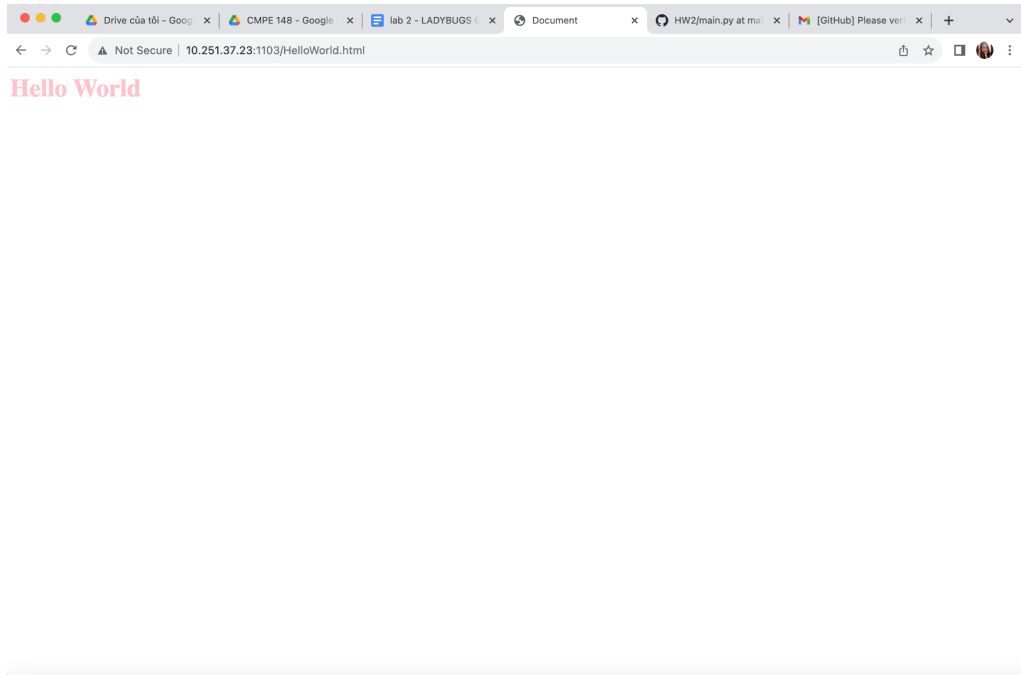
#Send one HTTP header line into socket
#Fill in start
connectionSocket.send(bytes('HTTP/1.1 200 OK', 'UTF-8'))
# connectionSocket.send(bytes(outputdata, 'UTF-8'))

#Fill in end
#Send the content of the requested file to the client
for i in range(0, len(outputdata)):
    connectionSocket.send(outputdata[i].encode())
    connectionSocket.send("\r\n".encode())
connectionSocket.close()
```

From another host, open a browser and provide the corresponding URL.

<http://10.251.37.23:1103/HelloWorld.html>

Client gets the html file and Status code: 200 OK



If we request the non-existing file, the server sends back 404 code. Status code: 404 Not found

<http://10.251.37.23:1103/HelloWorld1.html>

- Code

```
except IOError:

    #Send response message for file not found

    #Fill in start

    connectionSocket.send(bytes('HTTP/1.1 404 Not Found', 'UTF-8'))

    #Fill in end

    #Close client socket

    #Fill in start

    connectionSocket.close()

    #Fill in end
```

- Result:

