



Report

Assignment 2



Author: ANAS KWEFATI, LOÏC
GALLAND

Email: ak223wd , lg222sv

Semester: Spring 2020

Area: Computer Science

Course code: 1DV701

Contents

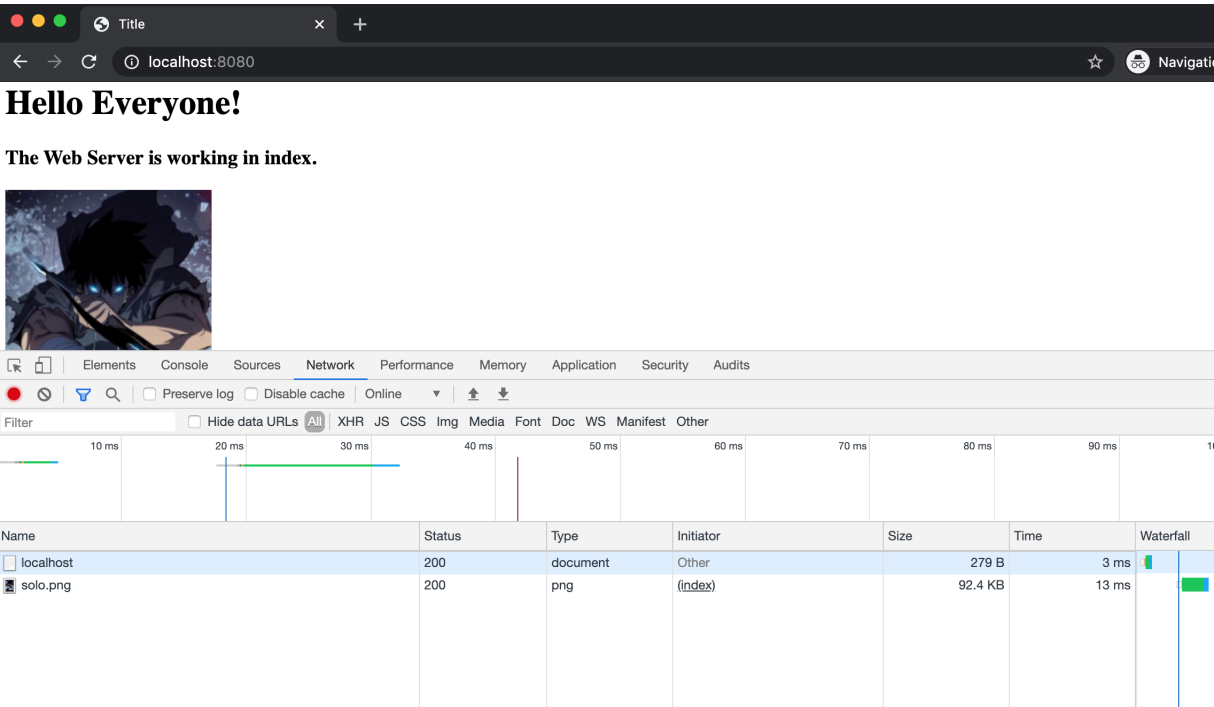
1	Summary	1
2	Problem 1	1
3	Problem 2 - Response codes	4
3.1	302 Found	4
3.2	403 Forbidden	5
3.3	404 Not Found	5
3.4	500 Internal Server Error	6
4	Problem 2 - VG1 POST	7
4.1	Discussion	8
5	Problem 2 - VG2 PUT	8
5.1	Discussion	9
5.2	Difference between POST and PUT	9
6	Problem 3	10
7	Problem 4	10

1 Summary

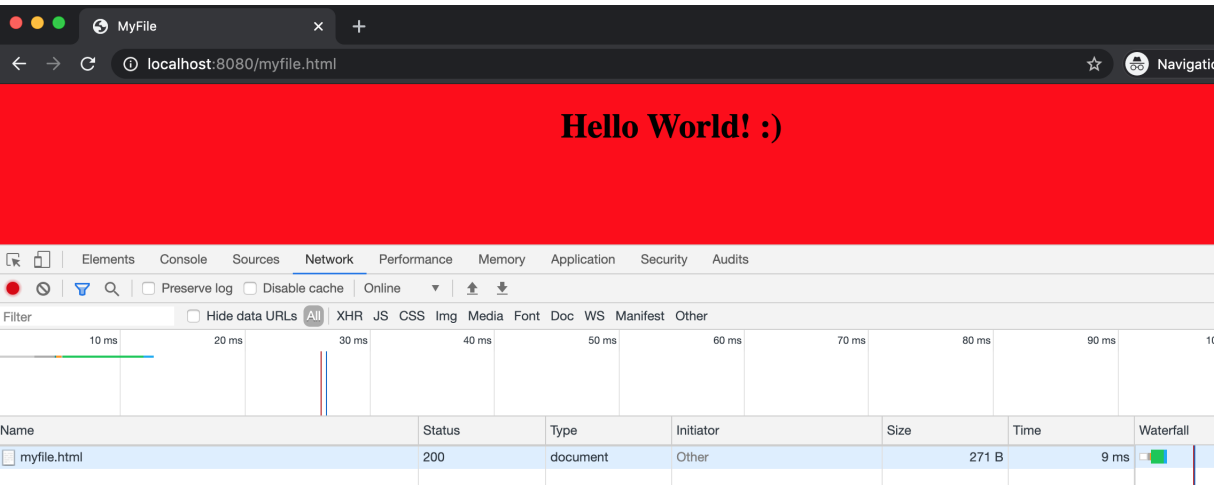
This below is a summary of the percentage for the workload: ak223wd= 55%, lg222sv = 45 %

2 Problem 1

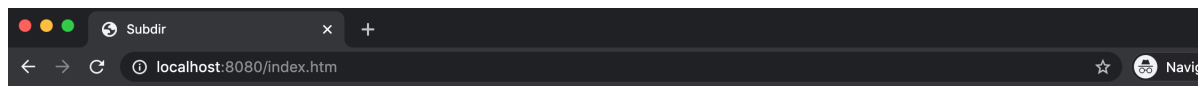
All of the screenshots below are to show all the different GET methods.



The Webserver works in index and successfully GET the image and html text.

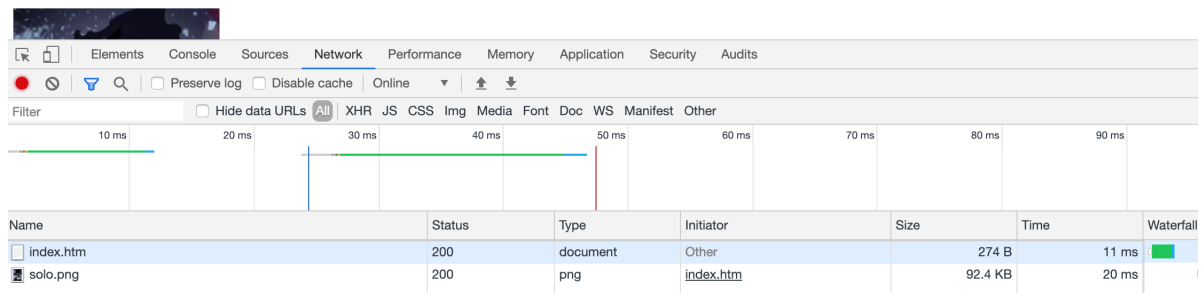


GET an html file by giving the path in the URL.

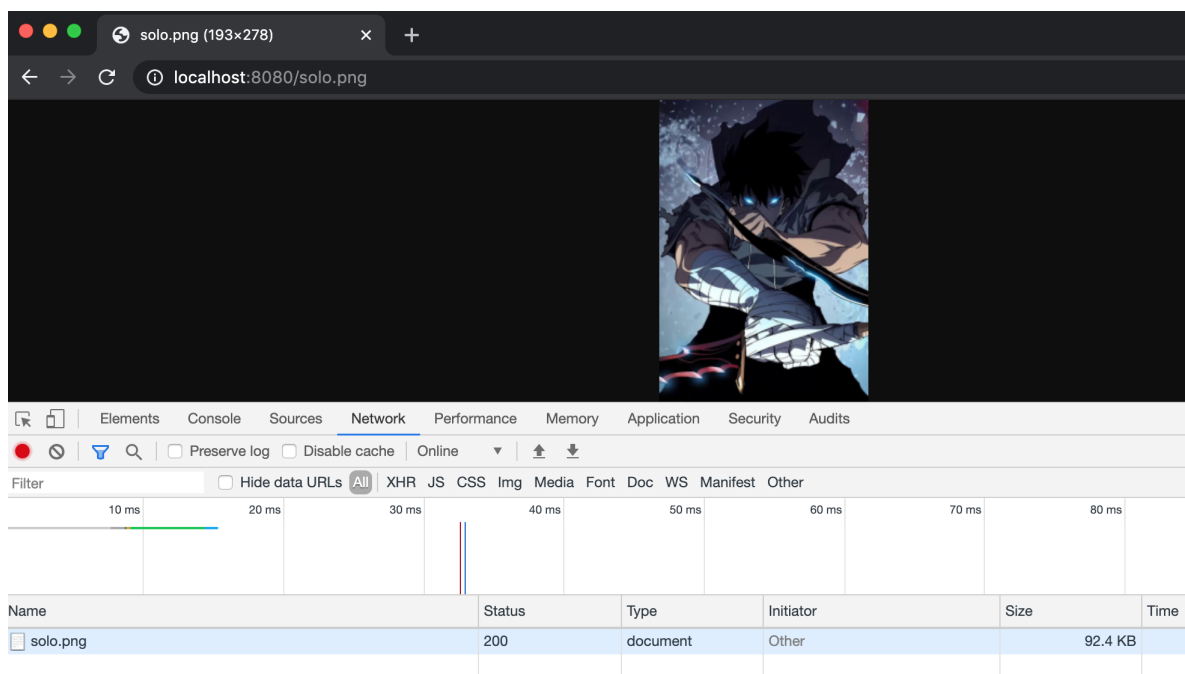


Hello Everyone!

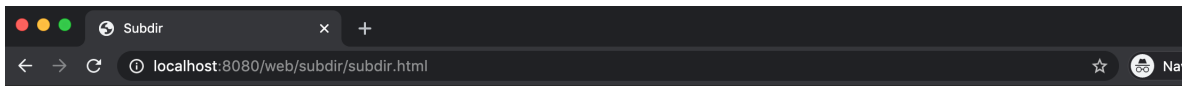
The Web Server is working in index.htm



GET an htm file by giving the path in the URL.



GET an Image file from the server.

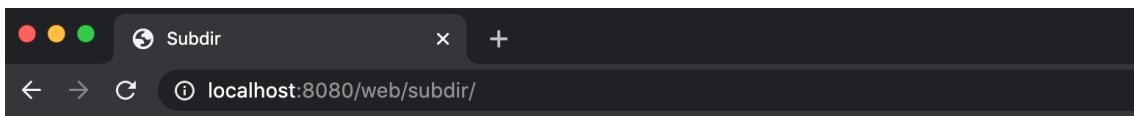


Hello Everyone!

The Web Server is working in subdir.

Network									
Filter									
10 ms 20 ms 30 ms 40 ms 50 ms 60 ms 70 ms 80 ms 90 ms									
Name	Status	Type	Initiator	Size	Time	Waterf			
subdir.html	200	document	Other	260 B	4 ms				

GET an html file from a subdirectory.

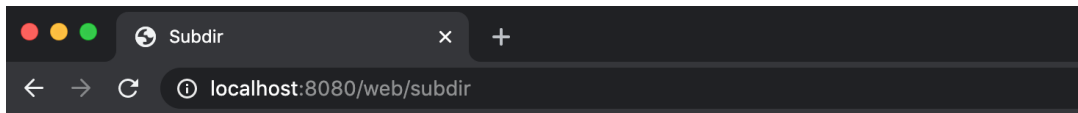


Hello Everyone!

The Web Server is working in index subdir with or without ./

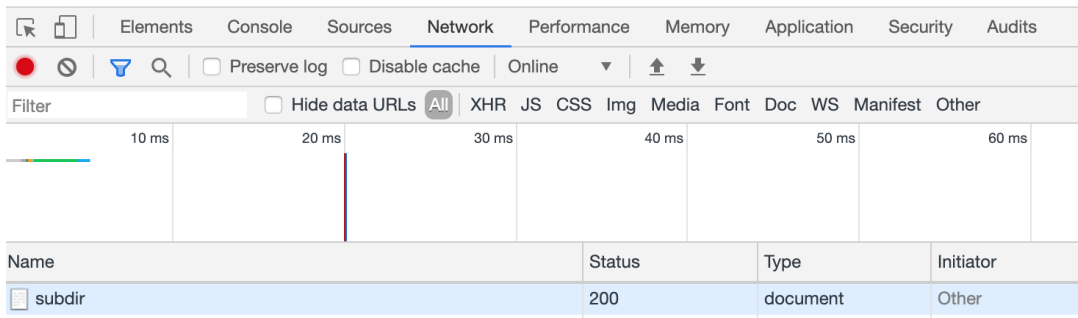
Network									
Filter									
10 ms 20 ms 30 ms 40 ms 50 ms 60 ms									
Name	Status	Type	Initiator						
subdir/	200	document	Other						

GET an index with "/" at the end of the directory.



Hello Everyone!

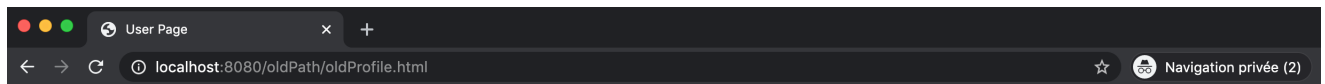
The Web Server is working in index subdir with or without /.



GET an index without "/" at the end of the directory.

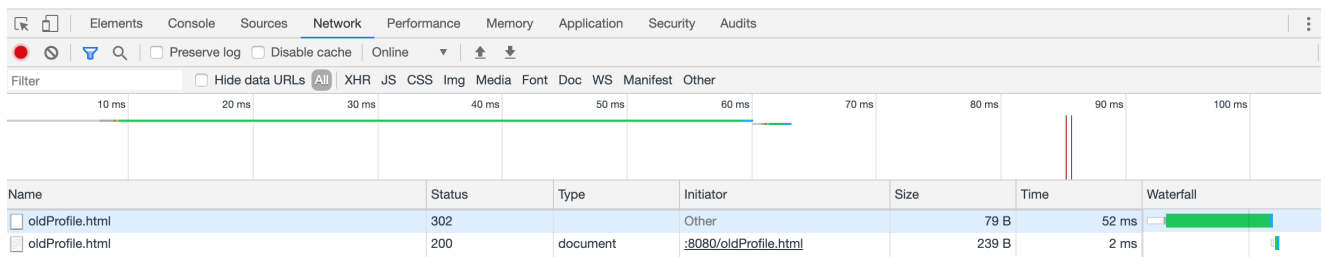
3 Problem 2 - Response codes

3.1 302 Found



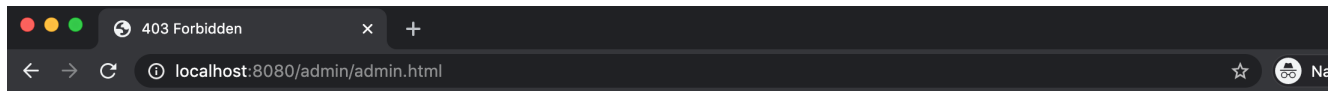
Hello Everyone!

Please login.



The response 302 Found is a way of doing URL redirection. When we write in the URL the html file that we want to find, it will redirect our search to the corresponding path. In this case, we wrote *localhost:8080/oldProfile.html* and we were redirected to *localhost:8080/oldPath/oldProfile.html*.

3.2 403 Forbidden



403 Forbidden

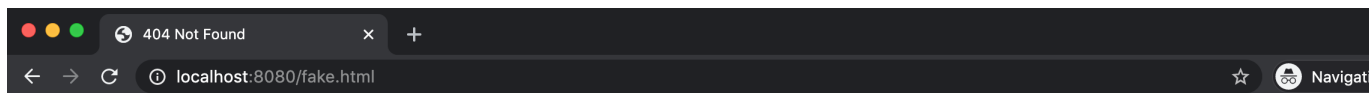
You don't have permission to access /admin on this server

A screenshot of the Chrome DevTools Network tab. The 'Network' tab is selected. A list of network requests is shown. The first request is 'admin.html' with a status of '403', type of 'document', and size of '292 B'. The table has columns for Name, Status, Type, Initiator, Size, Time, and Waterfall.

Name	Status	Type	Initiator	Size	Time	Waterfall
admin.html	403	document	Other	292 B	4 ms	

The response 403 Forbidden is used to restrict the access to a valid URL for some reason. In our case we are restricting all access to the *admin* directory. We decided to restrict to the admin directory because we believe that normal users should not be able to access this directory.

3.3 404 Not Found



404 Not Found

File Not Found

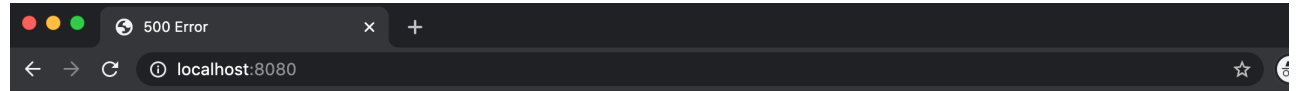
A screenshot of the Chrome DevTools Network tab. The 'Network' tab is selected. A list of network requests is shown. The first request is 'fake.html' with a status of '404', type of 'document', and size of '249 B'. The table has columns for Name, Status, Type, Initiator, Size, Time, and Waterfall.

Name	Status	Type	Initiator	Size	Time	Waterfall
fake.html	404	document	Other	249 B	6 ms	

The response 404 Not Found happen when the client is able to make a link with the server

but the URL could not find what is requested. In our case the URL was fake as there is no fake.html file that exist on this server.

3.4 500 Internal Server Error



500 Internal Server Error

The server encountered an internal error or misconfiguration and was unable to complete your request.

A screenshot of the Chrome DevTools Network tab. The 'Network' tab is selected. A single request is listed with the name 'localhost'. The status is '500', the type is 'document', the initiator is 'Other', the size is '347 B', and the time is '103 ms'. The table has columns for Name, Status, Type, Initiator, Size, and Time.

Name	Status	Type	Initiator	Size	Time
localhost	500	document	Other	347 B	103 ms

Above shows the response 500 Internal Server Error which happens when there is an unexpected error. Below shows the code that was changed to be able to get the 500 response code. GET was changed by GE which made the website not able to do GET and therefore it returned 500 Internal Server Error.

```
//readHttpRequestLine(new String(buf));
http.readHttpRequestLine(receivedString,result); //Will get all the data
//sendRequestedFile();
if(http.requestMethodName.equals("GE")){ //Changed Get into GE to get a 500
    sendToBrowser(directoryRoot);
} else if (http.requestMethodName.equals("POST")){
    //http.readPOSTFile();
    System.out.println("here");
    http.addFileToServer(directoryRoot);
    sendToBrowser(directoryRoot);
} else if (http.requestMethodName.equals("PUT")){
    http.updateCreateFile(directoryRoot);
    sendToBrowser(directoryRoot);
} else {
    internalError();
}

}

} catch (Exception se){
    System.err.println("Connection has been lost or the client " +id+ " closed it...");
    closeClient();
    internalError(); //Call the 500 Internal Server Error
} finally {
    closeClient();
}
```


4 Problem 2 - VG1 POST

Please upload an Image :

Select image: c.png

Transfert des données depuis localhost...

État	Méthode	Domaine	Fichier	Source	Type	Transfert
200	POST	localhost:8080	uploadImage.html	document	html	444 o

▼ POST http://localhost:8080/uploadImage.html 200 | 22ms
Show raw log

▶ Network

▼ Request Headers

User-Agent: "PostmanRuntime/7.22.0"

Accept: "*/*"

Cache-Control: "no-cache"

Postman-Token: "9fc540a2-f540-44f0-8075-abc27e739184"

Host: "localhost:8080"

Content-Type: "multipart/form-data; boundary=-----372753073181541606377878"

Accept-Encoding: "gzip, deflate, br"

Content-Length: 45840

Connection: "keep-alive"

▼ Request Body

▼ "": {...}

_events: {}

_eventsCount: 3

▶ _readableState: {...}

autoClose: true

bytesRead: 45642

closed: true

domain: null

fd: null

flags: "r"

mode: 438

path: "/Users/AK47/Postman/files/c.png"

readable: false

▼ Response Headers

Content-Type: "text/html"

Content-Length: "379"

▶ Response Body

POST http://localhost:8080/uploadImage.html Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings

1 pm.test("Status code is 200", function () {
2 pm.response.to.have.status(200);
3 });
4 pm.test("Successful POST request", function () {
5 pm.expect(pm.response.code).to.be.oneOf([200]);
6 });

Test scripts are written in JavaScript and run after the response is received.
Learn more about tests scripts

SNIPPETS

Response body: Contains string

Response body: JSON value checked

Body Cookies Headers (2) Test Results (2/2) Status: 200 OK Time: 15ms Size: 444 B Save

All Passed Skipped Failed

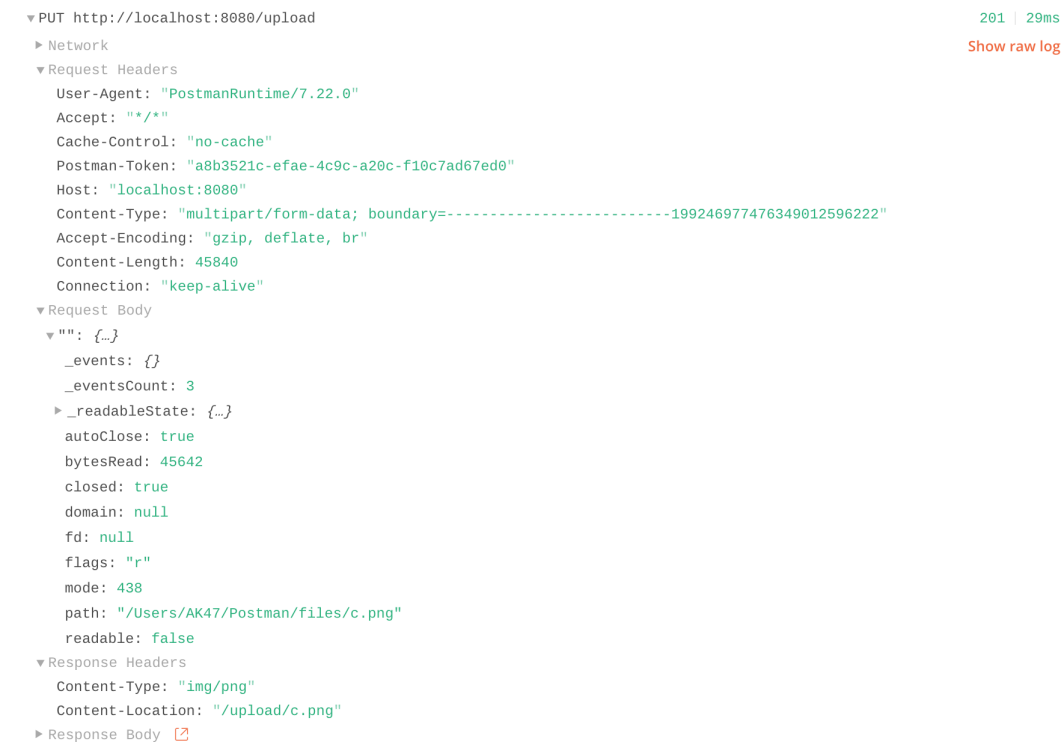
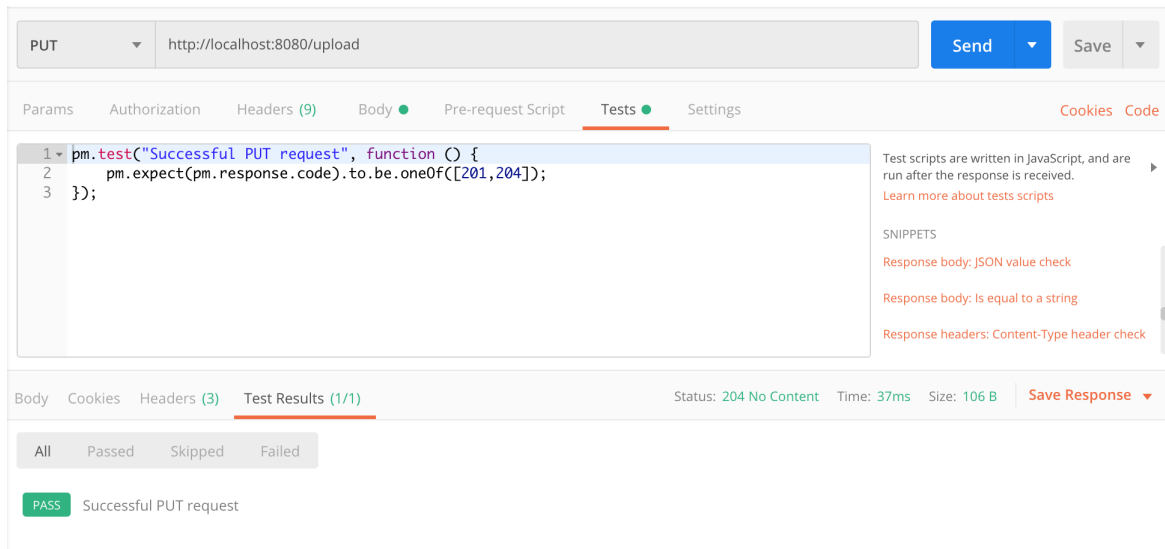
PASS Status code is 200


PASS Successful POST request

4.1 Discussion

Postman application was used to check if the POST method was working. The first screenshot shows the browser where the image was uploaded from. The second image is from Postman where we got the POST response and the last image shows how we have tested the POST method in Postman.

5 Problem 2 - VG2 PUT



```
▼ PUT http://localhost:8080/upload 204 | 69ms  
  ► Network Show raw log  
    ▼ Request Headers  
      User-Agent: "PostmanRuntime/7.22.0"  
      Accept: "*/*"  
      Cache-Control: "no-cache"  
      Postman-Token: "8610f931-3d64-40b8-929c-1a1064042f1a"  
      Host: "localhost:8080"  
      Content-Type: "multipart/form-data; boundary=-----563920084825708069876092"  
      Accept-Encoding: "gzip, deflate, br"  
      Content-Length: 45840  
      Connection: "keep-alive"  
    ▼ Request Body  
      ▼ "": {}  
        _events: {}  
        _eventsCount: 3  
        ► _readableState: {}  
          autoClose: true  
          bytesRead: 45642  
          closed: true  
          domain: null  
          fd: null  
          flags: "r"  
          mode: 438  
          path: "/Users/AK47/Postman/files/c.png"  
          readable: false  
    ▼ Response Headers  
      Content-Type: "img/png"  
      Content-Length: "49459"  
      Content-Location: "/upload/c.png"  
    ► Response Body 
```

5.1 Discussion

We have also used Postman to check if the PUT method was done correctly. The first picture shows the test of Postman that checks if the response code 201 and 204 are being returned. The second picture shows the PUT with response 201 Created which means that the server acknowledge that the request has been fulfilled and it created a new file. The third pictures shows the PUT method with response 204 No Content which means that the server processed the request but did not return any content.

5.2 Difference between POST and PUT

The difference between POST and PUT is that when a PUT receive an URI, it will replace the file at the location and if there is no file it will create it. PUT is idempotent. The POST method always create a new file if there is already one that exist. It is also a non idempotent method.

6 Problem 3

```
MacBook-Pro-AK:~ AK47$ telnet localhost 8080
Trying ::1...
Connected to localhost.
Escape character is '^]'.
GET / HTTP/1.1
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 214

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>
<h1>Hello Everyone!</h1>
<h3>The Web Server is working in index.</h3>

</body>
</html>Connection closed by foreign host.
MacBook-Pro-AK:~ AK47$

MacBook-Pro-AK:~ AK47$ telnet localhost 8080
Trying ::1...
Connected to localhost.
Escape character is '^]'.
GET /oldPath/oldProfile.html HTTP/1.1
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 174

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>User Page</title>
</head>
<body>
<h1>Hello Everyone!</h1>
<h3>Please login.</h3>
</body>
</html>Connection closed by foreign host.
MacBook-Pro-AK:~ AK47$
```

For this exercise telnet was used. On the left, it shows a GET from a directory , *GET / HTTP/1.1*. The screenshot on the right shows a GET with a named HTML page, *GET /oldPath/oldProfile.html HTTP/1.1*.

7 Problem 4

Here is a screenshot from the result we got when we tested the python code from the teacher.

```
MacBook-Pro-AK:morgan AK47$ python3.7 testa2.py
OK: Main index page
OK: Named page
OK: Named page
OK: Clown PNG
OK: Bee PNG
OK: World PNG
OK: Index a
OK: Page a
OK: Fake page b
OK: Index b
OK: Page b
OK: Fake Page
OK: Index c
OK: Page c
OK: Page fail
OK: Page in dir fail
OK: Dir no index fail
OK: Image fail
MacBook-Pro-AK:morgan AK47$
```