

Milestone Two Document

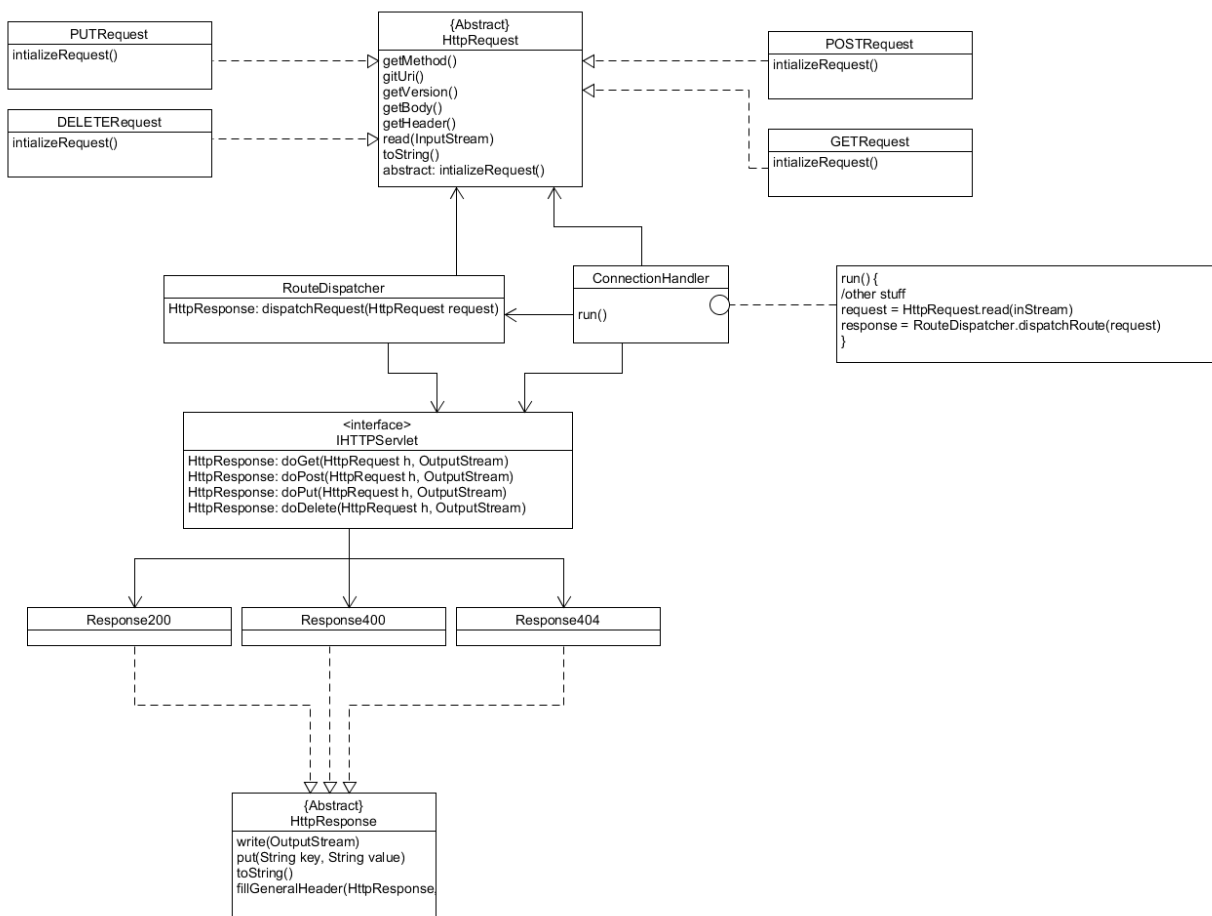
Gabriel Glenn and Jack Petry

Design Patterns

Command Pattern – The client sends request and responds. These are created as objects that are easily extensible.

Prototype Pattern – Our response and request classes are all abstracted to the same objects. This allows the user or later programmers to easily add both responses and request without changing the source code.

Updated UML



Change Log

IHTTPServlet – and extensible interface that allows user to create their own Servlets. These servlets uniquely handle requests based on the user's implementation.

RouteDispatcher – loads new Servlet plugins and maps their routes. Also handles incoming requests dispatching them to the correct servlet.

Possible Improvements

Currently RouteDispatcher both loads and dispatches requests. Those two jobs should be separated.

Currently adding request classes requires naming convention that follows [request name in uppercase] + Request. A proxy protecting the server could help to protect against DDOS. Adding a larger set of operations and responses would make the server better as well. Also, probably could make the request reflection area more robust.

Feature list

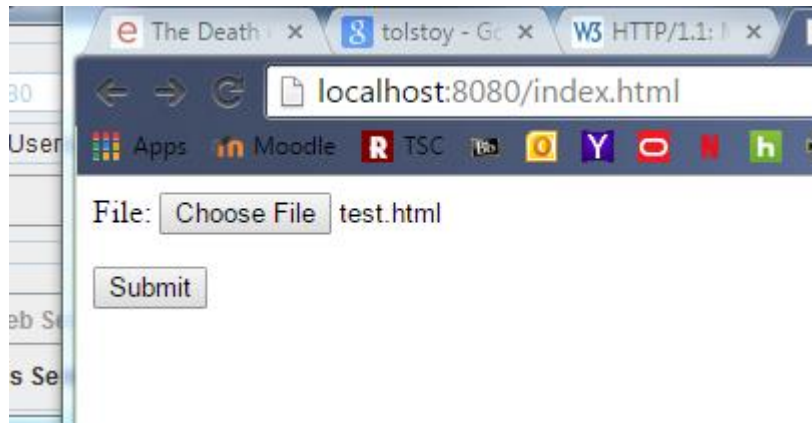
GET, POST, PUT, and DELETE – (Jack) the user can choose how and where each of these requests are handled

Servlet Creation – (Jack) allows the user to create their own unique Servlet

Dispatch Requests-(Gabe) Allows the user to dynamically add servlets and access them via their browser

Test Report – MS1

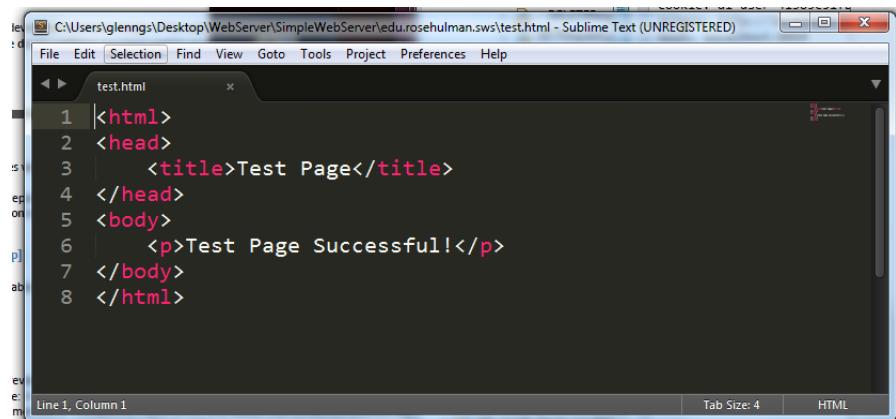
POST test



Submitting test.html



This is the request the server received after the submit button is pressed



```
1 <html>
2 <head>
3   <title>Test Page</title>
4 </head>
5 <body>
6   <p>Test Page Successful!</p>
7 </body>
8 </html>
```

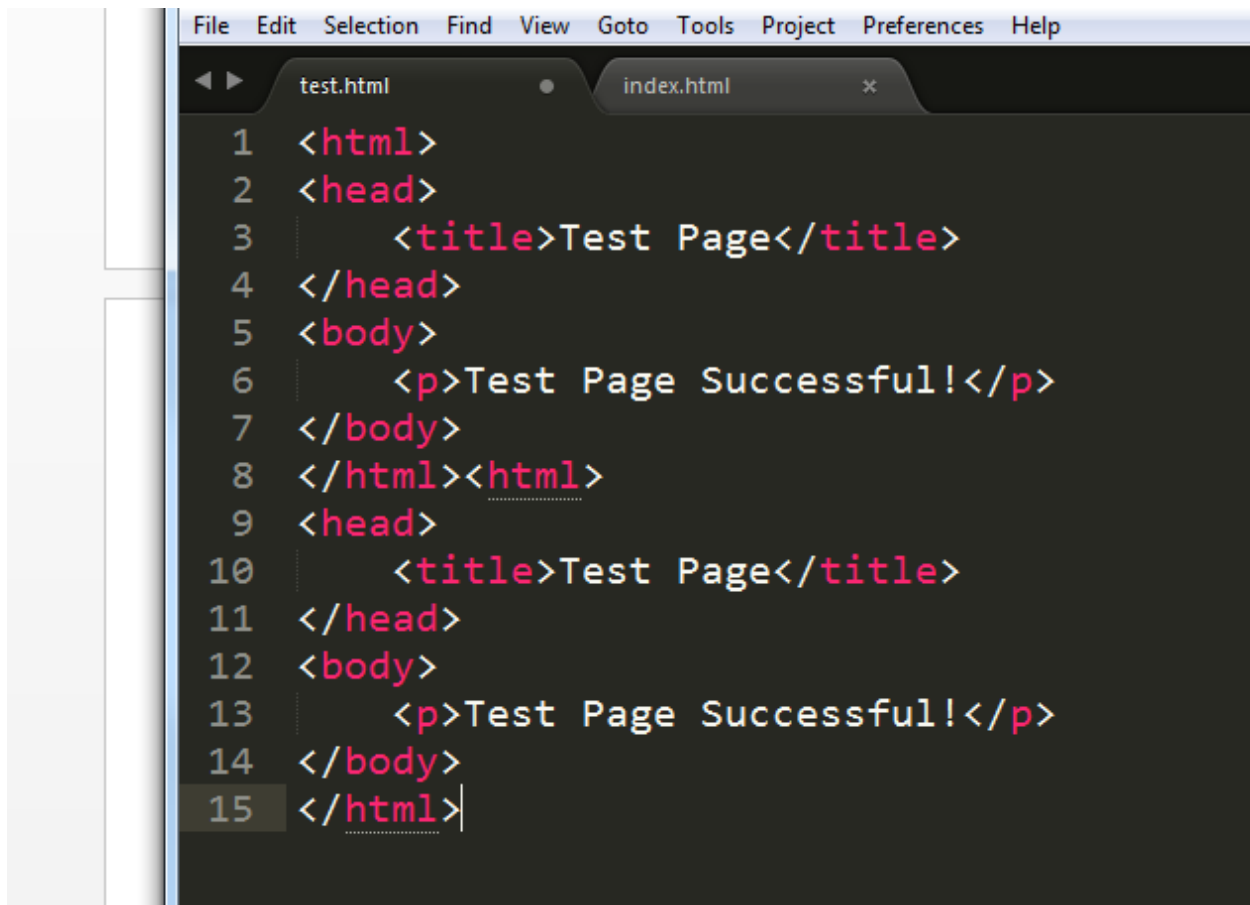
This is the file in the root dir that the request posted to

PUT Test



```
WebServer [Java Application] C:\Program Files\Java\jre1.8.0_40\bin\javaw.exe (Apr 27, 2015, 1:58:44 AM)
PUT / HTTP/1.1
content-length: 290
referer: http://localhost:8080/index.html
accept-language: en-US,en;q=0.8
cookie: ai_user=41385c51fb9f48bcafc8a542bb760c50|2015-03-12T20:08:47.7988973+00:00; ai_session=4b7b7bd7e2df4feb9d5e408cc70677ad|20:
origin: http://localhost:8080
accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
content-disposition: form-data; name="file1"; filename="test.html"
host: localhost:8080
connection: keep-alive
content-type: text/html
cache-control: max-age=0
accept-encoding: gzip, deflate
user-agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/42.0.2311.90 Safari/537.36
----- Body -----
<html>
<head>
  <title>Test Page</title>
</head>
<body>
  <p>Test Page Successful!</p>
</body>
</html>
-----WebKitFormBoundaryek6bbPf3aWbtMRJR--
```

Working with the same file we again have the request reaching the server



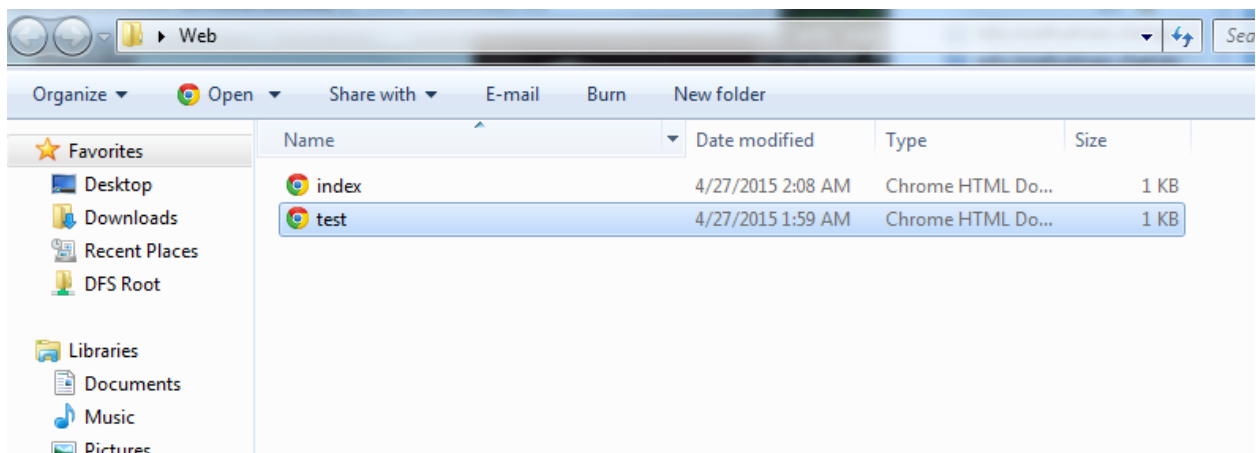
```
File Edit Selection Find View Goto Tools Project Preferences Help

test.html index.html x

1 <html>
2 <head>
3   <title>Test Page</title>
4 </head>
5 <body>
6   <p>Test Page Successful!</p>
7 </body>
8 </html><html>
9 <head>
10   <title>Test Page</title>
11 </head>
12 <body>
13   <p>Test Page Successful!</p>
14 </body>
15 </html>
```

And here we have the result

DELETE Test



There is the file we are going to delete

Sleep time (ms) Requests

Connection Request

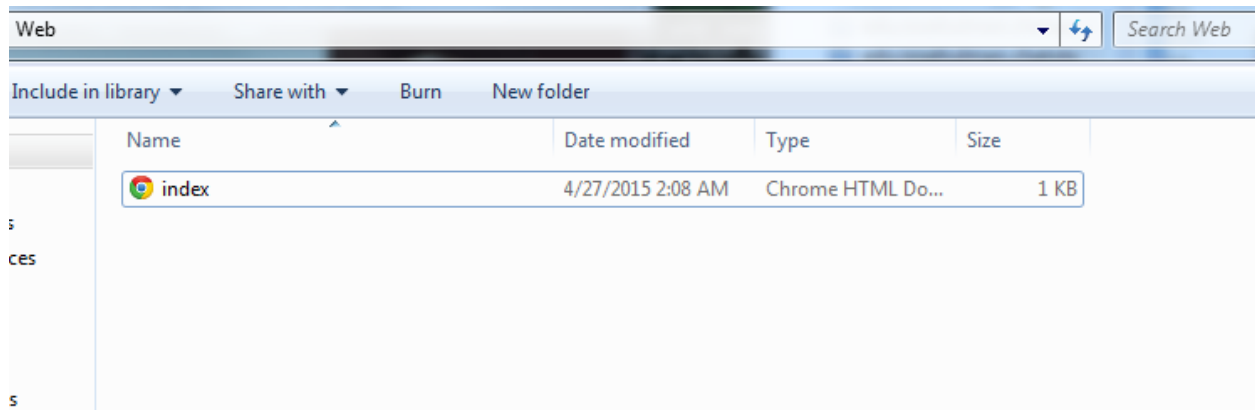
```
DELETE /test.html HTTP/1.1
Host: localhost
Connection: Keep-Alive
User-Agent: HttpTestClient/1.0
Accept: text/html,text/plain,application/xml,application/json
Accept-Language: en-US,en;q=0.8
```

Connection Command

Connection Response

Connection Established!

Here is the request



And now it has been deleted

Test Report - MS 2

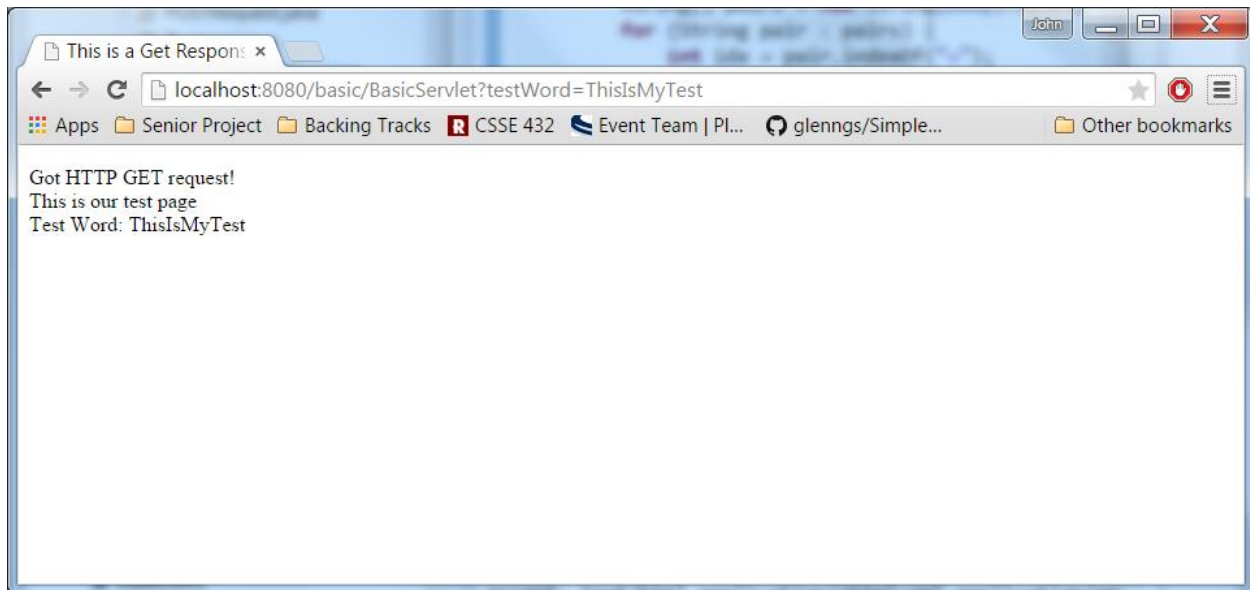
Start the server. In the directory where the server is running, there either initially needs to be basic.jar which is the creation of the TestServlet project or you need to add it after it starts running (and it will be automatically added).

GET.

There are a couple of different pages for GET.

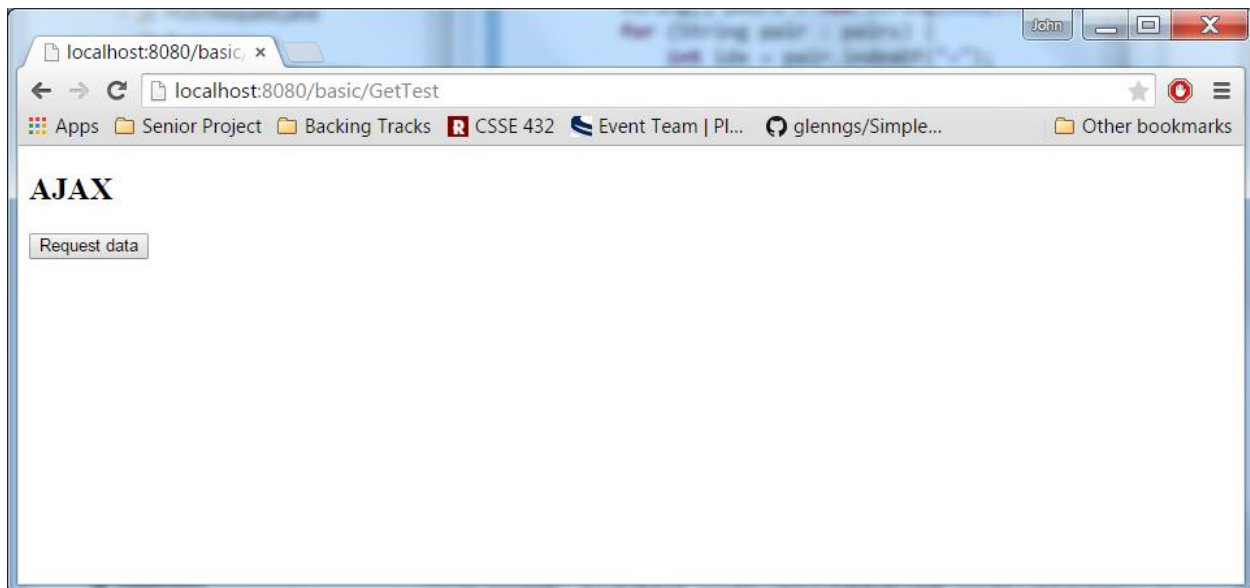
Go to <http://localhost:8080/basic/BasicServlet?testWord=ThisIsMyTest>

Gives a response like the following:



Another page that we have serves static GET content and useful for testing the POST command.

<http://localhost:8080/basic/GetTest>



POST.

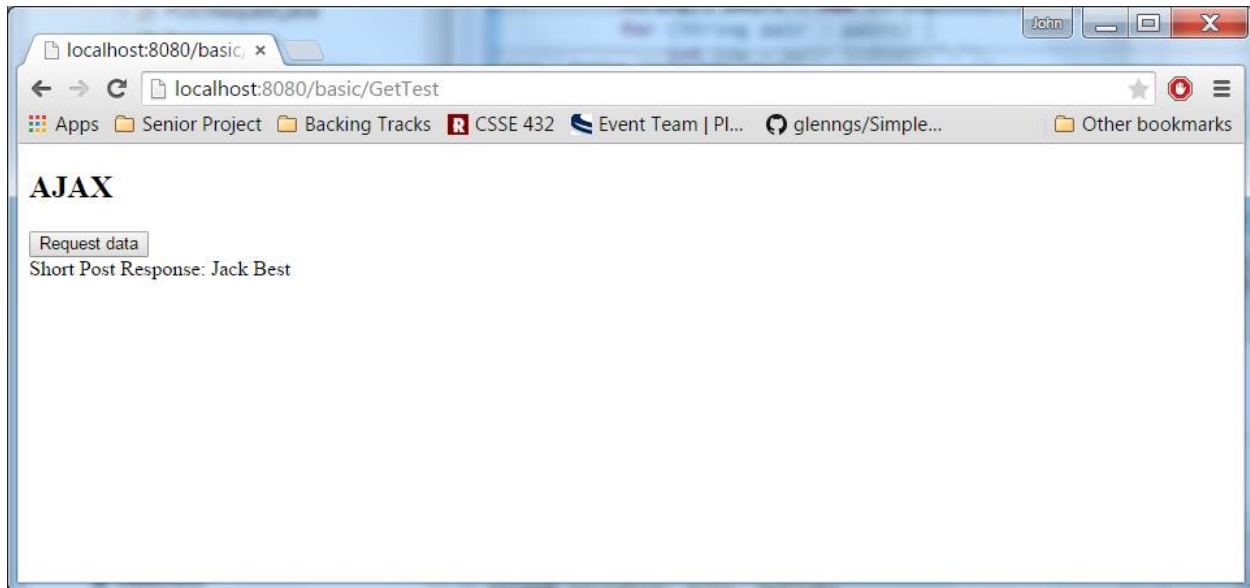
The POST command can be tested using the button of the last page.

It uses an ajax call like the following.

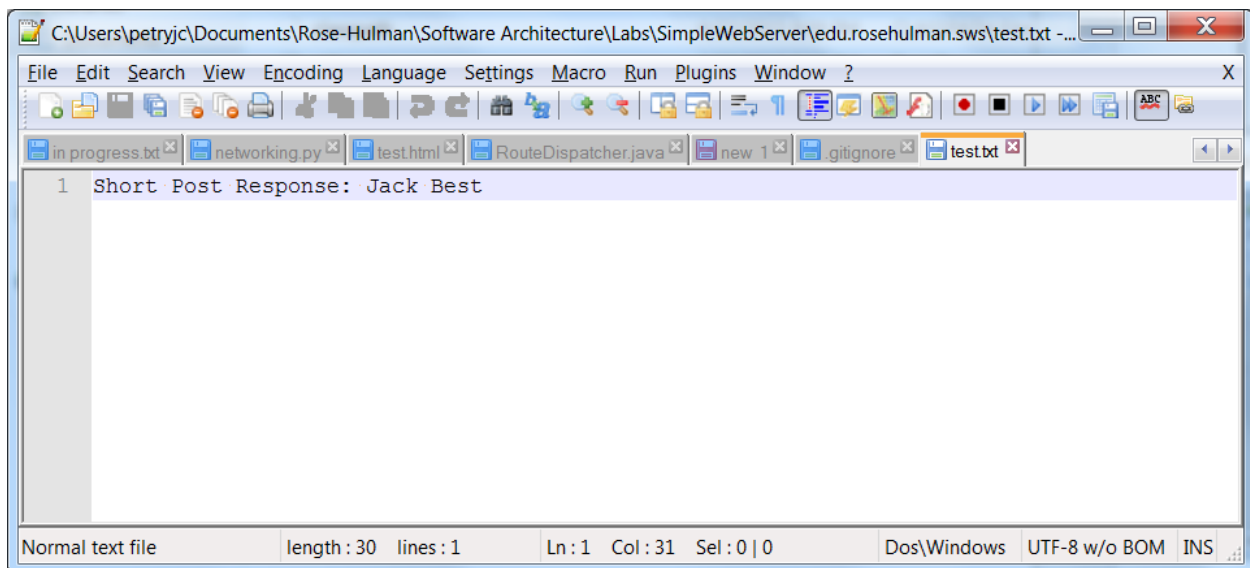
```
xmlhttp.open("POST","http://localhost:8080/basic/PostTest",true);
```

```
xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded");
```

```
xmlhttp.send("word1=Jack&word2=Best&filename=test.txt");
```

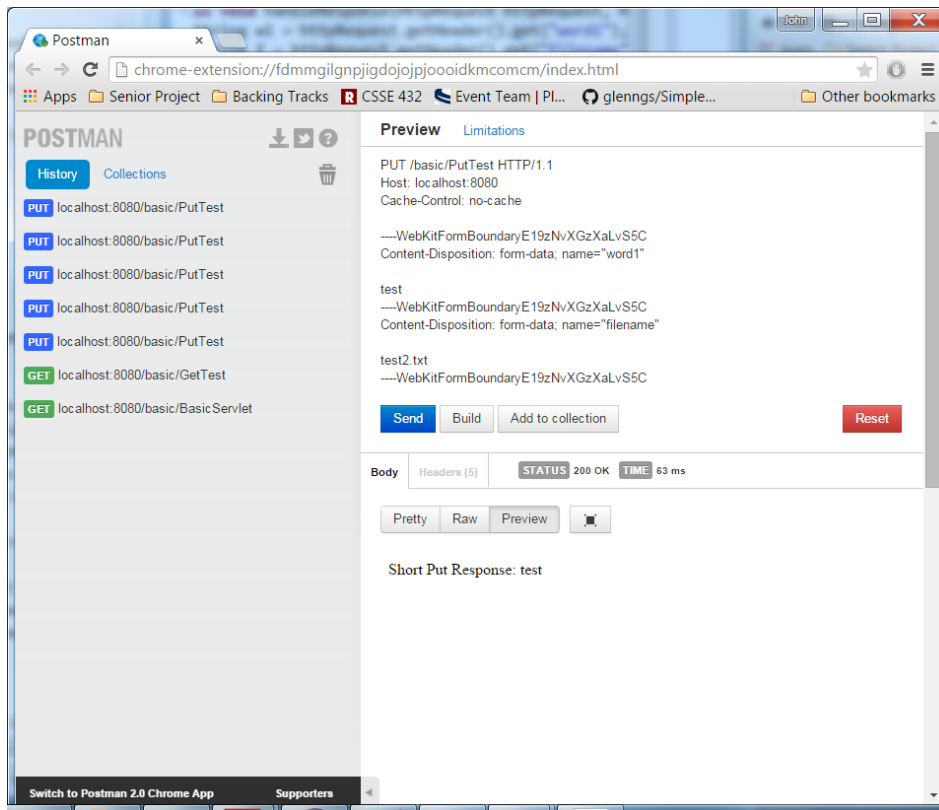


It also creates a file (test.txt) that has the same response contents.

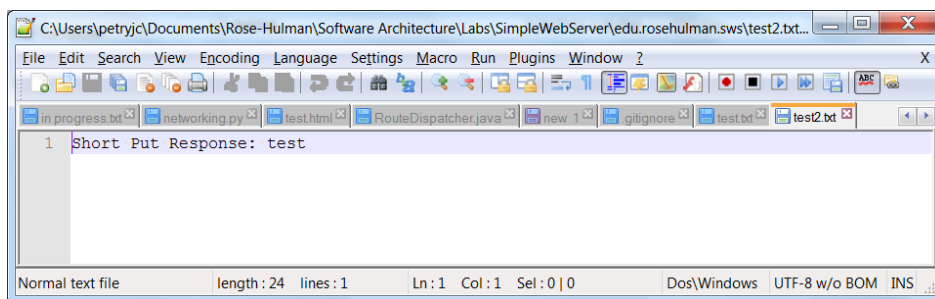


PUT.

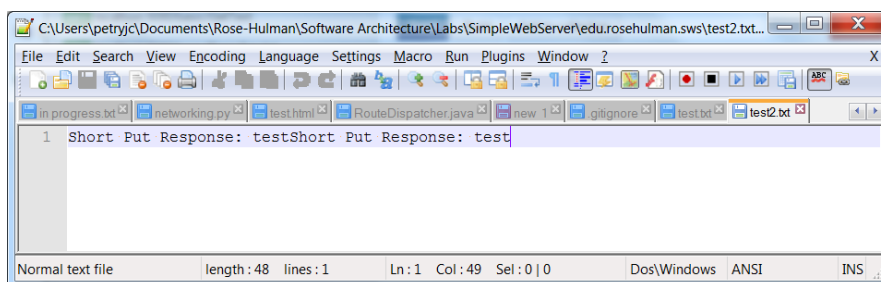
The PUT command can easily be tested using POSTMAN.



It also creates a file (test2.txt)

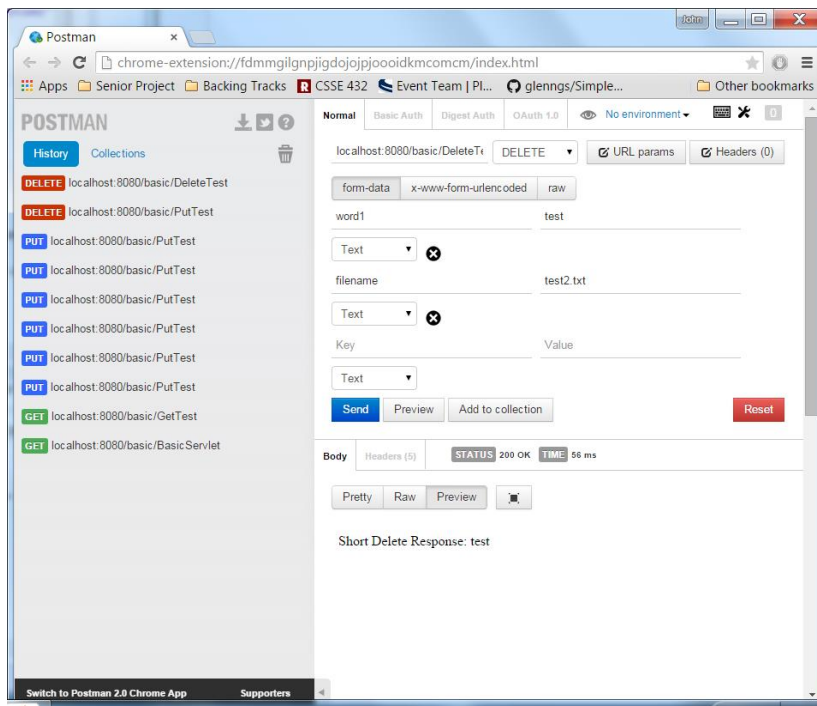


If you do the put again. It appends because I thought that would be nice of the servlet to do.



DELETE

DELETE can also be easily tested using POSTMAN.



It also deleted the given file because I thought that sounded cool. (Notice test2.txt is gone)

