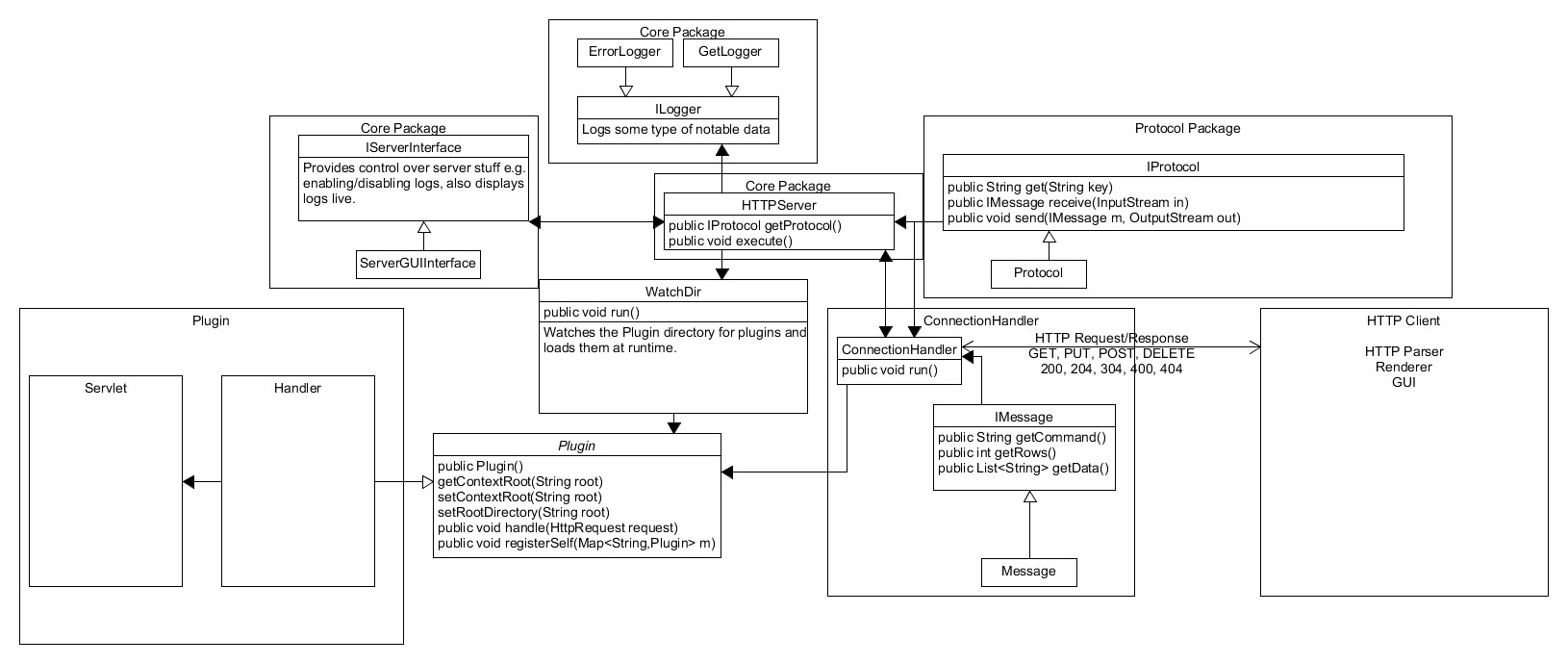
|  |
| --- |
| Rose-Hulman Institute of Technology |
| Simple Web Server |
| CSSE477 |

|  |
| --- |
| Gregory Bollivar and L.E. Davey  11-2-2015 |

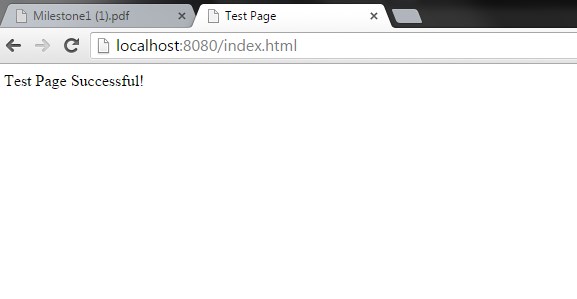
1.)    
Changes to the diagram have occured

2.) About the “detailed diagram” we believe that a.) our detailed diagram is very much the same as the diagram above, b.) it has not changed significantly.

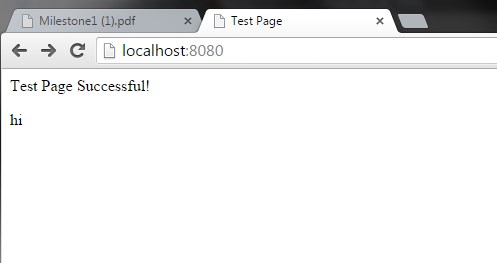
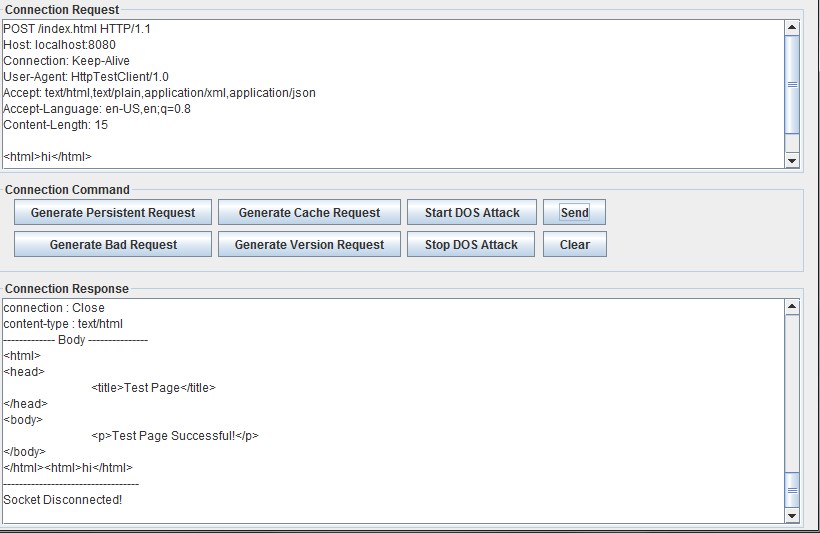
3.) Patterns: Delegating to an interface, Strategy, WatchDir, no additional patterns

4.) Improvements: Our counting, improve client to generate dummy post/put/delete, we increased the security of the system, we restricted access to the plugins folder, added logging, throttled high using clients and attempted to prevent threadbombs.

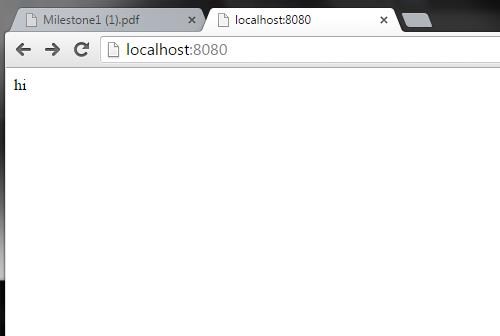
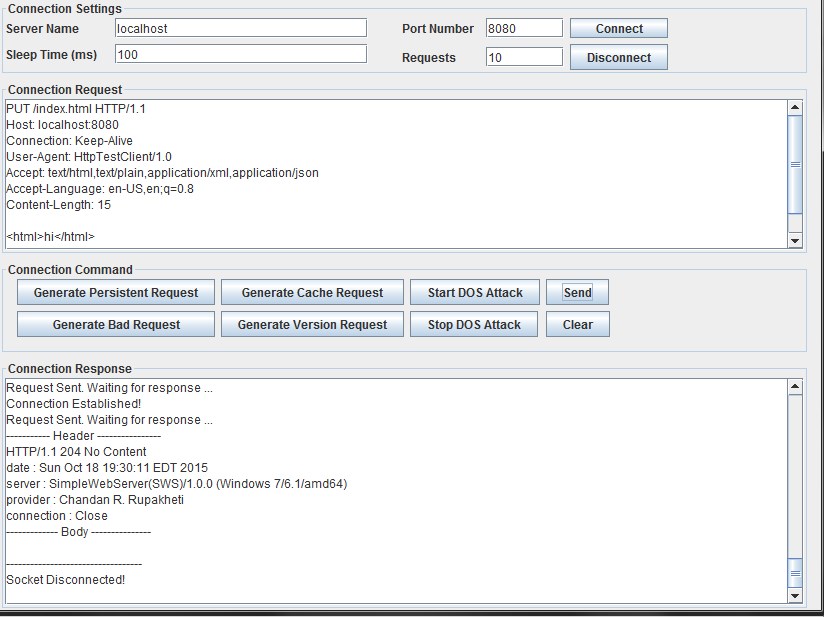
5.) GET:



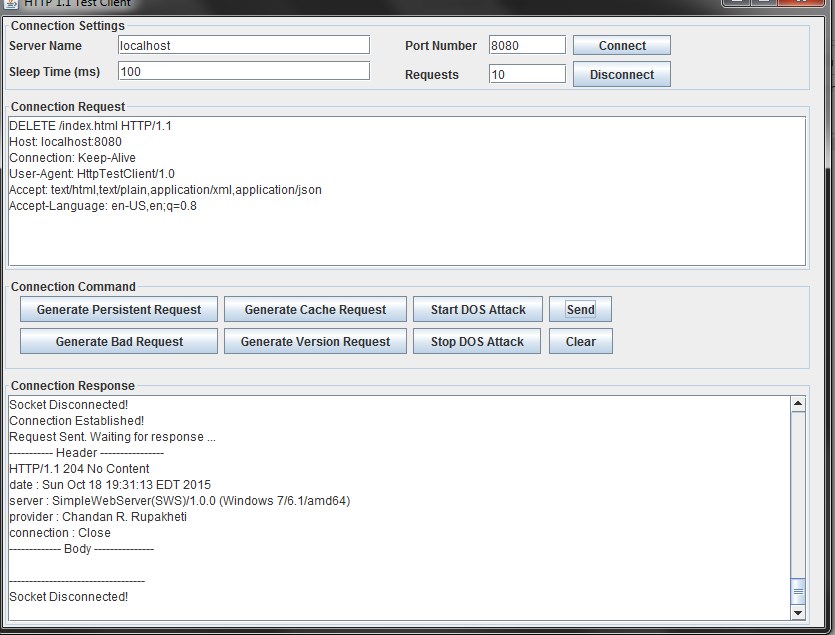
POST:

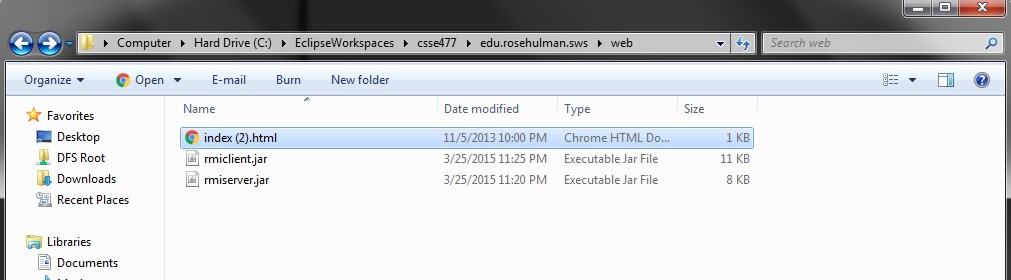
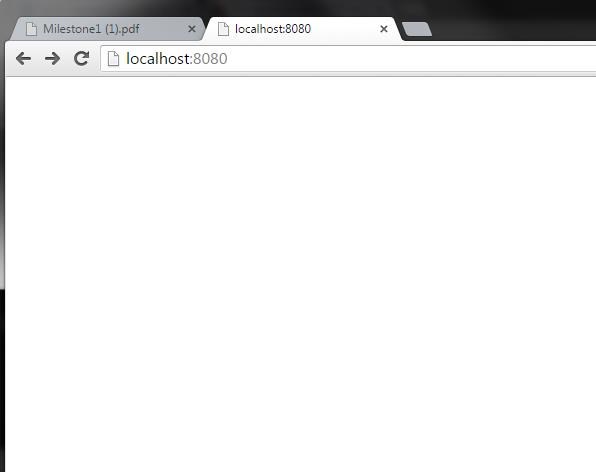


PUT:



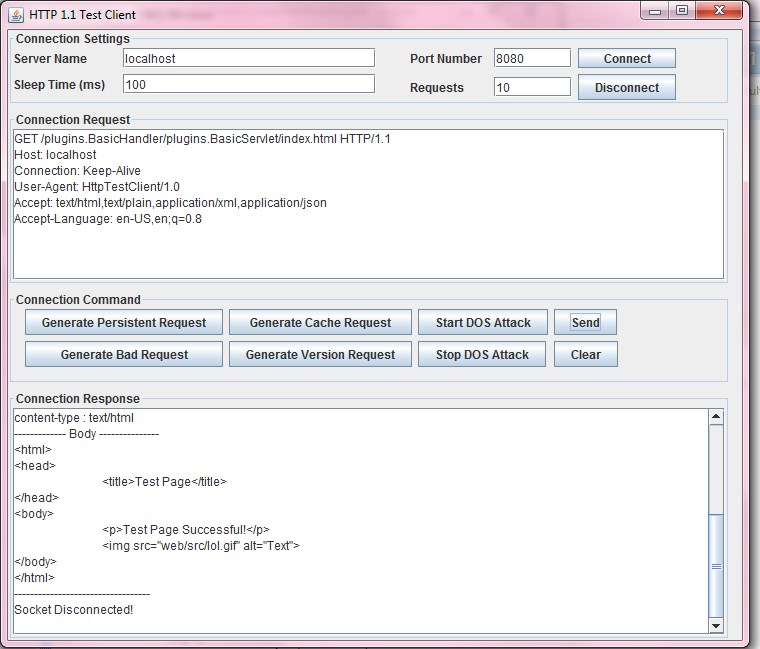
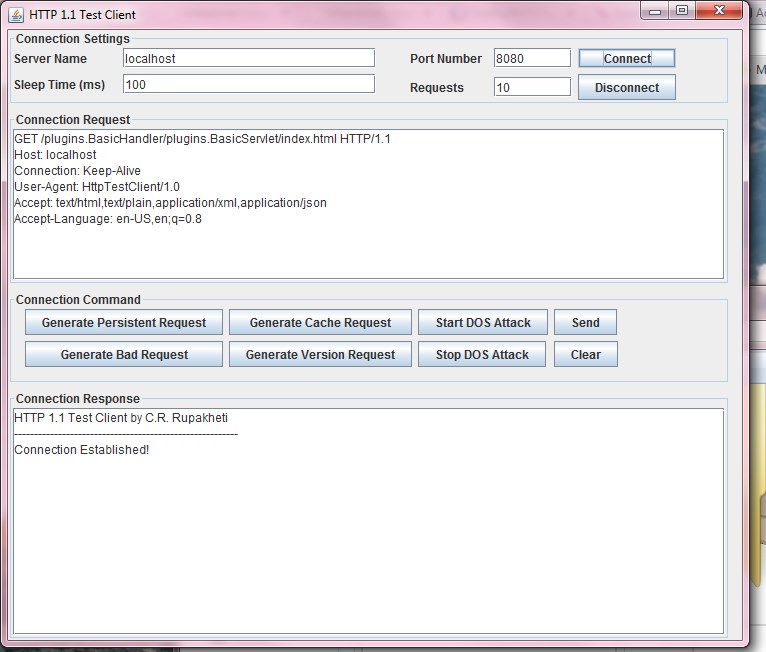
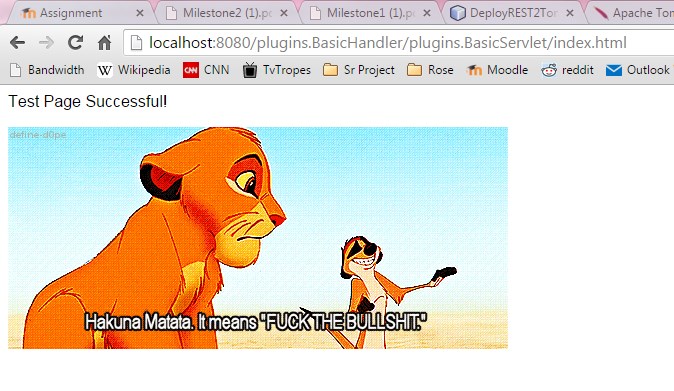
DELETE:



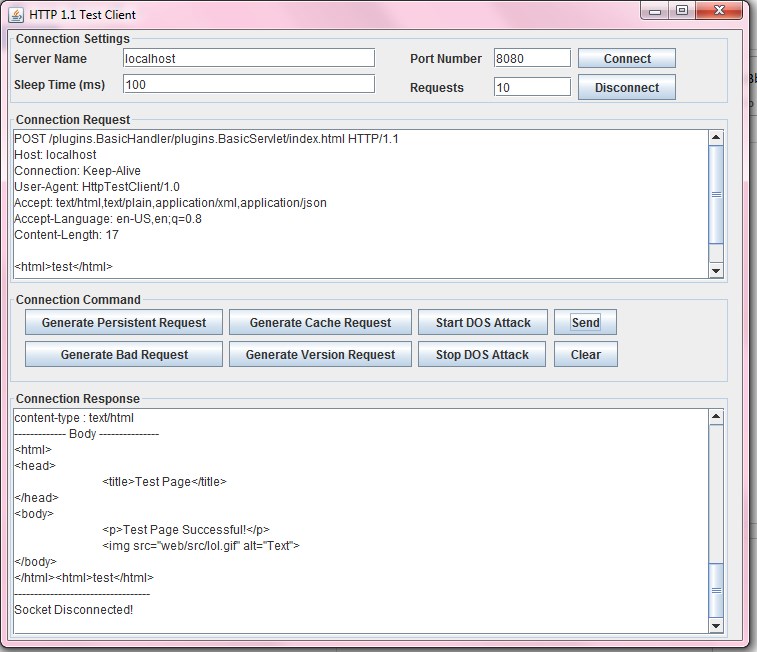
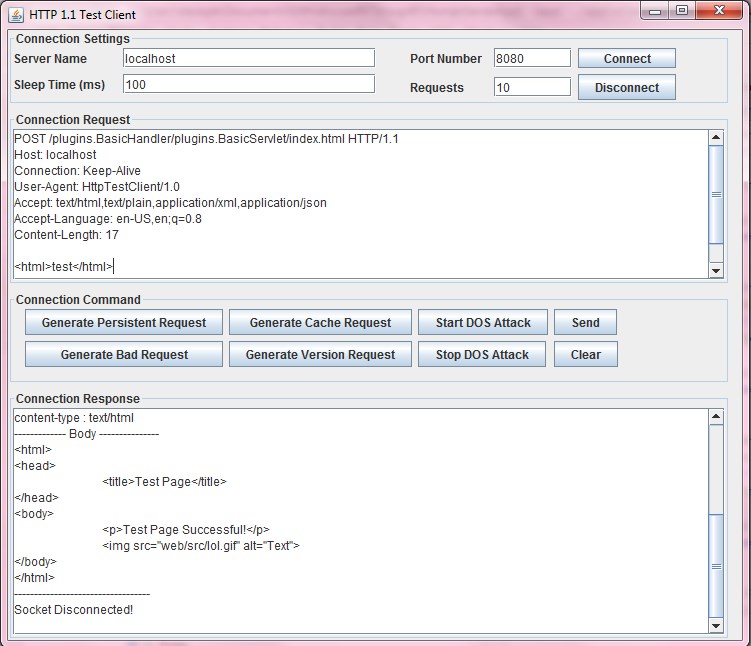


Original GPPD Handler:

GET:



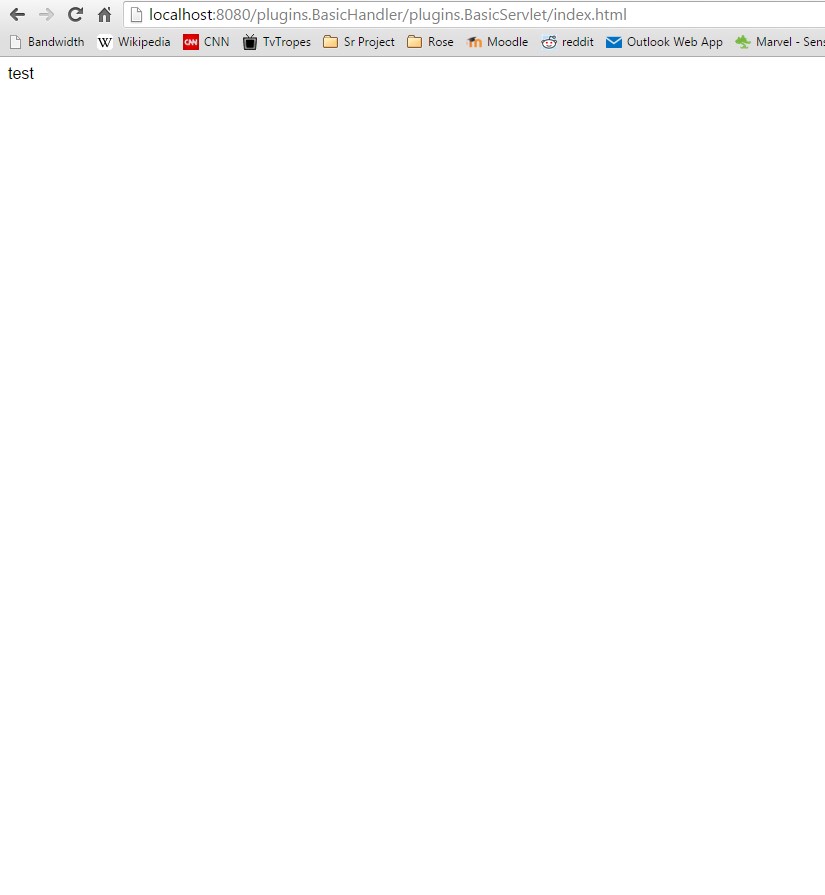
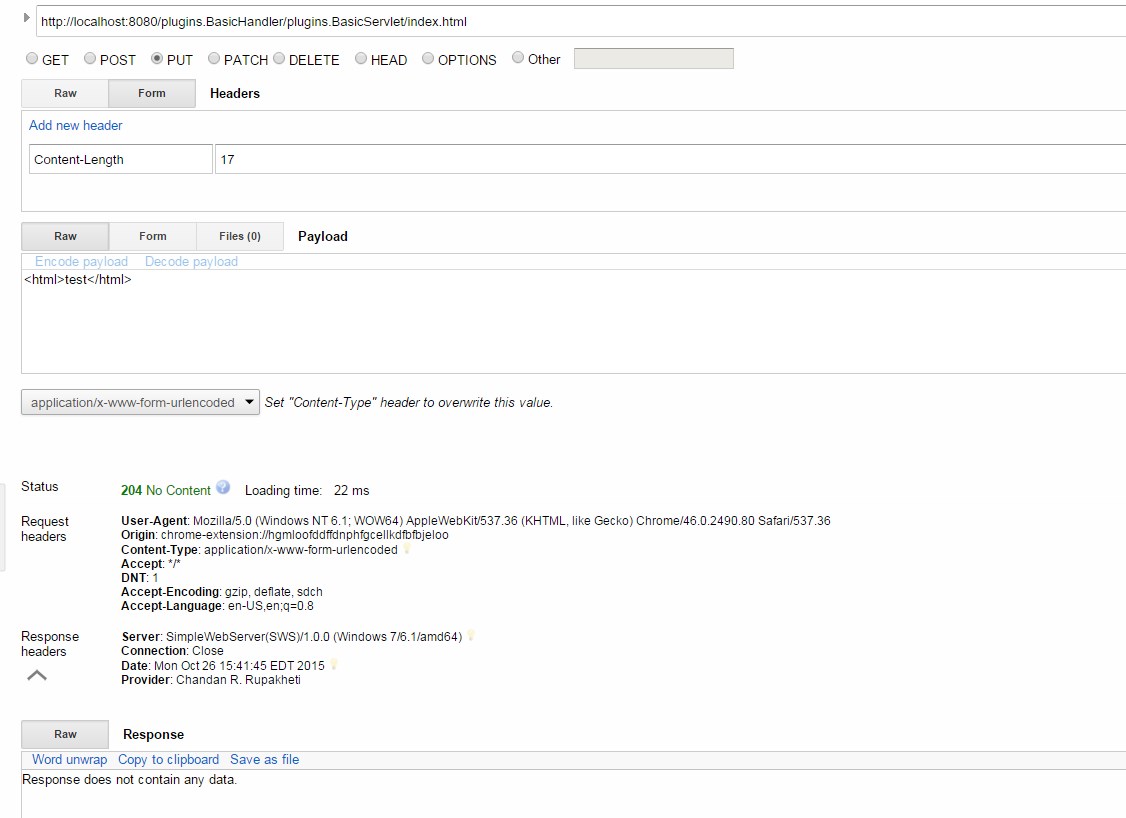
POST



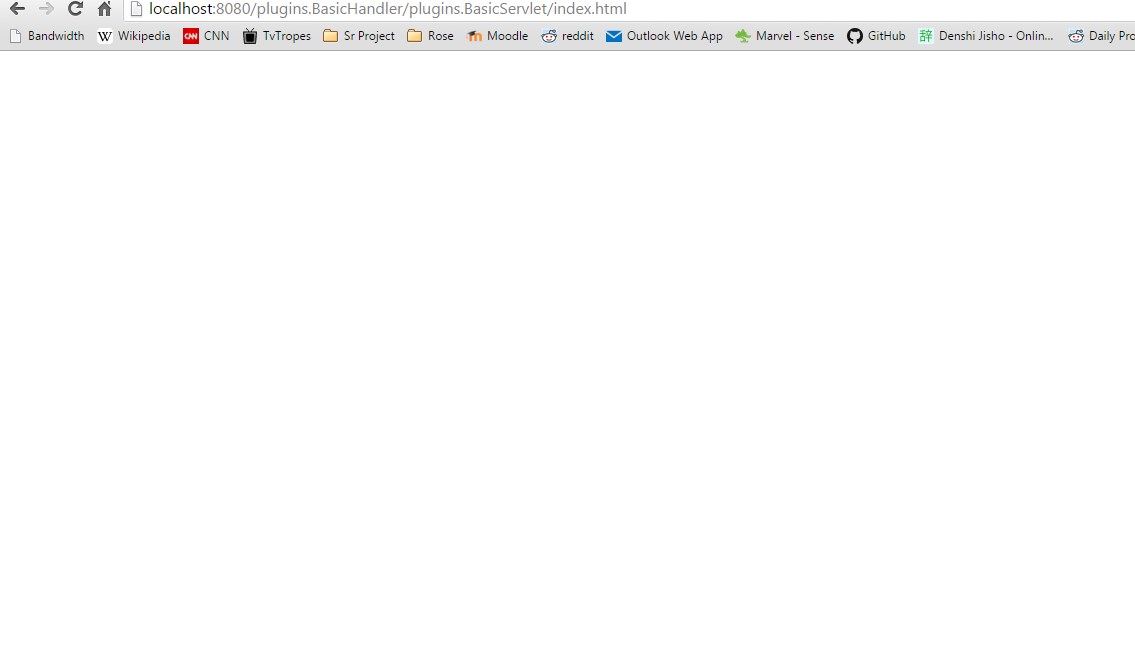
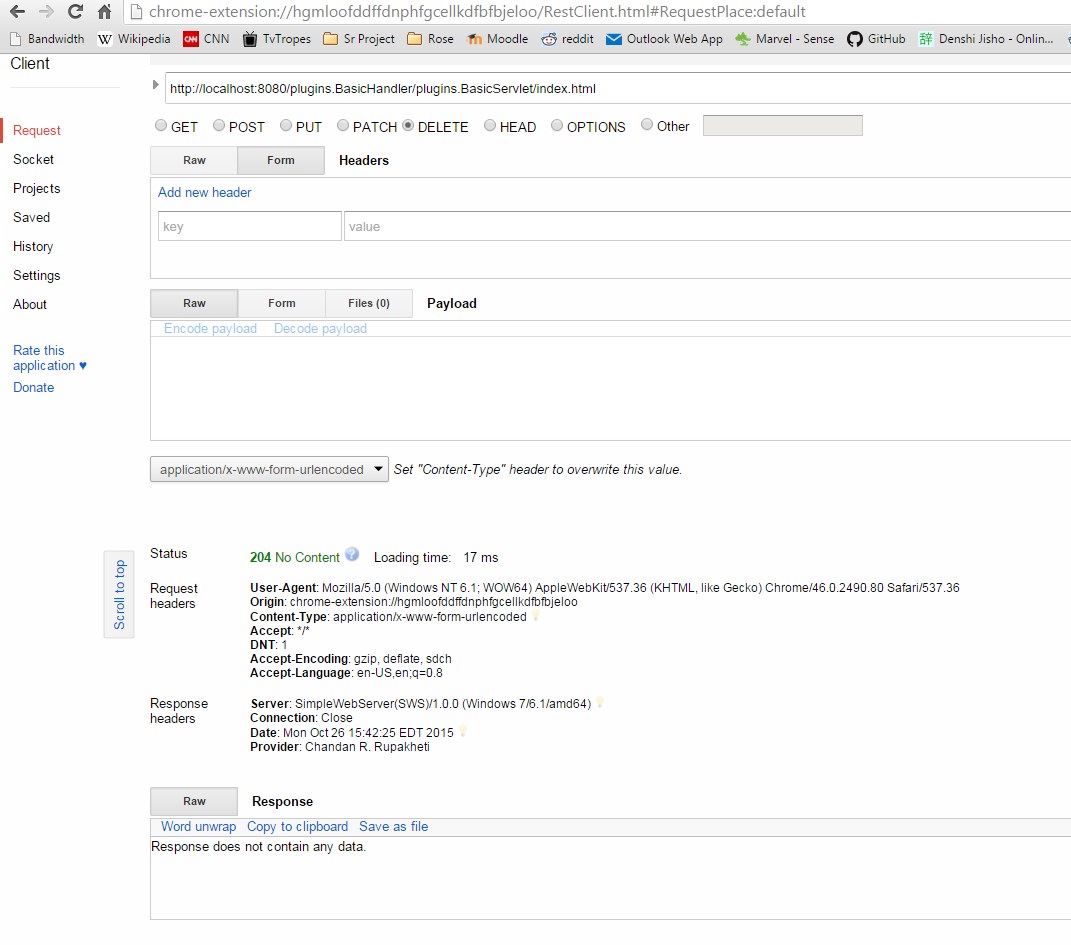
Demonstration of change in browser. Received via GET call.



PUT



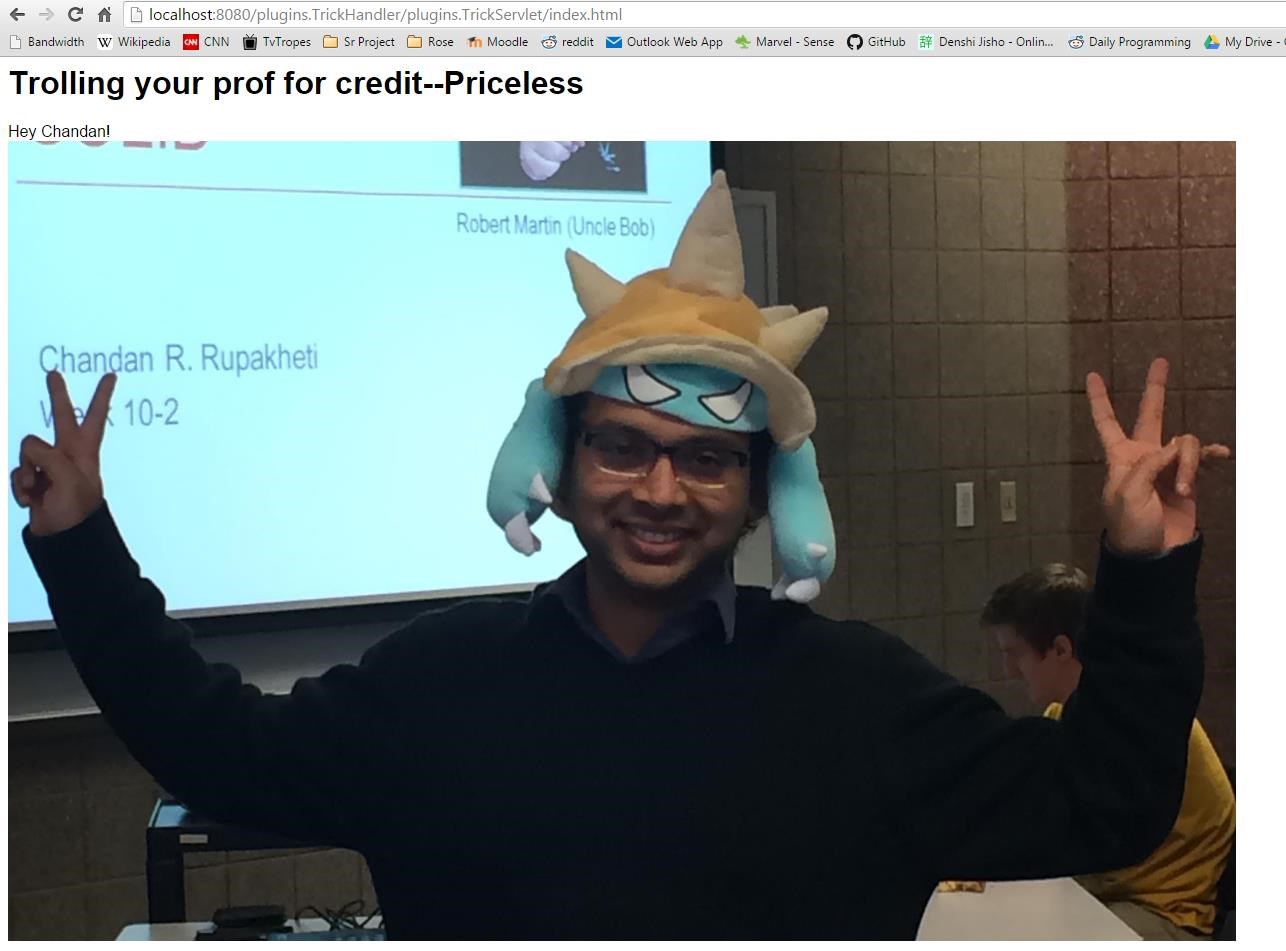
DELETE



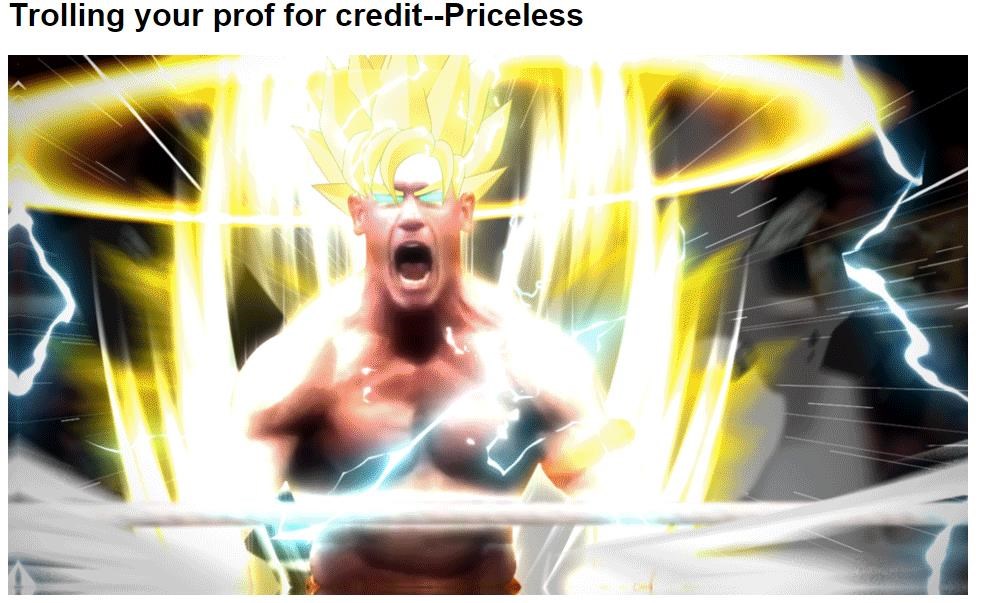
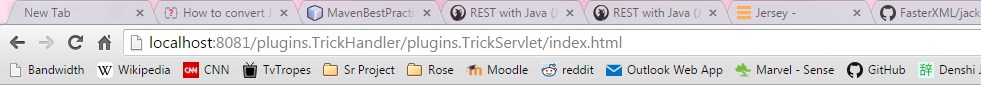
Trick Handler:

This handler redirects the user to one of 5 random webpages no matter what they ask for in a GET request.

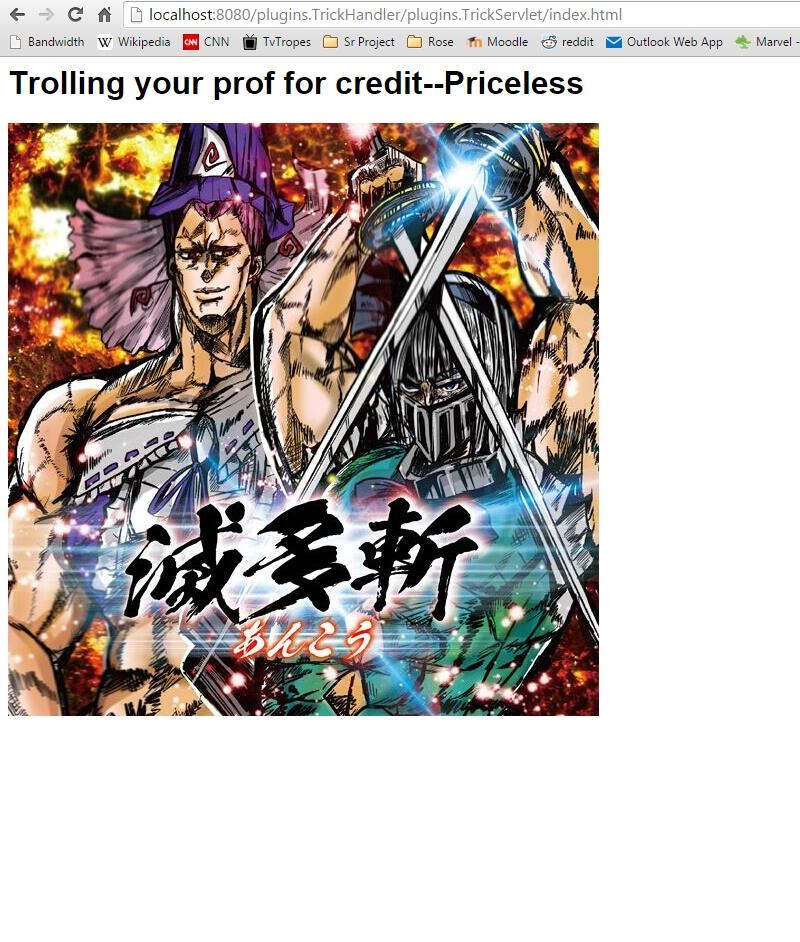
Page #1:



Page #2:

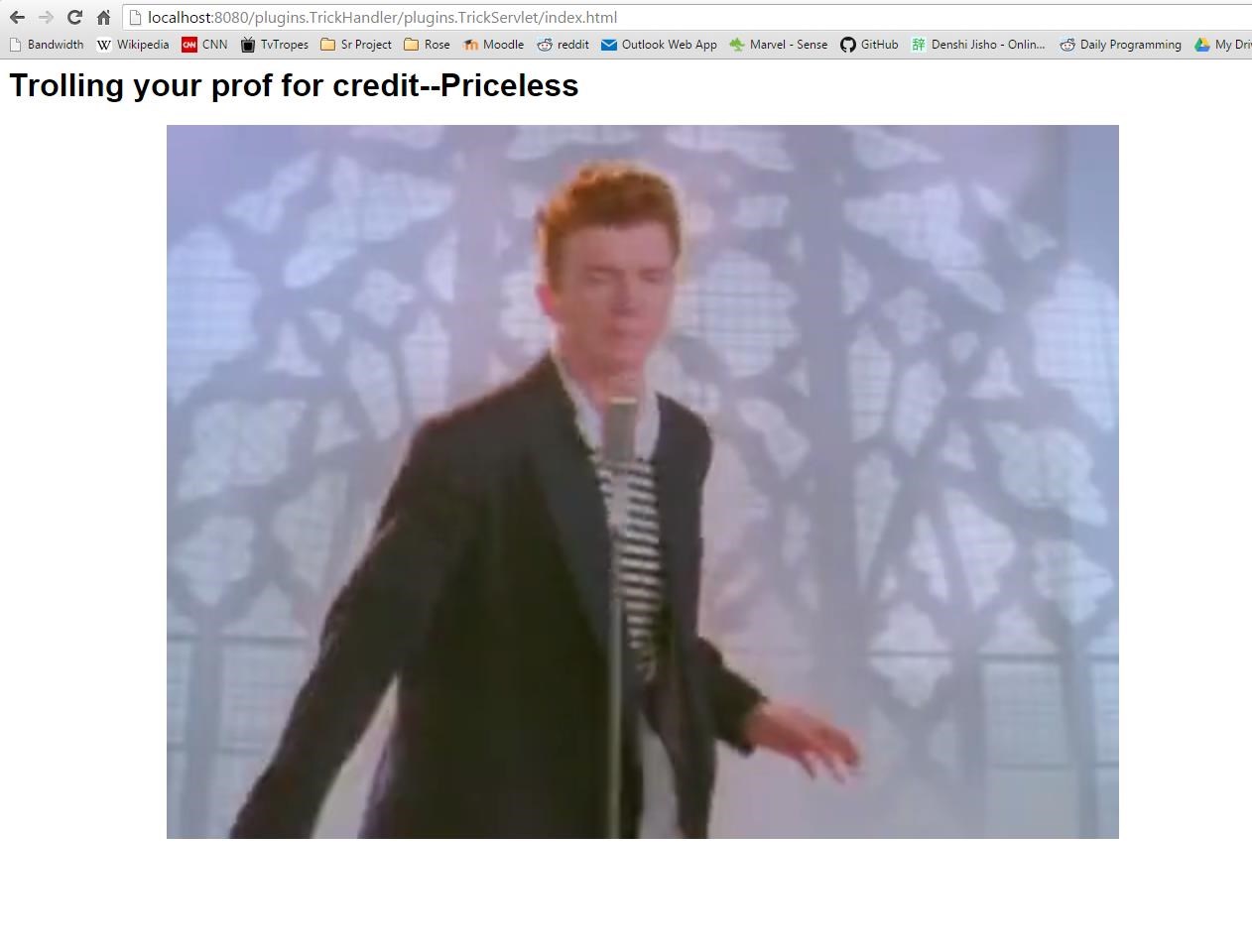
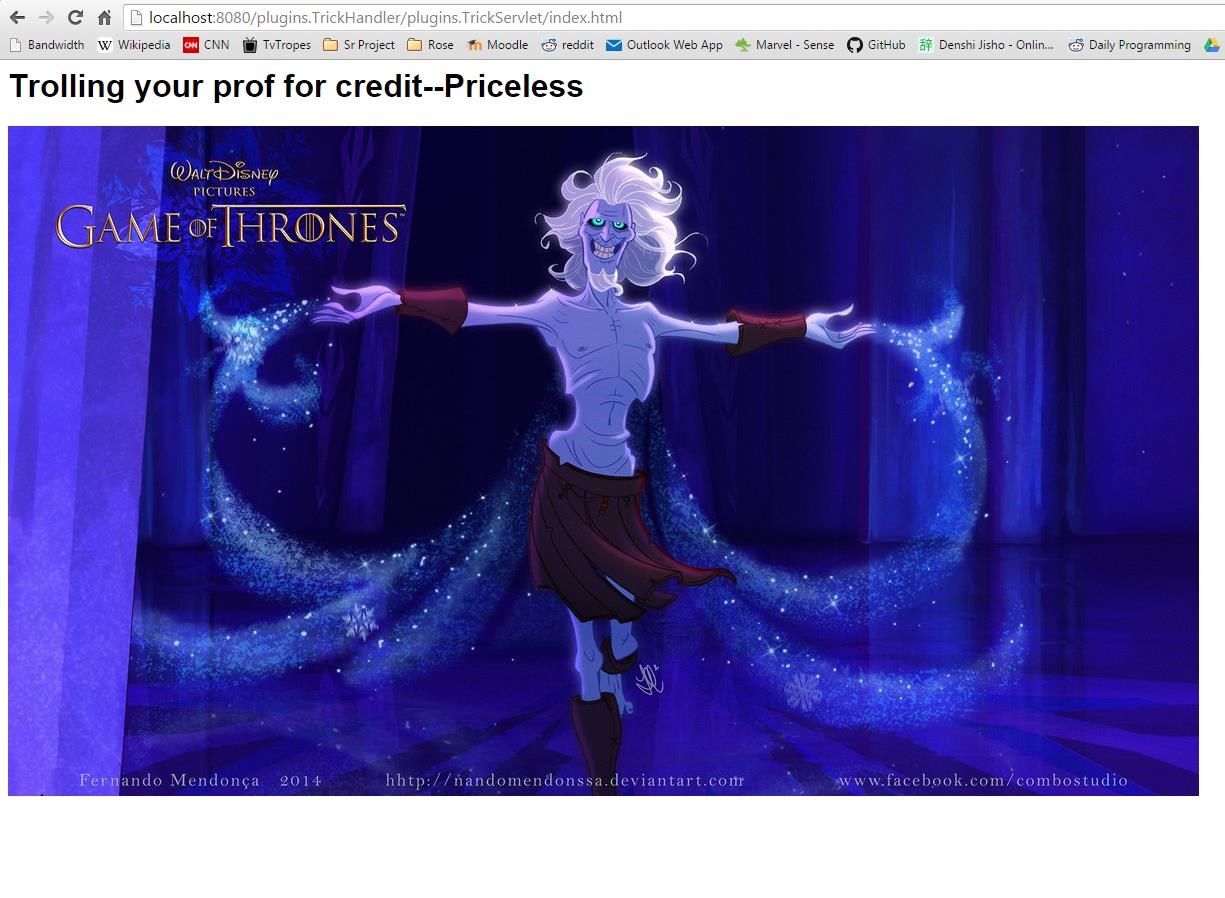


Page #3:

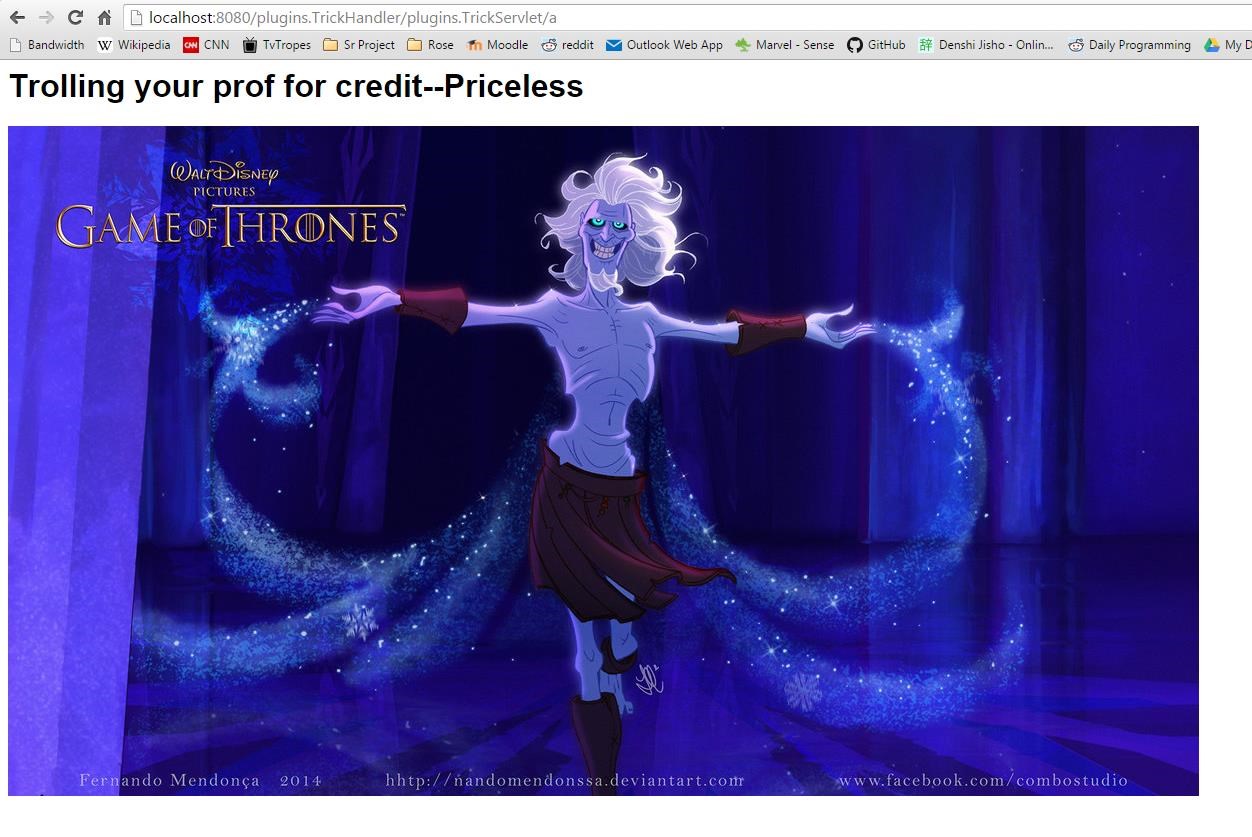


Page #4:

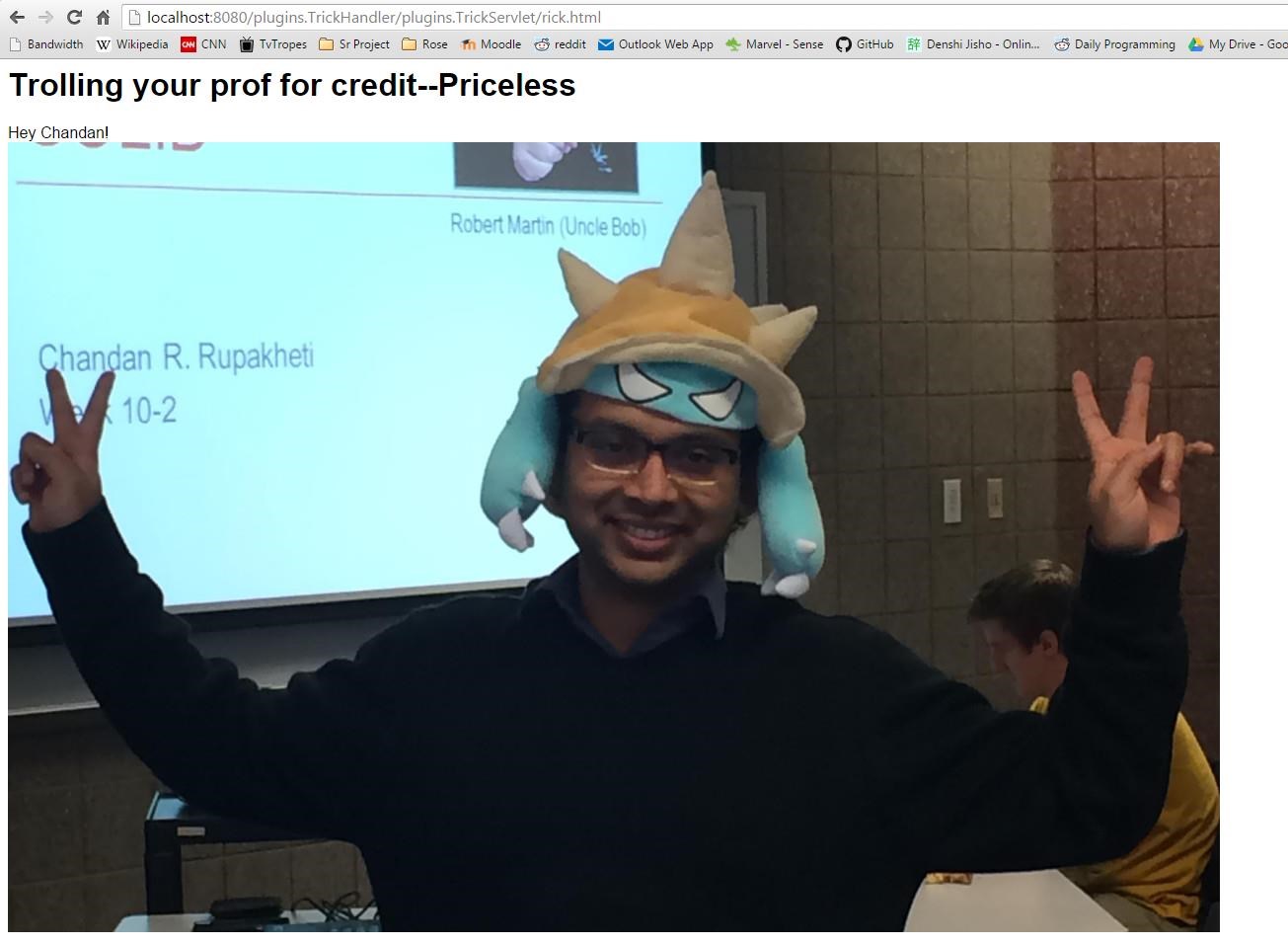
Page #5:



Notice that the URIs are the same for each one. But it doesn’t have to be “index.html”:



Funny thing even if you request the html that it redirects you to, you may not get that html:



4. Feature listing and assignment:

Assignment: Same-Room Separate-Coding, Mutual Despair

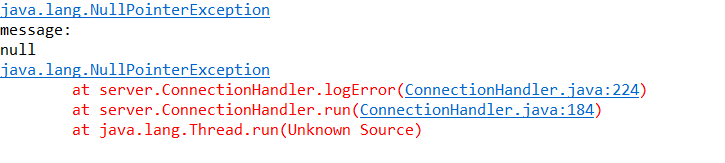
Features:

Testing Tools and Client--Greg

Server and Database

Change History:

All red and underlined text is changed.

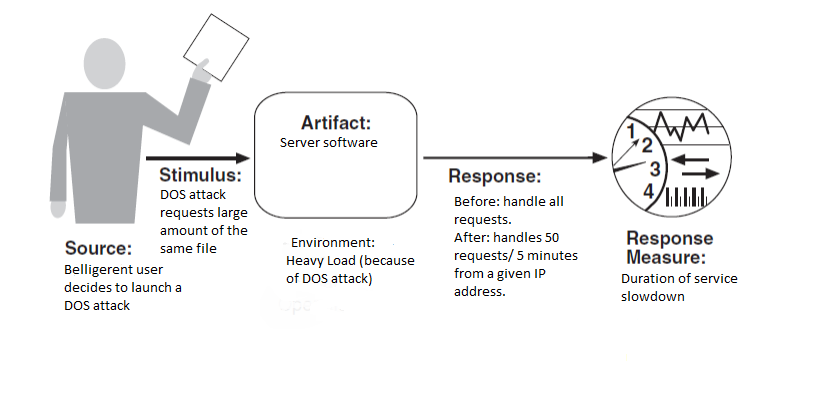


# Architecture Evaluation and Improvements

## Availability:

Name: DOSing

A1.1



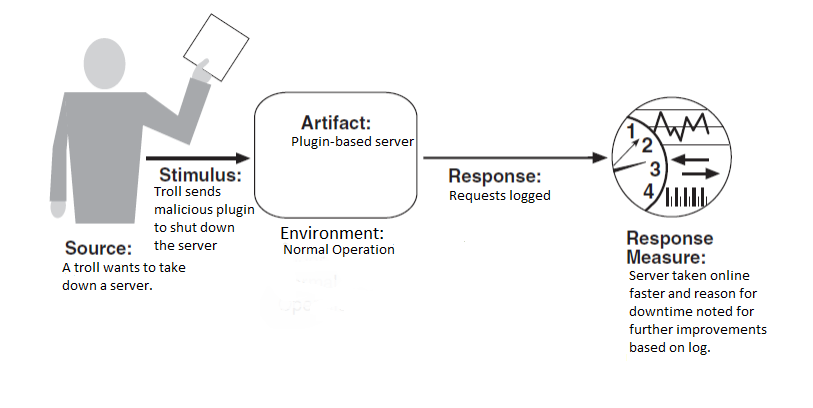
A1.2 We pointed our DOSing attempts at http://<ip-address>:8080/plugins.BasicHandler/plugins.BasicServlet/web/src/Untitled.gif. We used this because the .gif file was easy to test: it should take approximately 1.2 seconds to run through. Our base line number was 1.48 seconds (accounts for reaction time)

A1.3 When DOSing was performed the time to finish the gif jumped to 5.06s.

A1.4 We set it up to send no data if the client’s IP address has send 50 requests in the past 5 minutes. This was done by creating a ThrottleService class. We also logged the IP address of any throttled users.

A1.5 When throttling was applied, and DOSing was performed the gif load time was 1.54s, well within the margin of error. Additionally the DOSer’s IP address was logged so that we can know who is throttled in case of issues.

Name: Unexplained Shutdown

A2.1

A2.2 L.E. caused a shutdown in server and then Greg had to fix it without asking her how she broke it.

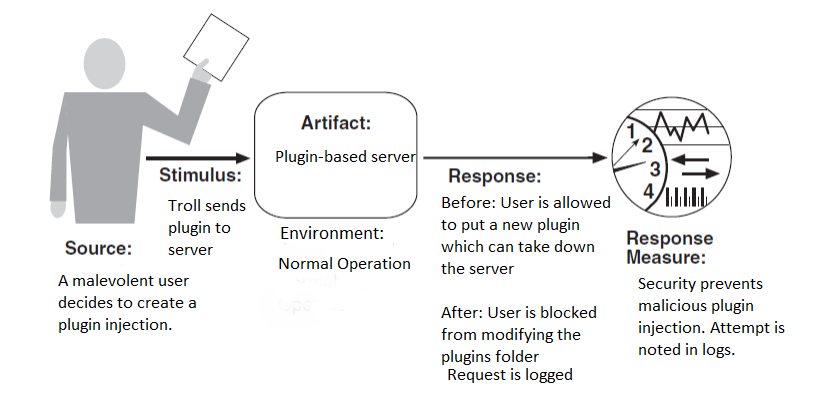
A2.3 It took 6m 35s for Greg to figure out the cause and restart the server from its broken state.

A2.4 We set the server up to log any error messages and any requests from users.

A2.5 When logging was implemented, Greg could see that L.E. had called the Malicious Plugin (see S1) and he was able to restart the server in 3m 9s.

## Security:

Name: MaliciousPlugin

S1.1

S1.2 We wrote a plugin that would force the computer into sleep, we then used a PUT request to deploy the plugin.

S1.3 Number of malicious plugins that can be added: infinite

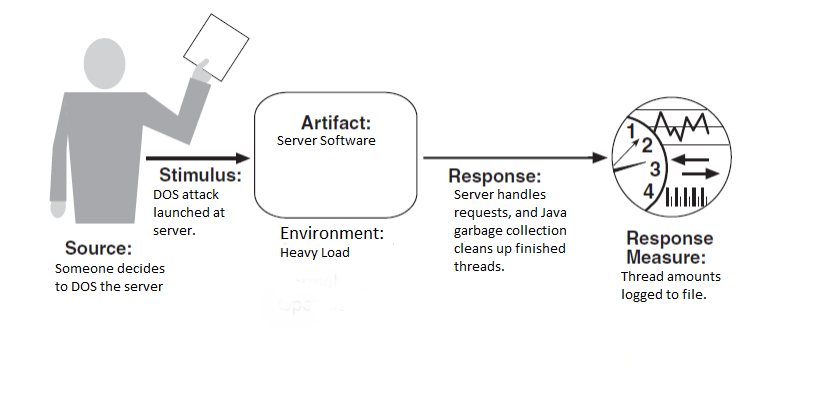
S1.4 We set the server up to prohibit any requests of any kind that wanted access to the plugins folder and returned a 403 Forbidden response.

S1.5 After we prohibited access to the plugins folder, the number of malicious plugins that can be added:

S2: See A1

## Performance:

Name: ThreadBomb

P1.1

P1.2 L.E. ran multiple DOSing clients against Greg

P1.3 When L.E. ran multiple DOSing clients against Greg, his thread count reached 223 and slowed down the server immensely.

P1.4 We discovered that a threadbomb was actually not a problem. Therefore we didn’t implement anything. If it were a problem, we planned to implement a system whereby by closing inactive connections, we would reduce the thread count.

P1.5 See P1.4

P2: See A1

## API Documentation

1. Marauders’ Database
   1. About the Application
      1. Our application is designed to provide possibly useful information to users who know how to access it, and to confuse, bewilder, or otherwise confound all other users.
   2. API Design:
      1. **F1: Retrieve a current Hogwarts Student’s Location**
      2. **Method:** GET
      3. **URI:** /Marauders/v1/student?passphrase=<passphrase>&name=”George Weasley”
      4. **Request Body:** <none>
      5. **Response Body:**  
         {  
            “payload”: {“name”: “George Weasley”,    “house”: “Gryffindor”, “location”:”The Kitchens”}
      6. }
      7. **Development Status:** DONE
      9. **F2: Retrieve a current Hogwarts Student’s Location Without Passphrase**
      10. **Method:** GET
      11. **URI:** /Marauders/v1/student?passphrase=<no/wrong passphrase>&name=”George Weasley”
      12. **Request Body:** <none>
      13. **Response Body:**  
          {  
          “code”: 394,  
             “message”: “Mr Raven presents his compliments to Donald Trump and begs him to keep his abnormally large weasel toupee out of other people's business.
      14. Ms Batty agrees with Mr Raven and would like to add that Donald Trump is an ugly git.”
      15. }
      16. **Development Status:** DONE
      18. **F3: Retrieve a list of nearby Hogwarts Students**
      19. **Method:** GET
      20. **URI:** /Marauders/v1/student?passphrase=<passphrase>
      21. **Request Body:** <none>
      22. **Response Body:**  
          {  
            “payload”: [{ “name”: “George Weasley”,    “house”: “Gryffindor”, “location”:”The Kitchens”},  
          { “name”: “Fred Weasley”,    “house”: “Gryffindor Ghosts”, “location”:”The Kitchens”}]
      23. }
      24. **Development Status:** DONE
      26. **F4: Retrieve a list of nearby Hogwarts Students with the wrong passphrase**
      27. **Method:** GET
      28. **URI:** /Marauders/v1/student?passphrase=<no/wrong passphrase>
      29. **Request Body:** <none>
      30. **Response Body:**  
          {  
          “code”: 394,  
             “message”: “Mr Raven would like to register his astonishment that an idiot like that ever became a Presidential Candidate.
      31. Ms Batty bids Donald Trump good day, and advises him to wash his weagle, the slime-ball."}
      32. **Development Status:** DONE
      34. **F5: Add a new Hogwarts Student**
      35. **Method:** PUT  
          **URI:** /Marauders/v1/student
      36. **Request Body:** {“passphrase”:”<passphrase>“, “name”: “James Sirius Potter”,    “house”: “Gryffindor”, “location”:”The Great Hall”}
      37. **Response Body:**  
          <none>
      38. **Development Status:** DONE
      40. **F6: Update a Hogwarts Student Location**
      41. **Method:** POST  
          **URI:** /Marauders/v1/student
      42. **Request Body:** {“passphrase”:”<passphrase>“, name”: “James Sirius Potter”, “location”:”The Great Hall”}
      43. **Response Body:**  
          {  
           “payload”: {“name”: “James Sirius Potter”,    “house”: “Gryffindor”, “location”:”The Great Hall”}}
      44. **Development Status:** DONE
      46. **F6: Delete a Hogwarts Student**
      47. **Method:** DELETE  
          **URI:** /Marauders/v1/student
      48. **Request Body:** {“passphrase”:”<passphrase>“, name”: “James Sirius Potter”}
      49. **Response Body:**  
          <none>
      50. **Development Status:** DONE