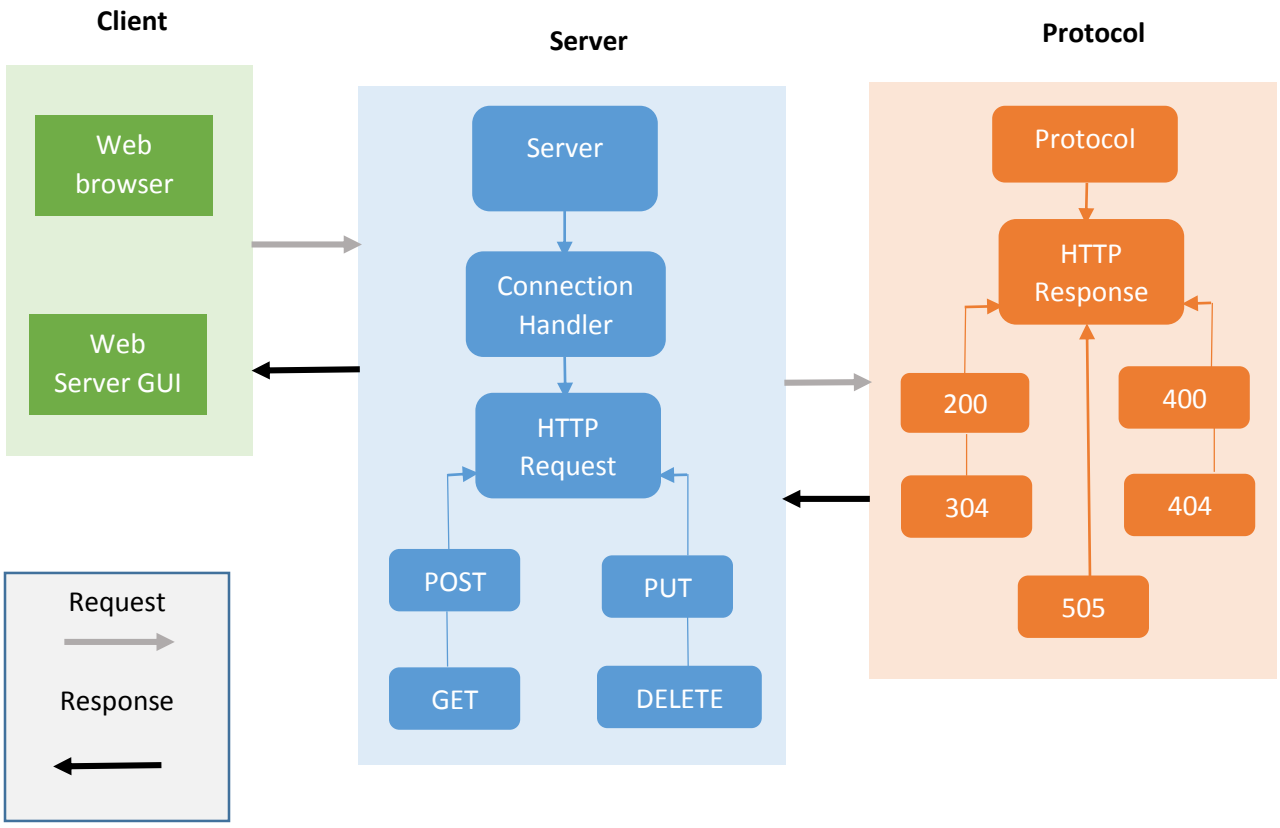
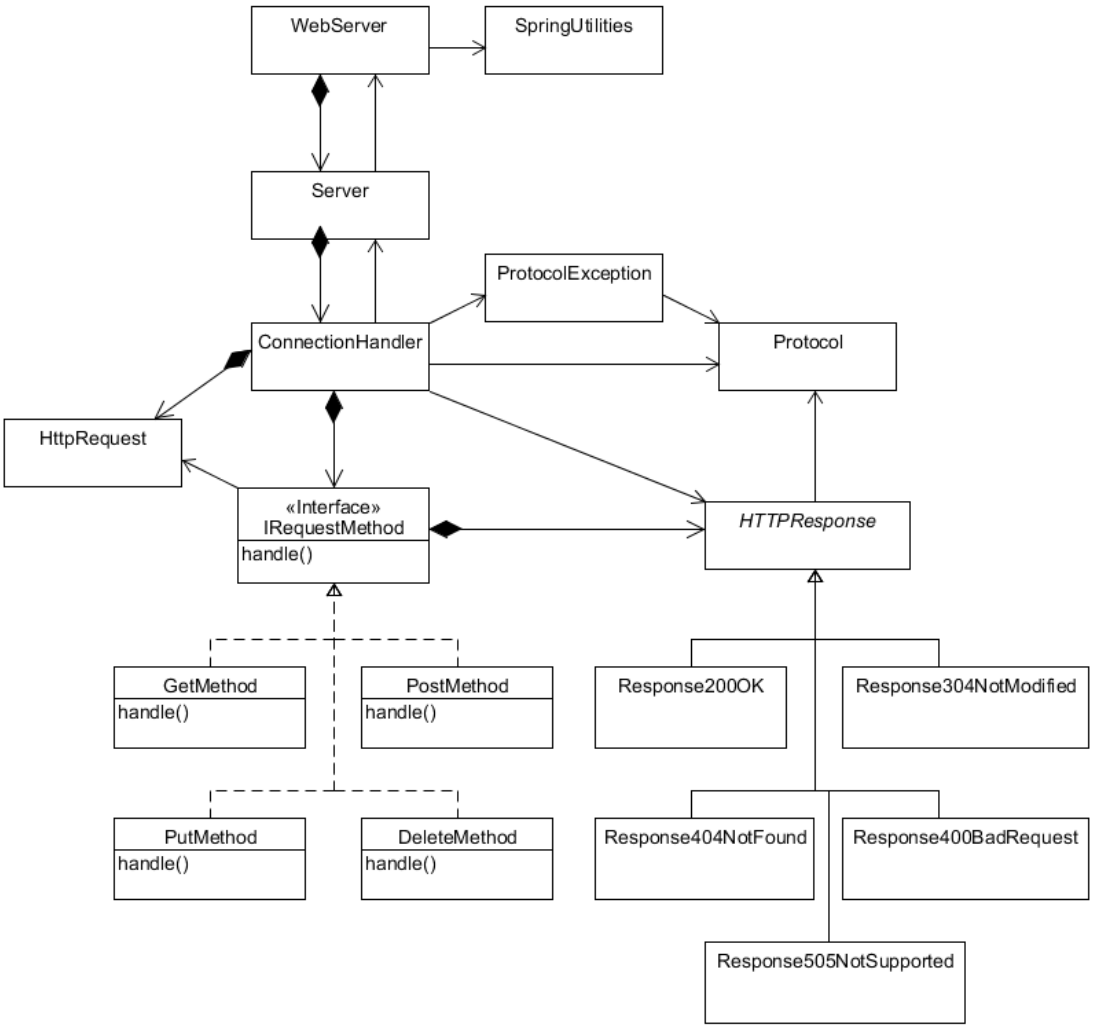
Milestone 1 – Web Server Report

Team RAJ – Angelica Rodriguez, John Krasich CSSE 477

Architecture Diagram



Detailed Design



Our refactoring of the web server utilized the following design patterns:

Strategy Pattern – The IRequestMethod interface allows for the various implementations of request handling to be completed in unique classes. This way, additional request handling can implanted with minimal changes to the ConnectionHandler class – simply add the new request to the ConnectionHandler's map of request methods.

Bridge Pattern – The abstract HTTPResponse class is used by the ConnectionHandler to write the generated response back to the client. However, the responses vary depending on the response code. Using a bridge pattern, each different response's implementation can be handled in separate classes without the ConnectionHandler needing to have any knowledge of how it is implemented.

Further Improvements

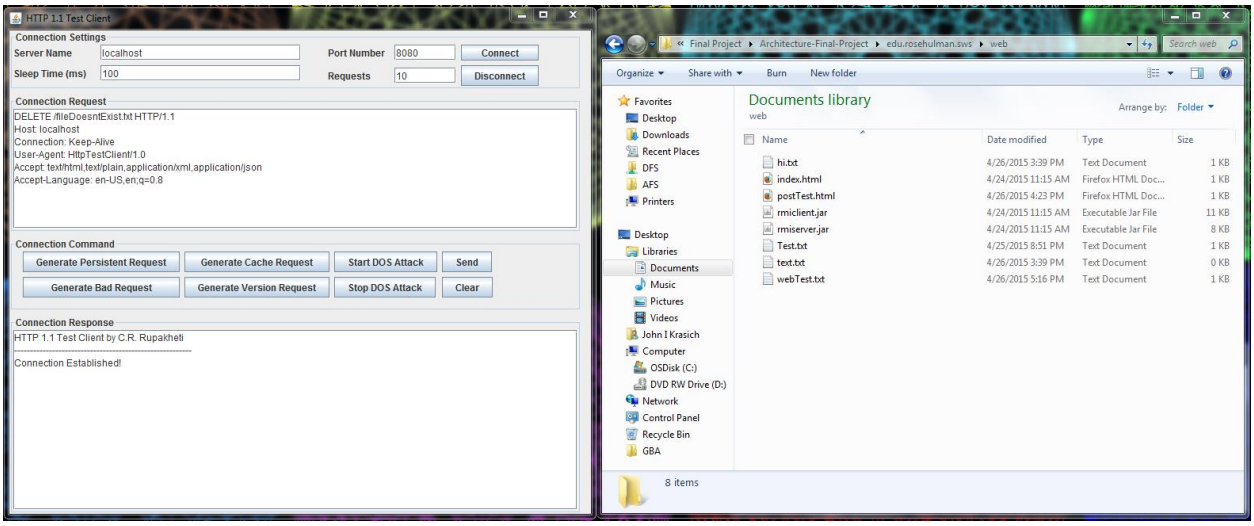
One area that can be further refactored would involve dividing the responsibilities of the ConnectionHandler class separately between requests and responses. The “run” method is rather long – breaking this up into different methods (or different classes) would make the code much more organized and easier to understand.

Test Report

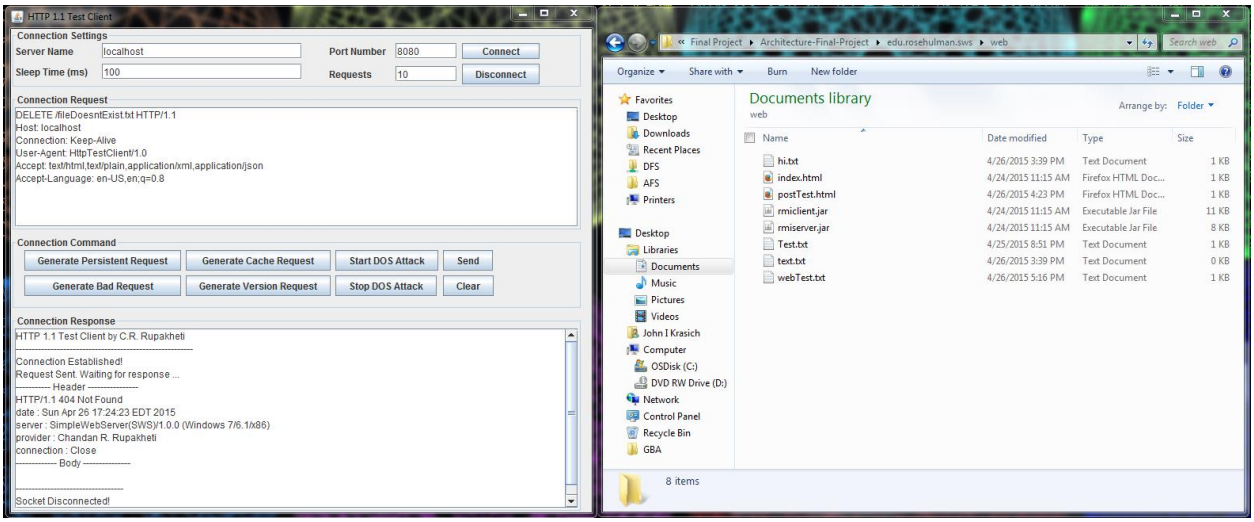
Testing Utility

DELETE of Non-Existent File

Before:

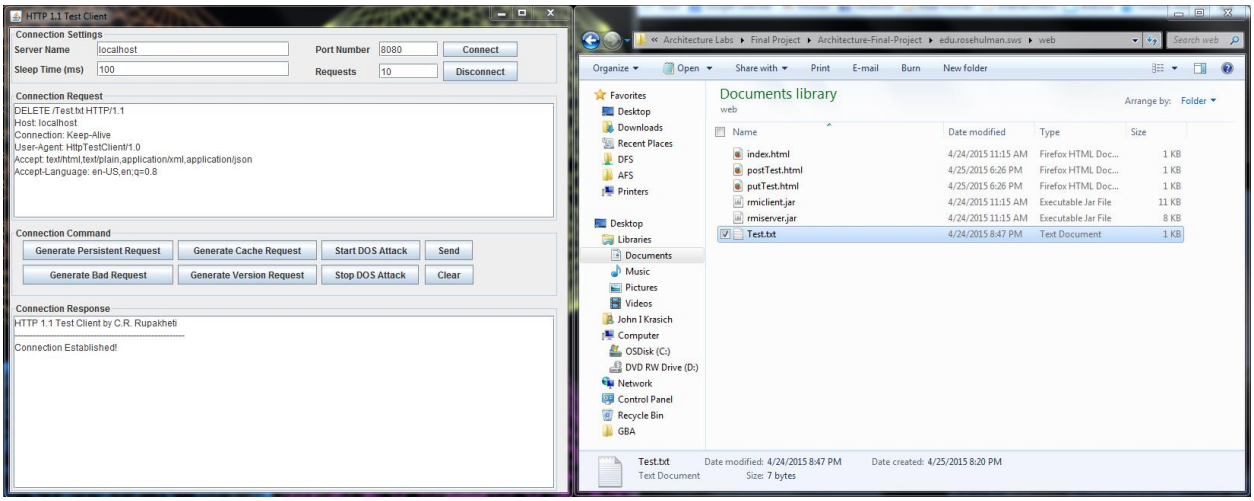


After: Response 404 Not Found

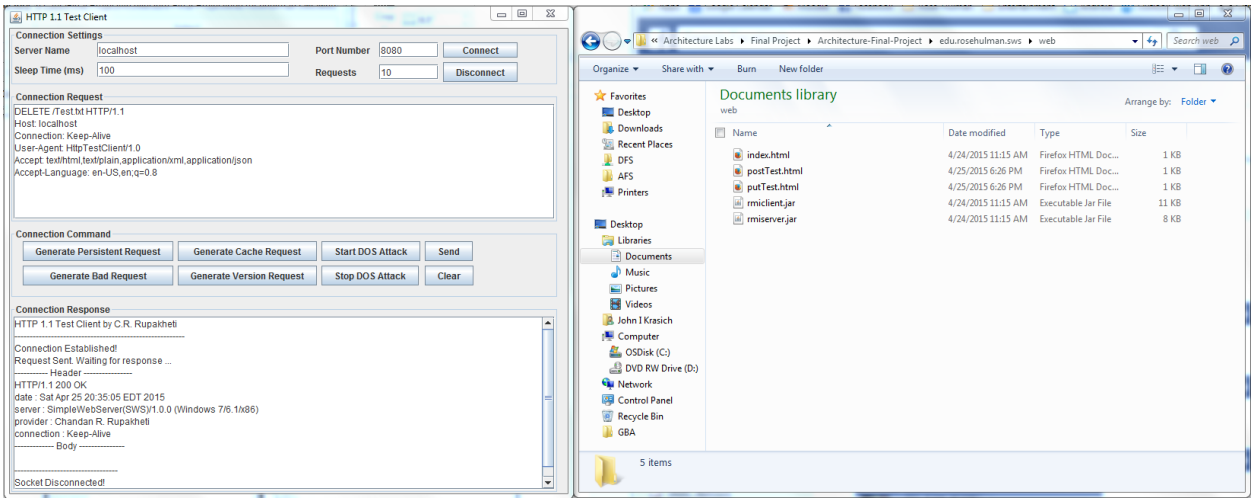


DELETE

Before:

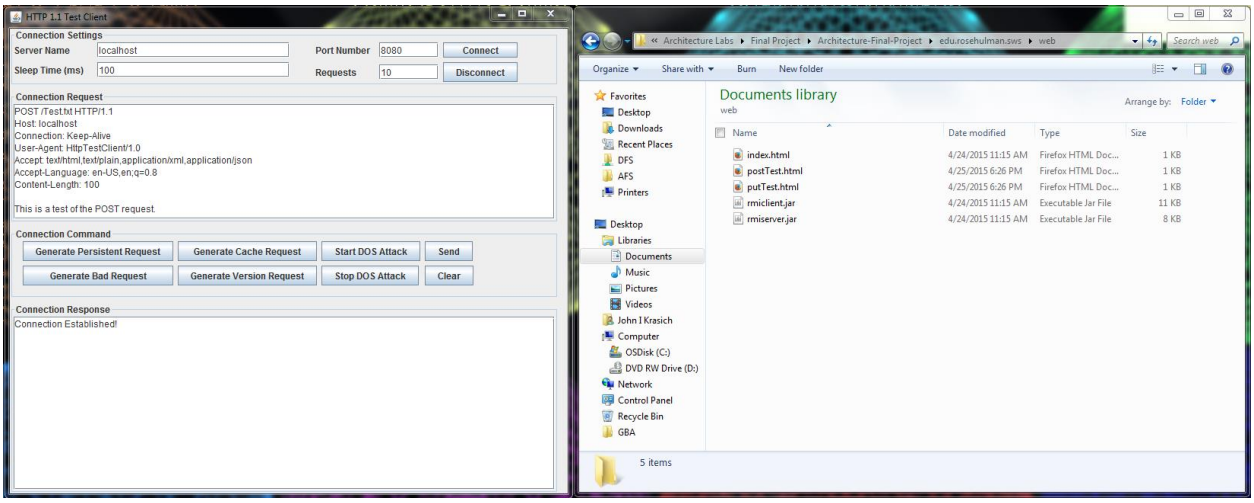


After: Response – 200 OK. File Test.txt has been deleted successfully

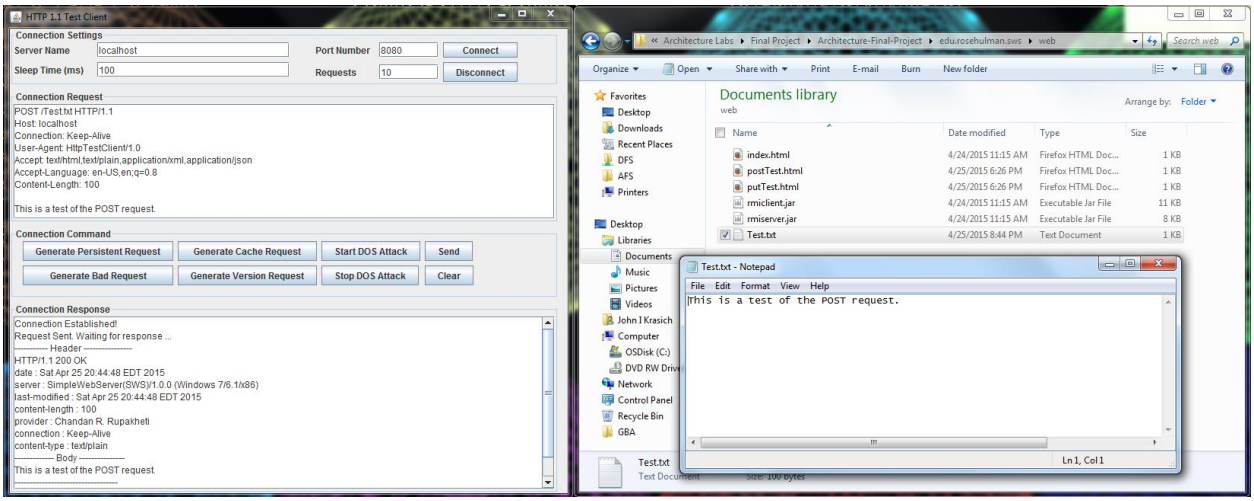


POST

Before:

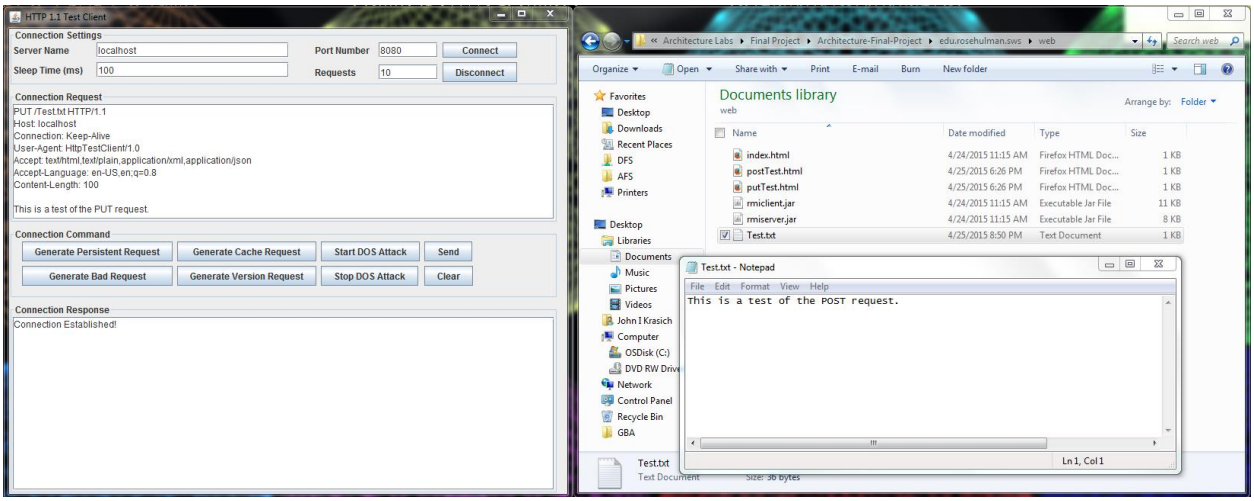


After: Response 200 OK. The file Test.txt has been created and filled with the body of the request.

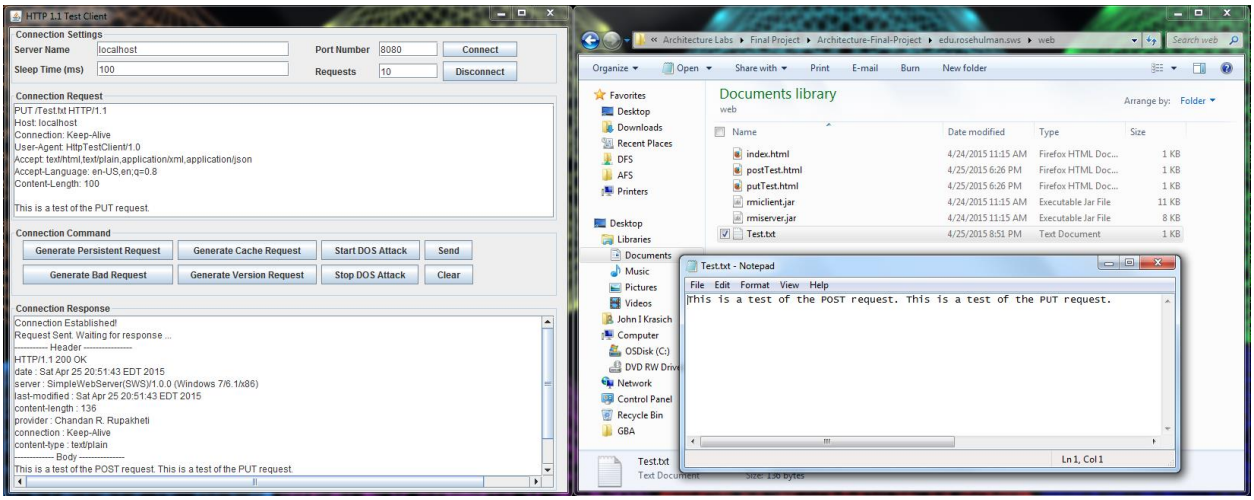


PUT

Before:



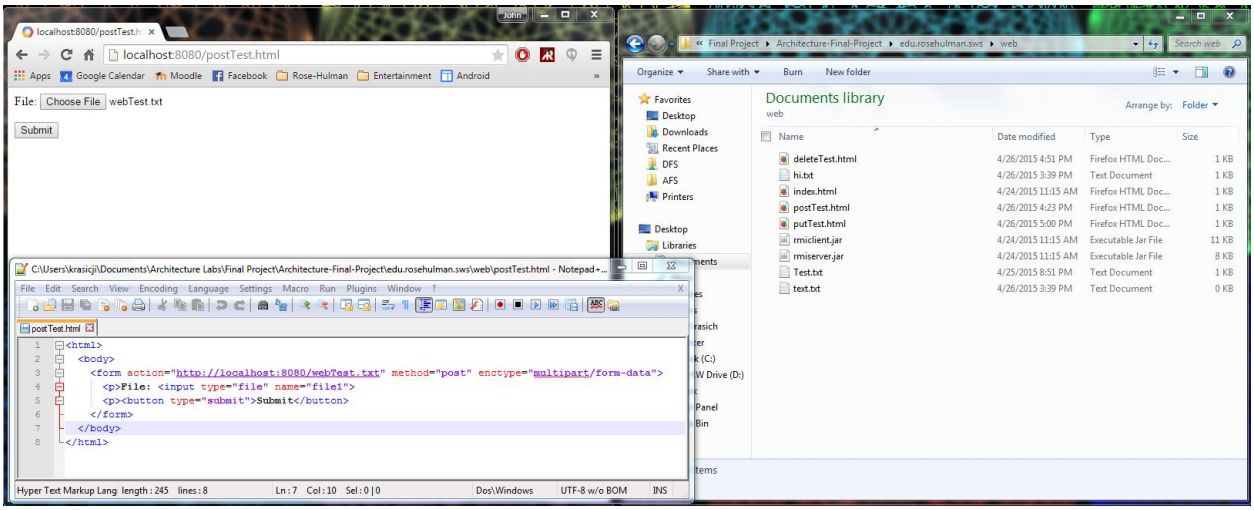
After: Response 200 OK. The body of the request was appended to the Test.txt file.



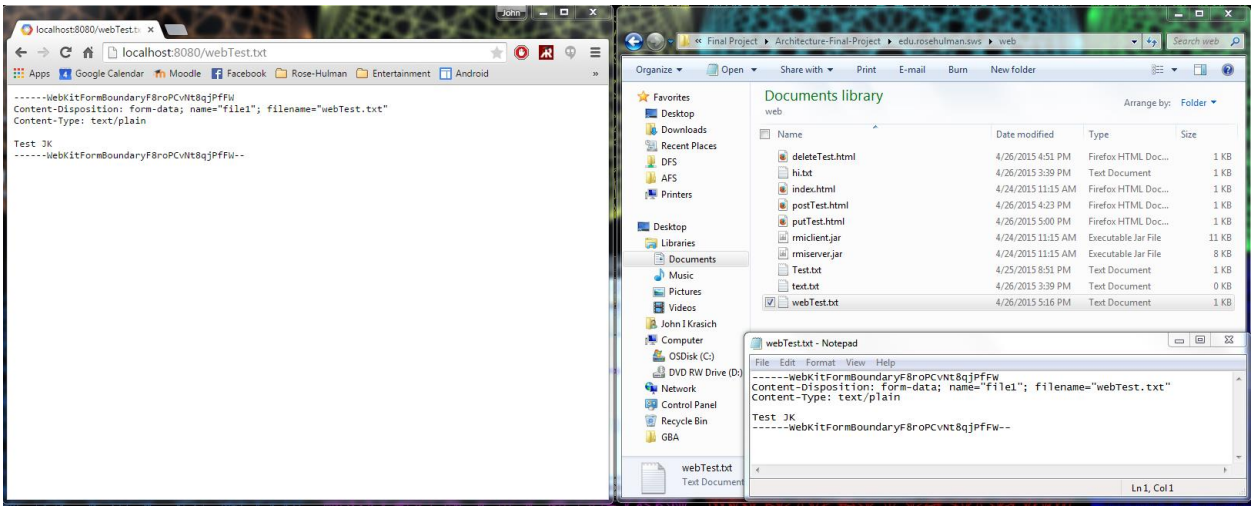
Web Browser

POST

Before:

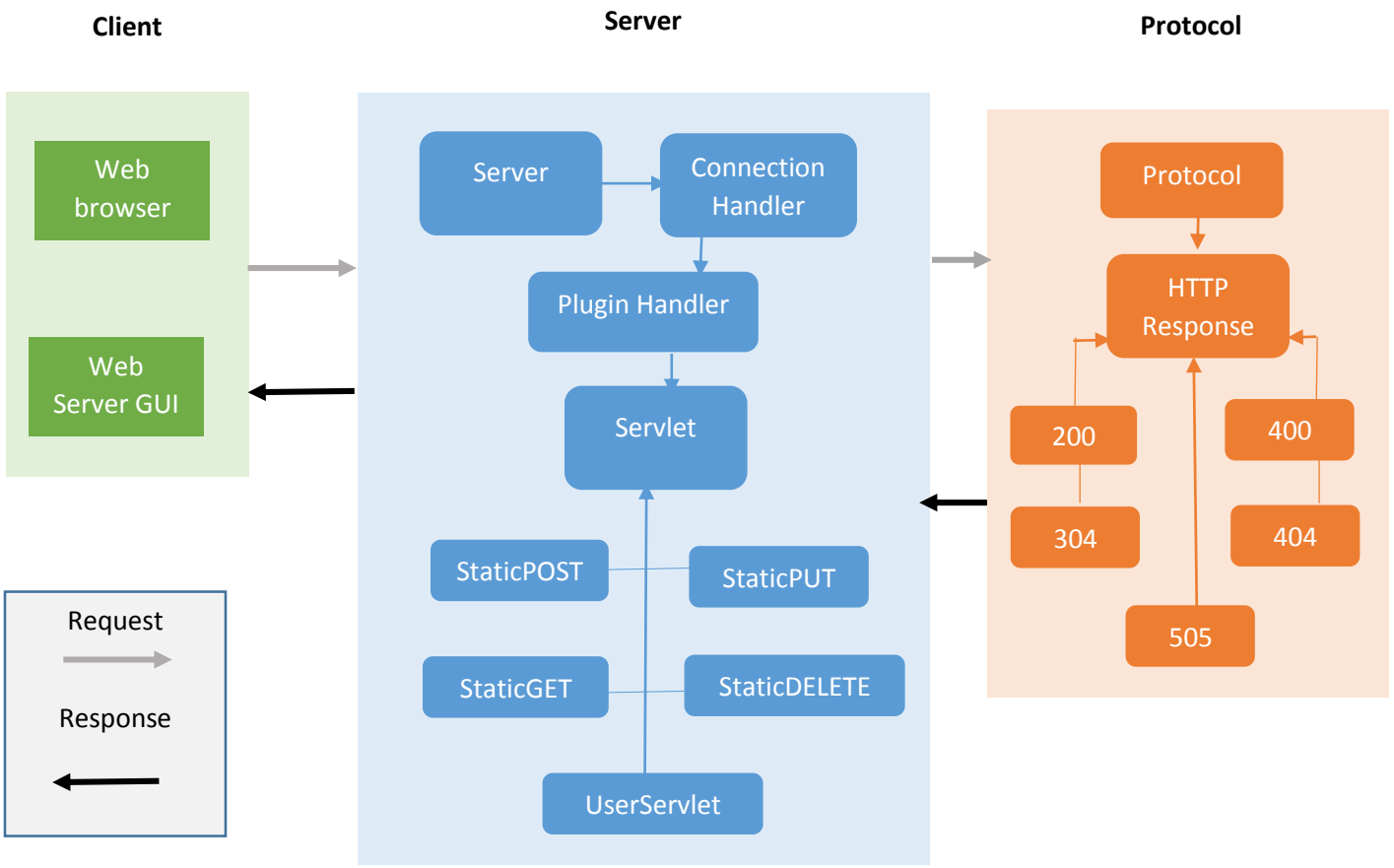


After: Response 200 OK – the text of the file was written into webTest.txt and returned as the body in the response.

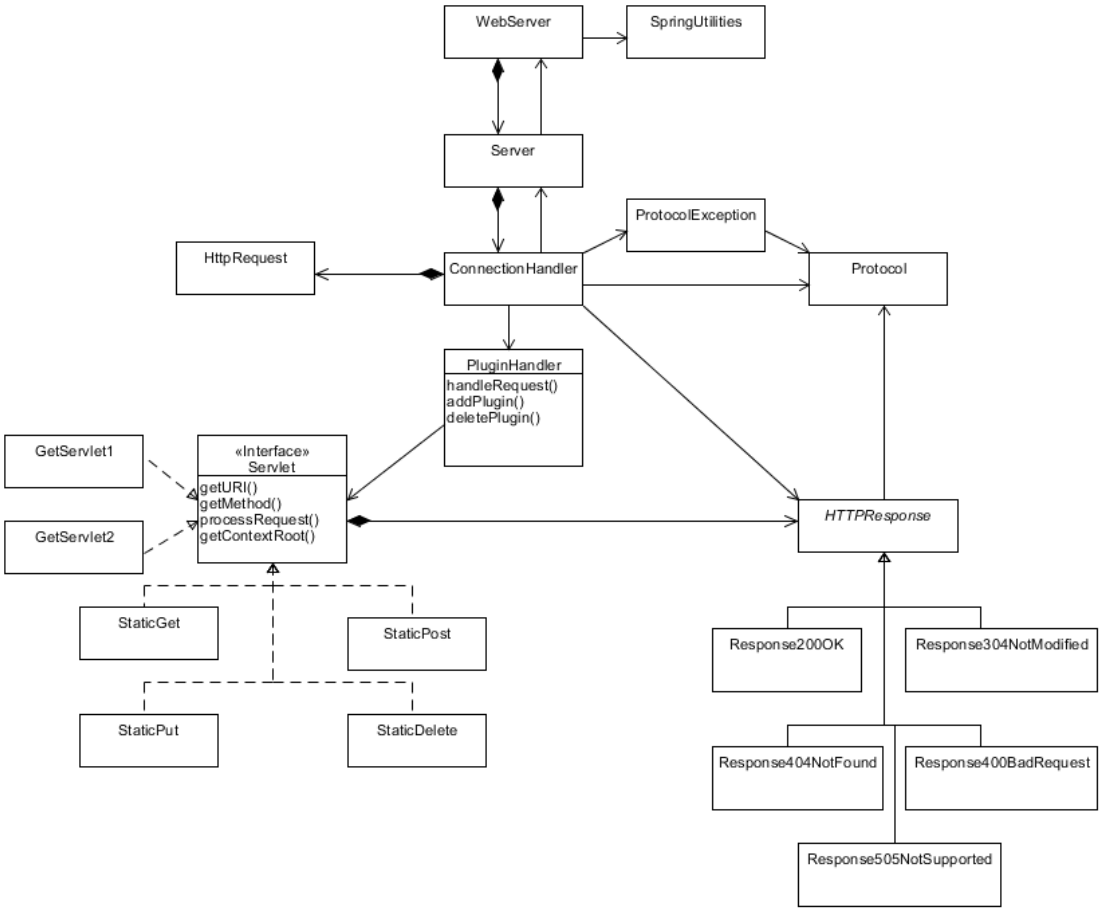


Change History – MS2

Updated Architecture Diagram



Updated Detailed Design



Brief Description:

The most significant changes made for this milestone were the addition of the PluginHandler class and Servlet Interface. The PluginHandler watches a Plugins directory for the addition of Jar files from which new servlets would be dynamically included into the web server. The ConnectionHandler communicates with the PluginHandler, passing along the request for the PlugHandler to process correctly. This is done through a HashMap, which relates the context root to a second HashMap that stores the servlets and their respective URIs. Any servlet must implement the Servlet interface, which contains information necessary for the PluginHandler as well as its custom request processing method. The basic GET, POST, PUT, and DELETE methods from MS1 became “static servlets” that will be run if no plugin is found for that kind of request.

Feature Listing & Assignment

Angelica Rodriguez

- W-1: GET Requests
- W-2: POST Requests
- W-3: PUT Requests
- W-4: DELETE Requests

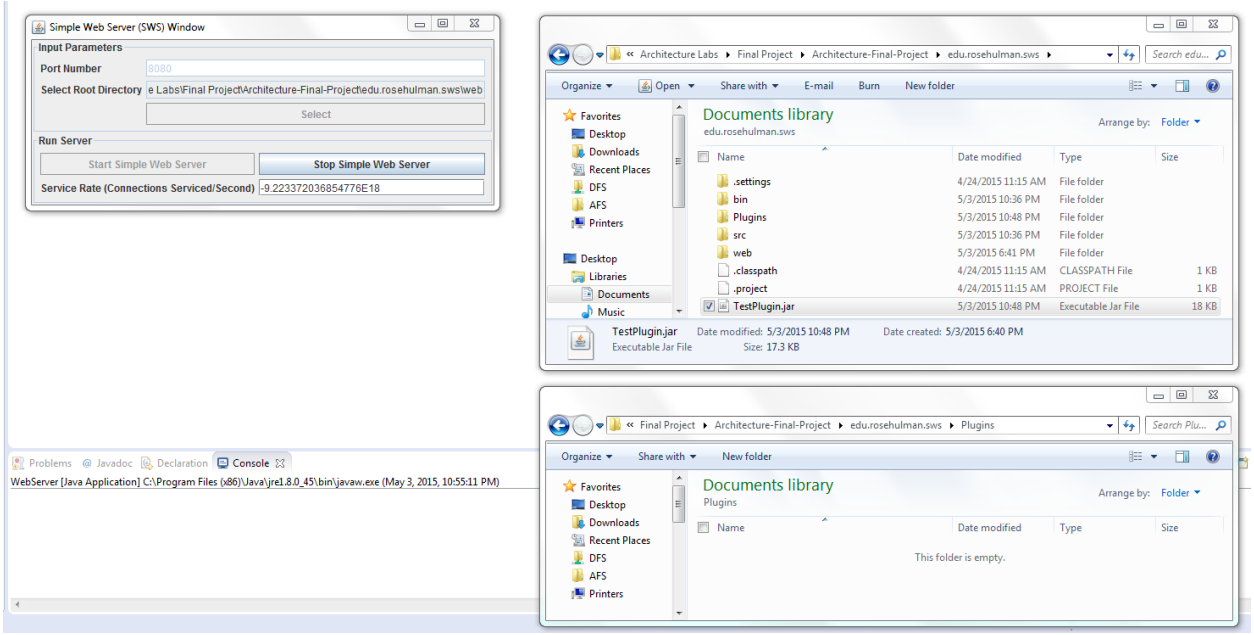
John Krasich

- P-1: Dynamic Loading
- E-1: Root Context and Configurable Route
- Test Report

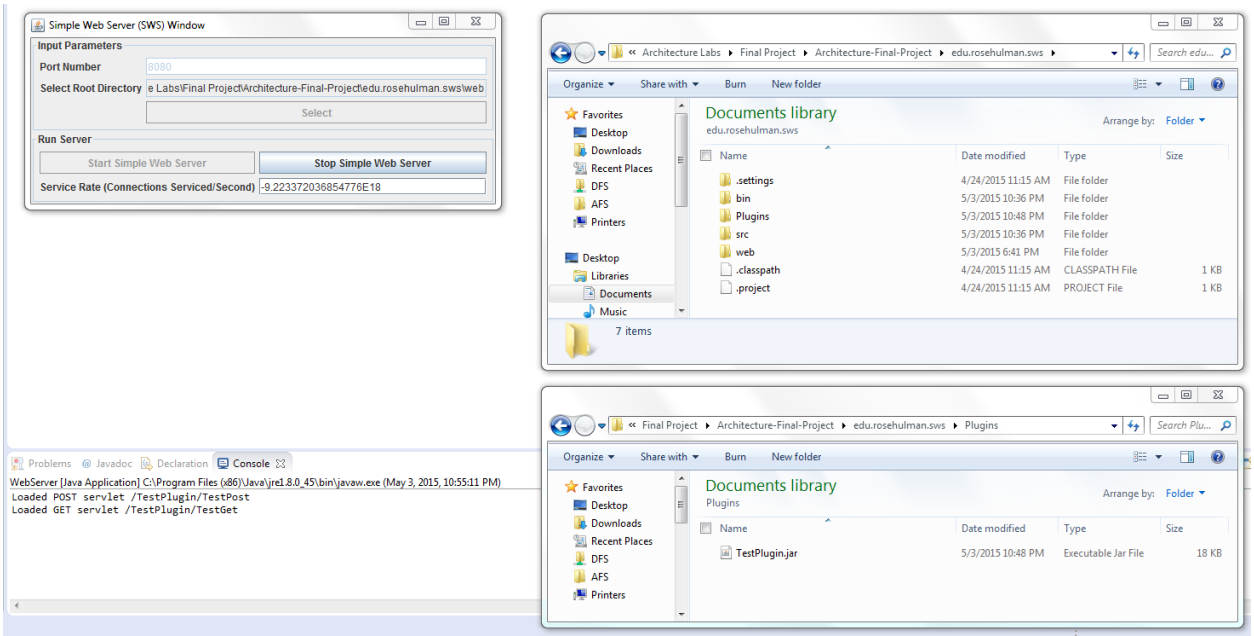
Test Report

Plugin Addition

Before:

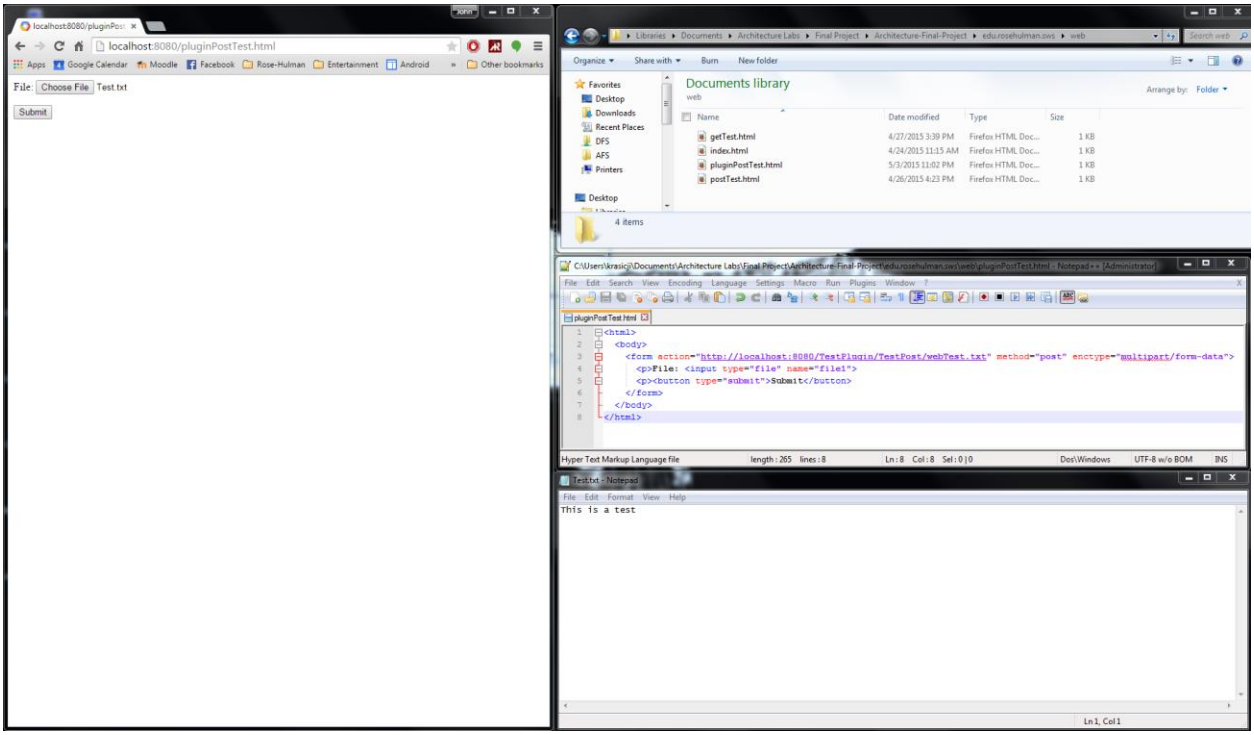


After: the plugin containing two servlets were dynamically loaded into the web server.

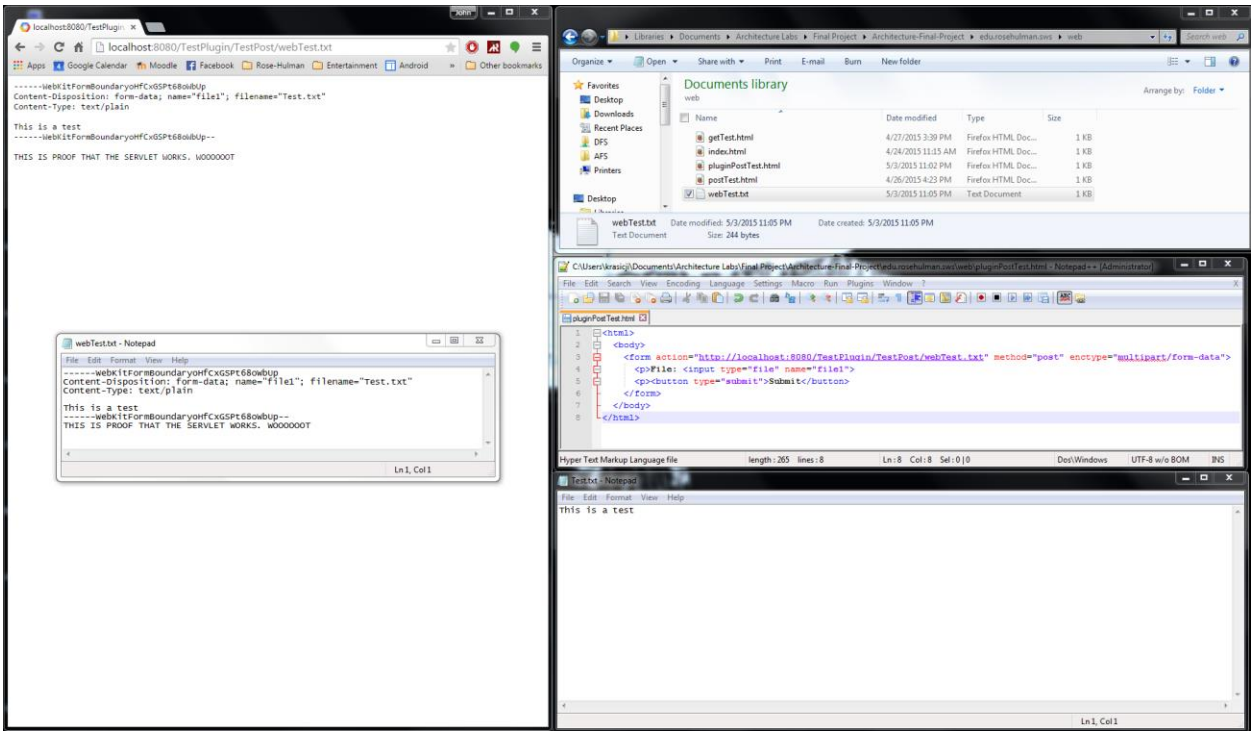


POST

Before:

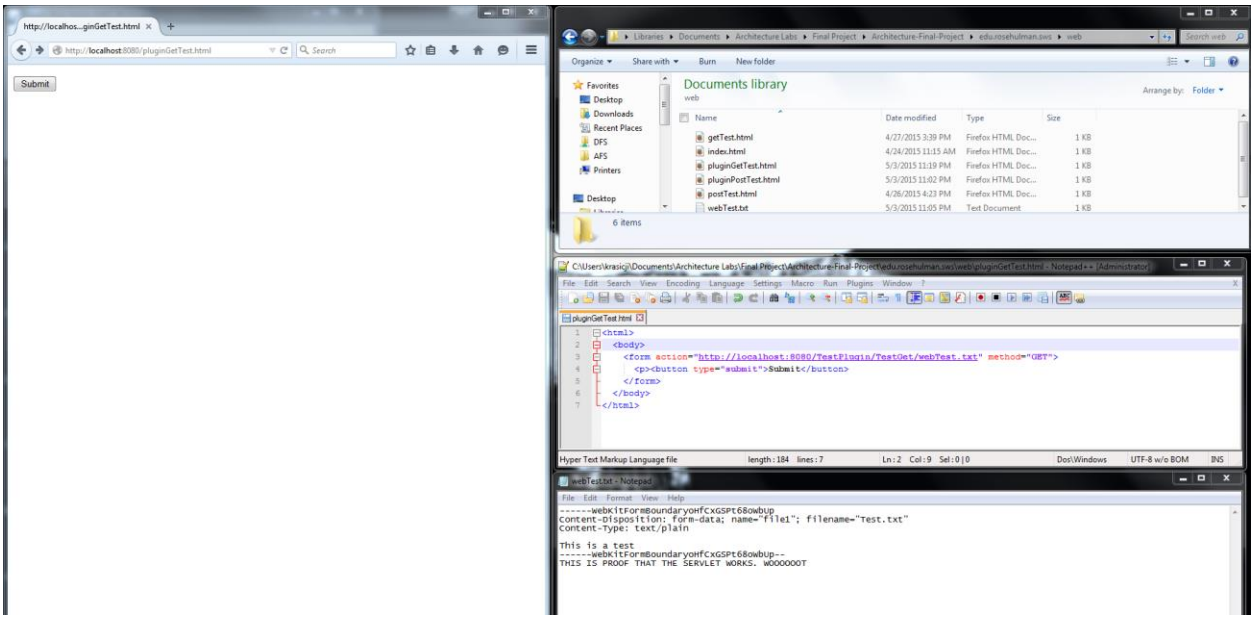


After: the servlet appended “THIS IS PROFF THAT THE SERVLET WORKS. WOOOOOT” to the file + body.

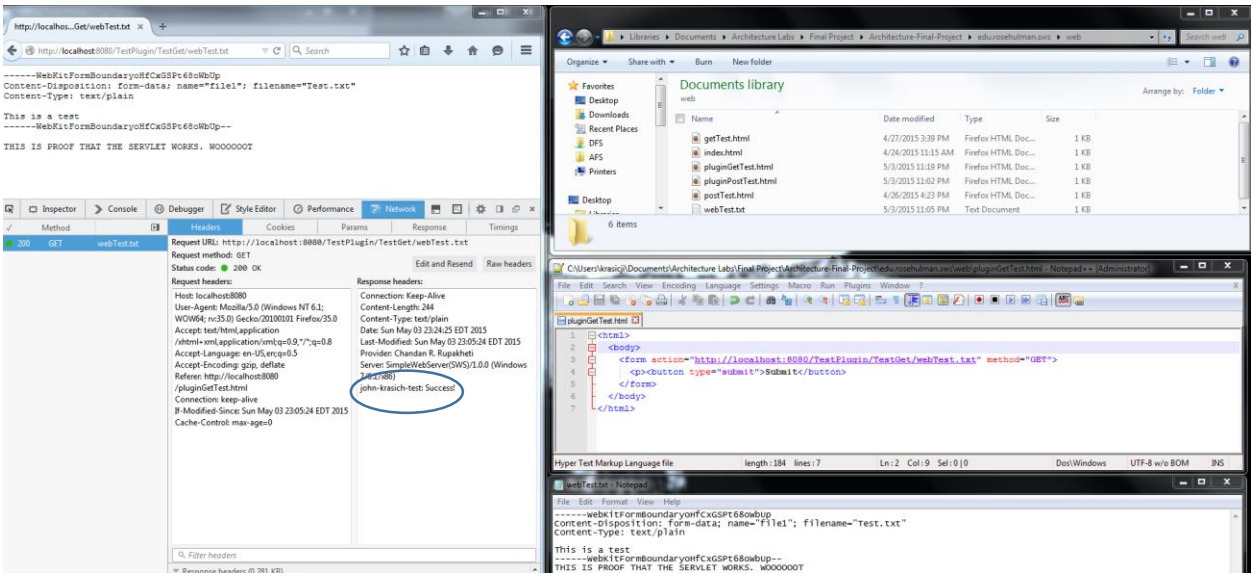


GET

Before:

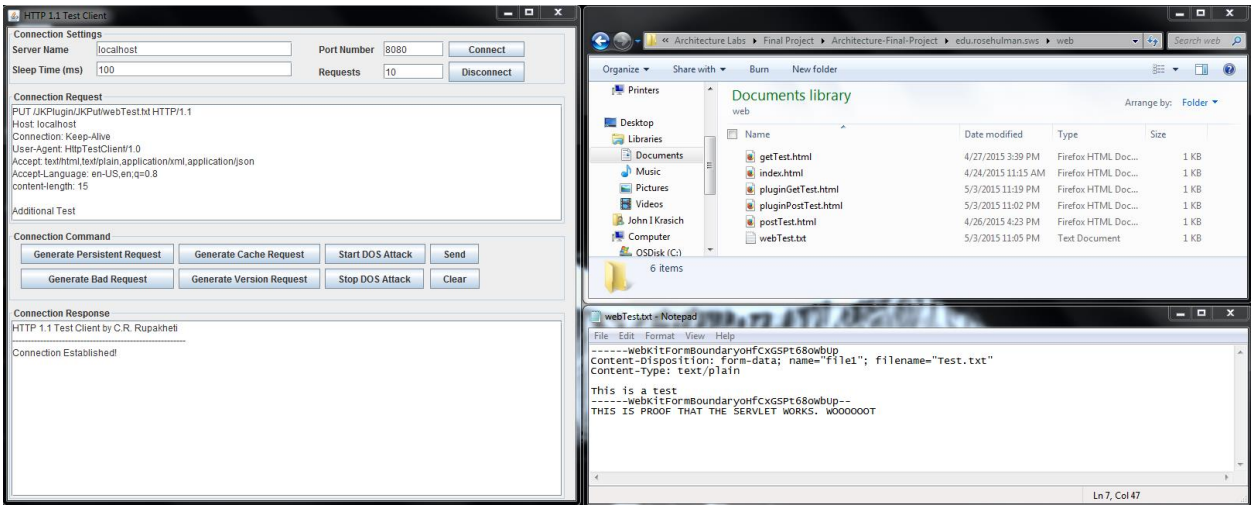


After: the servlet appended an additional header to the response.

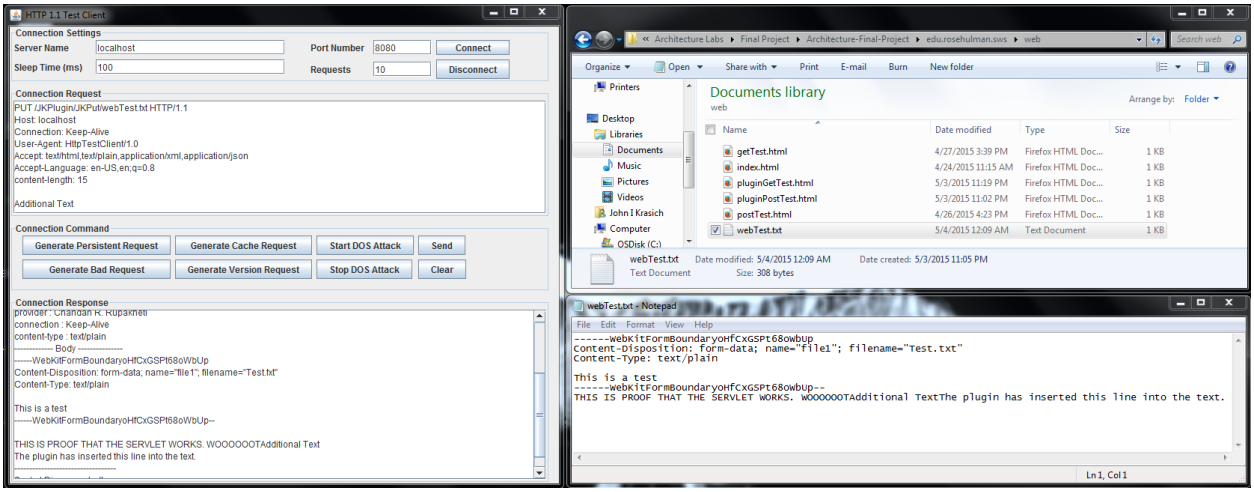


PUT

Before:

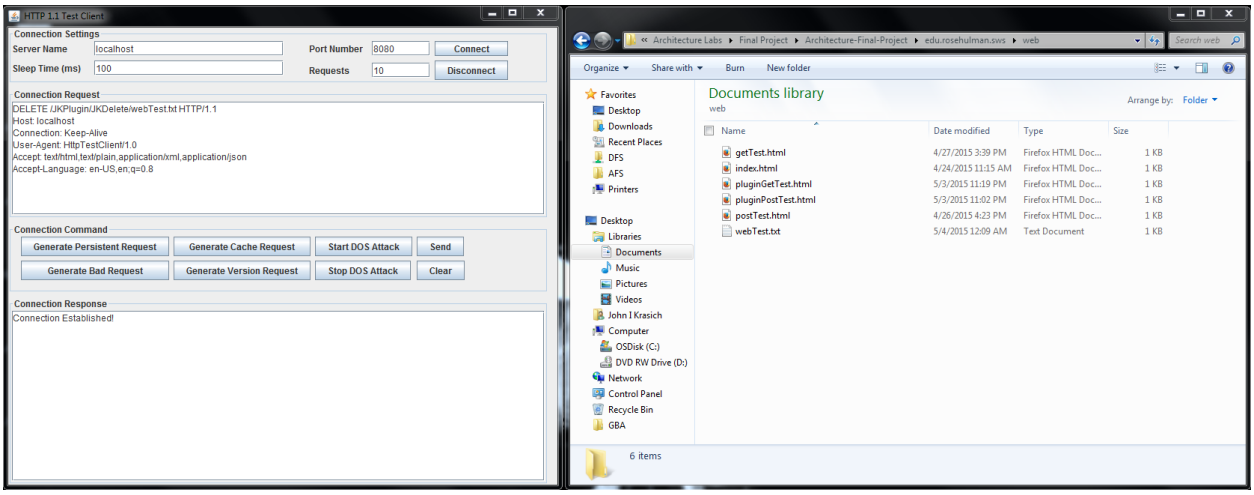


After: the servlet appended the extra text “The plugin has inserted this line into the text” into the file.

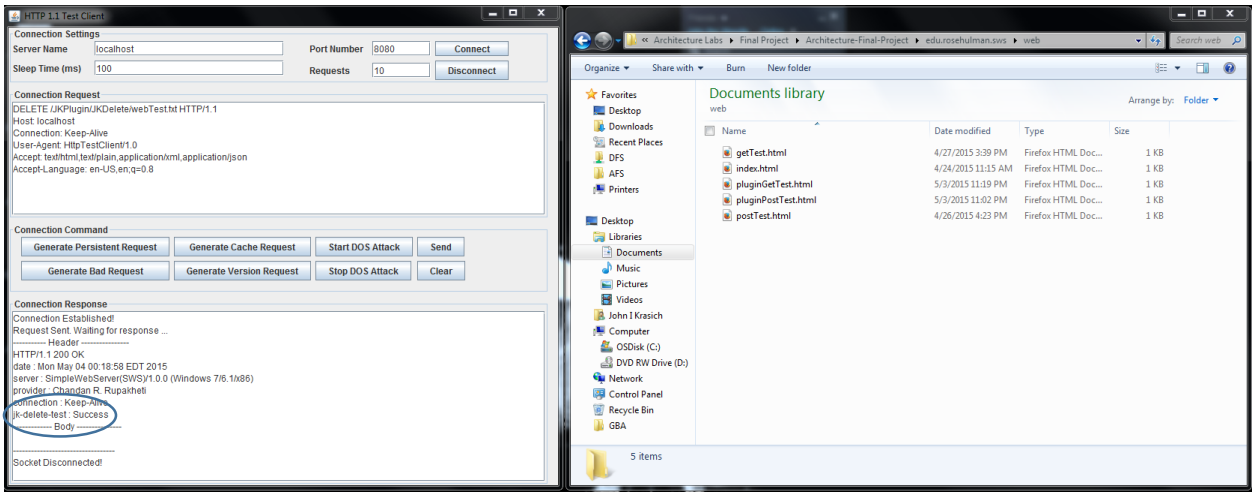


DELETE

Before:



After: the servlet appended an extra header into the delete response.



Future Improvements

One idea we did not get to try but wanted to was to have the users supply a configuration file for the servlets with the information necessary, rather than have them hardcode the request code and create the JAR file. This would be a significant improvement because it would allow it to be easily modifiable, since they could make changes dynamically.