CS290-400 Edmund Dea
3/4/17 ID# 933280343

# Activity: Practice!

This past week required a lot of learning through practice in using Node Express, sessions, and using all the key lessons from the past 8 weeks. When I first started learning sessions, I forgot to install the package for Express Sessions, so I resolved this by typing in “npm install express-session” at the Flip command prompt. After having downloaded a new version of the CS290-server-side-examples, I realized that I didn’t have the rest of the packages installed that was required for using. I was modifying the forms-demo.js in the express-forms example to add sessions, so when I tried to run Node, it threw an error saying that I didn’t have body-parser installed, so I resolved that by running “npm install body-parser” and, just to be sure I installed the rest of the packages, I ran “npm install express-handlebars.”

After installing all required packages, I added example code to add a session count based on the number of times a person visited the page. However, Node threw an error saying “Error: Failed to lookup view "counter" in views directory "/nfs/stak/users/deae/CS290/Week8/CS290-Server-Side-Examples/express-forms/views”. This error occurred because I forgot to create a handlebars template for the counter page. The counter page worked successfully after I create the template page. When I attempted the session To-Do list activity using GET/POST, I initially added all the example code for the handlebars templates and the GET request, but I didn’t add the POST request code yet. I first wanted to test the GET request code by itself. The newSession webpage would load a create a new session but, after I submitted, I kept getting a 404 error message. This happened because newSession.handlebars contained a form that was set to method=”post”, whereas I didn’t add the POST request code yet. After changing the form to a GET action instead, the page submitted back to itself. After changing the form back to POST method and adding the POST request example code, the To-Do list worked.

When I attempted to create two consecutive HTTP requests which get and display data using OpenWeatherAPI, I ran into one conceptual issue. I had trouble understanding exactly what “next” did in a Node request. I thought that next passed the error onto the next function to handle an error but I didn’t quite understand who would handle the error and how. After doing an internet search on next to supplement the lecture material, I found websites explaining Express Routing Middleware. I came to understood that next was a reference to a callback function to allow HTTP requests to be handled by subsequent items in the middleware stack. I’m still struggling to fully understand middleware but will continue to study this topic over the next week. Other than that, I was able to use the add code to express-forms to query OpenWeatherAPI and post to HttpBin successfully.