Claudia Gajes, Colleen Baksi and Sidney Mantwill

Project Deliverable 2

Team #5

Project 3

Our team's project objective was to get a Node.js webserver to interact with a MySQL database through Docker containers. The objective of our web server was to return a corresponding Pennsylvania city based on an area code entered by the user. On our web server site there is a list of valid options, but currently the user is only able to search by using the command prompt, and not the webpage itself. The idea is that the server should connect to the mySQL database and access a table containing Pennsylvania area codes mapped to the locations corresponding to the area codes. The server would run a query on the mySQL table and if the input area code matches an area code in the table, the corresponding city is returned. If the inputted area code does not exist in the table, an empty array output is then returned. We had no exception handling for non-numeric characters inputted by the user, so as a result an error occurs and the server crashes.

Our project functions on a local machine, working perfectly as intended with what we have so far. When running the server, the user needs to connect to localhost and the port number (8080). The server then connects to the mySQL database, prompts the user for an input, and runs the query. Unfortunately, we ran into a lot of trouble pushing the mySQL server to Docker, but we were able to clone the Node.js web server to Docker by uploading the JavaScript file, html file, and other file dependencies to a GitHub repository and cloning it to Docker. When we run our project in the Docker terminal on a Windows desktop, the server successfully runs and connects to the network. This is done successfully by using the default IP address that Docker is configured to, along with the port number (8080). Since the mySQL server was unable to be containerized, the web server throws an error when it attempts to connect to it. To circumvent this error when we did testing on Docker, we commented out the code where the server would attempt to connect to the database, and where the database value was defined. This allowed the html data to load, and allowed the Node.js server run for long enough to prompt the user for an input. The user can input a code into the command prompt, but no matter what they input an error is thrown. This is because the server attempts to run a query on the mySQL database, but since no database was ever defined or connected to, it fails. As a result, the server then crashes. We also had trouble getting our project to work on the Docker branch in CloudLab. We were able to successfully clone the GitHub repository and then run the server. Due to a misunderstanding regarding which web address we needed to use, when attempting to access the webpage in the browser, we got an infinite load and unsuccessful connection after each test in the environment had to force stop the container.

We believe that we, for the most part, successfully met the requirements of the project's second deliverable. Though we did not meet the full time requirement when presenting, our technical presentation met the necessary criteria and followed the listed guidelines on D2L. Though we experienced technical difficulties when trying to deploy one of our in-class demonstration, we had the corresponding screenshots of the web server deployment, and had a successful demo of the Docker desktop component. We were able to smoothly continue our presentation even though we experienced a slight technical difficulty. As a team, we were able to show that we utilized cloud computing infrastructures successfully when using Docker, because we created an image and container, and then attempted to run the container in the browser via the localhost and an IP address. We were able to articulate what issues we experienced and how we planned to continue our work in the third deliverable, which makes us believe that we satisfied almost all of the deliverable two requirements.

We feel that we worked really hard on this project. We struggled early on by misunderstanding the project outline and its requirements, and we struggled with deciding whether or not we wanted to continue with Project 3. Ultimately, after heavily researching our potential options and discussing them as a team, we decided to create the Node.js server with a mySQL backend. Although the idea at its core is simple the actual execution was a big challenge, and it took a lot of time, effort, research, and teamwork to get the server to this stage. For the final deliverable, we would like to get the mySQL server on Docker. Even if the server doesn't work as we intend by the time deliverable three is due, we still hope we can test the connection between the server and the database network connection to figure out how to solve the problem we faced. We hope to get this working for the final deliverable but there are unfortunately not many resources using mySQL and Docker so everything from this point forward will mostly consist of trial and error.

We have a couple plans that we will execute to get our web server to fully work on Docker and Cloudlab. One of the issues we have been having was with our mySQL container interacting with our web server container on Docker. Docker has a built in command that shows the logs of a specified container. We're hoping if we execute this command for the mySQL container we will be able to find potential errors or anything that looks unusual. Another idea that we plan on trying is using the bash terminal in the mySQL container to try and ping the web server container. This will show us whether or not the two containers are connecting properly. If the pinging is successful, the issue could be a separate problem with the database. Our hope is that once we resolve these issues, everything will run smoothly both on Cloudlab and Docker by the third deliverable. If for some reason these minor issues remain unresolved by the third deliverable, we still feel confident that both our project and our understanding of the material is strong despite a few small technical errors.