CSC 2610: Cloud Fundamentals and Web Programming

Fall 2024 – Project #2 (30 course points)

In this project, your goal is to develop your own ExpressJS-based application that manages all LSU parking lot data in a MongoDB database. All the following criteria should be met in to receive full credit for the project. So, please read these below instructions carefully. At the same time, you are encouraged to be creative in terms of how you design this web application. Try to think about how you would like your visitors to retrieve the parking information before embarking on designing your web application. It would be interesting to see how different teams approach this same problem.

1. **Data**: You should retrieve all the data from LSU's parking page (https://www.lsu.edu/parking/availability.php). You can write your own simple parsing code to retrieve this data using a process like what you have used when working on Assignment-2. If you make your parsing code output a JSON object, it should then be very easy for you to import this data into a MongoDB database. Instead of a native JS approach, you can also use NodeJS libraries such as Cheerio for parsing data out from the parking site. See more here if you are interested: https://cheerio.js.org/docs/intro

2. **Bootstrap**: Your web-application should have a Bootstrap-based menu with support for at least two pages:
    a. An "Availability" page that allows visitors to ***view*** the availability of various lots on various days of the week.
    b. An "Admin" page that allows visitors to ***add/delete entries*** of to your parking database.

    The above is just a minimum limit for the number of pages your web-app should have. You are allowed to have more than these two pages if you would like. For example, your team can decide to have two pages just for viewing the available lots: (1) A "view" page that allows visitors to search lots by name (2) another "view" page that allows visitors to search lots by minimum availability percentage.

3. **Forms and REST APIs:** All pages should have form fields that take some sort of input from the user. They should then be making queries to backend REST APIs that you need

to build. You are free to design these APIs as you see fit. Please make use of the Postman tool to test these APIs as we will do in class.

4. **MVC Architecture**: Your code needs to make use of the EJS view engine. Make sure all content that is displayed to user is routed from this view engine.

5. **Responsive site**: It is important that your web app work properly on all devices. Your web application will be viewed on multiple devices to make sure it gets rendered properly. Make use of CSS directives provided by Bootstrap for this purpose.

6. **AWS deployment**: A very important requirement is for you to deploy your web application on an AWS EC2 instance. Make sure to create a free-tier account as we discussed in class. We discussed in class, the procedure for deploying your apps on AWS. Make sure to leave your AWS EC2 instance "on" until you receive grades for your project on Moodle.

7. **No jQuery/client-side JS requirements**: It is not explicitly required that you use any front-end JS code for this project as we covered this in prior assignments. However, using front-end JS code as well as JS libraries (such as jQuery) can potentially make your web app be more user-friendly. For example, they can allow your app to look for information in the backend (via JavaScript-driven asynchronous REST API requests) as the user types in a search bar (i.e. search suggestions). Feel free to implement such features in your web application as they are fun to learn and very useful for you in the future.

8. **Teams:** Teams can have a maximum of ***two students***. But this project is also within the scope of a single student.

<u>Submission Requirements:</u>

1. You need to submit all code that you wrote as part of this project on Moodle. Make sure to include the AWS URL where you deployed your web application and leave the instance turned on.

2. Do include the admin page credentials as well in your submission if you decided to attempt the bonus.

3. On the last week of classes (12/3 and 12/5), you will also need to make a 5-10 minute presentation demonstrating the flow of your code as well as any salient features of your web application. If there are two students in a team, then, both students are required to participate by dividing their presentation time.

4. Please include references to any useful tutorials/blogs, Stack Exchange Network posts or other online forum posts that you came across in the course of working on this project. **Note**: If you need more time beyond 12/5, there will be a flat 10% penalty until December 9th. You will need to record your video online and share a link to it in this case.

Good luck to all!