CSE 316 Spring 2024: Project Final Code

Due: June 6, 2024 at 11:59 PM

Read This Right Away

The final project code is due at **11:59 pm on June 6, 2024 KST**.

Directions

You will be submitting your final code for your project in this assignment area (See submission instructions below).

One member of the team should invite me as a collaborator on your private github repo.

Keep In Mind

The code should build and run without errors. I will not have the time to troubleshoot and fix issues. Nor will I have time to email you and ask help in fixing it as I have done on some assignments. Make sure any needed packages are listed in the dependencies of the package.json file. One idea, is to move your zip to a different machine if you have a spare or to at least a new folder, unzip it, do the npm install, build the code, and run it. Verify all of this works so you will be 90-95% sure I will not have a problem.

If you don't have a spare machine or environment, at least make sure you have not installed any products as global (for all users). This is usually the cause of problems where the code is missing packages after the npm install command is run. If there is even a small possibility that one of the packages was 'globally' installed and may not be in package.json, at least mention the needed install commands ('npm i <pkgname>') in the README file. That way, I have a chance at running it without major issues.

I am testing on a macbook using Chrome. You should at least test your code against Chrome (even if it is on another platform like Windows).

The code should implement all the requirements you wrote and should pass the test procedures should I decide to use a few of them.

Submission

When the code is working properly, you should do the following before submitting:

1. Clean up any dead code or debug code (remove it)
2. Write some meaningful comments to make the code easy to follow (mainly in your javascript/typescript but it doesn't hurt to make some notes in the HTML as well.)
3. If there is anything important to know about how to build and start the system, please write a detailed README file and locate it at the top of the project tree. This should include how to install packages, build the code and run the server and the front end. Also, if a database must be created and/or populated with data, provide a sql script file or (for mongodb), a way to easily create and initialize the database collections.
4. **Delete the node_modules folder!!!** I can reproduce this with npm install as long as your package.json file is in place.

   [One member of the team]
5. Move your development folder to a top level folder called CSE316_<Group#>_ProjectFinalCode where *<Group#>* is replaced by your group name. If your team is Group2, then your folder name for submission is CSE316_Group2_ProjectFinalCode.
6. Compress your top level folder containing entire development tree (producing a zip file). Upload the zip file to the Project assignment page in Brightspace.
7. Navigate to the course Brightspace site. Click **Assignments** in the top navbar menu. Look under the category 'Assignments'. Click **ProjectFinalCode**.
   a. Scroll down and under **Submit Assignment**, click the **Add a File** button**.**
   b. Click **My Computer** (first item in list).
   c. Find the zip file and drag it to the 'Upload' area of the presented dialog box.
   d. Click the **Add** button in the dialog.
   e. You may write comments in the comment box at the bottom.
   f. Click **Submit**. 🡐 Be sure to do this so I can retrieve the submission!

Grading [Final Code]

I will be testing your submissions on Google Chrome under Mac OSX.

Grading will be based on the styling and functionality of the submission. The points are assigned as follows:

- **Static design**: Style matches mockups. Content is correctly displayed: **15 points**
- **Interactivity**: Checkbox, mouse click and menu functionality. This means the code can pass all of the test procedures when run as written: **35 points**
- **Responsiveness**: Good style when resizing page. Components are organized well at any page size. Multi column lists reduce column count gradually on narrower displays. Menu compresses to a menu button (if specified in design). **15 points.**
- **Version Control**: Project shows good version control practices. Checkins for major or minor features (or a few small bug fixes) with descriptive (1 or 2 sentence) comments. For a project of this size, there should be a significant number of checkins progressively made across at least 2-3 weeks of the project duration. Both/All team members should have at least a few checkins. If most of the code is checked in over the last 2 or 3 days, then it shows there was little work done ahead of time and it is not clear what changes were for since each checking will probably have a large amount of differences. **15 points**.
- **Quality Points:** This concerns the overall quality of the code and professional look of the resulting website. **20 points**

Quality assessment

For this project, I want to see that you are able to write requirements and implement the requirements properly following good coding conventions (e.g. consistent capitalization, meaningful naming, good abstraction).

Your sites are expected to either prevent errors from occurring, or otherwise handle and display basic error messages to inform the user when there is an error. These errors will typically involve user input (e.g. incorrect password, etc.).

The frontend and backend should not crash in an irrecoverable way from any standard usage scenario, which can include user's typing in incorrect input data for different fields.

Additionally, keep following good version control practices. Your team should have multiple commits and make sure all the commits descriptively labeled.

For all pages in the site, this (Quality Assessment) part of your grade is in how well you are able to create and execute a design. For these points, the grade will be categorized as below:

Top – (18-20 points) Superior pieces of work with excellent effort demonstrating a mastery of software engineering skills and a thoughtful use of concepts discussed in class; work that shows clarity of presentation, effort, and attention to detail.

Solid – (15-17 points) Good work demonstrating quality software engineering skills, with adequate preparation and clear presentation.

Adequate – (8-14 points) Work that is adequate but that would benefit from increased effort or preparation.

Needs Work – (0-8 points) Work that needs more effort.

To qualify for a ‘Solid’ or ‘Top’ levels, it is expected that you go above and beyond the bare minimum to make a professional quality design and have a consistent visual look across the website.

Note: Whereas static design, interactivity, responsiveness, and source control have specific critiques and point deductions which may be discussed in a re-grade request, the ‘quality points’ is a general assessment of the overall quality of the work and will not likely be adjusted. It is a personal/professional assessment of the overall attention to ‘detail’ and ‘polish’ of the final product. ***This separates minimal or adequate quality development from exceptional or superior quality development.***