Project 3

Start Assignment

- Due Thursday by 11:59pm
- Points 100
- Submitting a text entry box or a website url

Projects have played a key role in your journey to becoming a software developer. As you apply for development jobs, your portfolio is absolutely vital to opening doors to opportunities. Your portfolio showcases high-quality deployed examples of your work, and you can use your finished projects for that very purpose.

Interactive MERN Stack Single-Page Application

This project is a fantastic opportunity to show employers your collaborative skills and coding abilities, especially in the context of a scalable, user-focused MERN app. Remember that employers want to know what you can do, but they also want to know how you work with other developers. The more examples of deployed collaborative work you have in your portfolio, the more likely you are to get an interview and a job.

Project Requirements

With your group, you'll again conceive and execute a design that solves a real-world problem. In creating your first collaborative MERN-stack single-page application, you'll combine a scalable MongoDB back end, a GraphQL API, and an Express.js and Node.js server with a React front end, implementing user authentication with JWT to build a user-focused platform. You'll continue to build on the agile development methodologies you've used throughout this course. These include storing your project code in GitHub, managing your work with a project management tool, and implementing feature and bug fixes using the Git branch workflow and pull requests.

For this project, you should start from scratch. Doing so will allow you to revisit your front-end abilities in the context of React and solidify your understanding of working with multiple servers in a MERN application. Your skills have continued to improve since the first two projects, so naturally your approach will be different considering the experience you've gained with each new application you've built.

Your group will use everything you've learned throughout this course to create a MERN-stack single-page application that works with real-world data to solve a real-world challenge, with a focus on data and user demand. This project will provide you with the best opportunity to demonstrate your problem-solving skills, which employers will want to observe. Once again, the user story and acceptance criteria will depend on the project that you create, but your project must fulfill the following requirements:

- Use React for the front end.
- Use GraphQL with a Node.js and Express.js server.

- Use MongoDB and the Mongoose ODM for the database.
- Use queries and mutations for retrieving, adding, updating, and deleting data.
- Be deployed using Render (with data).
- Use at least one GitHub Action in the project repository.
- Have a polished UI.
- Be responsive.
- Be interactive (i.e., accept and respond to user input).
- Include authentication (JWT).
- Protect sensitive API key information on the server.
- Have a clean repository that meets quality coding standards (file structure, naming conventions, best practices for class and id naming conventions, indentation, high-quality comments, etc.).
- Have a high-quality README (with unique name, description, technologies used, screenshot, and link to deployed application).

CSS Styling

Instead of using a CSS library like Bootstrap, consider one of the following suggestions:

- Explore the concept of CSS-in-JS, which abstracts CSS to the component level, using JavaScript to describe styles in a declarative and maintainable way. Some popular libraries include **styled-components** (https://styled-components.com/) and **Emotion** (https://emotion.sh/docs/introduction).
- Try using a component library, such as **Semantic UI** (https://semantic-ui.com/), **Chakra UI** (https://chakra-ui.com/), or **Ant Design** (https://ant.design/).
- Create all the CSS for your application just using CSS.

Ultimately, it doesn't matter which of these options you choose—it just needs to look professional and be mobile-friendly.

Alternative Project Using Python

You also have the option to replace use Python as your back-end language, which would replace the following requirement:

- Use GraphQL with a Node.js and Express.js server.

IMPORTANT

If you choose to go this route, please note that this will likely require additional work from your group, as the curriculum doesn't cover full-stack Python applications. Please speak to your instructional team prior to choosing this option.

Presentation Requirements

Use this **project presentation template (https://docs.google.com/presentation/d/10QaO9KH8HtUXj__81ve0SZcpO5DbMbqqQr4iPpbwKks/edit?usp=sharing)** to address the following:

- Elevator pitch: A one-minute description of your application.
- Concept: What is your user story? What was your motivation for development?
- Process: What were the technologies used? How were tasks and roles broken down and assigned? What challenges did you encounter? What were your successes?
- Demo: Show your stuff!
- Directions for future development.
- Links to the deployed application and the GitHub repository. Use the **Deploy MERN Stack with Render and MongoDB Atlas (https://coding-boot-camp.github.io/full-stack/render/deploy-mern-stack-with-render-guide)** if you need a reminder on how to deploy to Render.

Grading Requirements

This project is graded based on the following criteria:

NOTE

If a project submission is marked as “0”, it is considered incomplete and will not count towards your graduation requirements. Examples of incomplete submissions include the following:

- A repository that has no code
- A repository that includes a unique name but nothing else
- A repository that includes only a README file but nothing else
- A repository that only includes starter code

Technical Acceptance Criteria: 30%

- Satisfies the following code requirements:
  - Application uses React for the front end.
  - Application has a GraphQL API with a Node.js and Express.js server, and uses queries and mutations for retrieving, adding, updating, and deleting data.
    - Or uses Python instead Node.js and Express.js.

- Application uses MongoDB and the Mongoose ODM for the database and protects sensitive API key information on the server.
- Application uses at least one GitHub Action in the project repository.
- Application includes user authentication using JWT.

Concept: 10%

- Application should be a unique and novel idea.
- Your group should clearly and concisely articulate your project idea.

Deployment: 15%

- Application deployed at live URL on Render and loads with no errors.
- Application GitHub URL submitted.

Repository Quality: 10%

- Repository has a unique name.
- Repository follows best practices for file structure and naming conventions.
- Repository follows best practices for class and id naming conventions, indentation, quality comments, etc.
- Repository contains multiple descriptive commit messages.
- Repository contains a high-quality README file with description, screenshot, and link to deployed application.

Application Quality: 15%

- Application user experience is intuitive and easy to navigate.
- Application user interface style is clean and polished.
- Application is responsive.

Presentation: 10%

- Your group should present using Google Slides, Powerpoint, or a similar presentation software.
- Every group member should speak during the presentation.
- Your presentation should follow the **Project Presentation Template (https://docs.google.com/presentation/d/10QaO9KH8HtUXj__81ve0SZcpO5DbMbqqQr4iPpbwKks/edit?usp=sharing)** .

Collaboration: 10%

- There are no major disparities in the number of GitHub contributions between group members.

How to Submit Your MERN-Stack Single-Page Application

Each member of your group is required to submit the following for review:

- The URL of the deployed application.
- The URL of the GitHub repository, with a unique name and a README describing the project.

NOTE

You are NOT allowed to skip this assignment. Project assignments are required.