Date Or Reflection

The date you're submitting this.

Oct/18/2021

Location of deployed application (not necessary for Junior Engineers)

Please provide the url where we can find and interact with your running application.

N/a

Instructions to run assignment locally

Please provide us with the necessary instructions to run your solution if it is implemented with technologies different from the starting repo.

In a new work space or make a new folder via any code editor. Make two folders “Client” and “Server”.

Install Node.js globally

Install the following dependencies in the “Client” folder:

npm i create-react-app -g

create-react-app intro-to-react-lab

npm install axios

npm i install node modules for react app

npm install --save react-router-dom

npm install enzyme enzyme-to-json enzyme-adapter-react-16

Enter “npm run start” in the terminal to intialize the app

Install the following dependencies in the “Server”

npm init-y creates new Json file

npm i express install express

npm install uuid

npm install cors

npm install --save-dev nodemon -g

npm i nodemon install nodemon

Enter “nodemon app.js” in the terminal to initialise the server

Time spent

How much time did you spend on the assignment? Normally, this is expressed in hours.

5 hours(aprx.)

Assumptions made

Use this section to tell us about any assumptions that you made when creating your solution.

1. Security :

File/Image uploads pose no threat therefore file type validation is not needed.

Payload: I am expecting a low amount of users to try the app; end users will not be using excessive payloads.

Server space is enough to store all the data from the users therefore file size validation is not needed.

2. UI: For the purpose of this application, the assumption is that the end-users would prefer to open this app in devices with relatively bigger screens such as tablets, laptops etc. Therefore it is responsive only for screens larger than 950px.

3. The end users find the blog site cathartic as they can be anonymous therefore no user authentication is required at the moment

4. Technology: It is React based app with a backend coded with Express.js. Data is currently stored on the hard disk with no connection to a database.

5. Geographical constraints: Currently the application is offer in Canada only with an assumption the entries will use english characters

Shortcuts/Compromises made

If applicable. Did you do something that you feel could have been done better in a real-world application? Please let us know.

1. Styling and Design: The color scheme is lacking. There are blank spaces on the pages that need to be covered up creatively using illustrations, color gradients, animation etc.

2. Responsiveness could use a bit of work.

3. Have a server side code to store images

Assume your application will go into production...

1) What would be your approach to ensuring the application is ready for production (testing)?

I would use the Exploratory Testing approach. I chose this method because the primary functionality of the app is to allow users upload data via form inputs; it is vulnerable to invalid data, using excessive payloads, etc. Following the Exploratory approach we will take the following steps:

a. Overwhelm the software with invalid data: Try feeding the system the input it doesn't expect and see how it handles them. We will start with trying with special characters like !@#$%^&*()_+={}[]<>,.?/~
b. Overwhelm the software by pushing beyond boundaries: We will send in data that will be correct but bigger than what we are expecting. We can then look for errors two ways: Processing errors, where the program itself chokes on the input, or rendering errors, where the output is re-displayed, but too large for the space available
c. Perform server-process interrupts: To check if the system gives proper indication that it is processing information and uploading on the servers, therefore avoiding uploading of multiple entries of the same kind from users.
d. Try the application from outside the firewall - from a public terminal, in secure mode
e. Try different browsers - Safari, Opera, FireFox, Edge

2) How would you ensure a smooth user experience as 1000's of users start using your app simultaneously?

1. Optimize page load speed:
    a. Start compressing images before using it on the website.
    b. Be careful of the images being used. Refrain from using Jpeg photos and prioritize the use of SVG formats for logos and illustrations.
    c. Get rid of unused JavaScript from the code
    d. Use CSS preprocessors like Sass or Less
2. Use Attractive Calls to Action(CTA):
    a. Use clearly marked visual cues to determine which content is important.
    b. Being mindful of the psychology of colors when choosing colors of (CTA) cues.
3. Use hyperlink differentiation.
4. Have a proper hierarchical structure for information relayed on the page. Structuring the content accurately will ensure that the users will consume the content and stay engaged.
5. Accessibility: Making sure the website complies with the current accessibility standards.
6. Make sure the app is responsive and is mobile-friendly.

3) What key steps would you take to ensure application security?

1. Get an application security audit carried out by people who have professional application security experience, who know what to look for.
2. Have proper logging implemented.
3. Use real-time security monitoring.
4. Use encryption to protect the application
5. Revaluate and harden everything. I will ask myself the following questions:
    a. Is my webserver using modules or extensions that the application doesn't need?
    b. Are my servers using security extensions?
    c. Where is the information being stored?
    d. What users are allowed to access the server and how is that access managed?
6. Keep my software updated.
7. Keep abreast of new vulnerabilities and never stop learning.

What did you not include in your solution that you want us to know about? Were you short on time and not able to include something that you want us to know about? Please list it here so that we know that you considered it.

1. The images are uploaded but cannot be saved in the backend. I know I had to send FormData as an object using axios and write server side code to receive it.
2. I did not include Edit, Delete, Share(via Email) functionality for journal entries.
3. Upload the app on a cloud such as Heroku
4. I could have added a comments section for each journal entry.
5. Validate file path, size, and type. Set min and max payload limits.
6. Styling: Could have added a header, footer, menu bar. I considered filling up blank spaces, and make the app responsive for screens as small as 200px. Adding logos,illustrations,images, and/or FontAwesome icons would have further added to the aesthetics of the application.

Other information about your submission that you feel it's important that we know if applicable.