Course Project

The project for FS1010 will build off of the contact us form JSON API from FS1020 and involve integrating in the existing HTML / CSS from FS1000.

Phase One - HTML/CSS Fundamentals

1. Modify your website's markup to make it more semantically correct. The expectation isn't to be perfectly semantically correct and iterated over throughout the length of the project. Ensure your setup is sound.
2. Make any improvments to your website's existing HTML and CSS to make it more maintainable.

Phase Two - Basic React

Using your semantic markup from Phase One, convert your website into React components, and using React Router, ensure your website works as a multi-page React application. For your contact form component, it does not need to be connected to state in this phase.

Phase Three - React with Backend

Note: All endpoints listed are ones developed in your FS1020 course. You will need your Express app running in another terminal session in order to interact with your React application.

1. Use the `useState` React hook in order to store your contact form's state. Feel free to use libraries like formik if you so choose to.
2. Make a successful AJAX request (`POST /contact_form/entries` - developed in the FS1020 Course Project) from the Contact Us React component that displays a success and error notification upon form submission.
3. a. Add a new page for your React application, which upon a successful submission to your login route (`POST /auth`), the token received will be used to gain access to protected pages in React. Error messages should be displayed appropriately on screen.

   b. Add a new page, which can only be accessed by a valid JWT, which leverages the `GET /contact_form/entries` endpoint to display all submissions. This page should be stylized appropriately. This page can be accessible through the login page. If a valid JWT is not available, or no token is set, then user should be redirected to the `Login` page from step 2.

Bonus:

4. In your protected area, add a new page which has a single form that allows to make a request to `POST /users` to create new users.

Rubric

Assessment Criteria	Not Good Enough *(0% score <60%)*	Good *(60% score <90%)*	Very Good *(%90 score %100)*	Marks
HTML Semtantics	Document is invalid HTML5 and mostly semantically incorrect.	Document is at least mostly valid HTML5, and fairly semantically correct.	Document is a valid HTML5 document and largely semantically correct.	**10**
Maintainability	Code and modules are not reasonably organized. The code is also mostly difficult to follow and reason about.	Most relevant files and code is easy to read and understand. All JavaScript is built with Webpack (`create-react-app`).	All code is well organized and easy to follow and understand. Code is easy to read and understand with reasonable formatting.	**10**
Contact Us Form Functionality	HTML and JavaScript for form is partially correct and a network request is attempted to be made. React is used for the form and renders something.	An HTTP request is sent within a React component and shows a success message upon receiving a successful response.	React shows an error upon an unsuccessful submission and JavaScript code is well organized.	**20**
Protected Area Functionality	HTML and JavaScript for form is partially correct and a network request is attempted to be made. React is used for the form and renders something.	Any HTTP request sent within a React component shows a success message upon receiving a successful response, and subsequently handles errors appropriately.	Protected routes are not accessible without valid JWT token. Care is taken to enhance experience by providing additional functionality such as logout	**30**