Final Project

The final project requires you to to write an application based on all the materials that you have learned during the course. The outcome will be a responsive public-facing web application deployed in the web.

Teamwork
You will be split into groups of 5-8 persons.

Timeline
The standard project duration is 2 month.

Setup
- One of team members creates a new public GitHub repository for this final project; name it as you see fit for your project.
- Add your team members and all mentors with developer access to the repository
- Forbid commits directly to master, it should be read only for team members
- Integrate all CI tools in GitHub repository settings (See **Work process details** section)

Process
- All work related processes will be managed using Trello. You will be provided with a board for your group
- Teamwork should be organized using Agile/Scrum techniques. One of your group members will be your Scrum Master
- Duration of 1 sprint is 1 week
- You will have regular stand-up meetings, grooming, sprint planning. Specific timing and intervals between the meetings to be decided by the team.

Technologies
- Backend part of the application should be launched using Node.JS and Express
- You may use MongoDB as a database. Database should be hosted in cloud
- Front-end part of the application should be written in React
- You shall use React Router v4 for managing front-end routes (choice of BrowserRouter / HashRouter to be done by the team)
- You shall use Redux for a front-end data storage
- Your web-application should be responsive and work well on all devices, including mobile phones
- You shall use SASS (SCSS syntax) for writing CSS markup for your application
- Using BEM for naming classes is not required, but strongly recommended
- You are allowed to use Bootstrap / Material UI or any other CSS framework

Work process details
Each team member will work on a specific feature in his own branch. Commits directly to master are prohibited. Steps to be followed when you are going to implement a feature:
1. Go to master branch and pull all latest changes
2. Create new branch based on master (i.e. feature branch)
3. Write some code, commit and push
4. Cover your code with tests
5. Create pull request on GitHub
6. Repeat steps 3 and 4 until your task is ready

7. Send pull request for a review to your colleagues
8. Two team members should QA, review pull request, and provide comments if your code should be improved
9. You fix issues reflected in pull request comments and repeat step 6 until ready
10. Two team members approve your pull request
11. Pull request is ready and can be merged to master (assuming all automated checks have passed, see below)

Before merging to master, each pull request should also pass an automated CI procedure with certain mandatory checks. This should be done using following technologies / tools:
1. Enzyme, **Jest** – unit tests for your components
2. ESLint - checking validity of Javascript code, compliance with Javascript code conventions
3. Stylelint - checking validity of SCSS code and code style, order of SCSS properties
4. Travis CI – Continuous integration platform that allows each pull request to pass the required CI flow. Can trigger react-scripts build (that would include running all unit tests) together with ESLint and Stylelint checks.

Application contents

Detailed description of application contents will be provided in a separate file.

Good luck, and may the Force be with you! :) It's up to you to make it real!