10/20/2018

Gavin Hanna - 17164974

Higher diploma in science in web technologies

Interim Report

Recipe Book

Table of Contents

[1. Project Overview 2](#_Toc527887561)

[2. Project Expected Deliverables 2](#_Toc527887562)

[2.1. Course Deliverables 2](#_Toc527887563)

[2.2. Self-Imposed Deliverables 2](#_Toc527887564)

[3. Project Actual Targets Met 3](#_Toc527887565)

[4. Deliverables Yet To Be Achieved 4](#_Toc527887566)

[5. Main Problems Encountered 5](#_Toc527887567)

[6. Expected Stumbling Blocks 5](#_Toc527887568)

[7. Focus on Rubric 5](#_Toc527887569)

[8. Summary of Project Progression 6](#_Toc527887570)

# 1. Project Overview

Recipe book is a social network web application focused on creating and sharing recipes, following other users and curating a personal recipe book of favourite recipes.

The application is designed with full-stack architecture in mind. Having a client-side Single Page App created using React.js, with state managed implemented in Redux.

The server will be built using express.js with several npm packages used to handle authorisation and object relational mapping among other features. This client-side application will communicate with the server-side application using authenticated requests using JSON Web Tokens.

The application will employ a manifest.json in such a way as to be capable of being used as a Progressive Web App, in other words it will be able to mimic the look and feel of a native mobile application (and desktop application on Windows 10 in Chromes latest versions).

# 2. Project Expected Deliverables

## 2.1. Course Deliverables

* Project Proposal
* Requirements Specification
* Project Analysis & Design Documentation
* Interim Report
* Project Presentation
* Project Final Report
* Project Code

## 2.2. Self-Imposed Deliverables

* Create basic Bootstrap mock-up of the website
* Plan the server API endpoints (e.g. /recipes/new)
* Finalise structure of database.
* Complete the server-side application
* Test server (continuously) using Postman testing utility
* Plan the (component) layout of the client-side React application
* Create client-side React application using the Bootstrap mock-up HTML
* Test client-server interaction using axios npm package to request data and test user authentication from the client side.
* Implement Redux state management in the React app
* Test Redux using the Redux dev tools chrome extension
* Create manifest.json file so the application can be used as a PWA (Progressive Web App, i.e. saved to, and launched from, the home screen of a device)
* Create logo and generate all necessary icons for the app to qualify for PWA status
* Create a service worker to handle any offline capability the app will have. Most likely by simply serving a “No Connection Detected” page, offering the user the option to reload.
* Clean up/refactor code

# 3. Project Actual Targets Met

To date, the following project deliverables have been met:

* Project Proposal
* Requirements Specification
* Interim Report (Current Document)
* Create basic Bootstrap mock-up of website
  + The main pages of the application have been created in HTML with Bootstrap 4. These pages are subject to change and may be added to or removed as the project progresses
* Plan the server API endpoints
  + A current plan for each endpoint, its access level (public or private) and what action will be taken upon hitting that endpoint has been planned. This is subject to change at early stages of development.
* Finalise the structure of the database.
  + A MongoDB instance was chosen as the database. The table structure has been decided, the values for the tables have been decided.
* Complete the server-side application
  + Partially completed. To date the User, Recipe and RecipeBook endpoints have been completed. User can register and login, with authentication being handles by Passport use a JSON Web Token strategy. Users can create, update and delete recipes. Users can follow and unfollow other users. Users can like and unlike recipes. Users can add recipes to their personal “Recipe Book”.
* Test Server
  + Partially completed. The above completed endpoints have been tested with the Postman API testing software.

# 4. Deliverables Yet To Be Achieved

The following deliverables are yet to be achieved:

* Complete the server-side application
  + Endpoints related to the Comments database table, which will implement CRUD functionality, have yet to be completed.
* Test server using Postman
  + This will continue until, and potentially even after, the server and it’s endpoints have been completed.
* Plan the (component) layout of the client-side React application
  + Create a diagram of the component based layout of the React application in order to help with the actual implementation, based on the mock-up.
* Create client-side React application using the Bootstrap mock-up HTML
  + Once the server has been completed, work will start on creating the client-side application in React
* Test client-server interaction
  + Once the basic structure of the React component has been put in place, data will be fetched from the server in order to test the interaction between the front and back ends of the application. Testing will include register users, logging in and out and retrieving and displaying data.
* Create manifest.json file so the application can be used as a PWA
  + A manifest.json file and a service worker can be used to implement a Progressive Web App. This allows a website to essentially mimic the functionality of a native mobile (and desktop, as in the case of Windows 10) application. A set of icons is also required. The end is result is that when a user navigates to the website in Chrome, they will be prompted to save the app to their home screen. If they choose yes, the application will be downloaded and saved to their home screen like any other app. It will then be launched in it’s very own full screen web view without any search bar etc.
* Create logo and generate all necessary icons for the app to qualify for PWA status
  + This can be done earlier or later in development, however to qualify for PWA status a certain minimum amount of icons need to be present in the application, and referred to in the manifest.json. A simple icon will be created in Inkscape, and an online icon generator will be used to generate all other required icons/sizes.
* Create a service worker to handle any offline capability the app will have
  + A simple service worker will be implemented. A service worker essentially uses JavaScript to hijack http requests send from the application, they can then be edited in any way the developer chooses. In the case of this application, in the event of no internet connection available an offline page will be shown, alerting the user to the lack of connection and offering a button to attempt to refresh the page and regain a connection.
* Clean up/refactor code
  + Once the app has reached a satisfactory level of efficacy, and if time allows, code will be cleaned up and refactored in order to attempt to increase both speed of the app and possibly improving readability of the code.

# 5. Main Problems Encountered

So far in development no major problems have been encountered. Some minor issues have come up. The design of the database was refactored several times as the structure of the server-side application was being designed and began to be developed.

The server’s endpoints were refactored several times in order to make better use of the Mongoose ORM (Object Relational Mapper). For example, originally there was a recipe/like route and a recipe/unlike route to be implemented. However after learning that Mongoose had the functionality to simply push and “pull” (removed an item from an array in Mongoose) an item from an array in the database, it was decided to amalgamate the two routes into a single route. This was achieved by first checking if the user’s id was already in the array of users who had liked the recipe in question. If it was, it was removed from the array using Array.pull(), if not, it was added using Array.push().

A similar technique was used during the implementation of the user/follow endpoint. A users list of followers is saved in an array in the User model schema in Mongoose. Therefore a simple check to see if the user is already in the array would dictate whether it was to be added or removed.

# 6. Expected Stumbling Blocks

At no point in the development is a major stumbling block imagined, however it’s possible a few issues could delay the completion of some aspects. It is entirely possible that once the server-side and client-side applications are being connected and tested, some of the API endpoints may have to be refactored or changed. It should not be a major issue as there is a lot of very helpful documentation associated with the Mongoose ORM, the express framework and the React library.

There is a possibility of unforeseen stumbling blocks as there are quite a few “moving parts” to this application, however at it’s current state of development, the developer believes the application will be completed by the end of term.

# 7. Focus on Rubric

In order to evaluate the adherence to rubric, each point in the rubric will be assessed separately:

|  |  |
| --- | --- |
| Communication | Refers to the final presentation and as yet cannot be assessed. |
| Writing & Presentation of Results | Difficult to evaluate as the developer has little experience with formal academic writing style, nor with technical writing. However the developer feels as if the content should be readily understood. Also confident that the ability to produce documentation will improve to a suitable point towards the end of the project semester. |
| Complexity/Coding Skills | Using modern full-stack web technologies and implementing a Progressive Web Application in a well-structured, modularised and clean fashion should demonstrate sufficient coding skills and management of complexity. |
| Innovation | While the user experience of the application may not be particularly novel, the implementation of the application with the range of modern technologies used and the marrying of client and server-side applications, coupled with offline and app-like features, should result in a technologically innovative application |
| Technology | As above, the developer believes that the application uses cutting edge modern web application technologies, interconnected in a suitable and scalable way. |
| Completeness | Difficult to assess during early stages of development, however the developer is working under the assumption that the application will meet minimum viable functionality for a commercial application upon completion of the semester. |
| Testing/Evaluation | The project will make use of several extremely helpful testing resources and applications. For example, the server will be fully tested using the Postman API testing software, which can send and receive any HTTP requests to the server and can therefore fully test all endpoints. On the client side, the React dev-tools can be used to fully debug React Components. This coupled with the Redux dev-tools provides a full-suite of efficient testing tools. |
| Project Management | Maintaining detailed notes and a clear and concise writing style should provide an easily understandable and complete project plan which will detail the project at each stage of development. |

# 8. Summary of Project Progression

As the project stands at the time of writing, the project is meeting deadlines and progressing as planned. The site “template” or mock-up, has been completed and is ready to be developed into the React application. The RESTful API server is almost complete, has gone through continuous testing and will reach it’s version 1.0 in the coming week. The database has been designed and implemented bar one remaining table (relating to “Comments”), which itself should be completed in the coming days. The server can currently register and log in users, users can then create recipes, follow and unfollow other users, like and unlike created recipes and add and remove recipes to/from their “Recipe Book”.

Documentation is being written and updated along with the development of the application itself.

The minor issues that arise are being swiftly dealt with using the excellent documentation associated with the chosen technologies. Small adjustments and improvements are also being made along the way as the developer explores the intricacies of the chosen technology stack.

Several pages of notes have been written and are being used to aid the writing of documentation as the project progresses.

Signature of author: