|  |  |
| --- | --- |
|  |  |
| CANTEEN APP |  |
|  |  |
|  | **PATRICK DUFFY  OISIN HEARNE MARY MCDONNELL** |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | TABLE OF CONTENTS  [CANTEEN APP 1](#_Toc163580886)  [PROJECT DETAILS 2](#_Toc163580887)  [INTRODUCTION 3](#_Toc163580888)  [REQUIREMENTS 3](#_Toc163580889)  [TECHNOLOGIES 3](#_Toc163580890)  [DESIGN METHODOLOGIES 3](#_Toc163580891)  [ARCHITECTURE 3](#_Toc163580892)  [PROJECT MANAGEMENT 3](#_Toc163580893)  [LIMITATIONS 3](#_Toc163580894)  [TESTING 4](#_Toc163580895)  [FUTURE WORK 4](#_Toc163580896)  [CONCLUSION 4](#_Toc163580897)   PROJECT DETAILS   **Project Title** – Canteen App  **Group Members** –   * Patrick Duffy – G00408635 * Oisin Hearne – G00405249 * Mary McDonnell – G00401878   **Supervisor** – Damien Costello  **Github** – https://github.com/CanteenProjectATU  **Screencast** – <link here> |  | |
|  |  |  |  |
|  | INTRODUCTION For our PPIT Project, we’ve been tasked with the creation of an app for the canteen to allow students to view nutritional information, canteen menus and more.  We created a client-side application for students to view all this information, as well as an admin application for canteen staff and the nutritional department to edit and upload information and a server-side application for managing the transfer of data between the client and the database. Project Components  |  |  | | --- | --- | | **Component** | **Contributor** | | Server-Side Application (server.js) | *Patrick Duffy* | | Client-Side Application (react) | *Mary McDonnell* | | Admin Application (react) | *Oisin Hearne* | | Front-End | *Mary McDonnell* | | Back-End | *Patrick Duffy* | | Database (MongoDB) | *Oisin Hearne* |  REQUIREMENTS  |  |  |  |  | | --- | --- | --- | --- | | Title | Priority | User Story | Acceptance Criteria | | See Menu | 10 | As a user, I want to be able to see the current offerings in the Canteen in detail. | When clicking the Menu option in the home page, a list of menu items for the canteen should appear. | | See Recipes | 10 | As a user, I want to be able to see and download nutritious/healthy recipes. | When clicking the Recipes option in the home page, a list of recipe PDFs should appear with a download option when clicked | | Upload Menu Items | 10 | As a canteen worker I want to upload menu items with names, prices, allergen information and nutritional information. | The Admin Panel should have an Upload Menu Item button that allows the to define and upload the details of a menu item. | | Menu Items Persistence | 7 | As an Admin I want menu items I’ve uploaded previously to be available in the future. | Menu items should be persistent so that they can be selected for future weekly menus with ease. | | Delete Items | 7 | As an Admin I want to delete menu items that will no longer be on offer. | Menu Items should have a delete button that removes them from the database. | | Upload Recipes | 10 | As an Admin I want to be able to upload recipes (PDF) so that users can view them. | The Admin Panel should have an Upload Recipe button that allows admins to define a title, and upload a recipe. | | Set Weekly Menu | 10 | As an Admin, I want to be able to select several menu items for each day of the week and set that as this week’s menu. | The Admin Panel should have a page that lists menu items previously uploaded that allows admins to select different ones for each day of the week that they’ll be available. | | Pantry | 8 | As a user, I want to be able to see the details of the food pantry. | There should be a Pantry page available to the user that showcases where the food pantry is, when it’s open and what it contains. |  TECHNOLOGIES This project uses the MERN stack, meaning MongoDB is used for storage, Express is used server-side and React is used client-side, with NodeJS used as the runtime environment.  We used this technology stack as the three of us each have some experience with it from previous projects, and we found it suitable for this particular use due to its ease of use for the following reasons:  **MongoDB** – The data being stored on the server is simple and lightweight and does not require much security due to there being no login system on the application. Because of this, we used MongoDB as it more easily translates to an Object-Oriented language like JavaScript and it is unstructured, so we did not need to create any schema.  **Express** – Express is something the three of us are all familiar with due to using it in multiple modules last year. It is flexible and allowed us to easily build the API for the application.  **React** – React allowed us to easily develop the front-end for both the client-side app and the admin-side app. It’s versatility also allowed us to develop the application for browsers, and then wrap it using Cordova into an application that can be used on Android devices. DESIGN METHODOLOGIES We used the agile methodology scrum to develop this app. We decided on this because we were working with changeable requirements, timelines, and we needed to get opinions on our development decisions from the customers. This allowed us to make a prototype of our app which we called the static version. ARCHITECTURE   Our architecture for this project features a client-server pattern, with the server written in express.js. This server communicates with the database (MongoDB, hosted on the same machine), and the client-side applications (The Admin & User versions, both of which are built in React).  Users request information from the server, which fetches it from the database. Admins push information to the server, which then passes it along to the database. From the perspective of these clients, they must only navigate to the correct page or input some data and it is added to the database without fuss. PROJECT MANAGEMENT We divided the workload into 3 sprints, The first sprint was for the static version of the app so try and get feedback. The second sprint was for database and server set up and using them together. The third sprint was getting more complex operations like recipes, menus for specific days and login functionality. We had daily 10-minute stand ups in person before our first class, 4 days a week. We did pair programming when there were issues with the code. LIMITATIONS  * While we originally intended for there to be only one app with a login system for admins, we instead decided to separate it into two due to the complication and security requirements that a login system would bring. * We were originally asked to implement a Loyalty System, where users could get stamps to get free coffees or other items. Due to unclear requirements as well as the complexity the system would require, we weren’t able to implement it in time.  TESTING We created a list of test cases and used those as a basis for testing the User & Admin apps. The excel spreadsheet containing these test cases can be found in the Documentation Repository on our Github. FUTURE WORK In the future, we may implement the aforementioned loyalty program as well as clean up the User Experience for both users and admins.  Implementing a loyalty program would also require implementing a user log-in system, so we could simultaneously merge the Admin App back into the User App to make it easier to maintain going forward. CONCLUSION |  | |