Project Description

Purpose

The purpose of the Community Cookbook app is to provide users with an efficient way to record and access recipes that they use often, as well as keep track of changes/edits they make on their recipes. This application takes away the excess time wasted on reading through recipe blogs so that the user can directly access the recipe without the "fluff". For beginner cooks, this application serves as a way for them to experiment with different recipes and track their progress.

Project Overview

The Community Cookbook is a localized web application (see Figure 1) that users can download on their devices. A simple, no-frills cookbook, where users can add, markup, modify, and share their recipes. Specifically, users can take notes, track historical changes and compare versions of previous recipes. They can divide recipes into various meal categories as well as create custom categories to place recipes in.

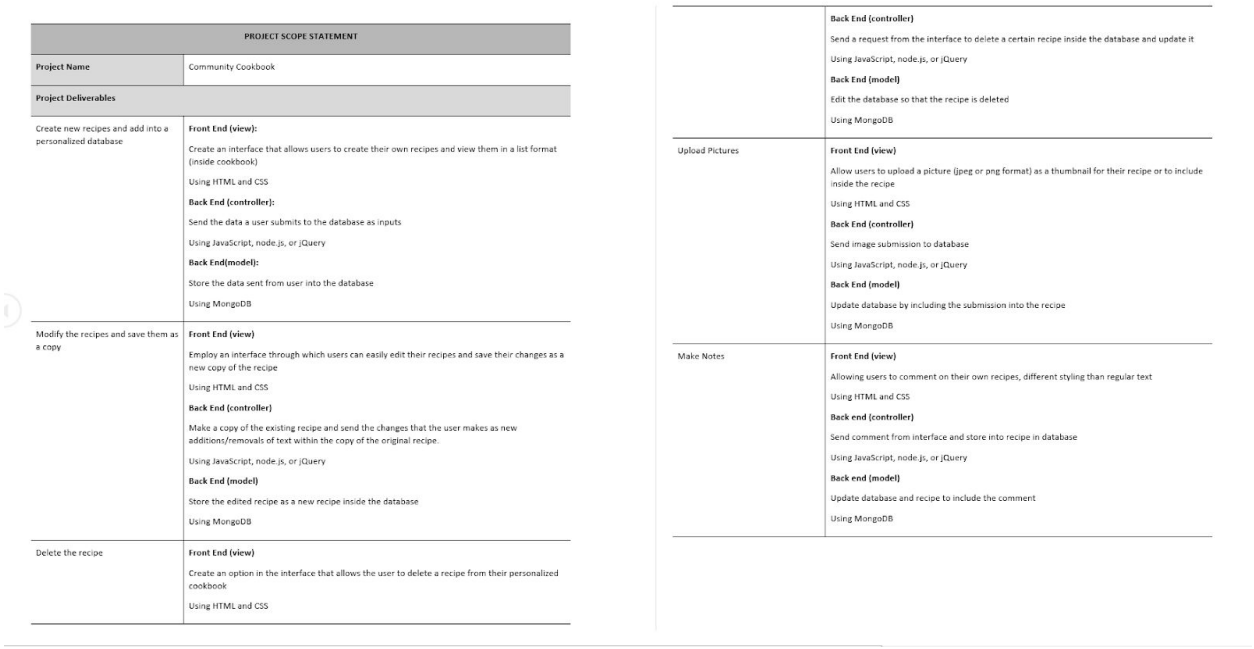
Recommendation
Localized Web App – This method will allow us to broadly distribute our application while minimizing our implementation requirements. We can focus on the core functionality without needing resources for web hosting and security. With more resources for core functionality we can produce a more feature rich product.

Figure 1: Benefits of Implementing the Project as a Localized Web App

Goals of the Project

Initially, the goals of the project included, but were not limited to, functionality to allow users to create recipes and add them to different cookbooks. Our envisioned web application would let the users create multiple custom cookbooks and add recipes inside the cookbooks. They could edit the recipes as well as make comments on the recipes to keep track of changes. Users could also highlight recipes by marking them as a "favourite", which they could access on the homepage easily. Otherwise, users would access saved recipes through the cookbook in which

they created the recipe.



PROJECT SCOPE STATEMENT	
Project Name	Community Cookbook
Project Deliverables	
Create new recipes and add into a personalized database	**Front End (view):** Create an interface that allows users to create their own recipes and view them in a list format (inside cookbook) Using HTML and CSS **Back End (controller):** Send the data a user submits to the database as inputs Using JavaScript, node.js, or jQuery **Back End(model):** Store the data sent from user into the database Using MongoDB
Modify the recipes and save them as a copy	**Front End (view)** Employ an interface through which users can easily edit their recipes and save their changes as a new copy of the recipe Using HTML and CSS **Back End (controller)** Make a copy of the existing recipe and send the changes that the user makes as new additions/removals of text within the copy of the original recipe. Using JavaScript, node.js, or jQuery **Back End (model)** Store the edited recipe as a new recipe inside the database Using MongoDB
Delete the recipe	**Front End (view)** Create an option in the interface that allows the user to delete a recipe from their personalized cookbook Using HTML and CSS **Back End (controller)** Send a request from the interface to delete a certain recipe inside the database and update it Using JavaScript, node.js, or jQuery **Back End (model)** Edit the database so that the recipe is deleted Using MongoDB
Upload Pictures	**Front End (view)** Allow users to upload a picture (jpeg or png format) as a thumbnail for their recipe or to include inside the recipe Using HTML and CSS **Back End (controller)** Send image submission to database Using JavaScript, node.js, or jQuery **Back End (model)** Update database by including the submission into the recipe Using MongoDB
Make Notes	**Front End (view)** Allowing users to comment on their own recipes, different styling than regular text Using HTML and CSS **Back end (controller)** Send comment from interface and store into recipe in database Using JavaScript, node.js, or jQuery **Back end (model)** Update database and recipe to include the comment Using MongoDB

Figure 2: List of Project Deliverables (Initial Goals of the Project)

Goals Achieved

While developing the project, our team came across various challenges that forced us to reevaluate our initial goals. For instance, we realized that it was too cumbersome for both the developers and users to have multiple cookbooks to place recipes in. We would have to consider the possibilities of users wanting to change which cookbook a recipe belongs in, or users wanting to place a recipe into multiple cookbooks, or users not wanting a recipe inside any cookbook. Faced with this complication, we decided to discard the idea of having multiple cookbooks, and instead implement the entirety of our application as a single cookbook. Users would create recipes and divide them into categories instead, and they could assign multiple categories to a single recipe.

For our project prototype, we created a minimum viable product. This includes the basic features of our application such as creating recipes, adding them to various categories, creating categories and making comments on recipes to track progress. As a prototype, this will allow users to experience the efficiency and tracking functionality, which are the main purposes of the application.

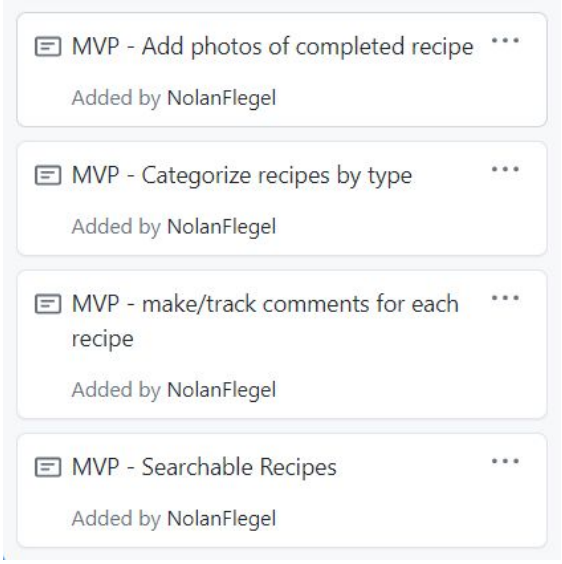


Figure 3: Current MVP Deliverables

In the future, we plan on improving the software to include a favourites section, allowing users to edit recipes and save them as a copy, as well as adding a feature for creating multiple cookbooks.

Figure 4: Future MVP Deliverables

Initial Design Process

Once we clarified the project idea and flushed it out, we used a Project Charter to create specific dates that we wanted to have parts of the project completed by.

Overall Project Milestones	Dates
Project Idea selection & introduction vlog preliminary.	October 9, 2020
Project Prerequisites/Planning and vlog presentation Preliminary (documentationing and requirement setup).	October 23, 2020
Project Design Architecture and presentation preliminary. - Development environment setup - Wireframe of our app UI - Prototypes of database structure - Implement Recipe Adding and Deleting	November 6, 2020
Project Observe, Analysis, and feedback preliminary. - Detailed style elements/UI - Navigation bar/menu/table of contents - database categories and comments prototyped - Search bar implemented	November 16, 2020
Project Storytelling and Live Zoom Demo/Presentation Preliminary. - testing app functionality - Comments and recipe version history added - bug fixing - Ability to add pictures	December 7, 2020

Figure 5: Project Charter

With the project charter in place, we had due dates to follow so that we would stay on track and be able to complete the project in a timely manner. To further organize our tasks, we created a roles and responsibilities chart. Using it, alongside our RACI chart, we were able to divide up our tasks evenly and know which tasks we were going to be held accountable for.

PROJECT ROLES AND RESPONSIBILITIES		
Project Name	Community Cookbook	
Name	**Role**	**Responsibilities**
Nolan Flegel	Developer	Backend framework Deployment (docker) Database and data structure design Coordinate MVC integration
Philip Anyuon	Developer	Front end implementation Build user interface using Bootstrap and Jquery Develop tools to insert and retrieve data from database
Aina Ulain	Developer	Direct the design of the user interface Provide the style elements for the interface Test the user experience Organizing our documentation
Timothy Maciag	Scrum Master	Manage team deliverables Facilitate communication Provide resources for development team

Figure 6: Roles and Responsibilities Chart

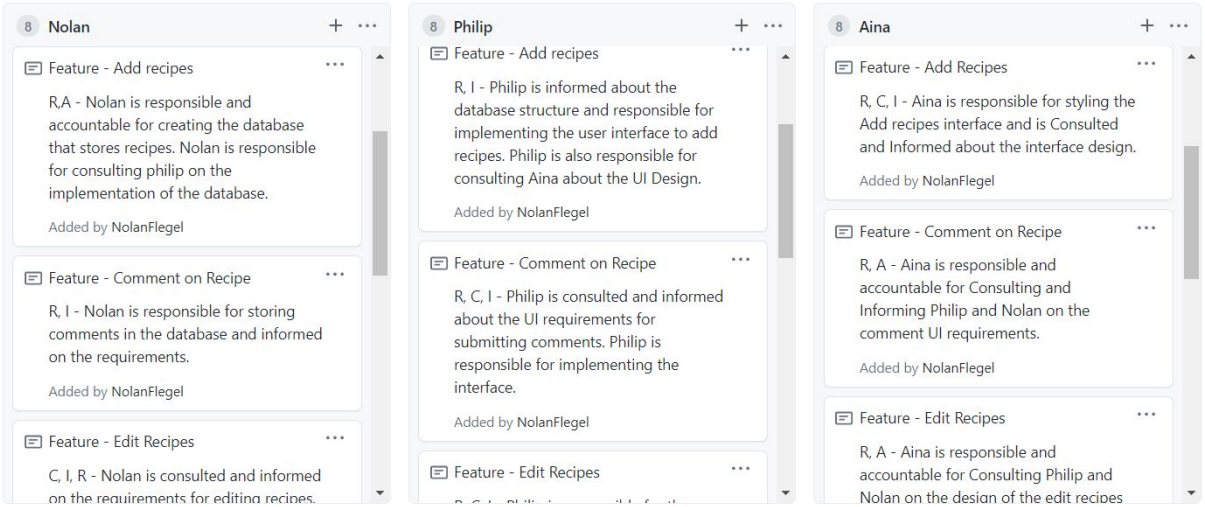


Figure 7: RACI Visual

With a good idea of what goals we wanted to achieve with the project and how we were going to achieve them, we were able to move forward with creating wireframes for the Community Cookbook. We made wireframes for each page we planned to design, but our most important pages were the home page and the add recipes page. This was because these two pages would be the main attraction for a user. If a user didn't like the home page or the functionality of adding recipes, they wouldn't use the application, so we had to make sure our wireframes for the two pages were impressive.

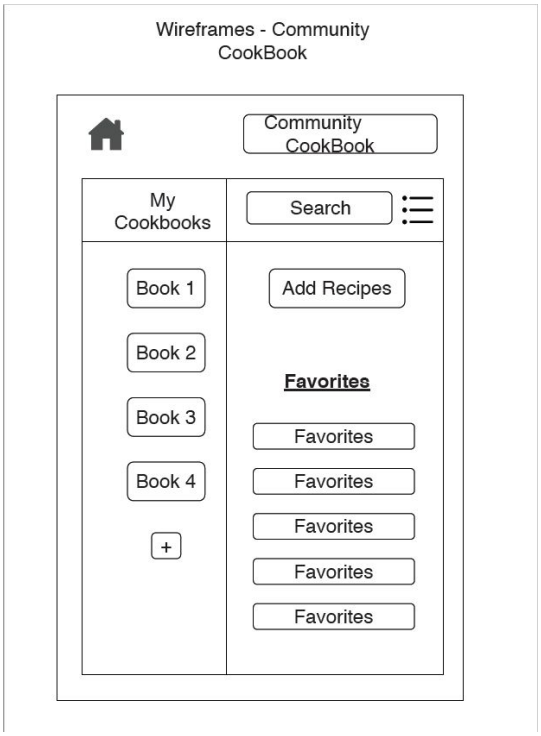


Figure 8: Homepage Wireframe

Figure 9: Add Recipes Wireframe

Phase 2

Once we had completed the planning phase, we were ready to start building our project. Referring back to our deliverables, we had split up each of our deliverables into the Model-View-Controller software design pattern. Building on that, we decided to begin by constructing different areas of our project based on our RACI charts.

Technology Used

The technologies we used for this project were Node.js, Github, ExpressJS, EJS, MongoDB, Mongoose, JQuery, and Bootstrap.

Node.js is a multi-platform, open source project that acts as a server environment for web projects. It runs on top of a Javascript runtime engine and allows developers to use javascript for standalone applications/services. The node package manager (NPM) is an ecosystem of open source libraries that can be incorporated into Node.js projects. Using this platform can streamline development and distribution of your projects through the use of node modules, which are reusable blocks of code that do not impact other modules.

Our project is built in Node.js and we used the packages for ExpressJS, EJS, Mongoose, MongoDB and JQuery.

For our Model we used MongoDB and Mongoose. MongoDB is a non relational database where data is stored as a collection of objects. A schema is not required for MongoDB and document fields can be created at the time of insertion. Each document can be different, in both size and content. Mongoose is a library for MongoDB that provides a schema structure to help programmers organize objects before storing them in MongoDB. Mongoose schemas can be used to build the structure of a document and provide an interface to the developer. These mongoose models can then be used to construct collections for MongoDB.

The front end View component of our project uses the technology EJS (Embedded Javascript) and ExpressJS. These are framework libraries in the Node Package Manager and are used for rendering HTML and Javascript for web applications. EJS can be combined with ExpressJS to help serve content from the Controller to the View. EJS can be used to create templates for HTML and these templates can be reused in other templates or pages in the web application.

JQuery is another library that we used in our project. This library simplifies javascript by condensing common tasks into methods that can be called in the scripts we write. There are

many plugins for JQuery that add further simplifications and functions to the default library. Some of the more frequent tasks that JQuery is used on are DOM manipulation, AJAX, and HMTL events.

Bootstrap is a web toolkit created by Twitter. It is used as a framework for HTML, CSS and Javascript. This front end development framework is used to provide many CSS styles and javascript animations, allowing developers to create responsive websites while saving programming time. We used Bootstrap classes and functions to implement our navigation menus and format our pages into a grid structure.

Git is a version control system that can be installed on your local development platform.
It creates a record of your programming and allows users to create branches of development. The tool provides simple controls for managing and maintaining projects. GitHub is an online cloud repository for hosting your git source control projects. GitHub provides an interface and visual tools for managing your repositories and providing a hosted storage solution that can be accessed anywhere remotely. These two tools are extremely useful in managing versions and merging work from multiple contributors, like group projects.

Feedback and Final Design

Right around the time we were finishing up creating a rough draft of our project, we received feedback from the Vinyl DeathStars group as per Activity 4. They provided some useful suggestions and gave us insight to important aspects of the user experience that we had overlooked. Based on these suggestions, we modified our application to fit these needs better.

The feedback and our response was as follows:

"A quick lookup on your MVC architecture and ERD as I would like to see if you can make it more integrated with adding users(_id) and associated it to your cookbook (bid)"

> Adding users to our app was a great recommendation and it would make the experience more personalized for the users. The current version of the application only runs locally, the cookbook is a single entity and represents your cookbook. As we scale up the product and move to a mobile application or website, we will need to integrate user accounts and a cookbook management interface.

"If your app is a localized webApp the time building your own cookbook from scratch, adding all the necessary details for a recipe makes me think that it takes a lot of time to do all these. Therefore the target audience/user which is the casual home cooks will lose interest in using the app since it takes a lot of time to make "

We agreed and would like to distribute the application with a number of our favorite recipes included. We added a future MVP to preload the database with a number of categories and recipes. Additionally, we would like to make it easy to share individual recipes between users.

"(LO-Fid document) - for my cookbook section, what if there are 100 of cookbooks would there be a scroller slider to look through"

We have decided to move the development of multiple cookbooks to a future MVP. We will need to reflect on the implementation of this feature. It would be possible to limit the number of cookbooks a user could add or we can explore other methods of presenting this feature.

"On the homepage/ Open Recipe page, I believe my cookbook/category list should be able to be minimized to maximize the screen space."

This feedback challenged us to evaluate how much information is presented to users on the homepage. Our original design called for the recipe to load within an area on the homepage. We changed our design to have the recipe open on a new page where the screen space could be dedicated to the recipe without other visual distractions.

"It might be wise to have a separate page dedicated to listing the recipes and categories in a specific cookbook, seeing as a user might have a lot."

With the removal of multiple cookbooks to a future MVP, our design was simplified. The user will only see categories for the single cookbook in the side menu. We also made the categories collapsible so it minimizes how much information is shown at one time.

"LOFI diagram- I think for the homepage you are showing way too much feature that may overwhelm the user(mostly mobile user). I think you should choose between whether to show favorites or books."

We also decided to remove the favorites option as per the last comment, it seemed to cause more clutter than desired, and instead we made the decision to load the user's five most recent recipes on the home page instead.

Project Demo and Reflection

The process of demoing our project and creating a presentation allowed us to reflect on the things we learned over the course of this class. Our team discovered that our biggest takeaway was how to efficiently work together. Not only did we get a first taste of various software languages we hadn't used before, but this class also taught us what project management is all about. The different activities forced us to reevaluate our progress at each milestone and make sure we hadn't strayed off track, and the different scrums with the class and professor gave us a chance to see and hear different perspectives.

One of our major challenges was communication. We learned that communication and regular check ins are important to success. When challenges and obstacles are communicated, together as a team we can reprioritize and redistribute work. These are also opportunities to collaborate and learn from each other as we worked through problems together. We saw team members step up and take on more work if one of us was falling behind or providing constructive feedback, we were able to really grow as a team.

We also found it difficult to put into practice the concept of RACI. While individuals had different things that they were responsible, accountable for and tasks that should be consulted and informed about; we found we did not have a mechanism to manage conflicts related to these tasks. What happens when a team member implements a feature without consulting or informing the group. If someone fails to produce a component that they were responsible and accountable for; how do you resolve that? We found that some of these situations can be avoided with regular check ins, status updates, and forthcoming communication.

Our team found it difficult at times to agree on certain ways to implement ideas, and that was an important learning experience because it forced us to consider each person's points and find a compromise. We were able to create a project that embodied the goals we had set out at the start of the project, while also focusing on the user experience and stories.

The final product encompassed several key MVPs. We were able to present an interface where users could view and navigate the cookbook. Users can add additional categories to the cookbook that are more personalized and add recipes to each of the categories. We also created an interface where users can add new recipes to the cookbook, adding a picture and managing their recipe. Lastly, one of the more important features we were able to implement was the ability to comment on your recipes, take notes and track small changes over time as a user makes the same recipe time and again. Overall we are proud of the product we produced and we can envision our future MVPs and what is needed to accomplish them.