Recipe Manager Final Report

Team: Ryan Herb, Dalton Smith, Bo Swartz, Nicolette Hart

CMSC 495

Current Trends and Projects in Computer Science

Instructor: Castillo

Table of Contents:

Overview…………………………………………………………......3

Project Plan……………………………………………………………4

Requirements and Specifications…………………………………………….…7

User Guide

Adding Recipes…………………………………………………………….8

Deleting Recipes…………………………………………………………...9

Modifying Recipes…………………………………………………………10

Searching Recipes………………………………………………………….11

Selecting a Recipe…………………………………………………………12

Test Plans and Results

Testing Summary…………………………………………………………14

User Stories and Testing Schedule…………………………...………………...15

Test Cases……….…………………………………………………………16

Other Testing………………………………………………………………19

Bug Report Template..……………………………………………………20

References…………………………………………………………………21

Designs and Diagrams……………………………………………………………22

Project History/Timeline……………………………………………..26

Conclusions………………………………………………………… 27

Overview:

Recipe Manager was created by team members: Ryan Herb, Dalton Smith, Bo Swartz, and Nicolette Hart.

The project was programmed using Java 8.0 and Spring Boot for the back-end. Database functionality was developed with Mongodb. The front–end was developed with jQuery, HTML, and CSS. The project is running in a Kubernetes cluster, hosted on the Google Cloud platform.

Recipe-Manager Source Code: https://github.com/rherb94/recipemanager

Kubernetes-Manifest Files: https://github.com/rherb94/kubernetes-manifests

Recipe-Manager can be accessed here: http://35.238.236.116/

Thank you for using Recipe Manager!

Project Plan:

Project Management:

Description and Objective of the project:

Our project is a recipe web-app where you can add, edit, and delete recipe entries. Additionally, we built a continuous Integration/Deployment pipeline that automatically builds, tests, and deploys our web-app to an active Kubernetes cluster hosted on the Google Cloud Platform. The goal of this project is to work in a team environment, utilizing the latest tools and technologies that are used in workplaces around the world.

The objective of the project is to learn how to develop a web-app in the scope of a real-life development team. We will be focusing our attentions on using a proper branching strategy. Additionally, we will be setting up a continuous integration/deployment pipeline that delivers code changes more frequently and reliably. This will allow us to automate the process of deploying our project. Finally, we will be building a scalable Kubernetes cluster, which is where it will be running. This is the latest technology available, and companies across the US are transitioning from deploying their applications on physical hardware to virtual instances such as Kubernetes. This will allow us to easily scale and manage our containerized web-app.

What problem is the project attempting to solve:

The problem that our project is attempting to solve is the inherit possibility of things to go wrong manually deploying software. Automating the process not only saves time and money, but it also minimizes the risk of production failures. Additionally, if things do go wrong, it will be a lot quicker to patch fixes with an automated process.

What project management methodology will you be using to manage your project (e.g. Scrum Agile, Kanban, Waterfall, etc.):

We choose to use the Agile methodology for managing our project. This will be achieved by assigning user stories to team members that will fulfill the functional requirements of our project. Additionally, we will be using project management tools like Slack to coordinate our efforts to document and track weekly goals and deliverables.

What tools will you use to capture Analysis and Design artifacts (e.g. UML tools, Visio, etc.)?

We will be using UML tools as well as Intellij to capture analysis and design artifacts.

How often will you communicate with your team:

We will be assigning weekly goals and communicating throughout each week for clarification and assistance as needed.

Who comprises your team:

Team Lead: Ryan Herb

Key Functions and Milestones: leading the group and directing the individual activity.

Project Manager: Dalton Smith

Key Functions and Milestones: updating the project plan weekly along with maintaining a count on defects and issues.

Tech Members: Nicolette Hart and Bo Swartz

Key Functions and Milestones: Completing individual user stories.

Project Analysis:

High level Epics and stories (if using Agile) describing the use cases you are trying to solve:

Epic: User-friendly, easy, and useful Recipe database App

Stories:

- Users need to be able to add recipes that persist in mongo.
- Users need an option to edit recipes.
- Users need an option to delete recipes
- Users need an option to search for a particular recipe
- Design front-end of website
    - Add tables and forms for each operation a user could perform
- Team needs to set-up a continuation integration/deployment pipeline
    - This will be defined by pipeline stages (checkout code, build project, and deploy jar and Docker image to package manager).
- Team needs to provision a Kubernetes Cluster
    - Need to have pod containing the project jar, as well as a pod containing a MongoDB instance.

Classes:

- Using SpringBoot/MongoDB for basic database elements (Add, Delete, Modify,…)
- Recipe Class Hierarchy:
    - Private string recipeID
    - Private String contributerName
    - Private String title
    - Private String instructions
    - Private String notes
- Main Page (index.html)
    - Add a recipe button
    - Search field
    - When a recipe entry is clicked on, options to modify or delete the recipe entry will appear to the user
- Project Information page
    - This page will contain information about the team. There will be a link included to our Git repository, as well as links to our continues integration/deployment pipeline and our Kubernetes cluster. We will also have our documentation files in this area, so that all we will need to turn in for our project is the link to access our web-app.

Actors in your system:

- Parents
- Active/Employed Individuals
- Chefs

Hardware and Software tools required:

Our project requires the use of a PC for project development. Along with this, our project will require either a PC or smart phone to run and test our project's progress and development. As for software, our Recipe app will include the use of Java JDK language, Spring Boot for development of the framework, and MongoDB for creating the data base of our application.

Requirements and Specifications:

Hardware:

- a PC/Mac or smart phone to run and test our project

System/Software:

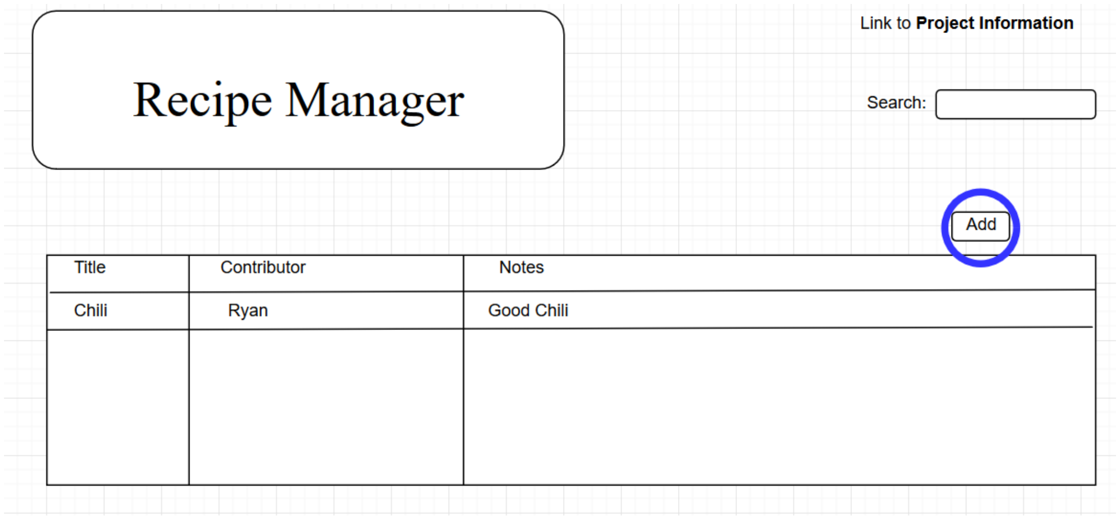
- Latest Installment of Java (Java 8.0)
- Windows (7, 8, or 10) **OR** Mac OS X
- Stable Internet connection

User Guide:

This guide is intended for users who have already followed the provided link to Recipe Manager. If you have not yet clicked the link, please refer the **Requirements and Specifications** page for the installation instructions.

1. **Adding Recipes:**

   On the main page of Recipe Manager, you will see a list of recipes for you to choose from as shown in Figure 1. Select the **Add** button located near the right side of the screen. (Indicated by the blue circle) A window titled "**Recipe Details**" will appear like Figure 2 shows.



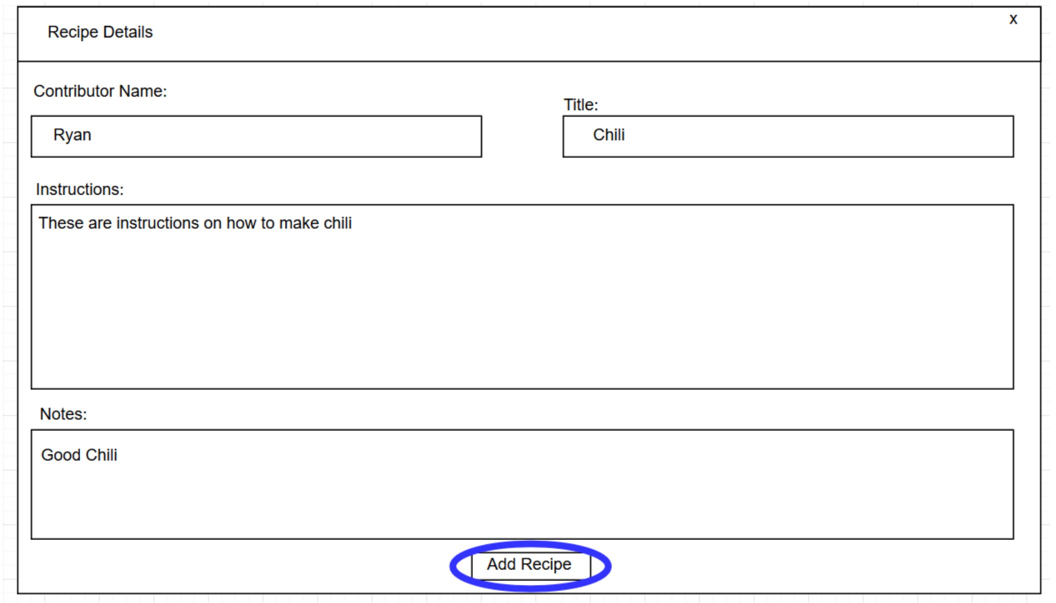
Title	Contributor	Notes
Chili	Ryan	Good Chili

(Figure 1)

The "**Recipe Details**" window (Figure 2) will show you all the components of creating a new recipe for Recipe Manager.

- **Contributor Name**: Name of the maker or provider of the recipe (Your name)
- **Title**: Title of the recipe you are adding
- **Instructions**: The recipe instructions
- **Notes**: Comments or side notes about the recipe

Once you are finished adding the details of the recipe, click on the "**Add Recipe**" button located near the bottom of the window, indicated by the blue circle. This will save your recipe to Recipe Manager. (Displayed in Figure 2)



Contributor Name: Ryan

Title: Chili

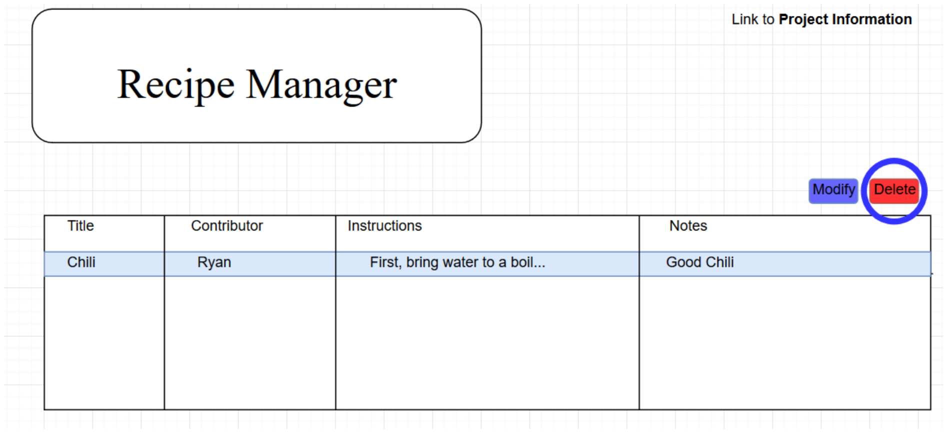
Instructions: These are instructions on how to make chili

Notes: Good Chili

(Figure 2)

2. **Deleting Recipes:**

   On the main page of Recipe Manager, you will see a list of recipes for you to choose from as shown in Figure 1. To delete a previously added recipe from Recipe Manager, simply click on the recipe listing from the list. Doing so will highlight the recipe like so:



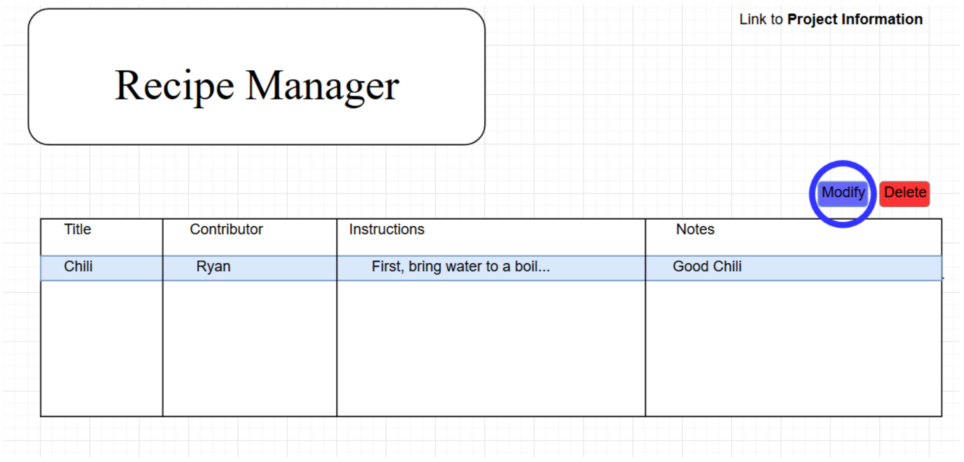
Title	Contributor	Instructions	Notes
Chili	Ryan	First, bring water to a boil...	Good Chili

(Figure 3)

To delete the selected recipe, click on the **Delete** button found near the right side of the screen and indicated by the blue circle. **WARNING:** Doing this will permanently remove the recipe from Recipe Manager. To regain the recipe, you must make a new recipe with the details again by using the **Add** button.

3. **Modifying Recipes**:

On the main page of Recipe Manager, you will see a list of recipes for you to choose from as shown in Figure 1. To modify a previously added recipe from Recipe Manager, simply click on the recipe listing from the list. Doing so will highlight the recipe like so:



Title	Contributor	Instructions	Notes
Chili	Ryan	First, bring water to a boil...	Good Chili

(Figure 4)

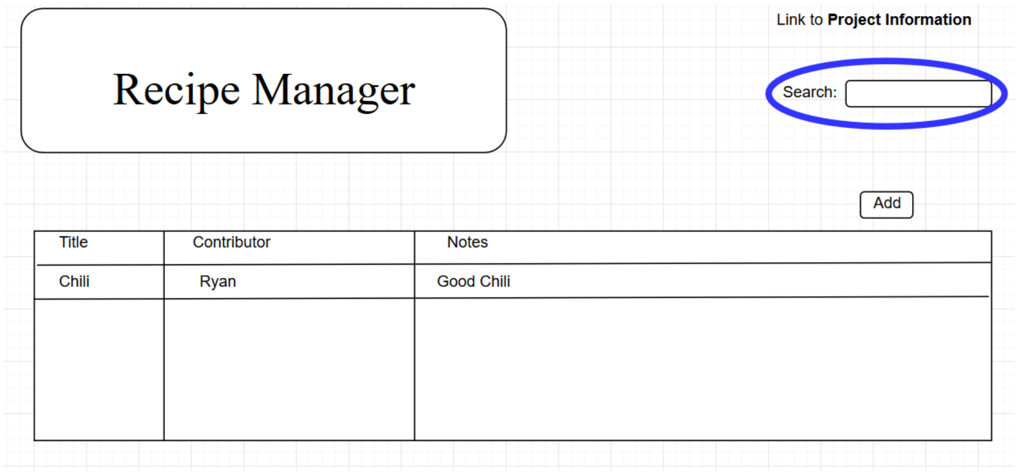
To modify the selected recipe, click on the **Modify** button found near the right side of the screen and indicated by the blue circle. Doing so will bring up the "**Recipe Details**" window. (Figure 2) From here, you can edit all the components of the recipe including:

- **Contributor Name**: Name of the maker or provider of the recipe (Your name)
- **Title**: Title of the recipe you are adding
- **Instructions**: The recipe instructions
- **Notes**: Comments or side notes about the recipe

When you are finished modifying the recipe, click on the "**Add Recipe**" button located near the bottom of the window, indicated by the blue circle. (Displayed in Figure 2) This will save your changes to Recipe Manager.

4. Searching Recipes:

To search for a recipe, click on the textbox next to the word "**Search**", located near the upper right corner of the screen, indicated by the blue circle. (Figure 5)



Title	Contributor	Notes
Chili	Ryan	Good Chili

(Figure 5)

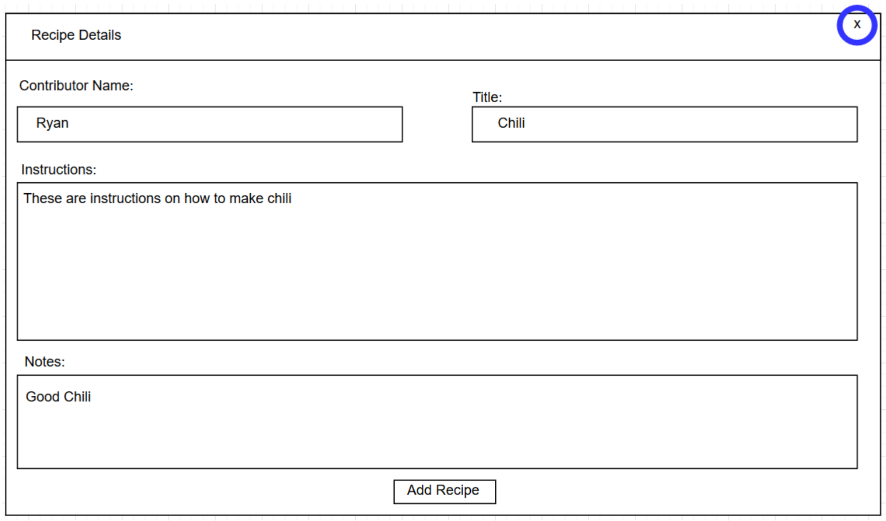
Type in a name, title, or word that corresponds to what recipe you would like to find. For example, if you wanted to just look at recipes added by Ryan, you would type in "Ryan" into the textbox and press the "**Enter**" key on your keyboard.

Recipe Manager will then give you a list of recipes that correspond to the word you entered. If Recipe Manager does not return any recipes after searching your chosen word or none of the recipes returned are what you were looking for then there are a few things you could do to help trouble shoot the problem:

- **Ensure the spelling is correct in the text box:** Recipe Manager does not come with a built-in autocorrect so correct spelling of words falls to the user.
- **Choose another search term**: For example, if you want to look for "Chicken Soup" but using the word "Chicken" comes up with too many unrelated results, a suggestion would be to use both words, "Chicken Soup" or to change the word from "Chicken" to "Soup"
- **Make sure the recipe exists**: If the recipe you are looking for does not exist in Recipe Manager, adding the recipe will be the best course of action. To learn how to do so, please review the **Adding Recipes** (pg.) section in the User Guide.

5. Selecting a Recipe:

On the main page of Recipe Manager, you will see a list of recipes for you to choose from as shown in Figure 1. To view a previously added recipe from Recipe Manager, simply double click on the recipe listing from the list. Doing so will highlight the recipe and bring up the "**Recipe Details**" window for you to view. (Figure 6) When you are done viewing, close the window by clicking on the "**X**" symbol in the upper right corner of the window, indicated by the blue circle.



Contributor Name: Ryan

Title: Chili

Instructions: These are instructions on how to make chili

Notes: Good Chili

(Figure 6)

1. Test Plan and Results

This document outlines the test plan for our Recipe Manager project. This is a preliminary test plan and will likely change and as we continue to work on the project. As new issues and challenges will inevitably arise throughout development, it is necessary to have an adaptable and flexible test plan. Therefore, it is important to start with a basic and solid foundation of testing that can be built on as the project evolves.

We will be utilizing the follow software testing principles:

Principle	Explanation
1. Exhaustive testing is not possible	Any project with a deadline does not have time to test every possible case. You must consider the risks and priorities of your project and focus on testing the most defect-prone parts.
2. Defect clustering	According to the Pareto Principle, 80% of the problems are due to 20% of the causes. In software development, this means that 80% of defects are most likely related to just 20% of the project.
3. Pesticide paradox	By running the same tests repeatedly, chances are that no new bugs will be discovered. As the project evolves through the development cycle, the test plans must evolve as well.
4. Testing shows presence of bugs	A test result with no defects does not guarantee that there are no other errors. It is important to design test cases to be as comprehensive as possible. Even with extensive testing, there can still be undiscovered bugs.
5. Absence of error	Just because a test found no defects does not mean the software is ready for deployment. A program that is 99% bug-free may still be unusable. It is important to make sure that the program is fully functional on the users end.
6. Early testing	The sooner you start testing, the better. This way any defects in the development cycle can be caught early on. It is much easier to deal with bugs early on than at the end of the development life cycle.
7. Testing is context dependent	The way you test your software should represent the consumer the software is intended for. For example, testing video game software vs. medical software or a low traffic website vs. a high traffic website.

2. Testing Summary

2.1. Testing Objectives

2.1.1. Usability

A. Layout

a. Homepage should grab user’s attention
b. User should not feel lost, navigation should be simple
c. All available functions should be obvious and easy to find
d. Contact information is displayed on the page
e. Responsive to browser window resizing
f. Consistency among layout between pages
g. All functions are responsive to user interactions
h. Existing recipes are displayed on the homepage

B. Navigation

a. Navigation options consistent between pages
b. User knows what page they are on at all times
c. Returning to home page is easy from any location
d. Site navigation options are specific and use descriptive names

2.1.2. Interface Funtionality

A. Add function

a. Recipe Details window opens when selected
b. All Recipe Details fields are responsive to user input
c. Adding the new recipe saves it to be viewed later

B. Delete function

a. All recipes should have a delete option
b. Deleting a recipe should remove it permanently

C. Modify function

a. All recipes should have a modify option
b. Selecting the modify option opens the Recipe Details window
c. User can edit the existing recipe and save their changes

D. Search function

a. Search box is responsive to user input
b. User can search by title, author, or keywords
c. If there is a match, the recipe will be displayed
d. If there is no match, no recipe will be displayed

E. Select function

a. Double clicking a recipe will open the Recipe Details window

2.1.3. Compatibility

A. Web Browser

a. The project is compatible with all commonly used web browsers (Chrome, Safari, Firefox, Internet Explorer, Edge, Opera, etc.)

B. Mobile Devices

a. The project is compatible and responsive to mobile phones and tablets.
b. Resizing occurs seamlessly on smaller screens

3. User Stories and Testing Schedule

3.1. Test Phase 1 – Basic Interface and User Experience

a. As a user, I want to use the given URL to access the website with no problems.
b. As a user, I want to see a simple and clean homepage with the Recipe Manager name displayed.
c. As a user, I want to see my list of recipes on the home page (unless I have not added any yet).
d. As a user, I do not want to feel lost on the website, so I should know page I am currently on at all times.
e. As a user, I should be able to navigate back to the homepage from any other pages.
f. As a user, I want to see the contact information for the developers of the project somewhere on the homepage in case I have questions.
g. As a user, I want the website to be responsive to me resizing my browser window or using my mobile device to access the site.
h. As a user, I want to see the add and search functions clearly and distinctly on the homepage because I will be using them very often.
i. As a user, I want to see modify and delete options on my existing recipes in case I want to make any changes or deletions.

3.2. Test Phase 2 – User Interaction and Funtionality

a. As a user, I want to click the add recipe button and be prompted with the Recipe Details window where I can input the information for my recipe.
b. As a user, I want my added recipes to show up on the homepage after I finish adding them and click add recipe.
c. As a user, I want to be able to search my existing recipes by author name, title, or keyword by typing into the search bar on the homepage.
d. As a user, after searching I should be given a list of recipes filtered by my search input.
e. As a user, I should be able to double click on an existing recipe to open the Recipe Details window.
f. As a user, I want to be able to modify a recipe by selecting it from the list and clicking the modify button.
g. As a user, I want to be able to delete a recipe by selecting it from the list and clicking the delete button.

3.3. Testing Schedule

Task Name	Duration	Week 4	Week 5	Week 6	Week 7	Week 8
Development Phase	5 weeks	🟦	🟦	🟦	🟦	🟦
Test Phase 1 – Interface	2 weeks	🟨	🟨			
Test Phase 2 - Functionality	2 weeks			▨	▨	

This schedule is an estimate based on our pre-development expectations. It may change as development progresses and new restrictions and requirements are introduced. Our testing period will last for a total of 4 weeks and we expect to complete all testing before Week 8. This gives us time in the final week to deal with any last-minute issues and prepare for our final presentation and submission.

4. Test Cases

4.1. User Story Test Cases

Phase	Test Case	Test Objective	Pass/Fail
1	URL leads to website with no problems	User should be able to access site with URL and very little loading time	Pass
1	Homepage is simple, clean, and appealing to users	UI should be clean and appealing to the eye. Team members can give input on this and we could also get first impressions of outside parties (other project groups or friends/family)	Pass
1	User knows what web page they are on at all times	Every page is clearly titled and defined so the user knows where they are and the purpose of the page they are currently on	Pass
1	Existing recipes are displayed on the homepage	All existing recipes are listed on the homepage to represent an online cookbook	Pass
1	Add and search functions are displayed on the home page	Add and search functions are displayed on the home page and are clearly named and identifiable, as they are the main user tools	Pass
1	Modify and delete functions are displayed when a recipe is selected	Modify and delete options are displayed when the user selects an existing recipe. They are not displayed on the homepage until a recipe is selected because add and search are the primary functions.	Pass
2	Clicking the add recipe button opens the Recipe Details window. Entering information and saving adds the recipe to the homepage. (add RTC1, RTC2)	Add function is the primary tool for the user, this test case will ensure that the add button and Recipe Window function correctly and the new recipe is displayed on the homepage	Pass

2	Entering a word into the search bar will search the existing recipes and return any recipes containing a match. (RTC1 - search for Bo, Buffalo, Dip)	Search function allows the user to search through their online cookbook instantly. They can search by author, title, or keyword. The existing recipes will be filtered based on the search input.	Pass
2	Double clicking on a recipe will open the Recipe Details window containing a full list of instructions and notes	The recipe information will be displayed in full, including the instructions and notes. The recipes are still displayed on the home page, but to avoid clutter the full recipe will be accessed through the Recipe Details window.	Pass
2	Selecting a recipe and clicking the modify button will open the Recipe Details window and allow the user to make changes (RTC1 – modify instruction to “Bake for 25 minutes”)	The modify function allows the user to make edits to an existing recipe in their cookbook.	Pass
2	Selecting a recipe and clicking the delete button will remove the recipe permanently	The delete function allows the user to permanently remove a recipe from their cookbook.	Pass

4.2. Recipe Test Cases (RTC)

Test #	**RTC1**
Author	Bo Swartz
Title	Buffalo Chicken Dip
Instructions	• Cook 1 pound of chicken and shred. • Mix in 16oz cream cheese, 1 cup Ranch dressing, ½ cup buffalo sauce, ½ cup shredded cheese. Stir until combined. • Move mixture into large baking dish. Top with ½ cup shredded cheese • Bake for 20 minutes at 350°F • Let cool and serve with tortilla chips
Notes	Perfect for parties

Test #	**RTC2**
Author	Bo Swartz
Title	Iced Tea
Instructions	• Heat 8 cups of water to a rapid boil • Remove from heat and drop in 3 tea bags of your favorite tea • Cover and let the tea steep for 1 hour • In a large pitcher, combine the tea and ¾ cup white sugar • Add ½ cup lemon juice and stir well until sugar dissolves • Refrigerate until chilled
Notes	Great for a hot summer day

5. Other Testing

5.1. Stress & Volume Testing (S&V)

We can use the following site for this: https://loadimpact.com/

5.2. Connectivity Testing (CT)

We can use the following site for this: https://www.webpagetest.org/

5.3. REST Endpoint Testing

We used Postman to test the functionality of our backend code. Our project supports the REST functions of POST(Create a resource, PUT(Modify a resource), and DELETE(Delete a resource). We used Postman to send mock requests for each of our REST endpoints, and it provided us with the server's response and the corresponding http status code. Additionally, we used Chrome Developer tools to test. This allowed us to see what's going on behind the scenes and provided us with data corresponding to the resources that we are creating/modifying/and deleting at the time to ensure everything is working properly.

6. Bug Report Template

Bug ID #	#000
Reporter	Name
Submitted Date	MM / DD / YYYY
Summary of Bug	Short description
Screenshot	Link to image
Operating System / Browser	Example: Windows 7, Google Chrome 70
Priority	Low / Medium / High

Description:

Steps to Reproduce:

Expected Result;

Actual Result:

Notes:

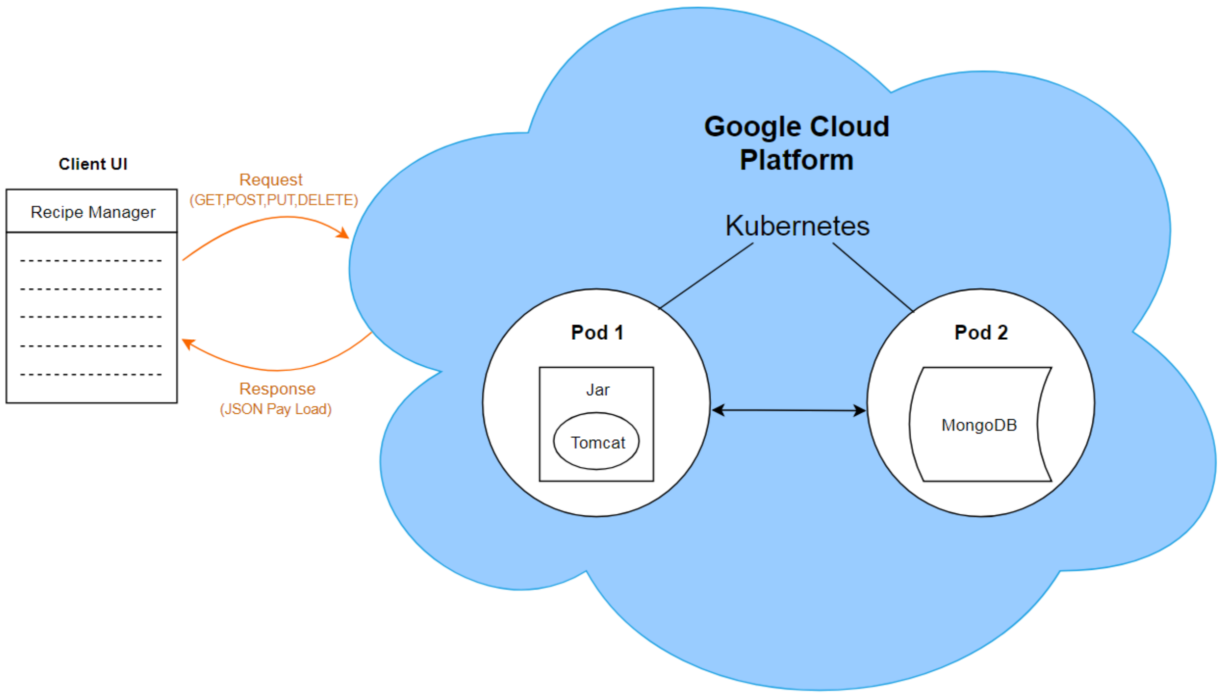
7. References

The following documents have been used to assist in creation of this document.

#	Reference name	Comments
1	https://strongqa.com/qa-portal/testing-docs-templates/test-plan	Adapted from Test Plan Template-04
3	https://www.lifewire.com/rules-of-effective-website-navigation-1697492	Site navigation priorities
4	https://www.testingexcellence.com/seven-principles-of-software-testing/	Test plan principles
5	https://www.guru99.com/software-testing-seven-principles.html	Test plan principles
6	https://www.mountaingoatsoftware.com/agile/user-stories	User story definition
7	https://marker.io/blog/bug-report-template/	Adapted bug report template
8	https://www.guru99.com/performance-vs-load-vs-stress-testing.html	Performance testing

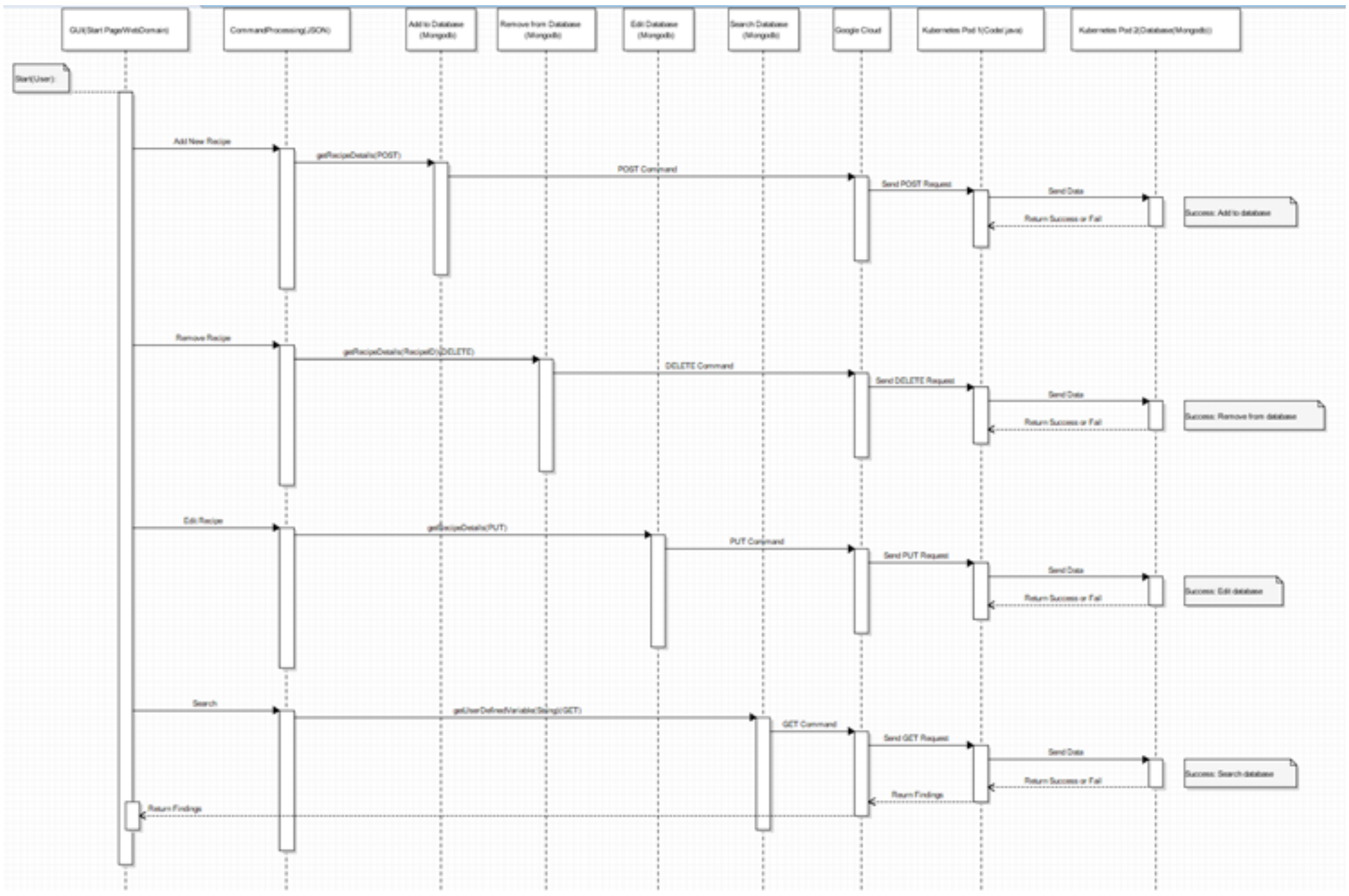
Designs and Diagrams:

High Level Architectural Diagram:



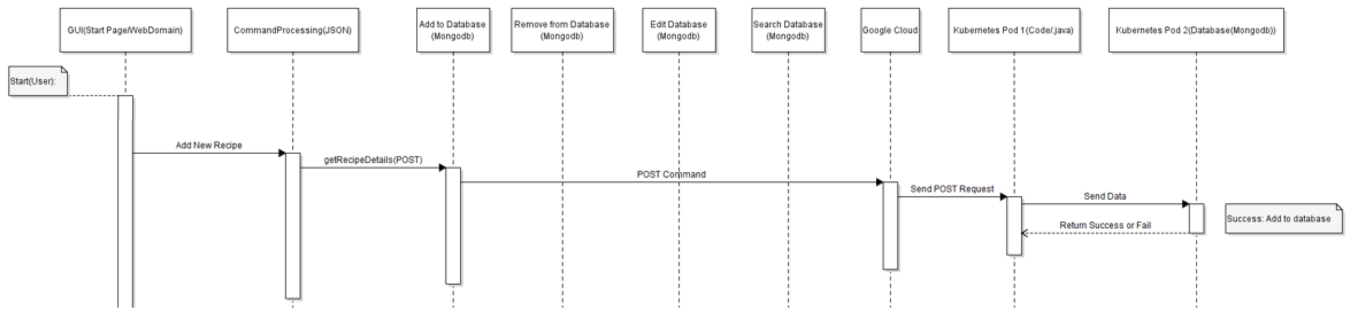
Sequence Diagram:

- **Full Program View:**

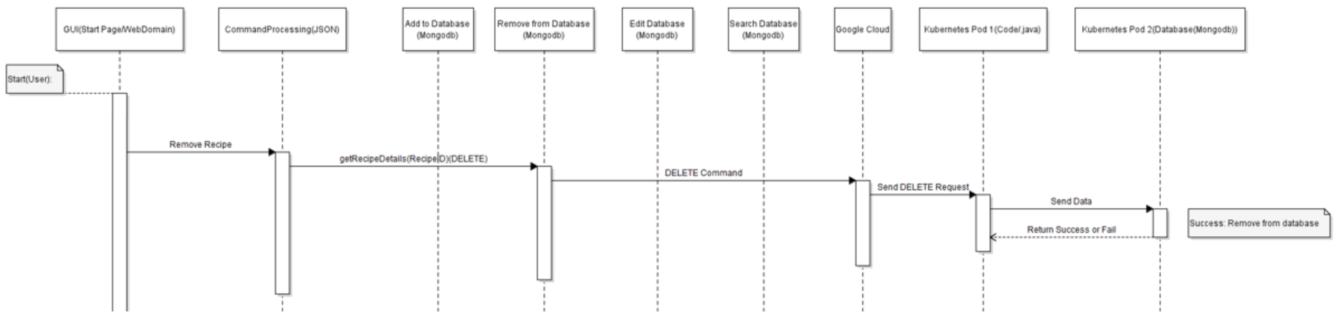


NOTE: Please note that the detailed views of the Sequence diagram are created to better show the sequence of events for each component (POST, DELETE, PUT, and GET) of the Recipe Manager program. For each detailed view, the event objects (headers) are added as a reference for the viewer and do not represent the sequence diagram as a whole. To view the full and accurate sequence diagram, please refer to the Full Program view in this section (the diagram pictured above).

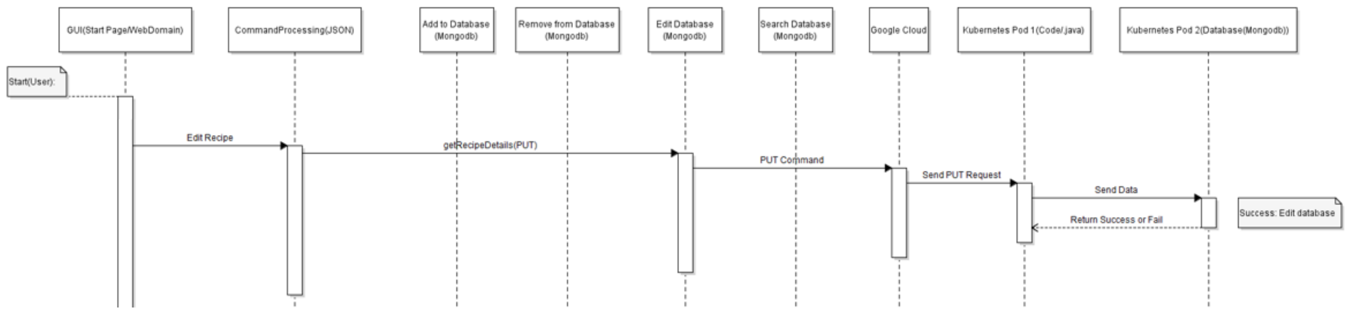
- **Detailed View of POST(Add Recipe):**



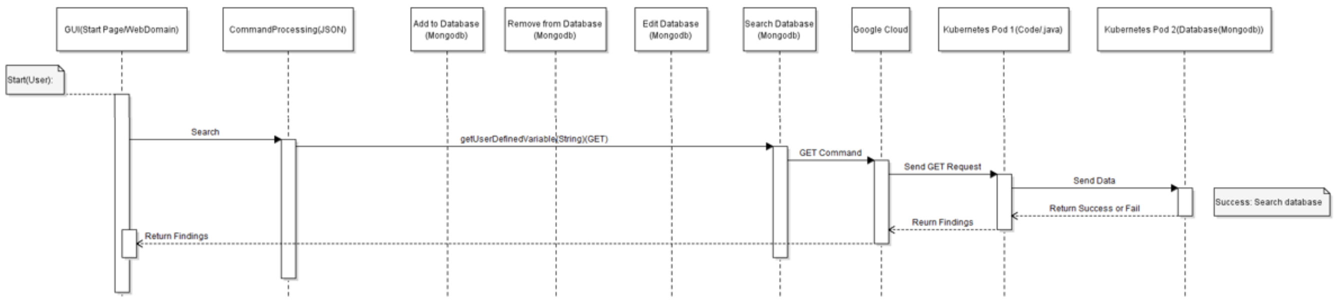
- **Detailed View of DELETE(Remove Recipe):**


GUI(Start Page/WebDomain)
CommandProcessing(JSON)
Add to Database (Mongodb)
Remove from Database (Mongodb)
Edit Database (Mongodb)
Search Database (Mongodb)
Google Cloud
Kubernetes Pod 1(Code/.java)
Kubernetes Pod 2(Database(Mongodb))
Start(User):
Remove Recipe
getRecipeDetails(RecipeID)(DELETE)
DELETE Command
Send DELETE Request
Send Data
Return Success or Fail
Success: Remove from database

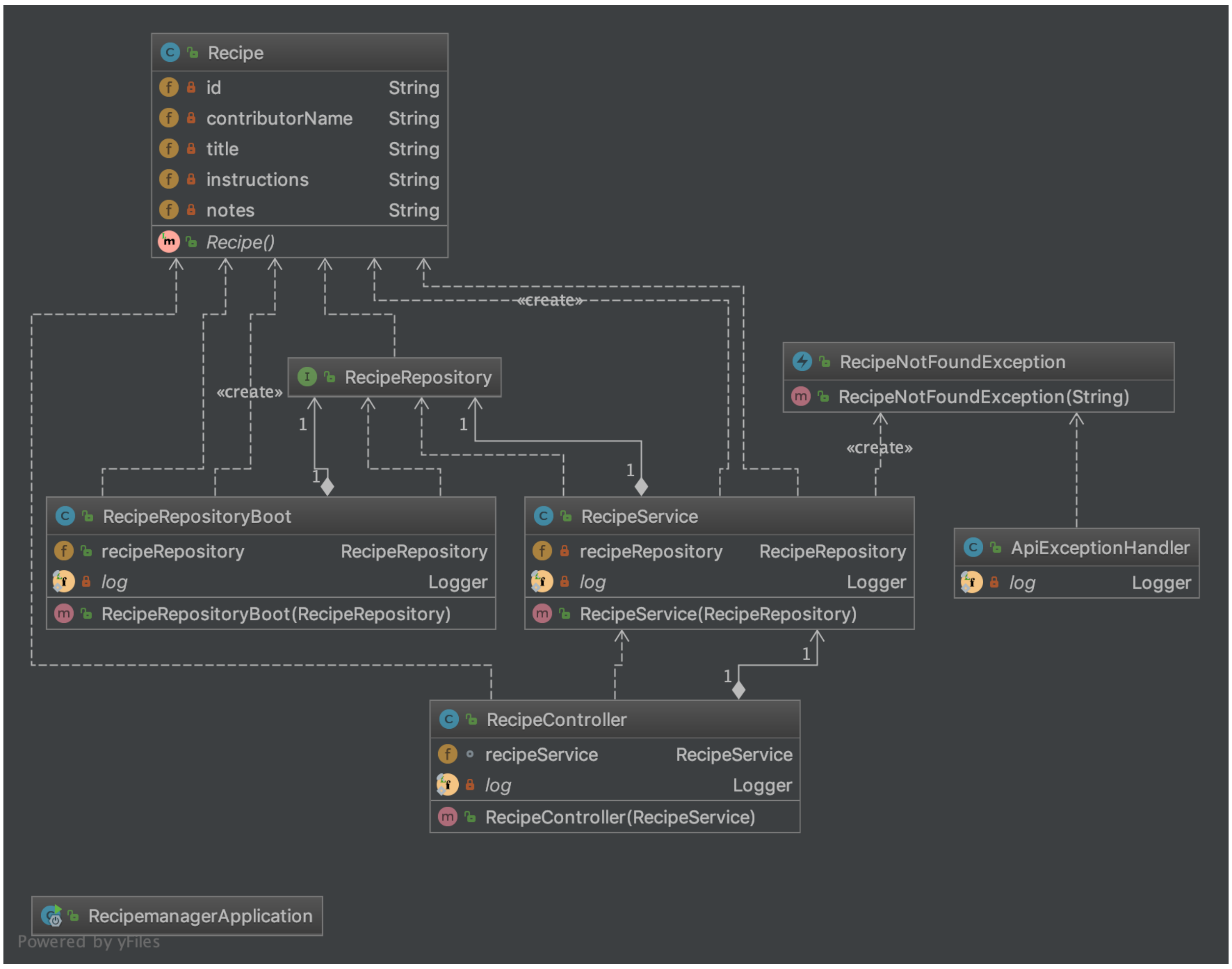
- **Detailed View of PUT(Edit Recipe):**



- **Detailed View of GET(Search):**

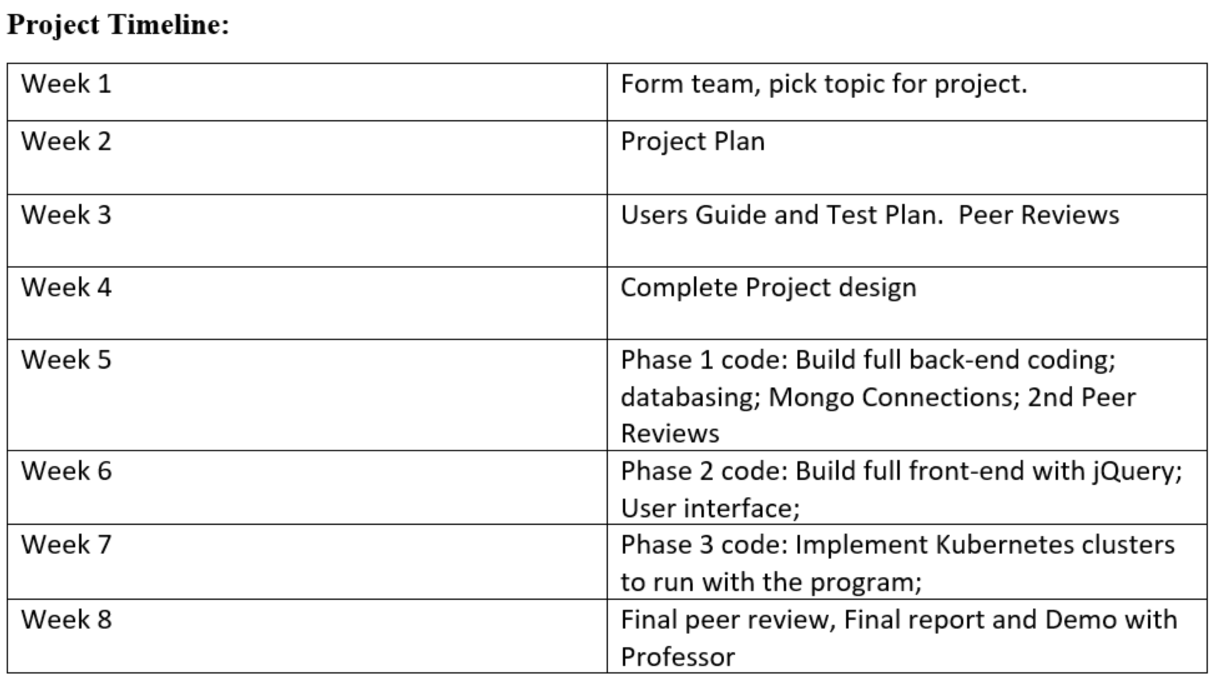

GUI(Start Page/WebDomain)
CommandProcessing(JSON)
Add to Database (Mongodb)
Remove from Database (Mongodb)
Edit Database (Mongodb)
Search Database (Mongodb)
Google Cloud
Kubernetes Pod 1(Code/.java)
Kubernetes Pod 2(Database(Mongodb))
Start(User):
Search
getUserDefinedVariable(String)(GET)
GET Command
Send GET Request
Send Data
Success: Search database
Return Success or Fail
Reurn Findings
Return Findings

UML Diagram:



Project History/Timeline:

Over the course of the eight weeks, our group planned out what the project needed and when each goal should be met. Once the programming weeks started (week 5), we realized our previous timeline needed to be updated due to new programming information and project goals. Figure x shows the final timeline we used in order to keep track of our progress.



Project Timeline:

Week 1	Form team, pick topic for project.
Week 2	Project Plan
Week 3	Users Guide and Test Plan. Peer Reviews
Week 4	Complete Project design
Week 5	Phase 1 code: Build full back-end coding; databasing; Mongo Connections; 2nd Peer Reviews
Week 6	Phase 2 code: Build full front-end with jQuery; User interface;
Week 7	Phase 3 code: Implement Kubernetes clusters to run with the program;
Week 8	Final peer review, Final report and Demo with Professor

Conclusions:

Lessons Learned:

Throughout this session, our team learned very valuable lessons both as a whole and from one another. Early on, we learned that group/project organization was incredibly important to the project's progression. Along with this, we found that conducting weekly meetings as a group helped us outline jobs, goals, successes, problems, and solutions for the week which made planning much easier than the first few weeks that we did not hold meetings. Our team consisted of differing levels of programming experience and with that, there were many conversations in teaching one another and providing more information on subjects such as Spring Boot, JSON, Maven, GitHub, Mongodb, jQuery and many more. All in all, this course provided us not only lessons in group organization but also lessons in how to become better in our fields of study.

Project Strengths:

We utilized a modern approach to develop our project, and used the industries best tools to help us. We architected a scalable Kubernetes Cluster that has the capacity to support more volume than the scope of our project demands.

The structure of the Kubernetes cluster we implemented for recipe-manager consists of three pods. We utilized pod replication to ensure that our webapp pods hosting our spring boot project always has 2 running pods. The final pod is our mongodb pod which holds the database of our application. We architected a persistent volume claim in our mongo pod deployment, which allows mongo to write to an SSD drive in the cloud.

Project Limitations:

The only thing that we didn't include in this project was a Continues Integration/Deployment Pipeline. This would have been attached to the master branch of our repository via a Service Hook, and would automatically build, test, and then deploy to our Kubernetes Cluster. The reason we choose not to implement this was that there were no decent free tools available to build this pipeline.