CS5704 Software Engineering

Semester Project Report

Fall 2021

Good Bite

Ishaan Gulati
Jack Sloane
Neha Surana

Department of Computer Science
Virginia Tech
Blacksburg, VA 24061

App's URL: https://cs5704gulati.live/GoodBite/

Date: 8th December, 2021
Team Number: 5
Instructor: Prof. Osman Balci

USER ACCOUNTS

Using the deployed software, each team member is required to create an account and generate *meaningful content* in the database for evaluation and testing. **You will be penalized for not having sufficient content**.

In the case of two-factor authentication, your software is required to provide a Bypass Two-Factor Authentication option for grading.

List the usernames and passwords below to use for grading the app.

Username	*Password*
ishaangulati97	Ishaan@2021
nehasurana	Password123#
sloanej	Password123!

EXECUTIVE SUMMARY

This document provides an in-depth outline of the GoodBite web application developed by CS5704 Software Engineering Team 5. Section 1 outlines our software engineering practice using the software engineering life cycle. Section 2 contains the description of our objective in undertaking this project. Section 3 outlines our project specification, including a high-level description of the problem our application aims to solve, why this problem is important, the expected functionality which we intend to include in our application, and a description of features in our application that were learned from previous examples in the CS5704 course, plus the features that are newly developed outside of our class experience. Section 4 outlines the requirements specification of our application in terms of functional and non-functional requirements with each requirement listed as a responsibility of a team member. Section 5 outlines our architecture specification. In this section we describe the network-centric / cloud-based architecture of our software-based solution system using the DoDAF models. Section 6 outlines our design specification, presenting the UI design and Database design with annotated graphics. Section 7 describes all the functionalities of our deployed application by using screenshots of user interfaces annotated with in-depth explanations. In section 8, we provide some concluding remarks of our software development journey, with reflections on what we learned and how we could have improved the software development process.

List of Implemented Cloud Software Features Learned

1. Create, Read, Update, Delete (CRUD) operations in a MySQL database.
2. Template-based User Interface (UI) with header and footer implementation.
3. Use of JPA to interact with MySQL.
4. Obtain JSON from an API and process JSON data.
5. Use PrimeFaces content flow to display images on the homepage.
6. User login flow, reset password flow, and sign up flow.
7. Create cloud storage for the user to upload/download files.
8. The use of Datatable, Datagrid and Datalist for displaying data.
9. Responsive Dialogues with Scrollable contents
10. TextBox Scrollable Vertically
11. Database search for geocodings
12. Custom CSS sheets

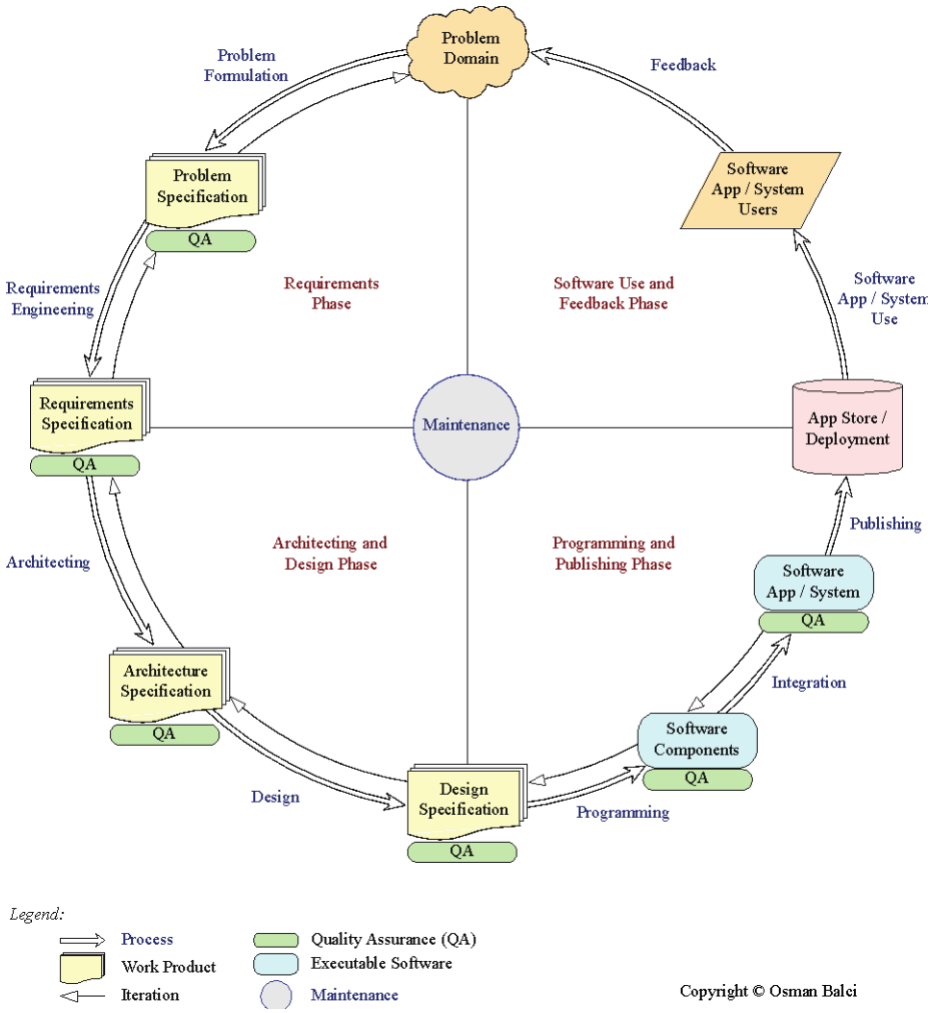
List of Implemented New Cloud Software Features

1. Used Prime Faces 11 RC-2 version
2. Prime Faces 11: Carousel -to display recipes on the home page
3. Prime Faces 11: Galleria with Thumbnail - to display food facts on home page
4. Prime Faces 11: Simple & Advanced Cards- to display information on various pages
5. Prime Faces : SelectCheckboxMenu (Basic) - to display and select the list of ingredients from a dropdown menu
6. HTTPS implementation using Apache web server and let’s encrypt.
7. A network forum for users based on the concept of ‘table joins’.

TABLE OF CONTENTS

1. SOFTWARE LIFE CYCLE

A good software engineer develops software by following the software life cycle shown below.



A programmer (hacker or ad-hoc developer) develops software by looking at the problem and directly coding in an IDE. This approach is known as the *Build-and-Fix Approach*, which must never be used!

2. OUR OBJECTIVE

The objective of our team project is to **demonstrate how capable the team members are** in engineering a Jakarta/Java EE cloud software application to solve a complex problem. The cloud software application is created for the purpose of showing how *learned* and *new* complex functionalities and features the team members are capable of developing.

3. PROBLEM SPECIFICATION

3.1 What is the problem?

Our project addresses some of the major problems surrounding planning and preparing meals. One such problem is the problem of monotony when it comes to cooking. People are indecisive about what meals to cook, or unaware of the possible meals that can be prepared given what is in their pantry. This can lead to a lack of cooking variety as people stick to cooking a few meals they are comfortable with. To address this problem, our application recommends multiple possible meals to prepare given the ingredients one owns, or the type of food one is interested in.

Another problem is ignorance or disregard of health and nutrition information when it comes to cooking. People are unaware of the health/nutrition information for the food they prepare - often because nutrition information is too much of a pain to keep track of. To address this problem, our application provides clear and consolidated nutrition information for each recommended meal.

Another problem that our project addresses is the lack of clarity of where to shop before cooking a meal. People often want to cook a meal that is new to them, but they do not know where to shop for the proper ingredients. To address this problem, our application recommends nearby stores that sell the proper ingredients for a given meal.

3.2 Why is the problem important to solve?

We all “know” the importance of a balanced diet and we know about food nutrition of course, right? Yet, today four of the top ten leading causes of death in the United States are directly linked to diet! In the fast-paced world that we are in, we have evolved and revolutionized every aspect of our lives, even to the point where we made food -“fast”. According to a recent report from the National Center for Health Statistics, every day, more than one out of three adults in the United States eat some type of fast food.

Fast-food consumption has been associated with excess intake of calories, fat and sodium which can lead to numerous health issues. The problem is how do we know the nutritional value of everything that we consume? Not everyone reads the “Nutrition Facts” on our tasty unhealthy fast food packages.

A major target audience for us through this project would be ‘indecisive’ people who don’t know what to eat for a meal, or who want to search delicious recipes online. Eighty-five percent of consumers don't know what they're having for dinner until hours before mealtime, according to a

survey from Acosta and Technomic reported by Food Navigator. This leads to choosing convenient meals from restaurants, meal kits or grocery deli.

3.3 Expected Functionality Description

Our cloud software application shall include the following features:

- Allow the user to search recipes based on desired characteristics and add recipes to saved recipe list
- Allow the user to save a recipe to be viewed again later
- Allow the user to view the nutritional value of a selected recipe
- Allow the user to export recipes with nutritional value to pdf, json, csv, xls, and xml formats from a Recipe List page.
- Allow the user to search for recipes using a "virtual pantry" which stores ingredients that the user owns.
- Allow the user to find nearby grocery stores
- Allow the user to upload meal pictures and descriptions to the drive
- Allow the user to share uploaded meals from the drive to a public page where different users can see each other's uploads
- Allow the user to sign up / sign in

3.4 Learned and new cloud software features to be implemented

Our proposed app shall implement the following *learned* and *new* cloud software features.

Table 1. List of Learned Cloud Software Features to be Implemented

No.	Description of the Learned Cloud Software Feature to be Implemented
1	Create, Read, Update, Delete (CRUD) operations in a MySQL database.
2	Template-based User Interface (UI) with header and footer implementation.
3	Use of JPA to interact with MySQL.
4	Obtain JSON from an API and process JSON data.
5	Use PrimeFaces content flow to display images on the homepage.
6	User login flow, reset password flow, and sign up flow.
7	Create cloud storage for the user to upload/download files.
8	The use of Datatable, Datagrid and Datalist for displaying data.
9	Responsive Dialogues with Scrollable contents
10	Textbox Scrollable Vertically
11	Database search for geocodings

12	Custom CSS

Table 2. List of New Cloud Software Features to be Implemented

No.	*Description of the New Cloud Software Feature to be Implemented*
1	Used Prime Faces 11 RC-2 version
2	Prime Faces 11: Carousel -to display recipes on the home page
3	Prime Faces 11: Galleria with Thumbnail - to display food facts on home page
4	Prime Faces 11: Simple & Advanced Cards- to display information on various pages
5	Prime Faces: SelectCheckboxMenu (Basic) - to display and select the list of ingredients from a dropdown menu
6	A shared meals page based on the concept of ‘table joins’.
7	Dynamic checking of user ownership of publicly shared files: The user can remove an item from the shared meals page if and only if that item was originally shared by the currently logged-in user.

4. REQUIREMENTS SPECIFICATION

This section specifies the Functional and Non-Functional Requirements under which our Jakarta/Java EE cloud software application will be developed.

4.1 Functional Requirements

Jack

1. The application shall allow the user to upload meal images, where an associated meal name and description are required.
2. The application shall allow the user to view all their uploaded meals such that each meal has a share and delete button attached to it.
3. The application shall allow the user to delete a meal they uploaded by clicking the attached delete button.
4. The application shall allow the user to share a meal to a public page of shared meals where the shared meals of all users can be viewed by clicking the attached share button
5. The application shall not allow the user to upload meals or view, delete, or share uploaded meals unless they are logged in.
6. The application shall allow the user to view publicly shared meals with or without being logged in.
7. The application shall allow the user to remove meals from the shared meals page if and only if that meal was originally shared by the currently logged-in user.

Neha

1. The application shall allow the user to sign up using the same credential fields and password complexity requirements as the CloudDrive tutorial.
2. The application shall allow the user to edit their profile, change their profile photo, change their password, and delete their account with the same functionality as the CloudDrive tutorial.
3. The application shall allow the user to log in using a username and password.
4. The application shall allow the user to add, remove, and edit ingredients in a virtual pantry.
5. The virtual pantry shall display ingredients, with associated quantity, unit of measurement, nutrition information, and calorie count in the form of a table.
6. When the user adds an ingredient to the virtual pantry, the application shall automatically append nutrition and calorie information to the ingredient's table entry by querying a nutrition API.
7. The application shall allow the user to search for a recipe by selecting ingredients from the virtual pantry.

Ishaan:

1. The application shall allow users to search stores based on Zip code and distance.
2. The user shall search for a recipe using ingredients or dish names.
3. The application shall allow the user to save a recipe from the search results list.
4. The application shall allow the user to download the list of saved recipes in CSV, PDF, JSON, XML, and XLS file formats.
5. The user shall be able to search the saved recipes.
6. The user shall be able to change the profile picture.

7. The user shall be able to delete their account.

4.2 Non-Functional Requirements

Each team shall satisfy the following non-functional requirements:

1. The UI template footer shall contain the following statement:

   CS5704 Software Engineering course semester project application developed by studentName1, studentName2, and studentName3.

   Course title and each student name above shall be hyperlinked to show the corresponding homepage *in a new window / tab*. Use VT website if you do not have one for yourself.

2. The cloud software application name in IntelliJ IDEA and when deployed on the server shall be a meaningful name without course number or team number. The app name shall reflect what the app does such as MeetingScheduler. Do **not** name it as "group", "Team4App", or "CS5704Team4Project".

Each team's non-functional requirements:
Jack:

1. The meal information of upload and shared meals shall be presented in uniformly sized cards where the contents are scrollable, and the meal images are all uniformly sized.
2. The page that displays shared meals shall be structurally the same as the uploaded meals page, except the name of the person who shared each meal at the top of the meal card, and the meal cards lack a share button.
3. When the user presses the delete button on an uploaded meal card or the remove button on a shared card, the page shall be refreshed immediately to show that the meal was removed.

Ishaan:

1. The recipe search results page should have both grid and list view.
2. The recipe search shall not allow the user to search for recipes with an empty string.
3. Stores search shall return an error for an invalid zip code.
4. The store search result page should bear the stores' results according to distance.
5. Store search shall only allow a five-digit zip code.

Neha:

1. The home page shall have a carousel.
2. The user shall not proceed for recipe search without selecting at least one ingredient from the pantry.
3. The website shall have a footer on every page.
4. All page icons shall be unique and represent the functionality of their respective pages.

5. ARCHITECTURE SPECIFICATION

Team Member 1 shall create the following 3 DoDAF Models
1. **OV-1**: High-Level Operational Concept Graphic 2. **OV-2**: Operational Resource Flow Description 3. **OV-5b**: Operational Activity Model
Team Member 2 shall create the following 3 DoDAF Models
1. **SV-1:** Systems Interface Description 2. **SV-2**: Systems Resource Flow Description 3. **SV-4**: Systems Functionality Description
Team Member 3 shall create the following 3 DoDAF Models
1. **SvcV-1**: Services Context Description 2. **SvcV-2**: Services Resource Flow Description 3. **SvcV-4**: Services Functionality Description

Notes:

- Study DoDAF Volume 2 to learn about each type of DoDAF model (diagram).
- See Examples of DoDAF models for a Campus Situational Awareness and Emergency Response Management System.
- **Microsoft Visio** (recommended) or a tool of your choice can be used to create the DoDAF diagrams.
- Each student is required to create 3 DoDAF models (diagrams) as indicated above. A team of 2 members shall create the models for Team Members 1 and 2 or 1 and 3 or 2 and 3.
- **Extra credit** will be given for additional DoDAF diagrams. Extra credit will be determined with respect to the complexity of each extra diagram.

5.1 OV-1: High-Level Operational Concept Graphic

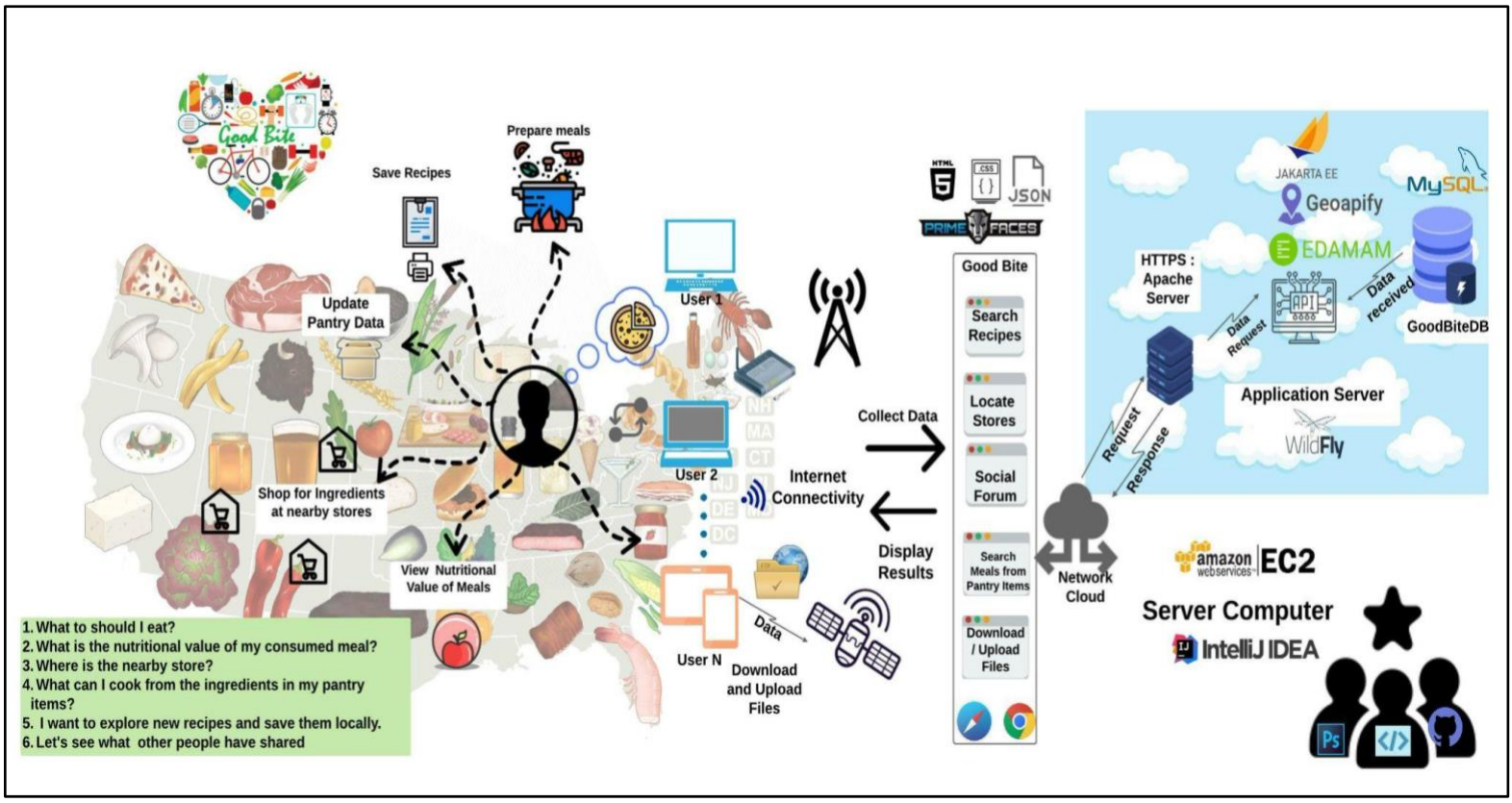


Fig. 1: High Level Operational Concept Graphic for GoodBite depicting our main operational concepts.

5.2 OV-2: Operational Resource Flow Description

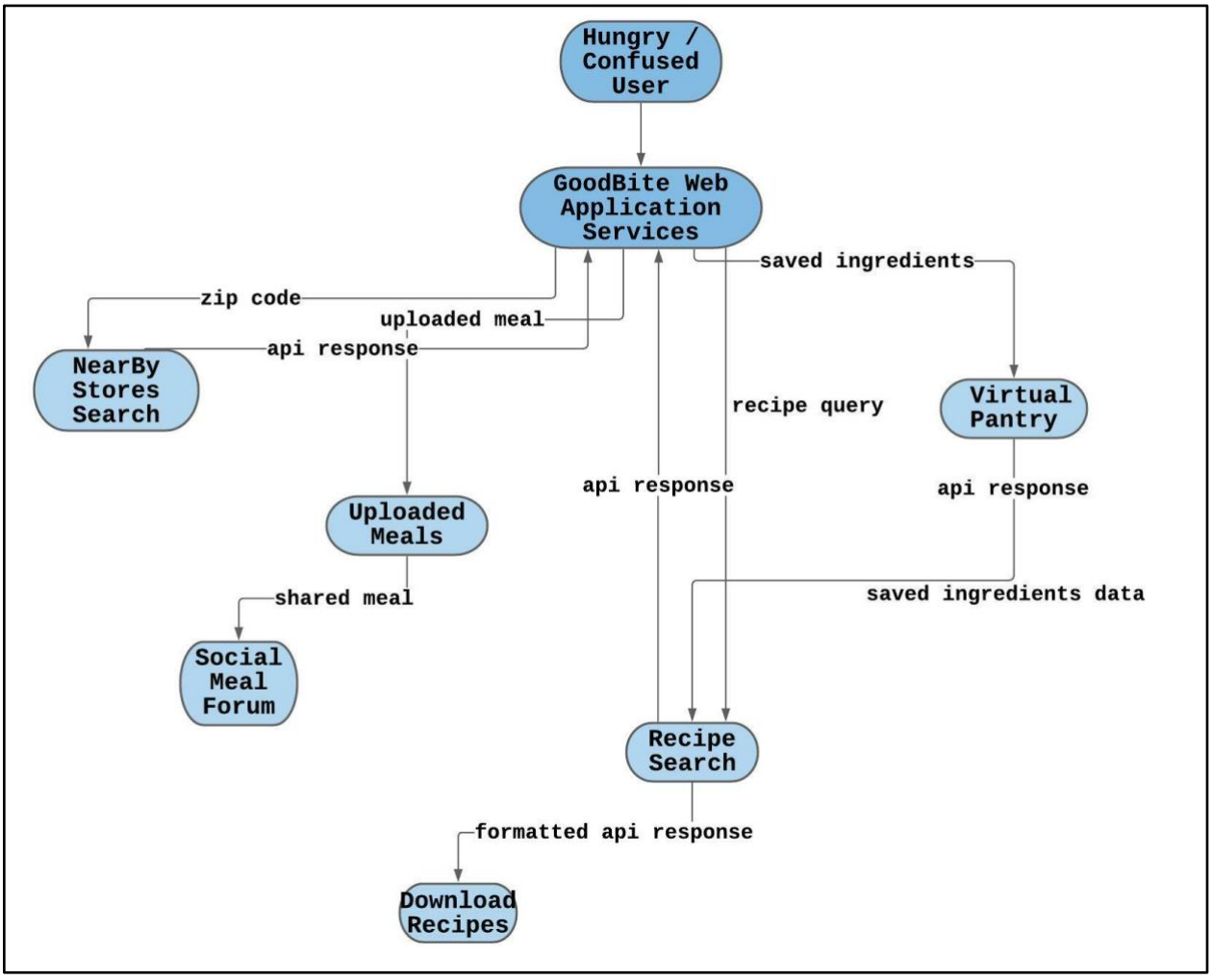


Fig. 2: Operational Resource Flow Description for GoodBite depicting the operational capability for our system users.

5.3 OV-5b: Operational Activity Model

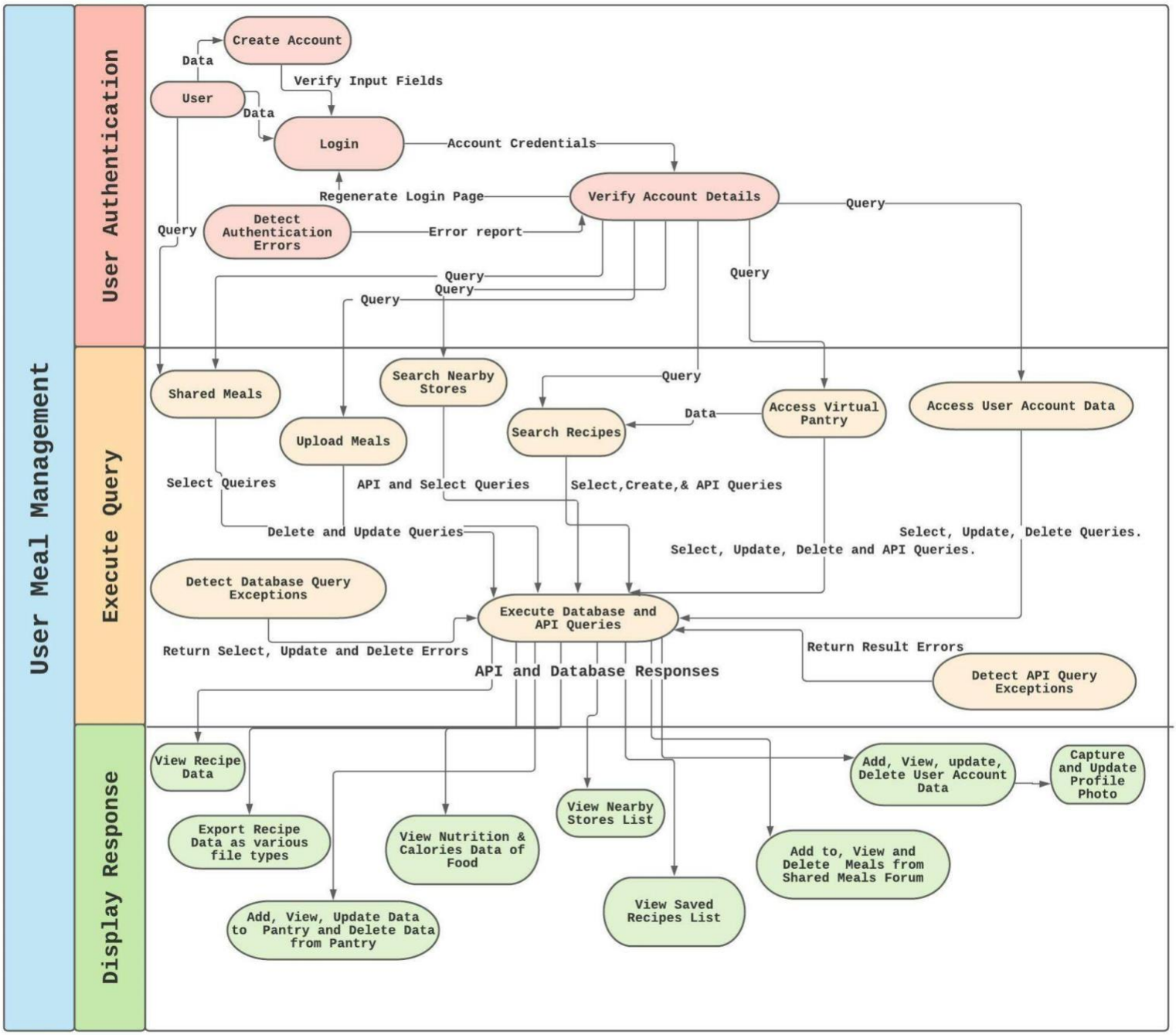


Fig. 3: Operational Activity Model for GoodBite depicting the operational activities for our system.

5.4 SV-1: Systems Interface Description

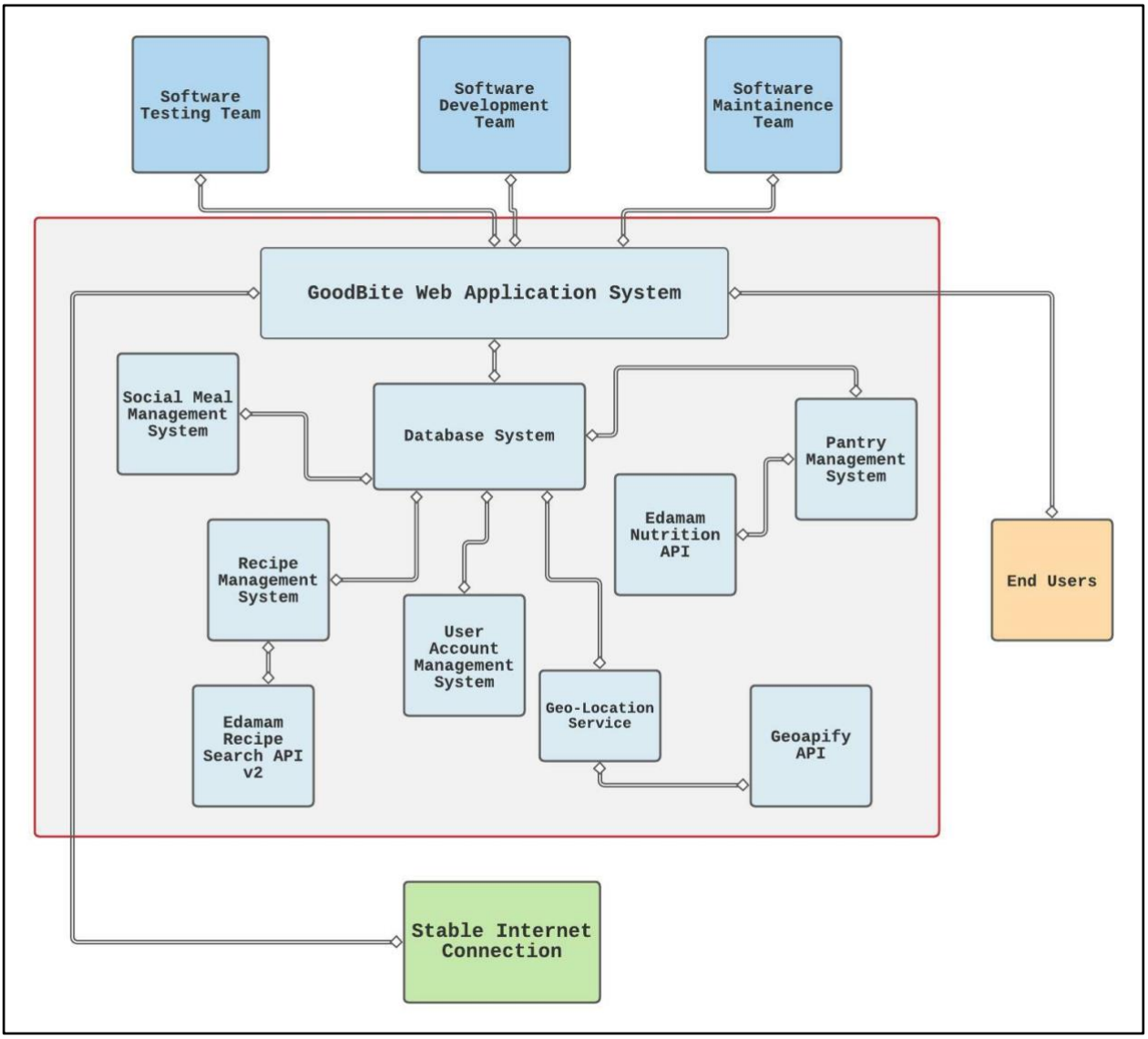


Fig. 4: Systems Interface Description for GoodBite depicting the composition and interactions of our system.

5.5 SV-2: Systems Resource Flow Description

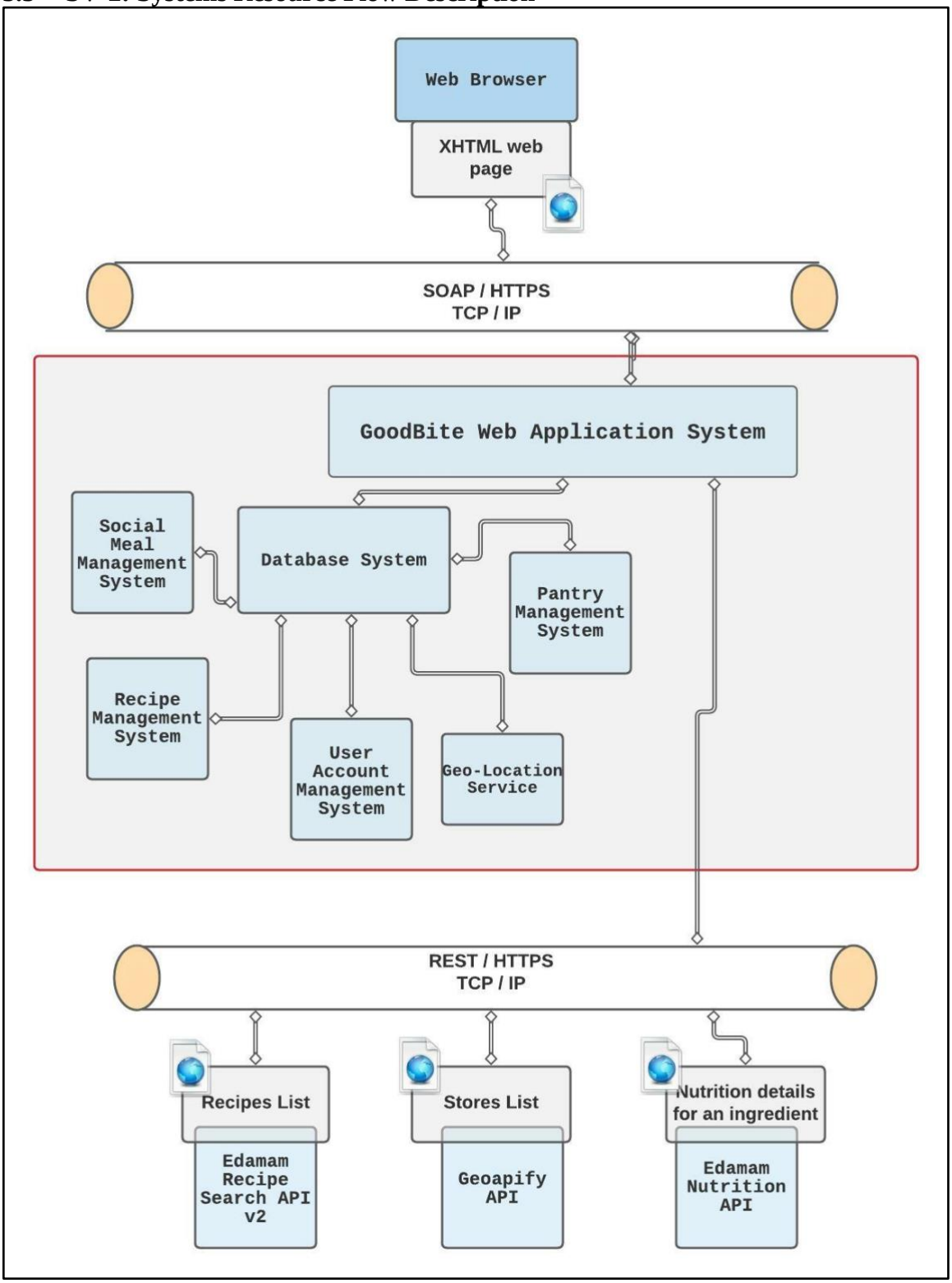


Fig. 5: Systems Resource Flow Description for GoodBite depicting the resource flow of our system.

5.6 SV-4: Systems Functionality Description

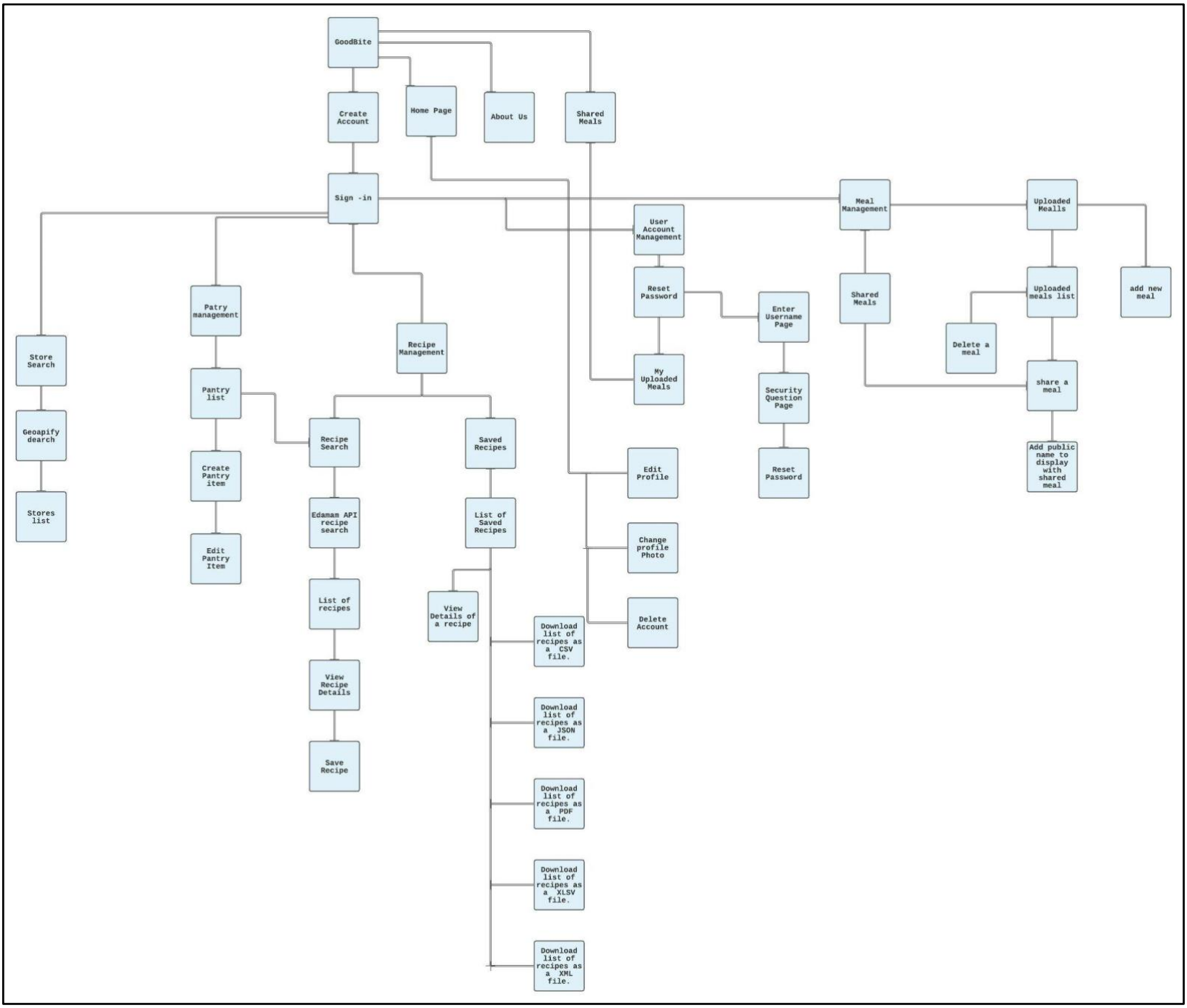


Fig. 6: Systems Functionality Description for GoodBite depicting the functional requirements of our system.

5.7 SvcV-1a: SOA Conceptual Layers

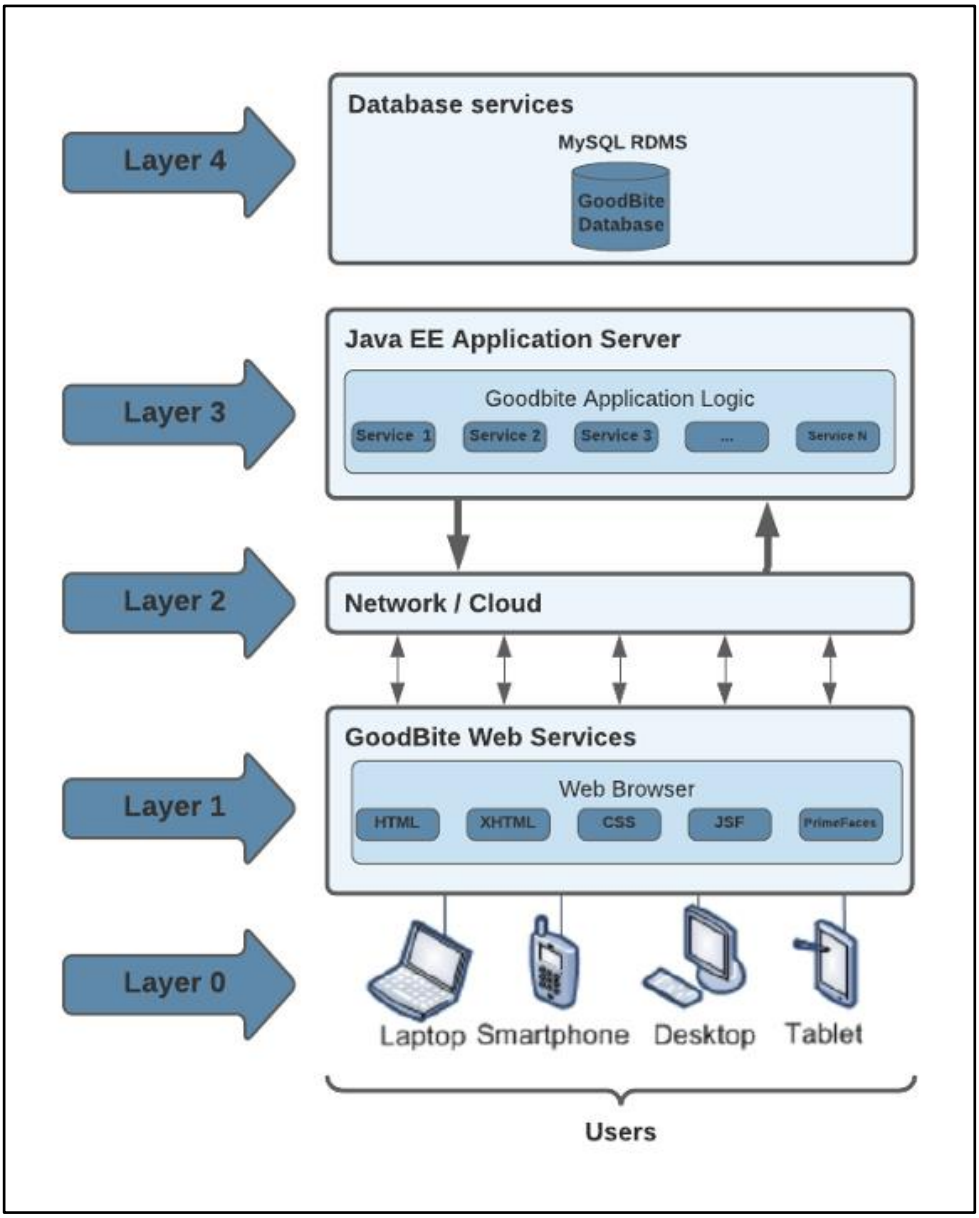


Fig. 7: SOA Conceptual Layers for GoodBite depicting the composition and interaction of services of our system

5.8 SvcV-1c: Services Context Description

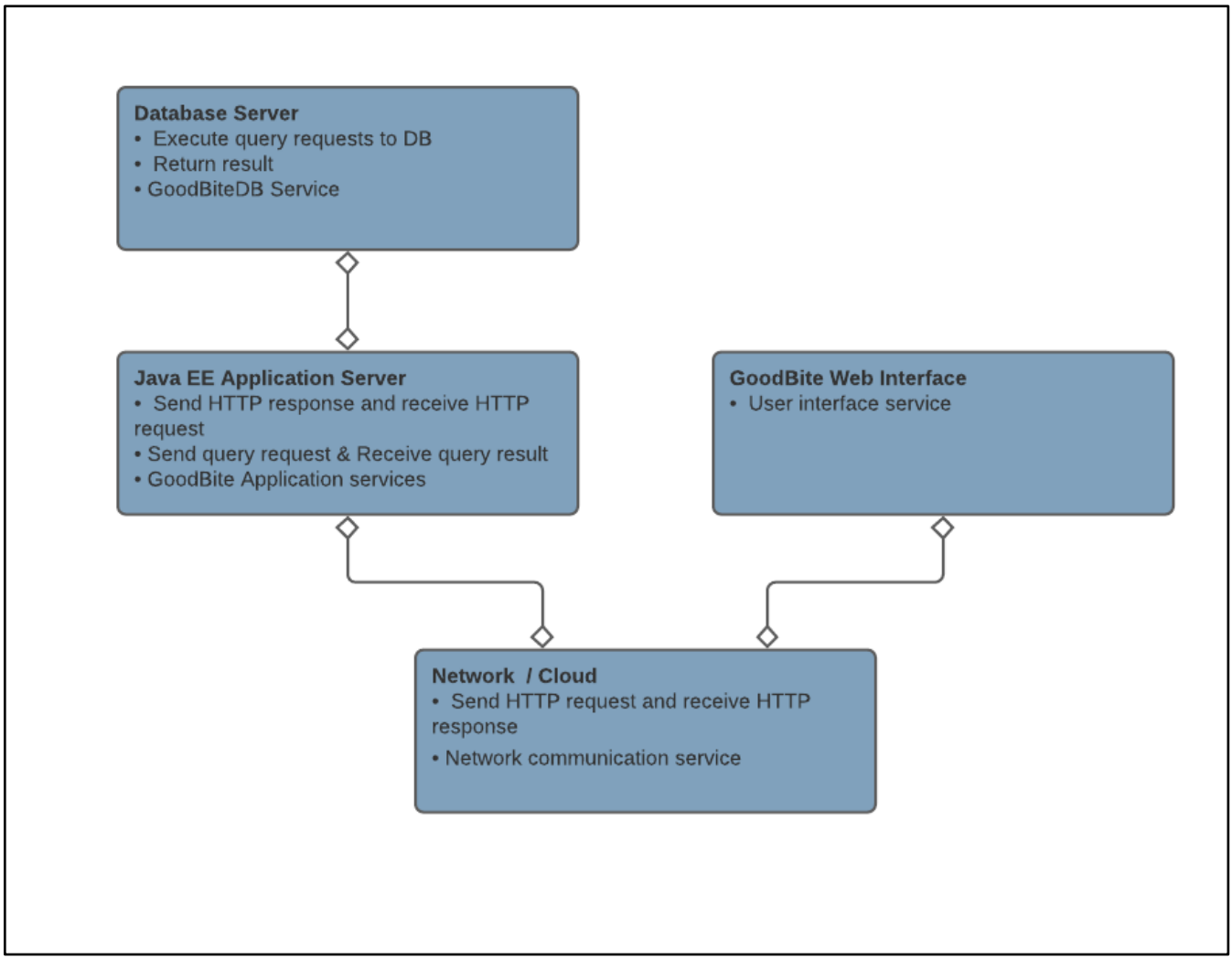


Fig. 8: Services Context Description of GoodBite.

5.9 SvcV-2: Services Resource Flow Description

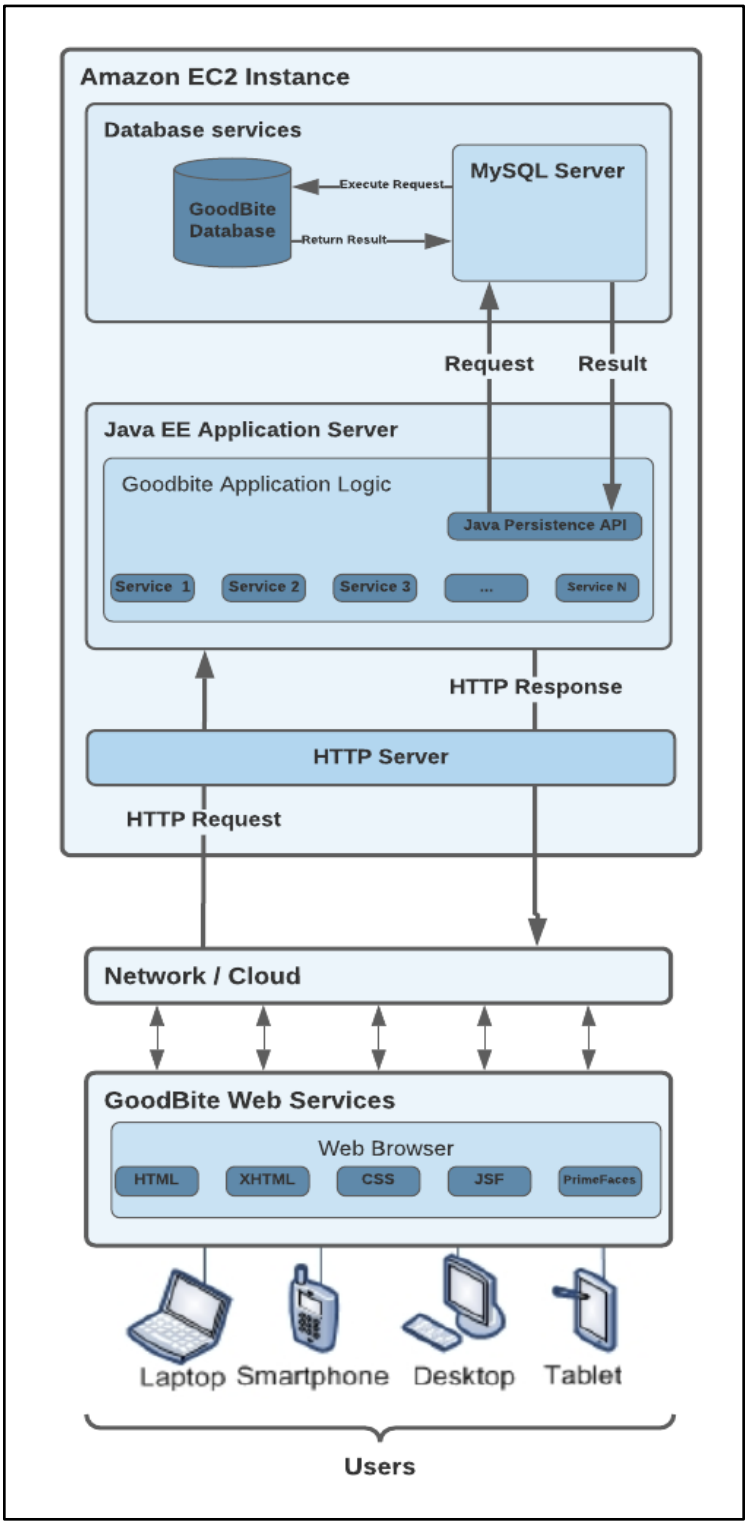


Fig. 9: Services Resource Flow Description of GoodBite.

5.10 SvcV-4: Services Functionality Description

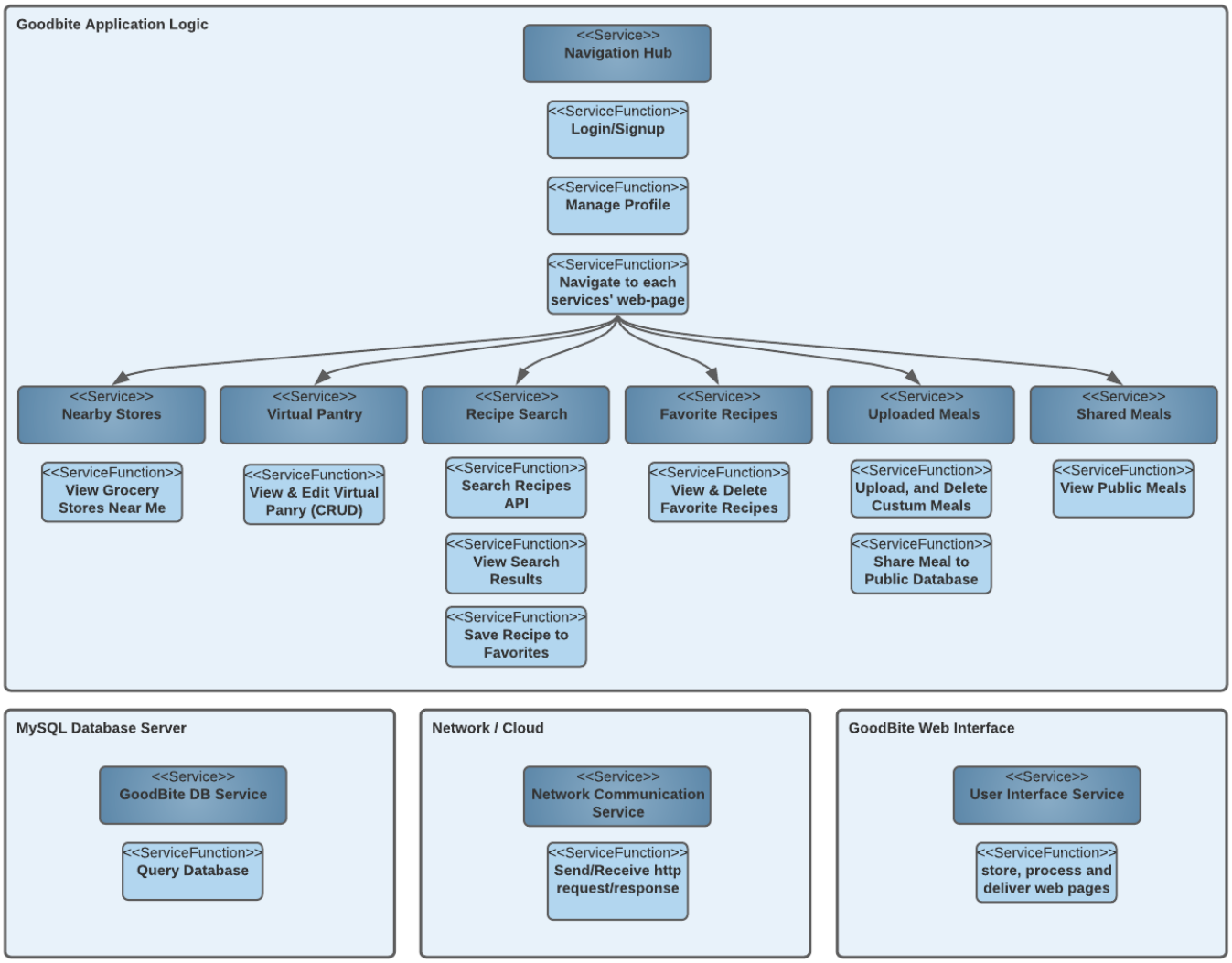
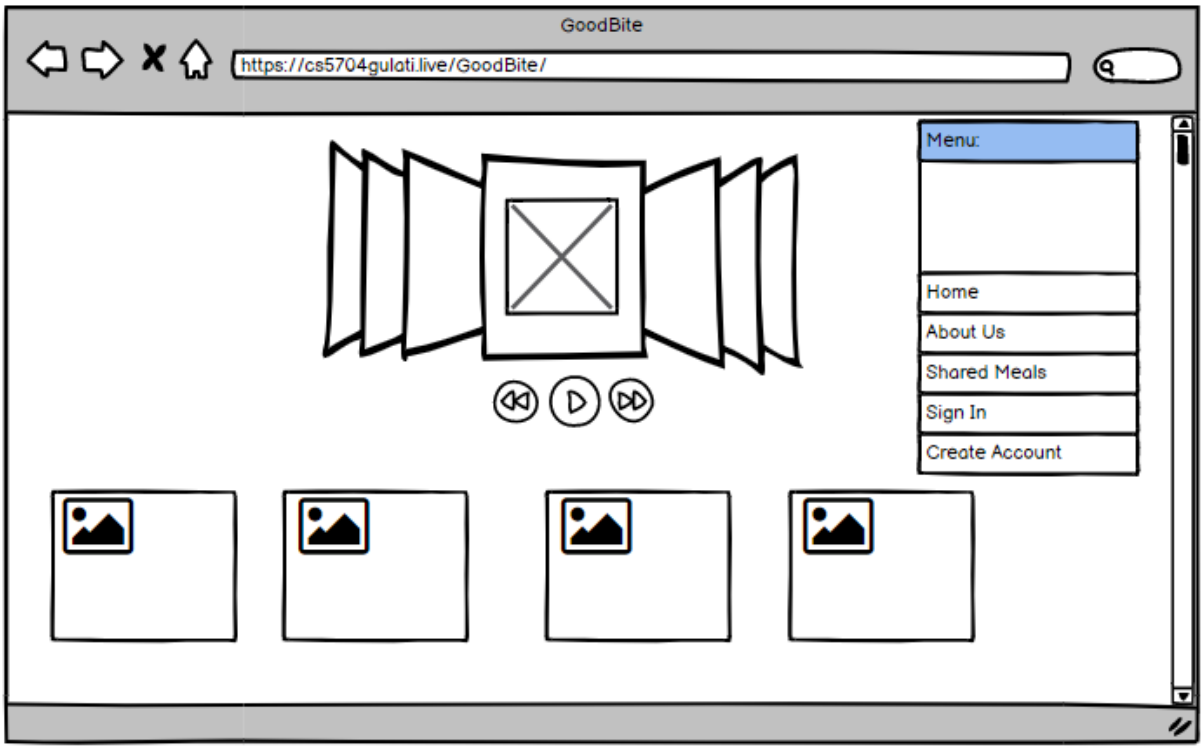


Fig. 10: Services Functionality Description of GoodBite depicting the human and service functionality of our system.

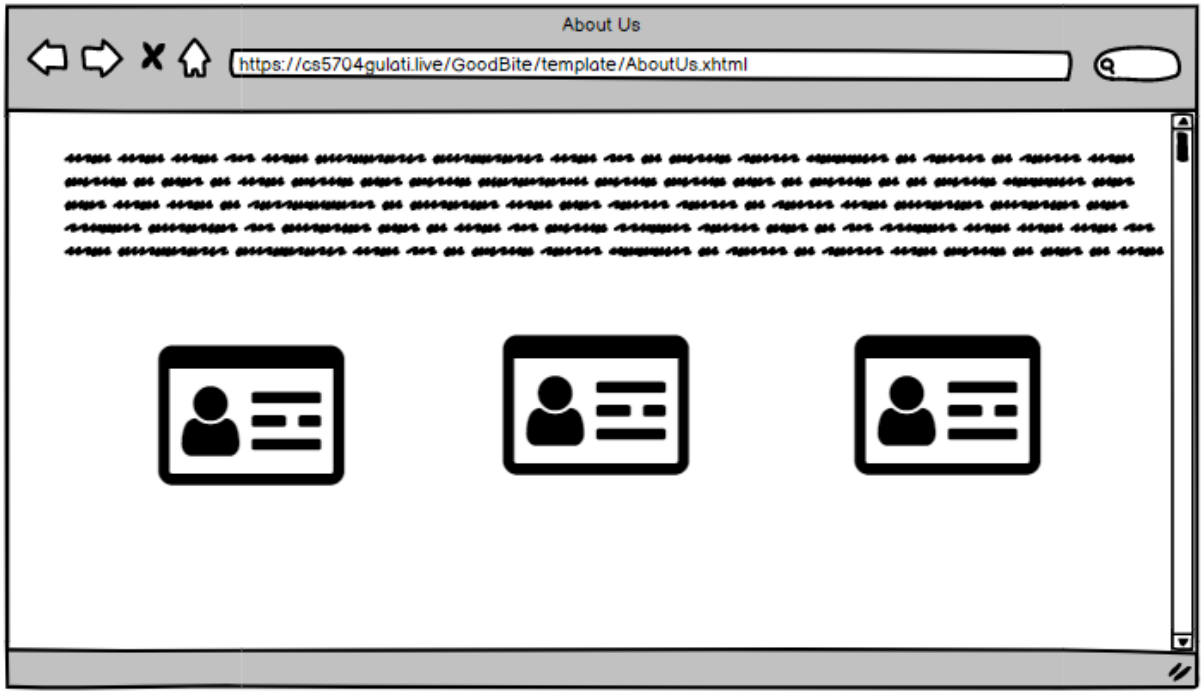
6. DESIGN SPECIFICATION

The below series of Images represent our system's wireframe designs created on Balsamiq's website.

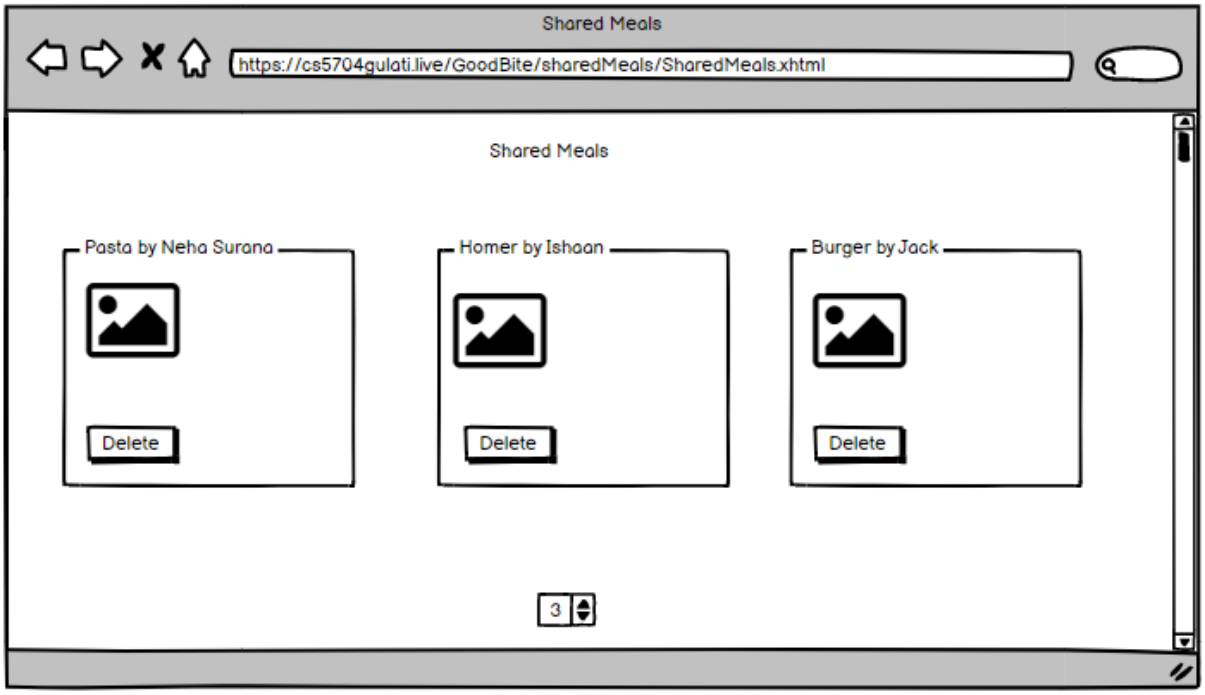
Home Page:



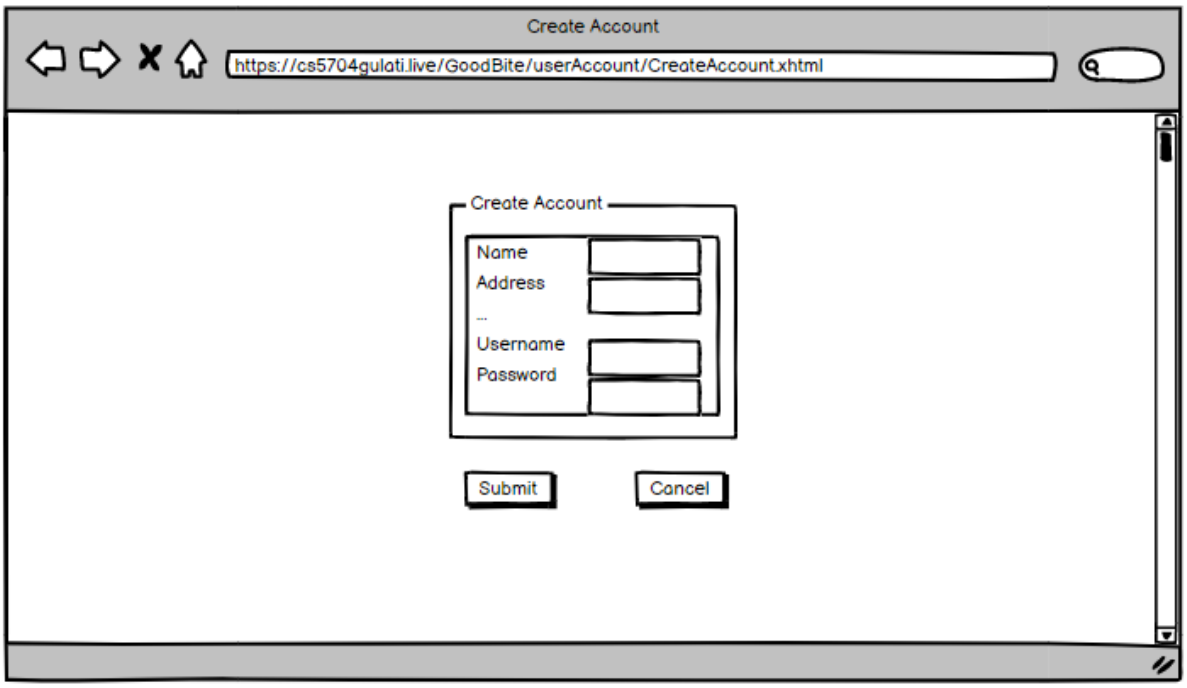
About Us Page:



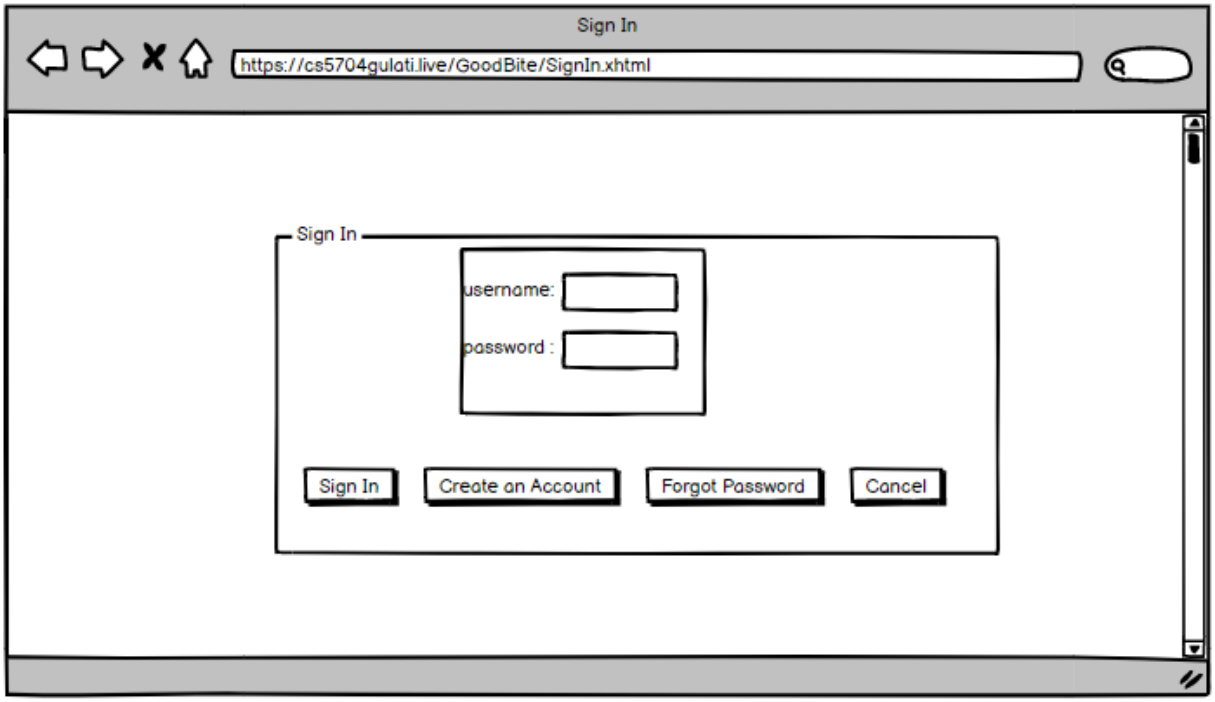
Shared Meals Page:



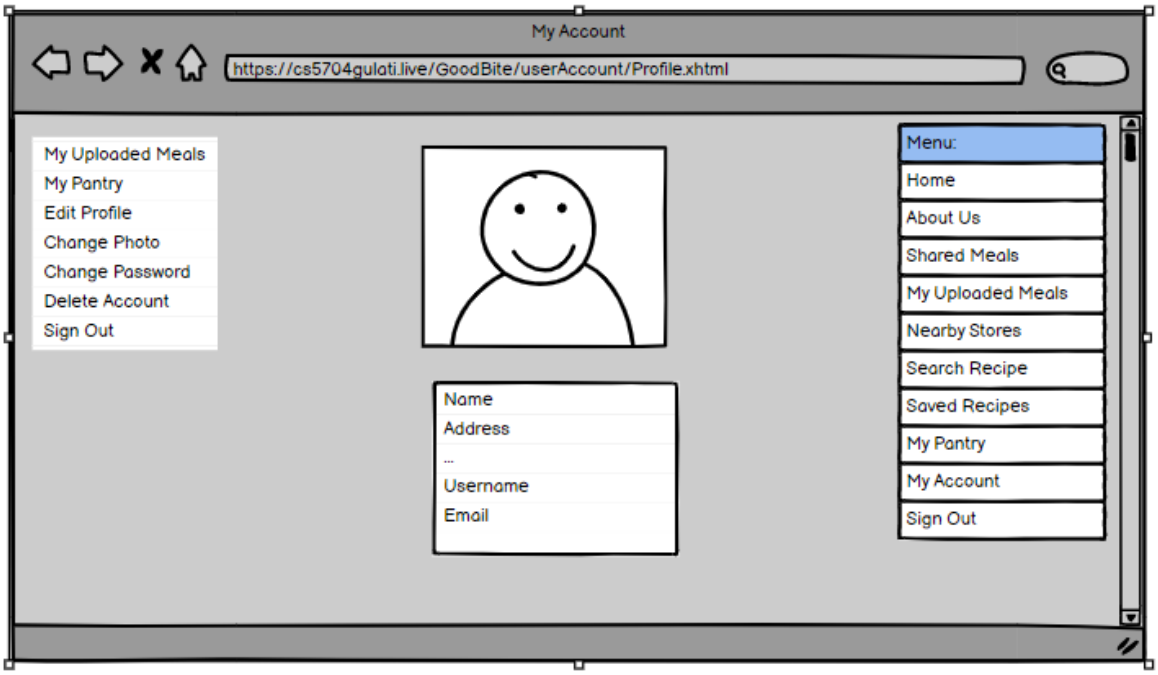
Create Account Page:



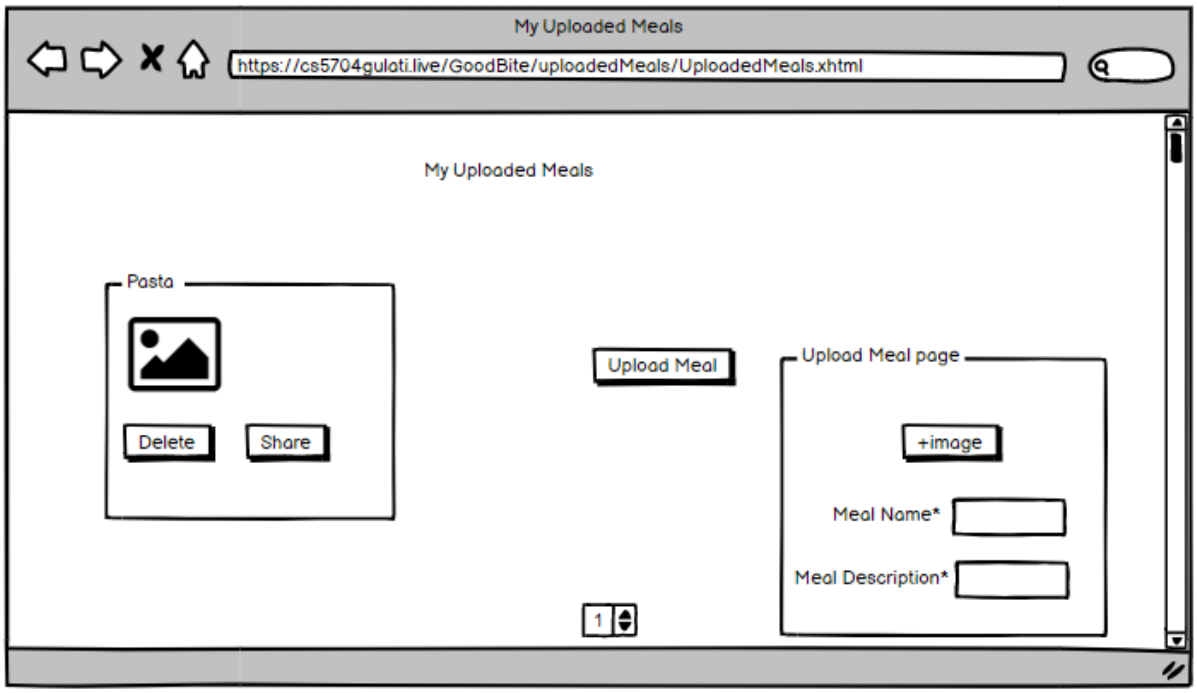
Sign In Page:



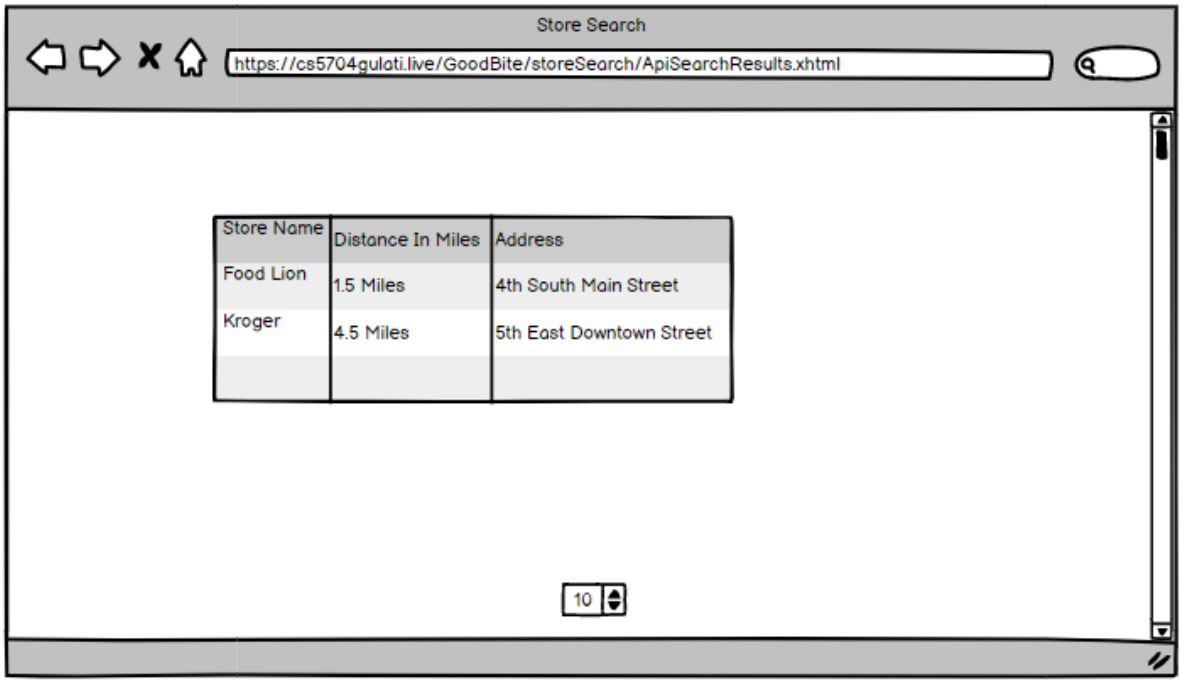
User Account Page:



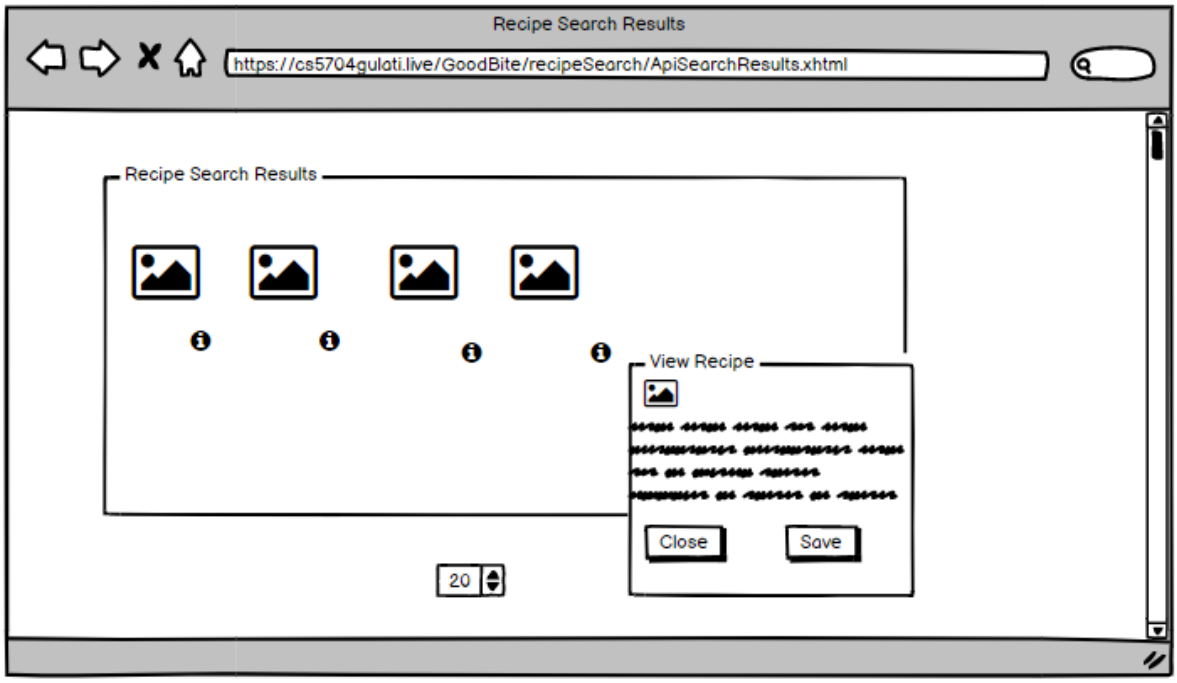
My Uploaded Meals Page:



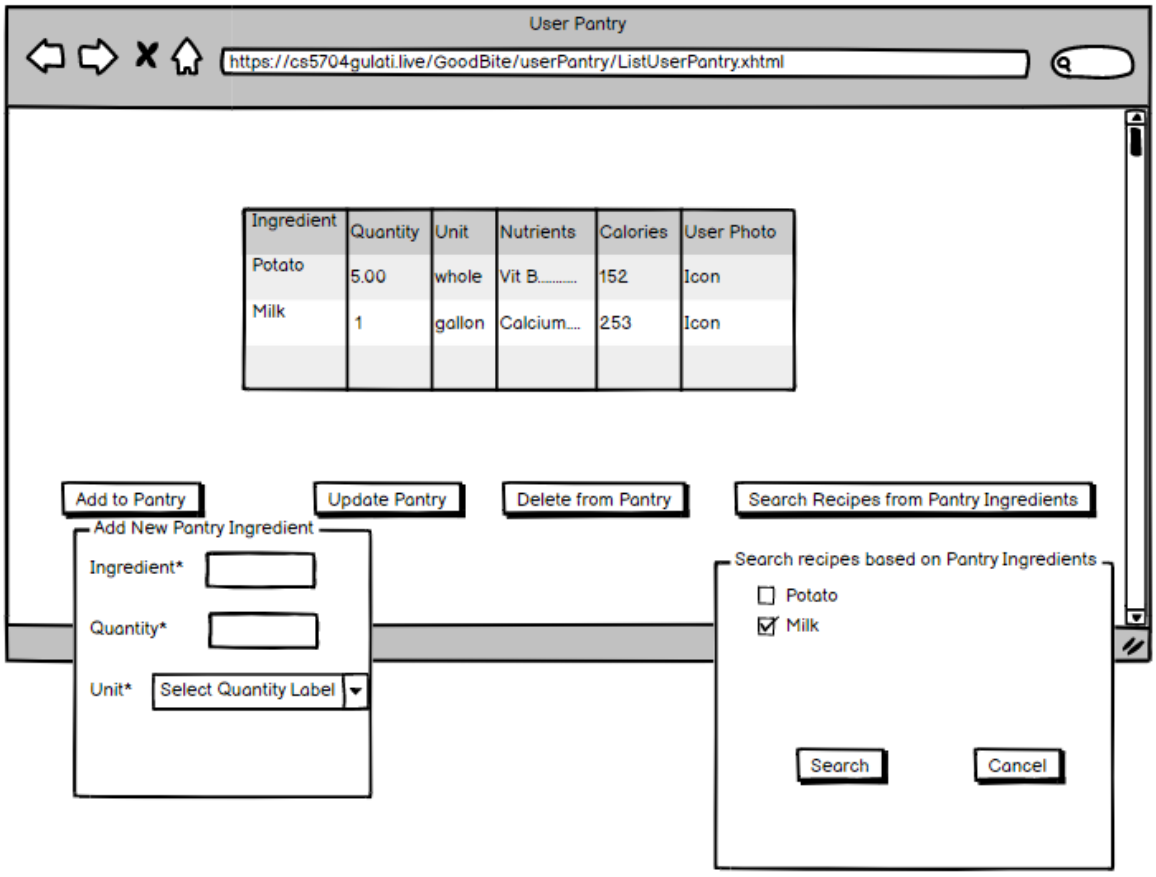
Nearby Store Page:



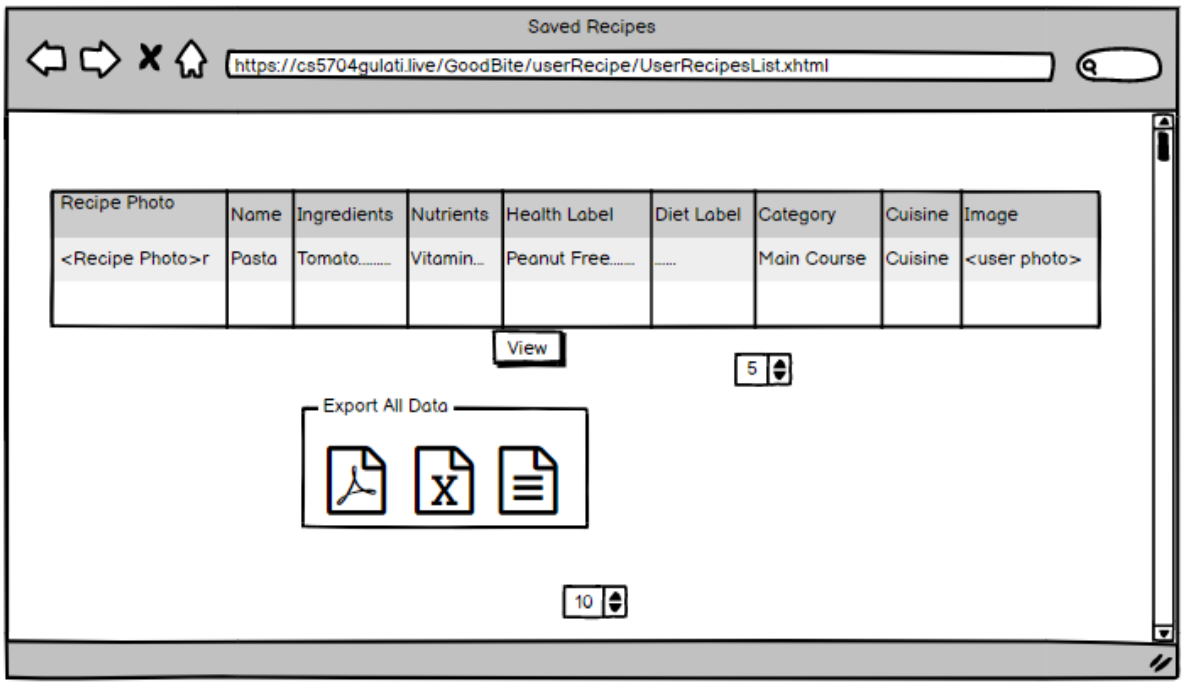
Recipe Search Results:



List of Pantry Items:



Saved Recipes Page:



Relational Database Design

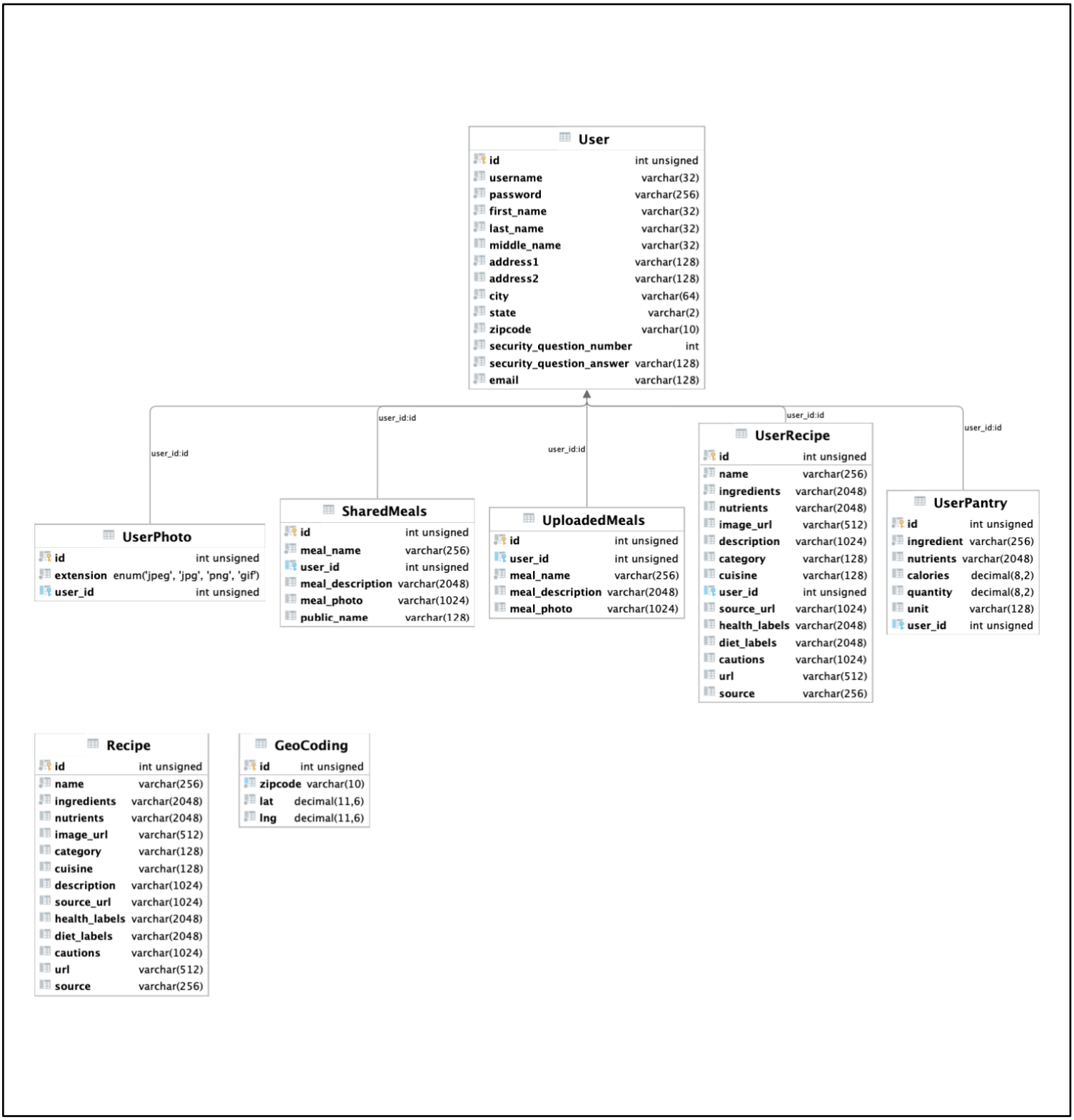


Fig. 11: The above image represents GoodBite's relational database design.

7. DELIVERED SOFTWARE FUNCTIONALITY

Here is a detailed walk-through of our software functionality. We start at the home page of our application website:

https://cs5704gulati.live/GoodBite/

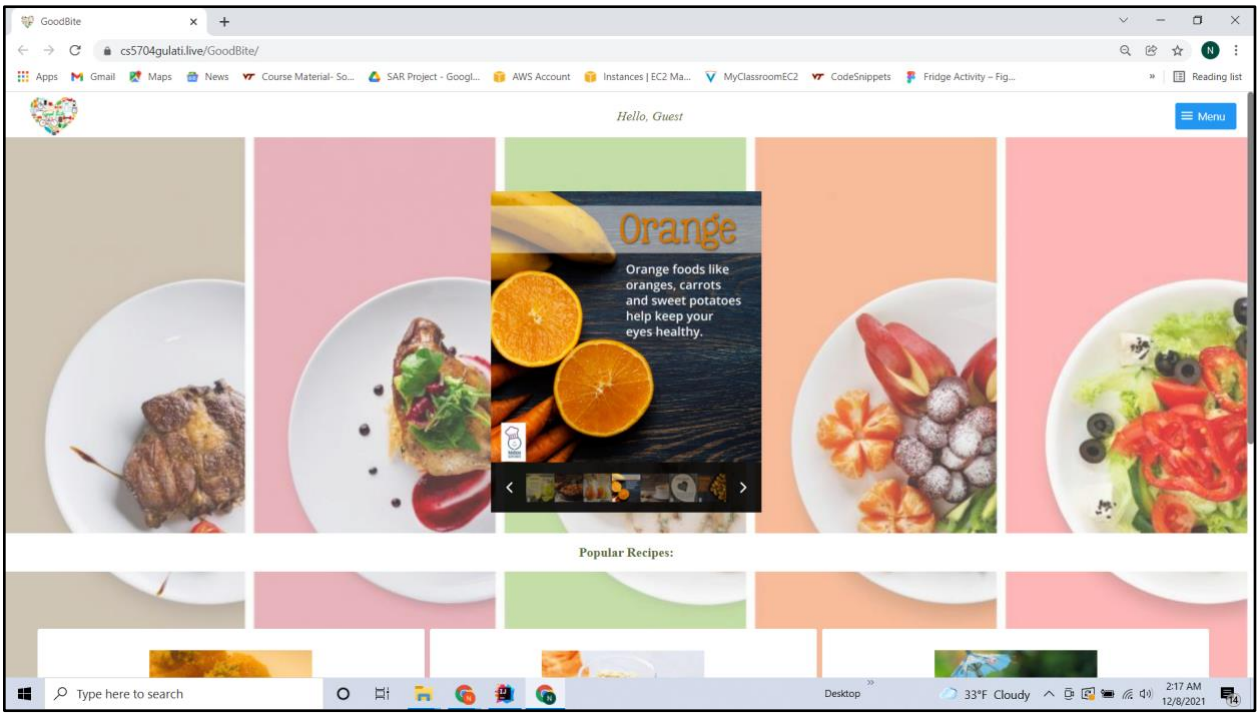


Fig. 12

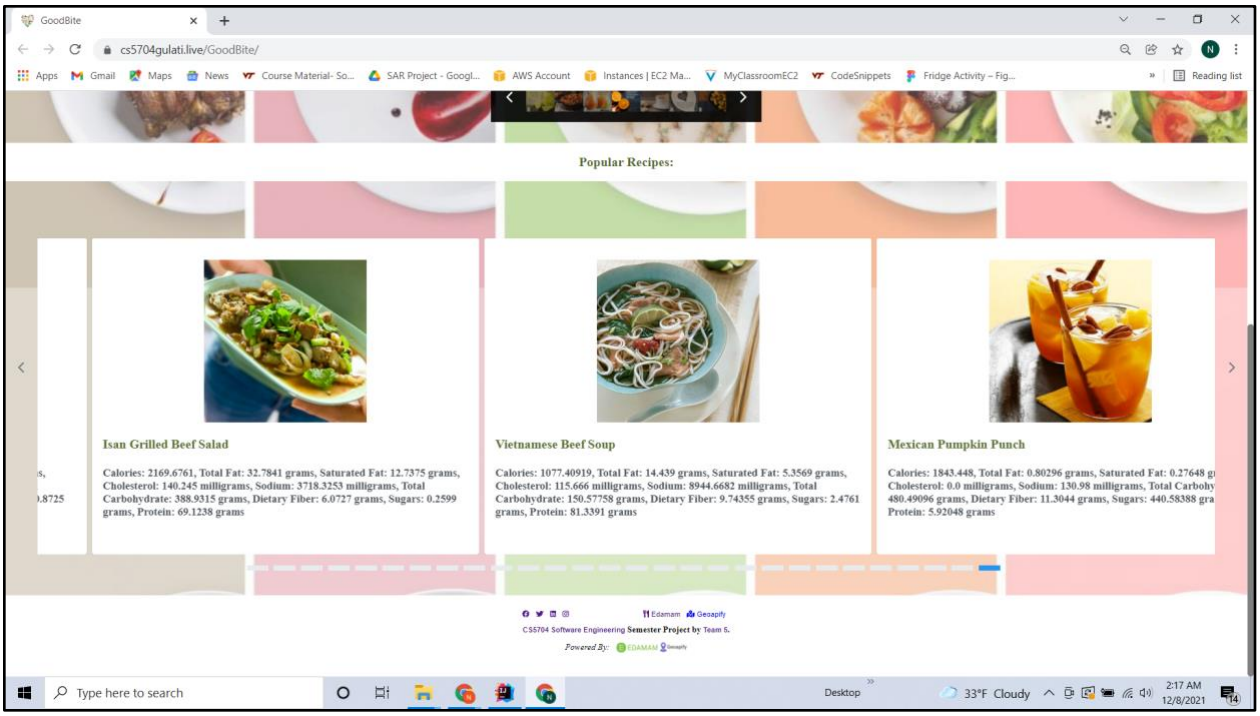


Fig .13: The home page (**Fig. 12&13**) displays a galleria of nutrition fact images and a carousel of meals from the recipes database with associated nutrition information.

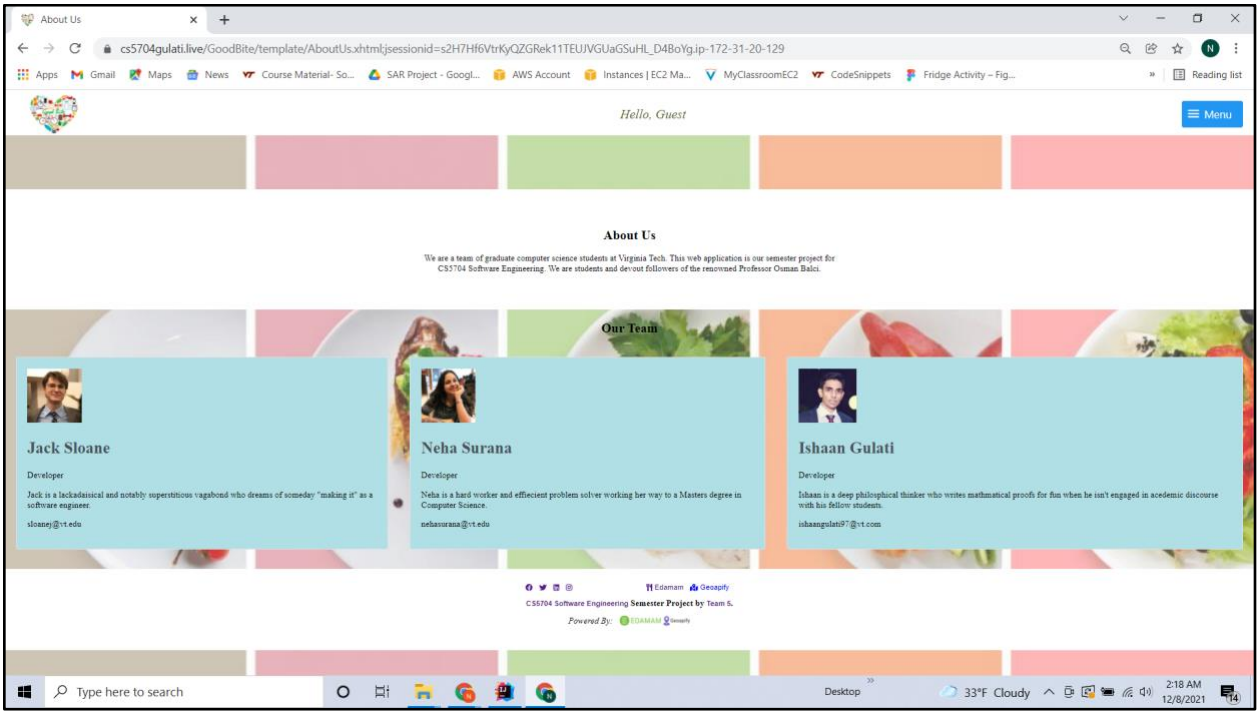


Fig. 14: The About Us page shows a description of the team members of the project.

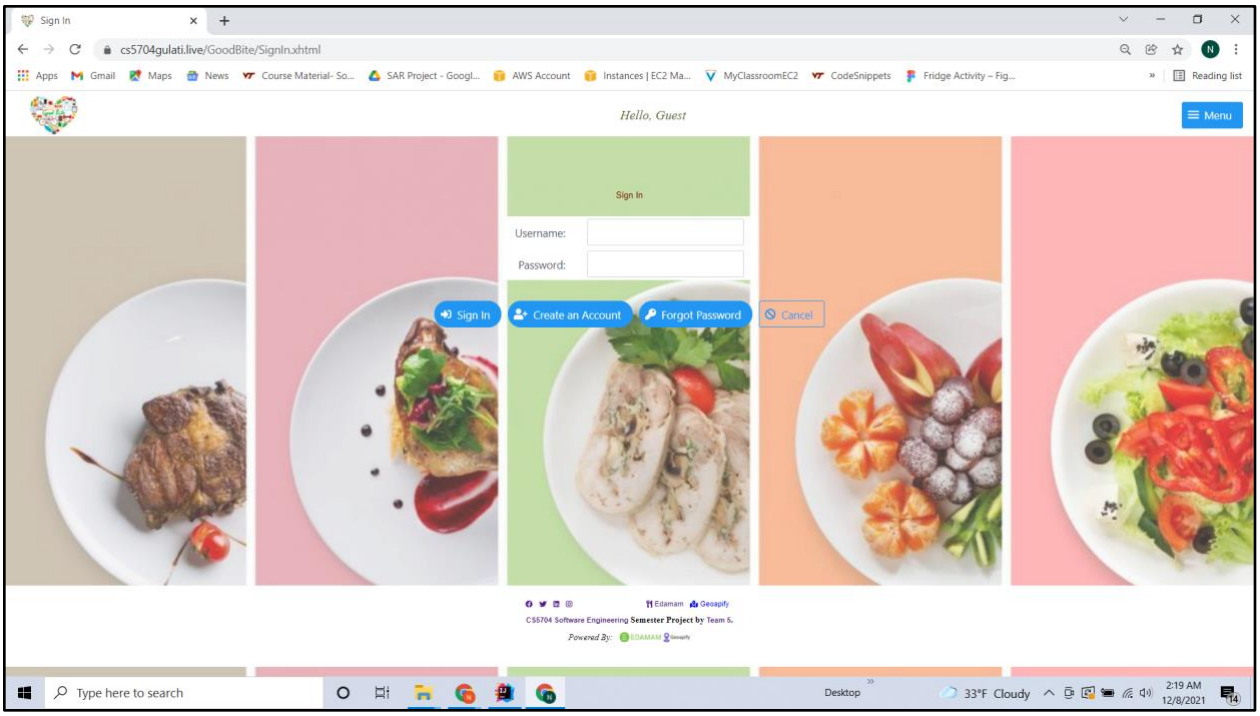


Fig 15: The sign in page allows the user to sign in using a username and password - mirroring the functionality of the CloudDrive tutorial.

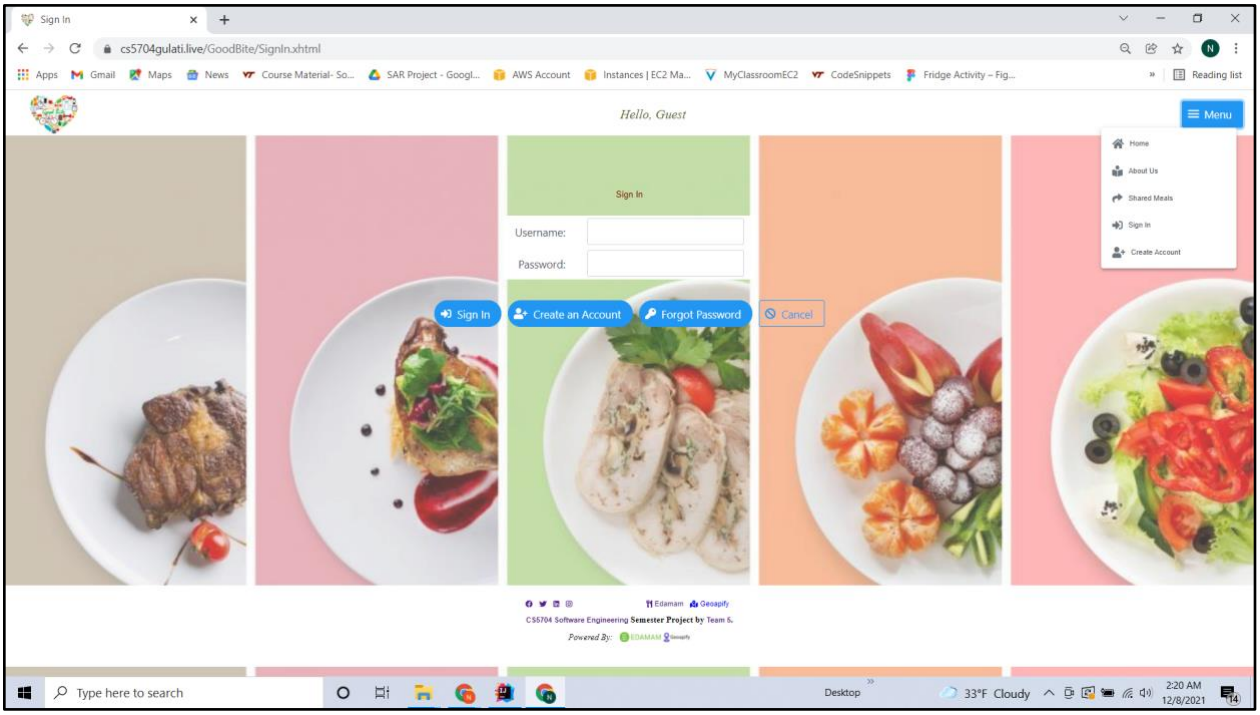


Fig. 16: The navigation dropdown menu in the header lists links to different pages. When not signed in, the dropdown menu lists the Home page, the About Us page, the Shared Meals page, the Sign In page, and the Create Account page.

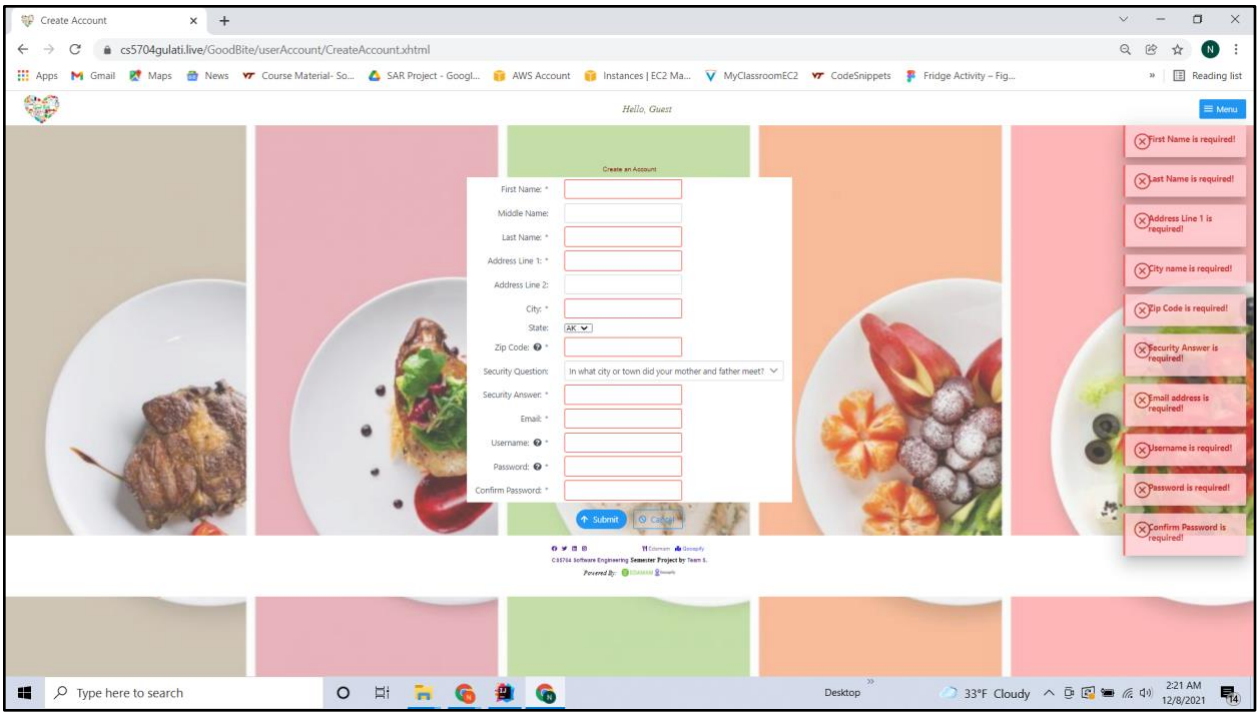


Fig. 17: The create account screen mirrors the functionality of the CloudDrive tutorial's account creation.

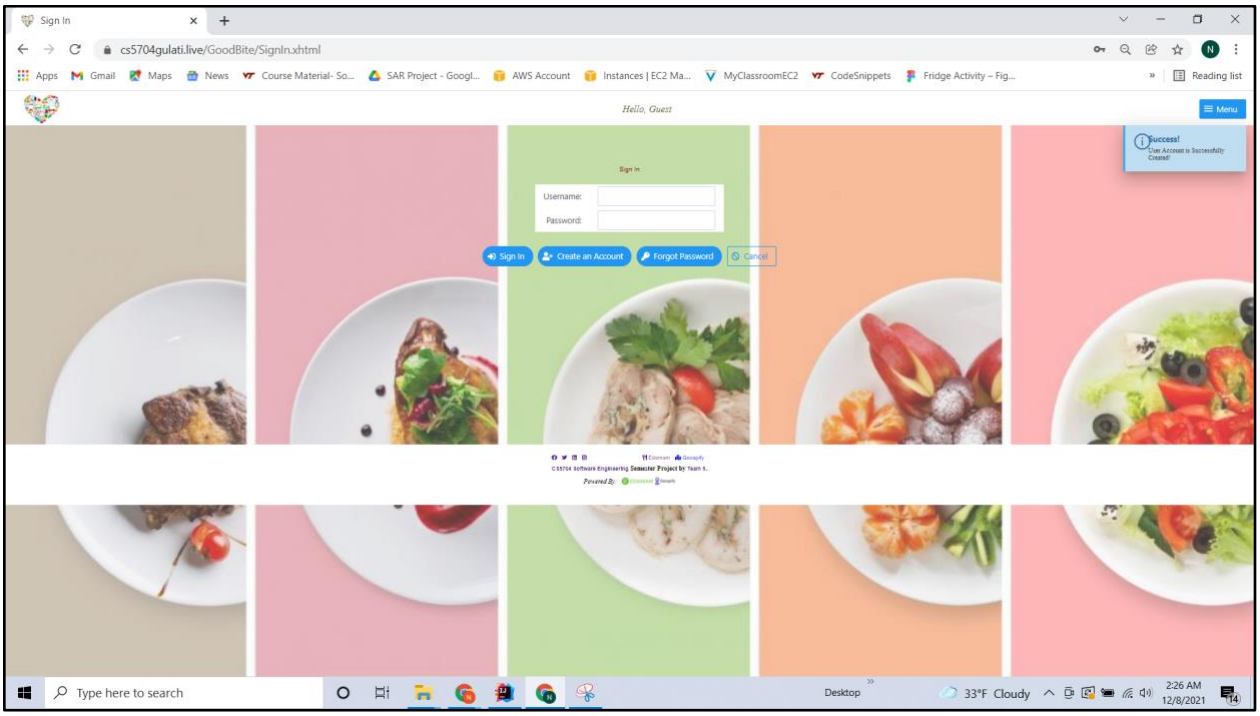


Fig. 18: When an account is created, the user is redirected to the Sign-In page.

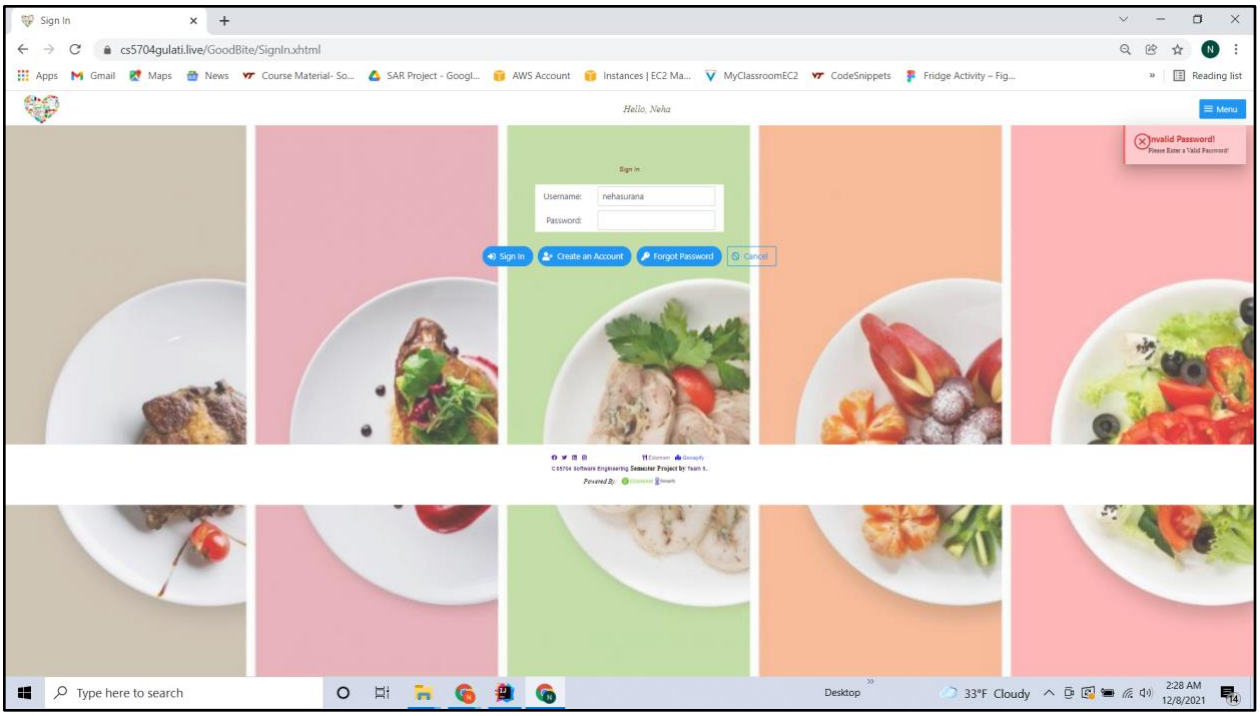


Fig. 19: If the user enters invalid credentials into the username and/or password field of the Sign-In page, an appropriate error message is displayed; this functionality mirrors that of the CloudDrive tutorial.

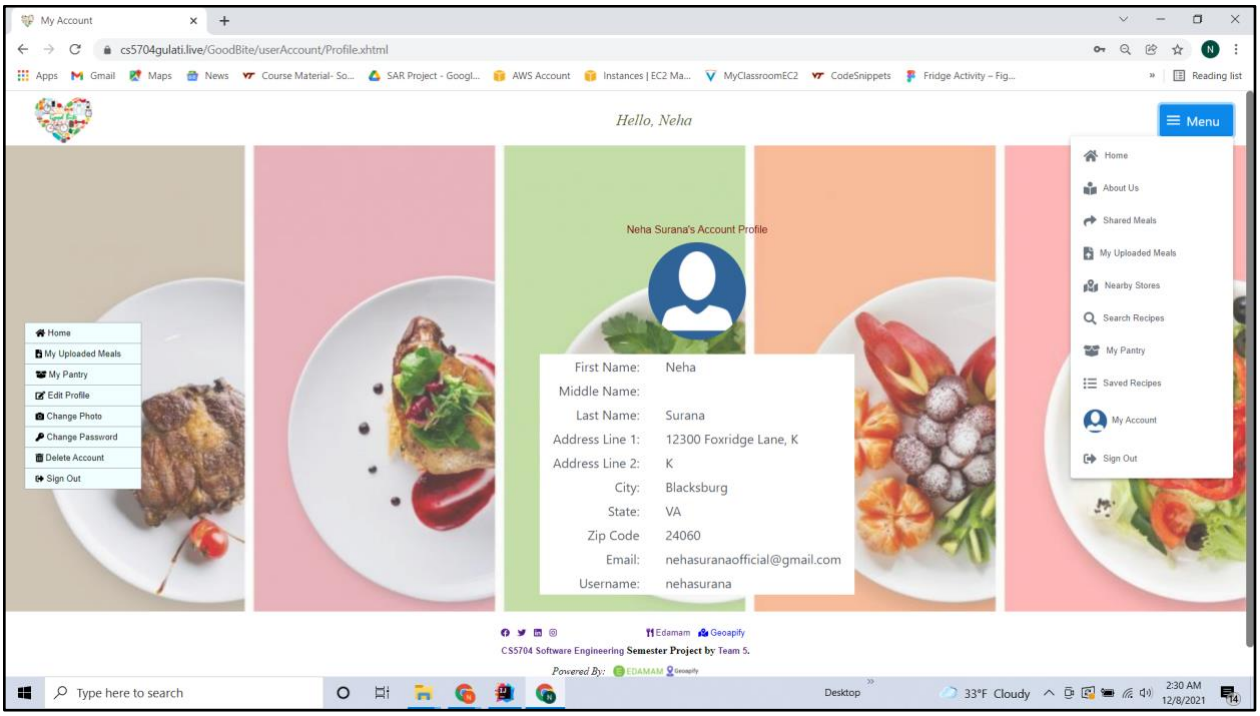


Fig. 20: The Profile page allows the signed in user to view and edit their account information, mirroring the functionality of the profile page in the CloudDrive tutorial. Additionally, the user can navigate to the My Uploaded Meals page from the Profile page. When the user is signed in, the navigation dropdown menu in the header is changed, allowing the user to access more pages

and dialogs. Now, the user can additionally access the My Uploaded Meals page, the Nearby Store search dialog, the Search Recipe dialog, the My Pantry page, the Saved Recipe page, and the Sign Out button.

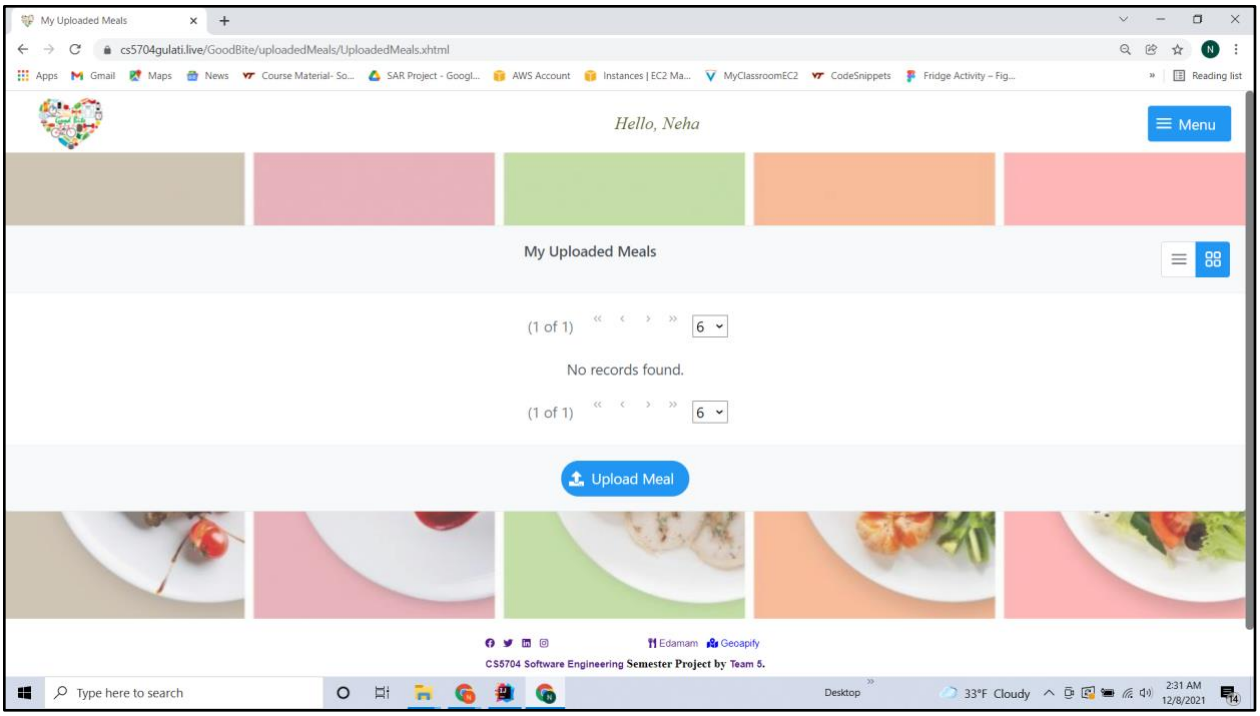


Fig. 21: The Uploaded Meals page is where the user can upload pictures and descriptions of meals they made. The user presses the Upload Meal button and is redirected to the Upload Meal page.

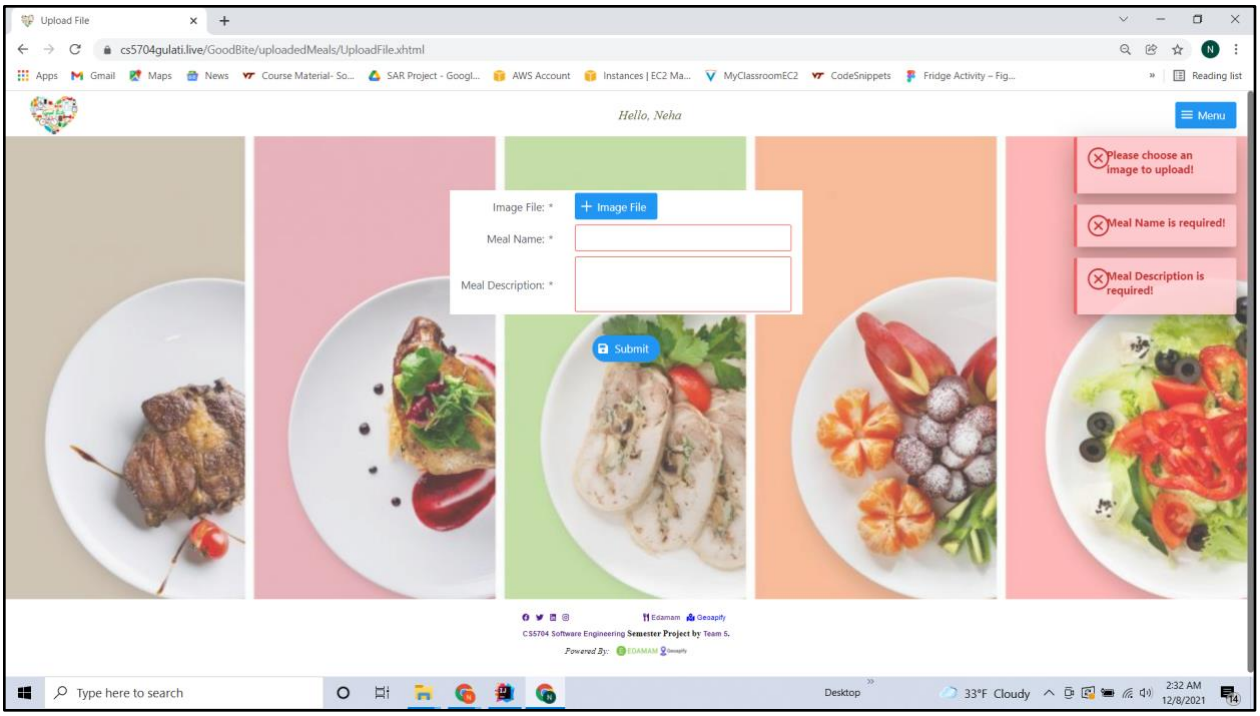


Fig. 22: In the Upload Meal page, the user selects an image file to select from their local files and enters a meal name and description. All three fields are required, and only image files are accepted.

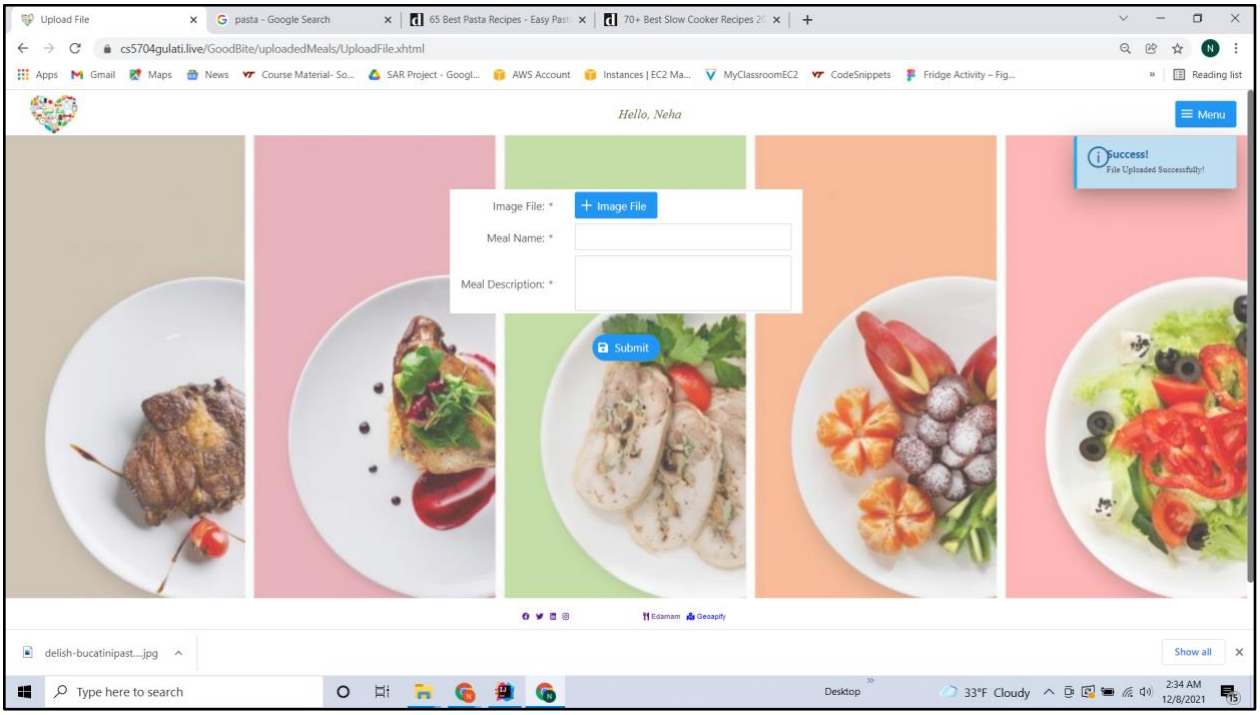


Fig. 23: If all the fields are satisfied and the user presses submit, the file is uploaded, and a success message is displayed.

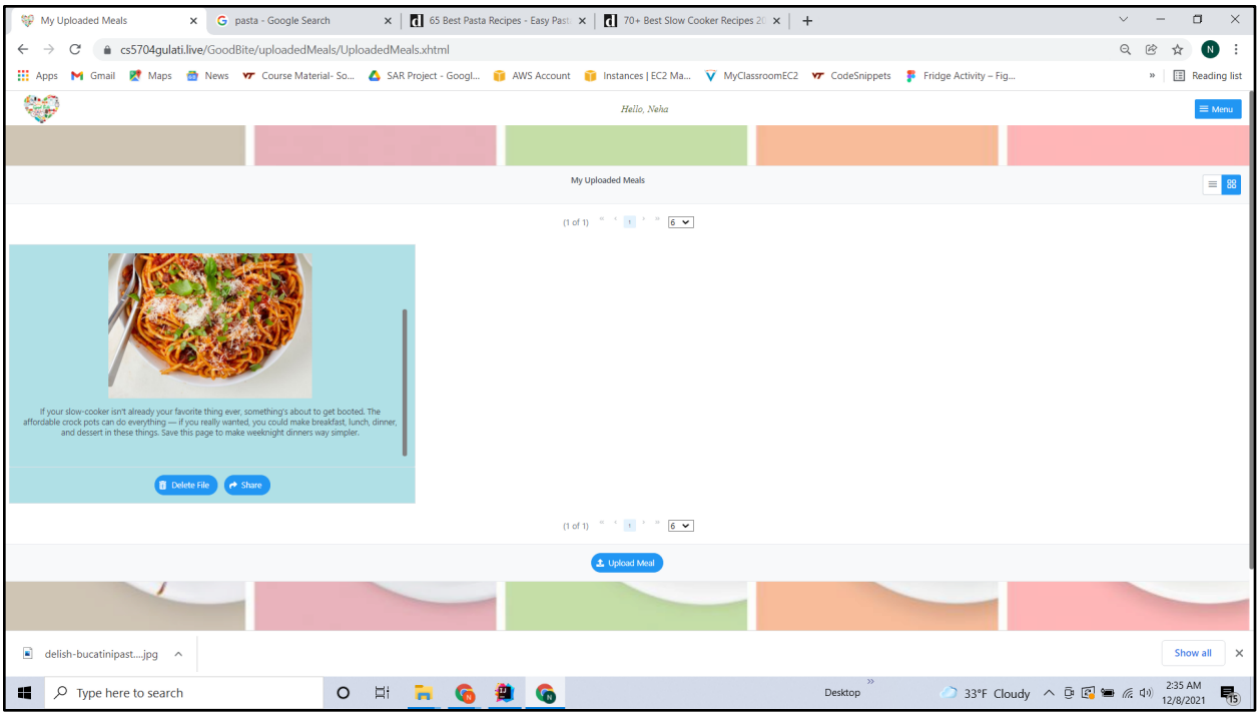


Fig. 24: The user can now view the uploaded Meal in the Uploaded Meals page. Each meal is displayed in a card with a delete and share button attached.

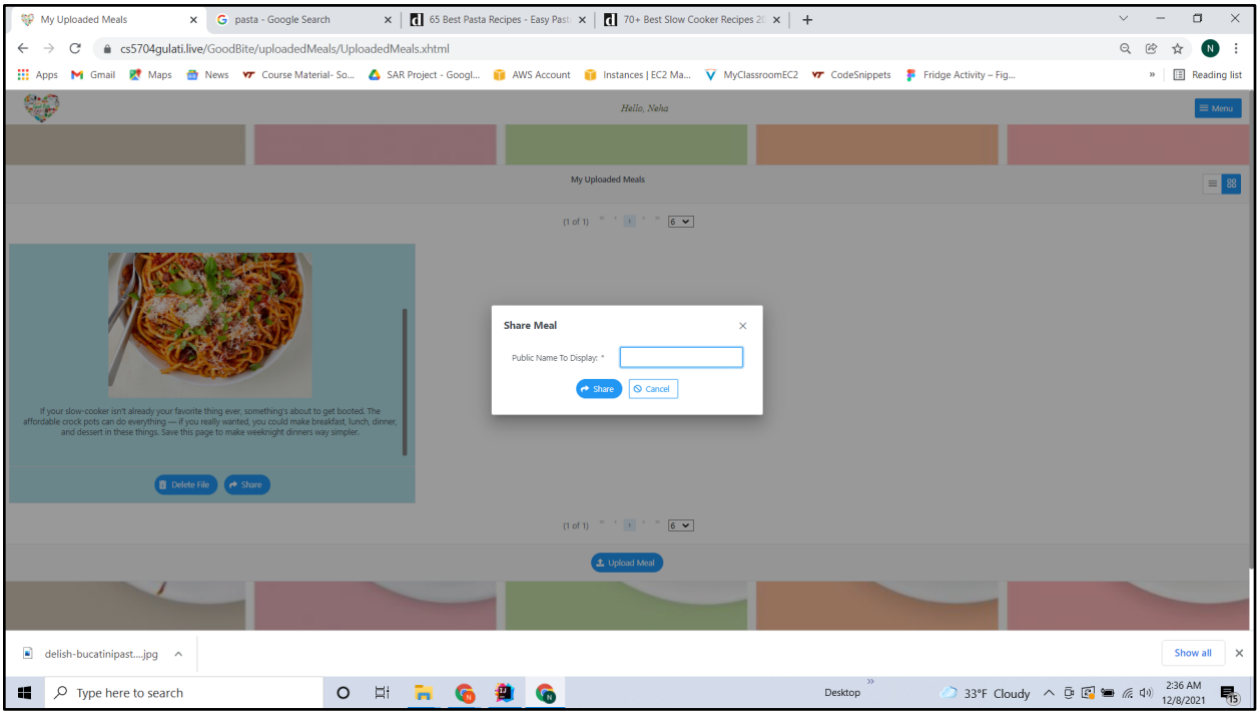


Fig. 25: Clicking the share button prompts the user to enter a name to be publicly displayed with the meal in the Shared Meals page. This field is required.

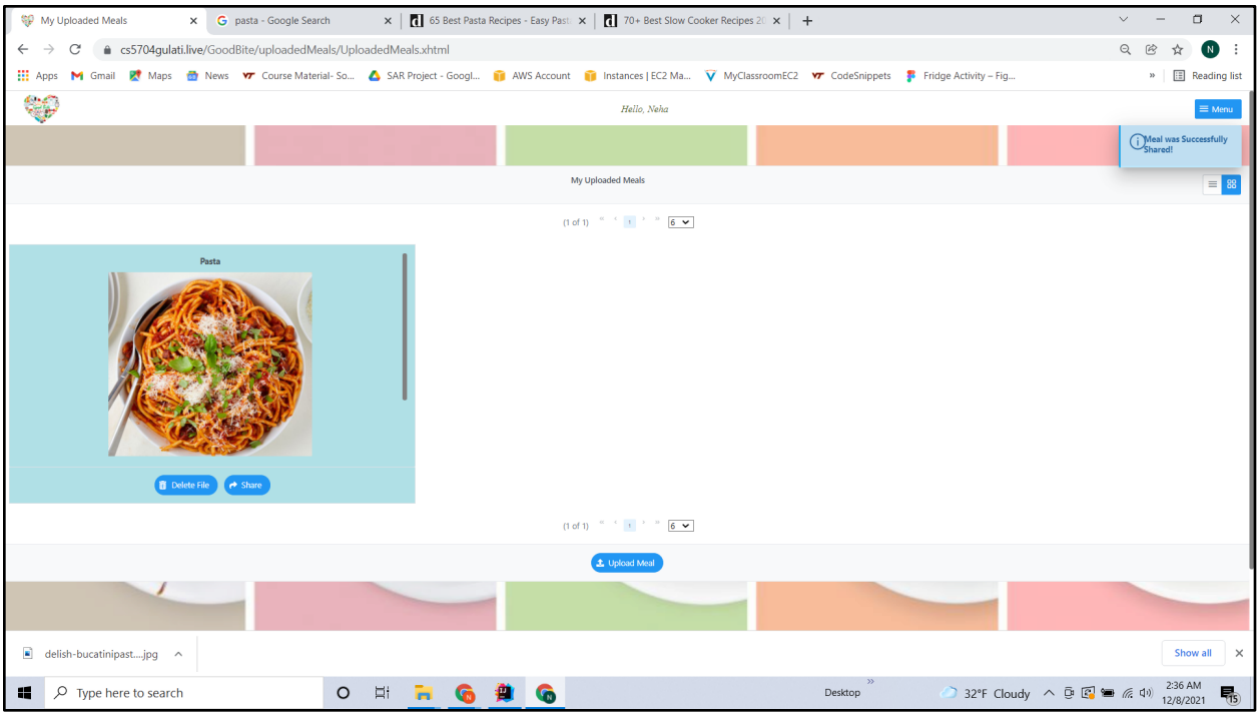


Fig. 26: Upon entering the public name and clicking share, a success message is displayed if the file was successfully shared.

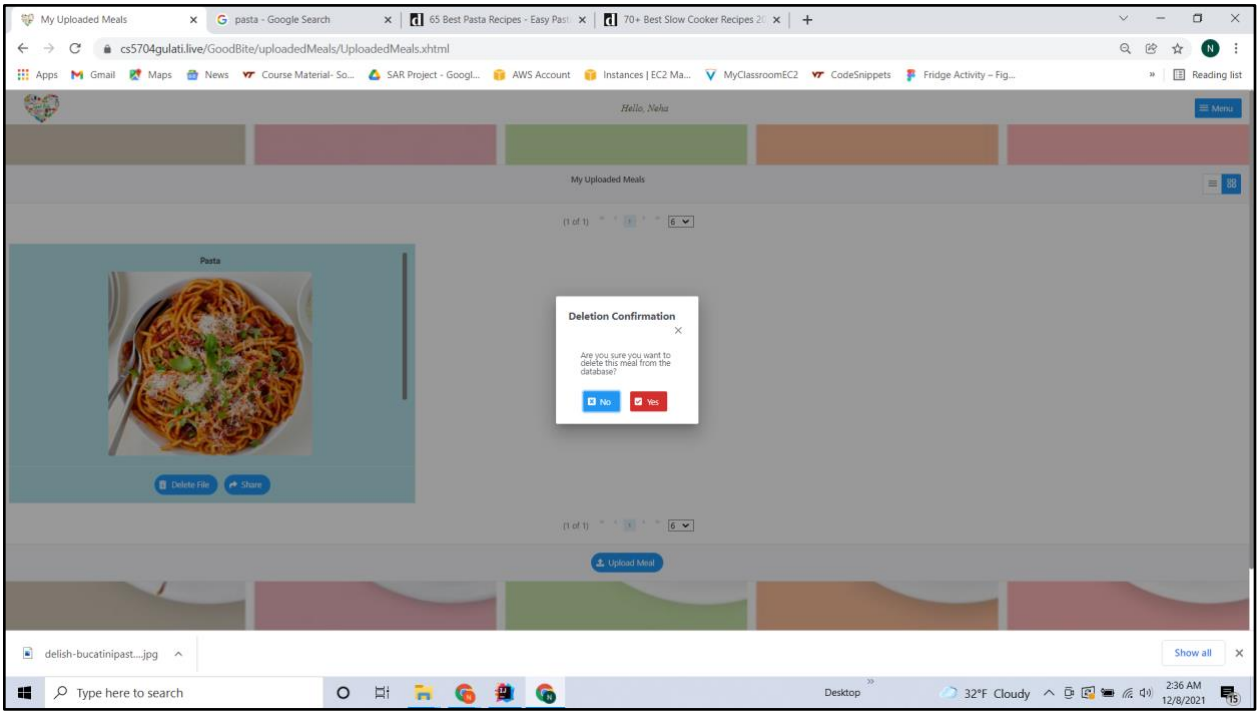


Fig. 27: The delete button causes a confirmation prompt to be displayed, wherein the user can confirm or cancel the deletion of the uploaded meal.

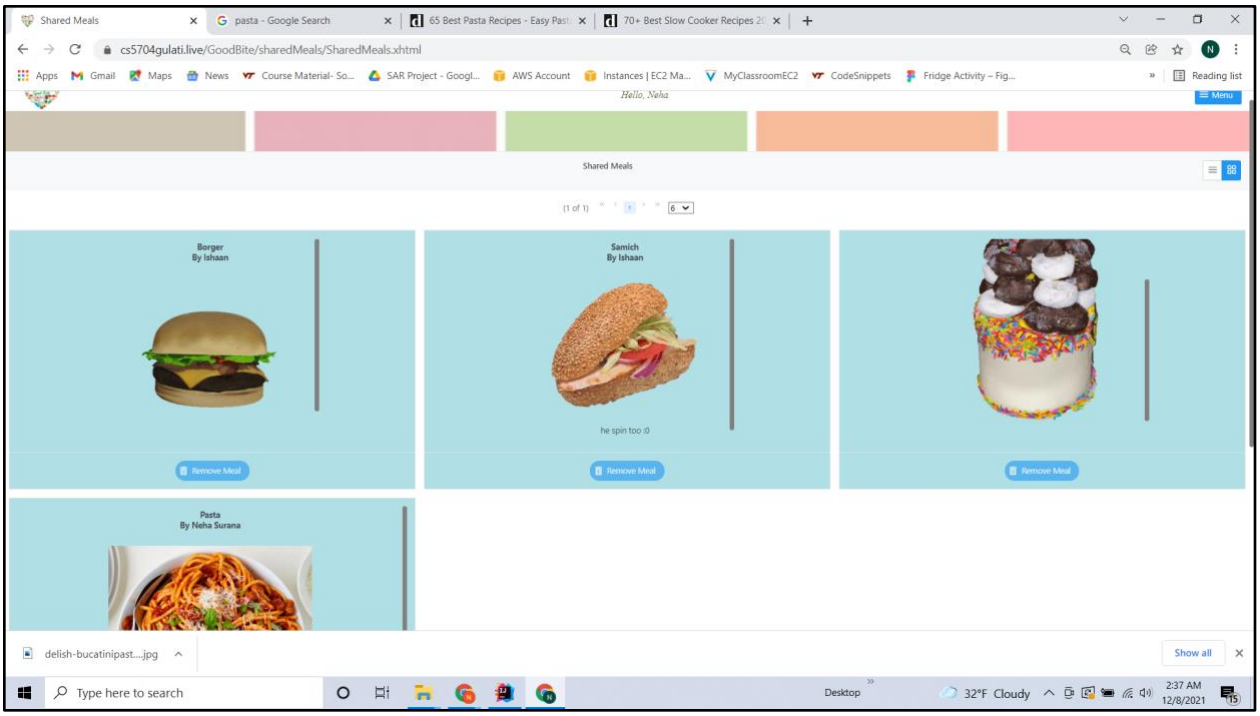


Fig. 28: The Shared Meals page displays the shared meals of all users and their associated names. Each meal card on this page has a "remove meal" button, but the button is greyed out unless it's attached to a meal that was uploaded by the same user that is currently signed in.

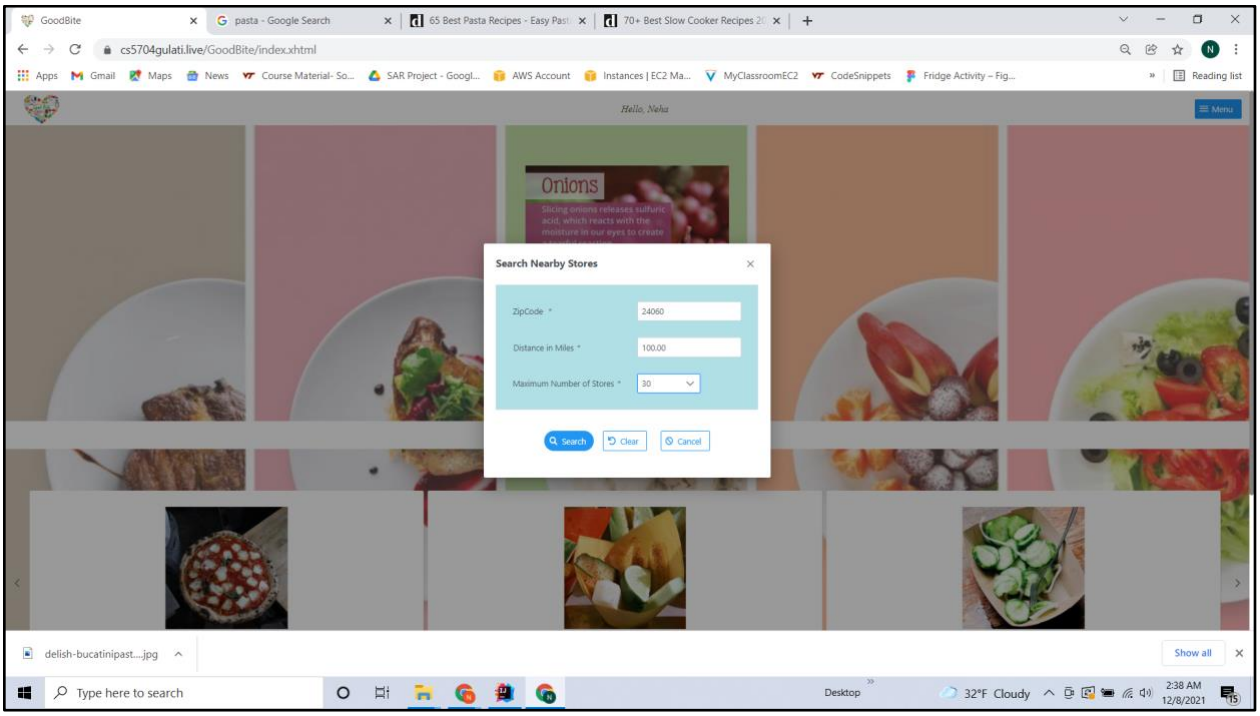


Fig. 29: The Search Nearby Stores dialog prompts the user for a zip code, a distance radius, and the number of desired results. All three fields are required. The dialog includes a button to cancel which closes the dialog, a button to clear the contents of the dialog, and a button to search.

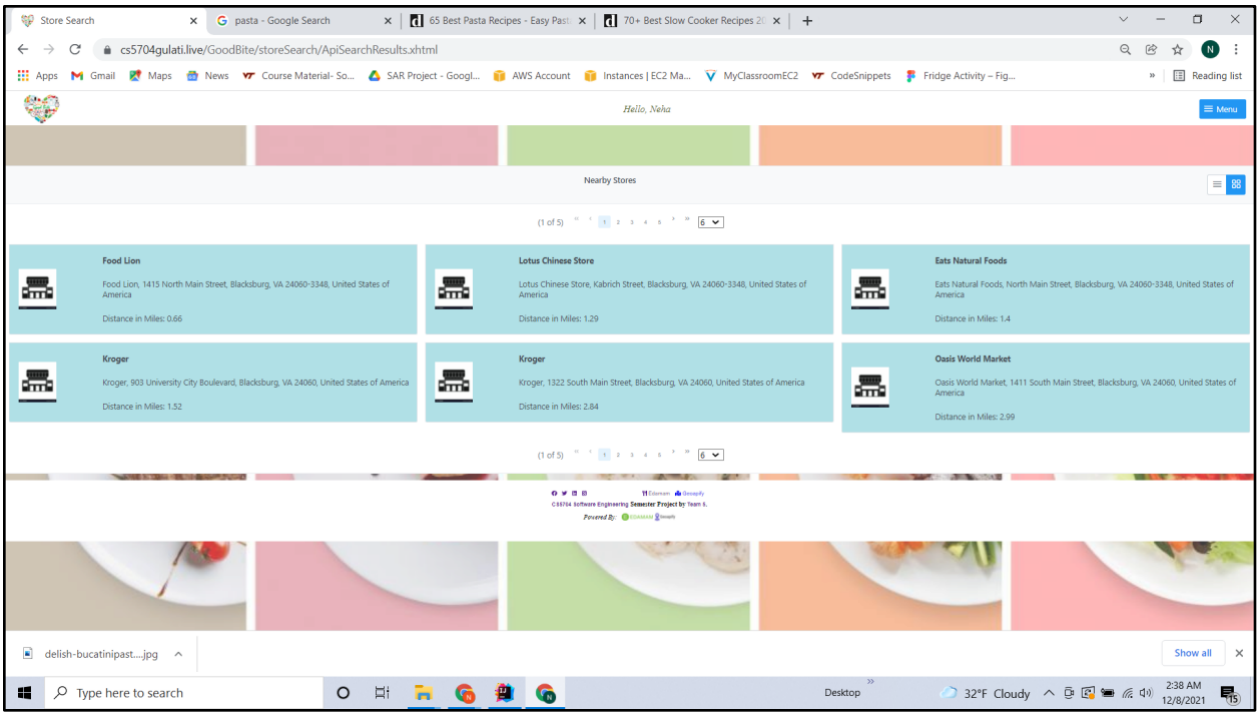


Fig. 30: Clicking the search button in the previous dialog redirects the user to the Nearby Store Results page. This page displays all nearby grocery stores, obeying the parameters the user selected in the search dialog. Each item in this results page includes the name of the store, the address, and the distance.

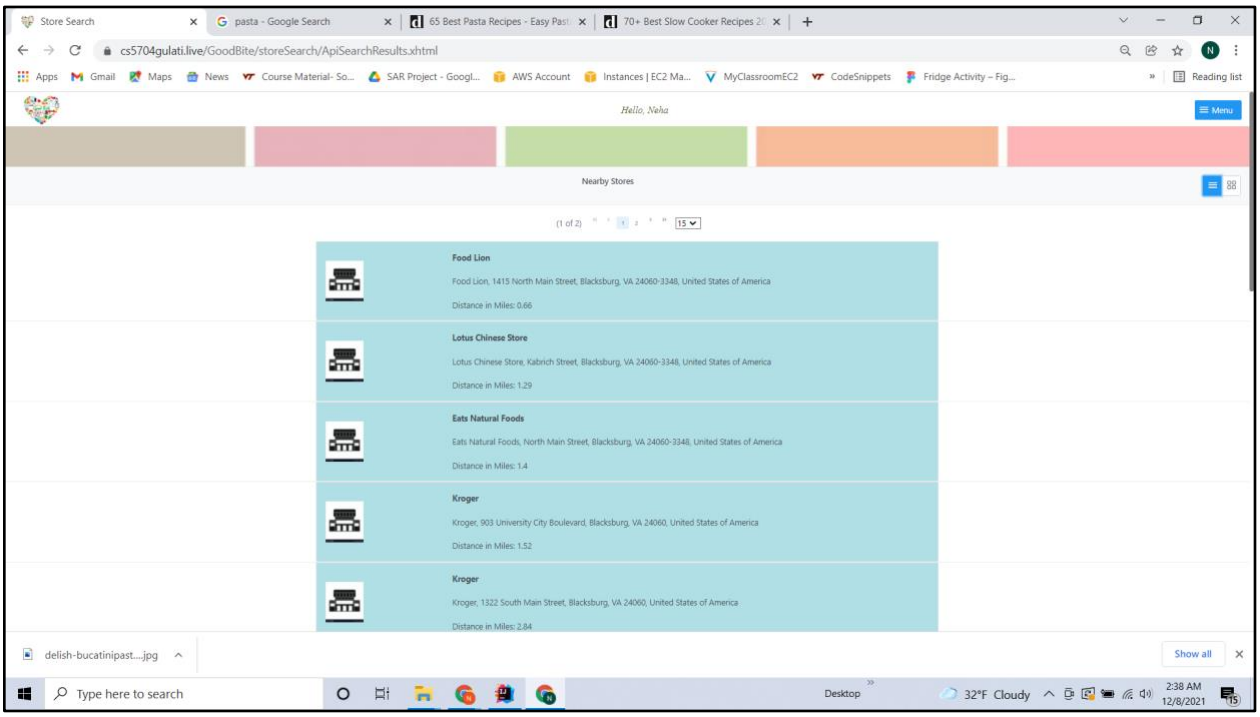


Fig. 31: The Nearby Store Results page can be toggled to display in grid or list format.

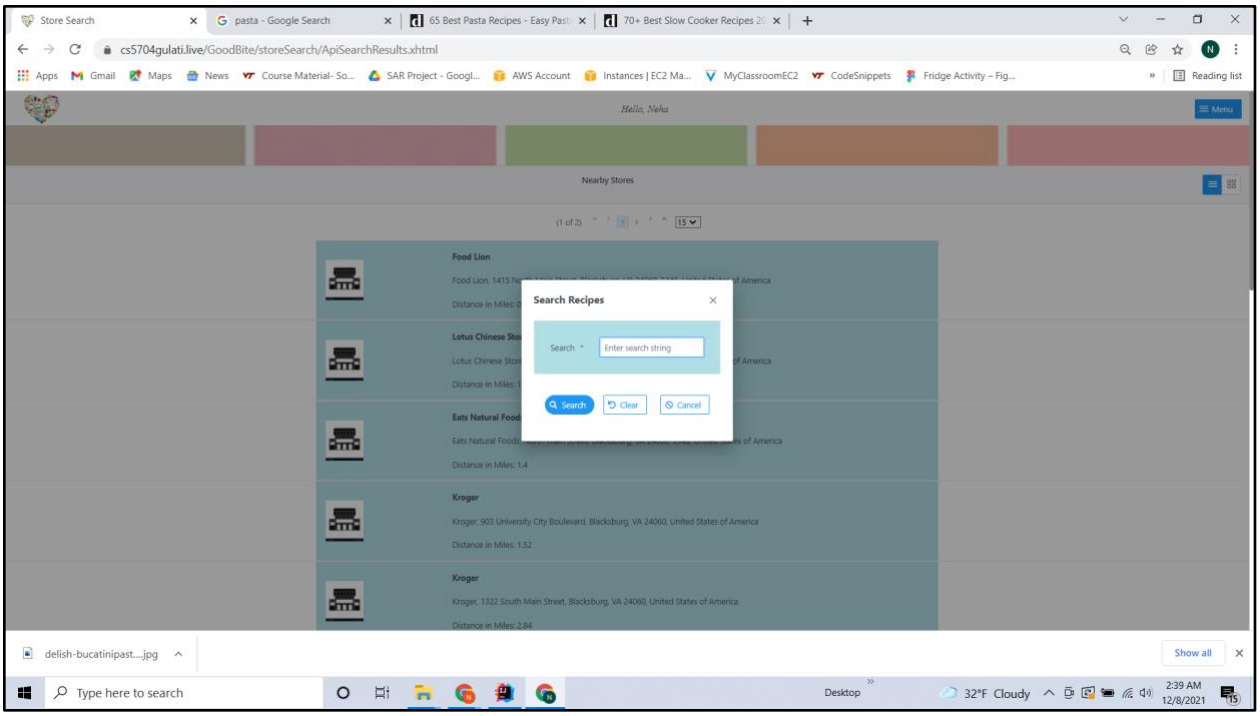


Fig. 32: The Search Recipes dialog prompts the user to enter a search string, which queries the recipes API for relevant recipes. Once pressing the search button, the user is redirected to the API Search Results page

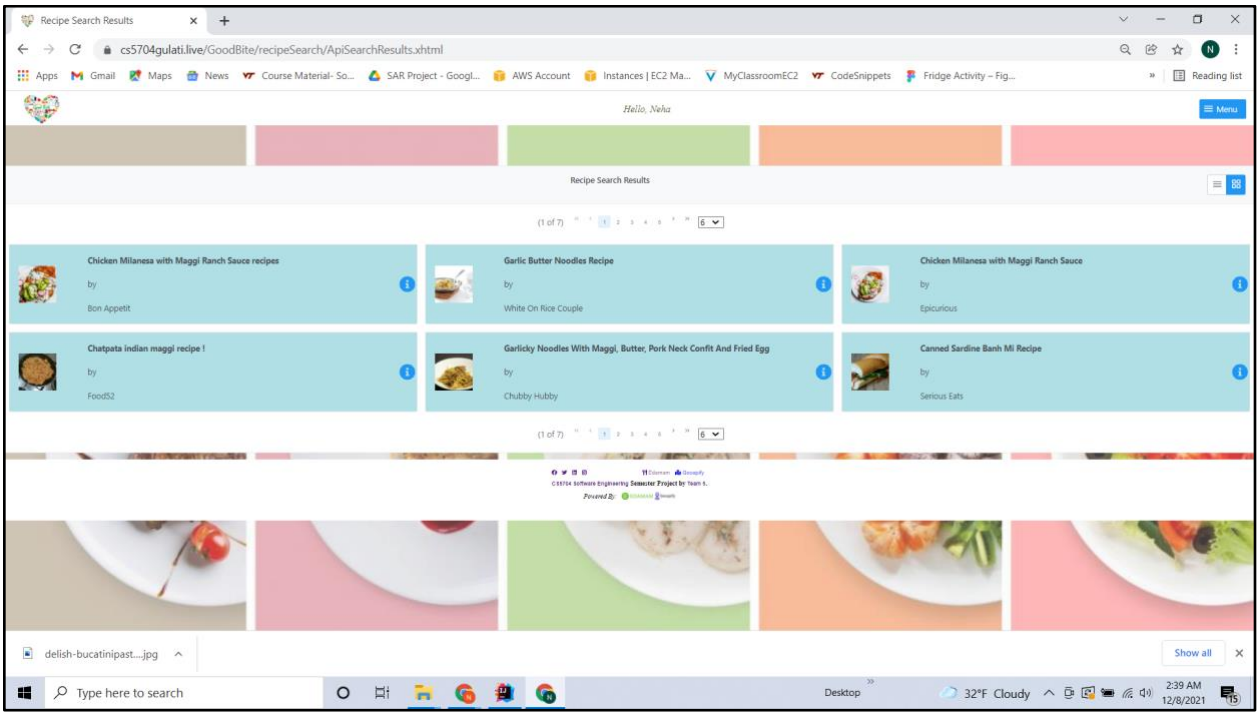


Fig. 33

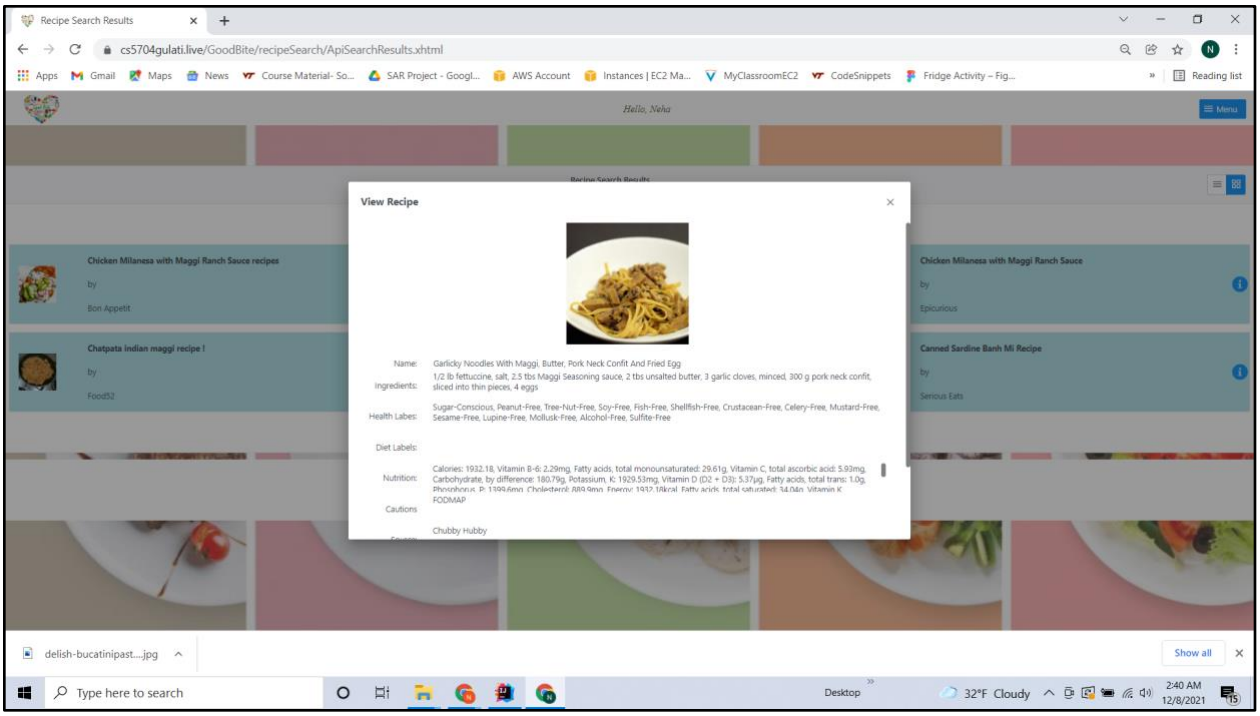


Fig. 34

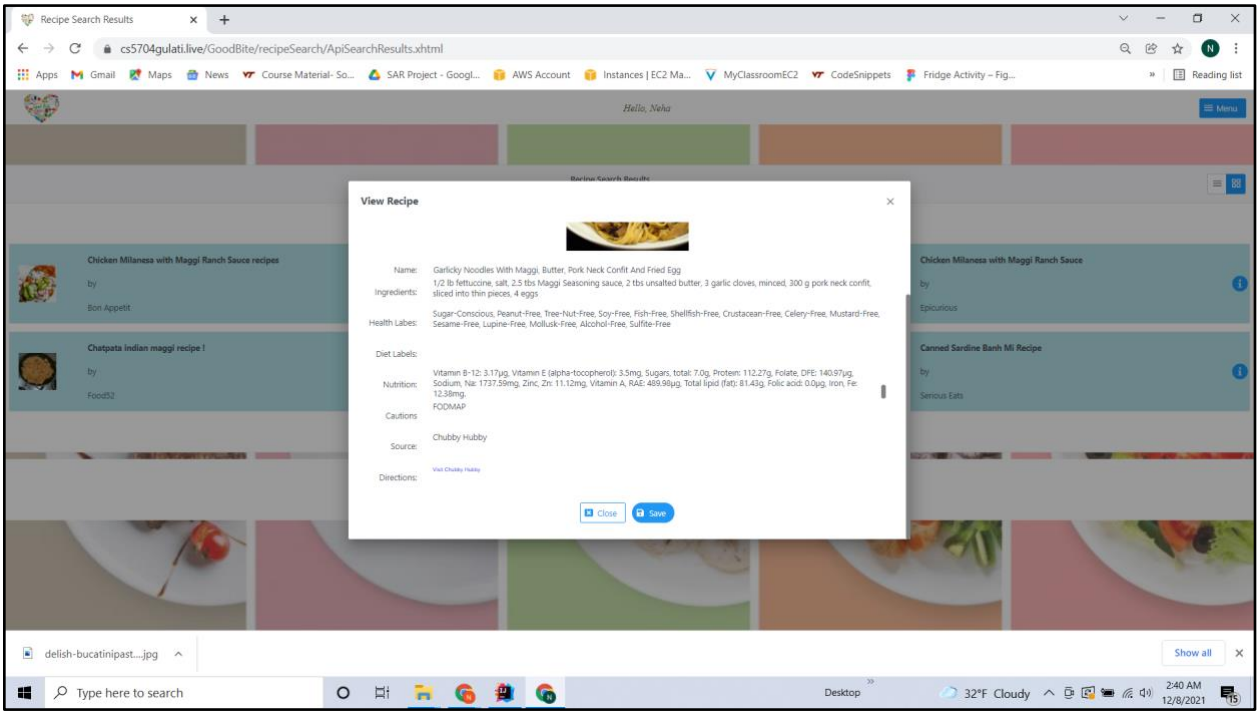


Fig. 35: The API Search Results pages (Fig. **33,34,35**) displays relevant recipes with a recipe name and author displayed in each card. Pressing the information icon, will display detailed information about the recipe, including the ingredients, health labels, diet labels, nutrition information, calories, and a link to cooking directions. From here, the user can save the recipe to the Saved Recipes page.

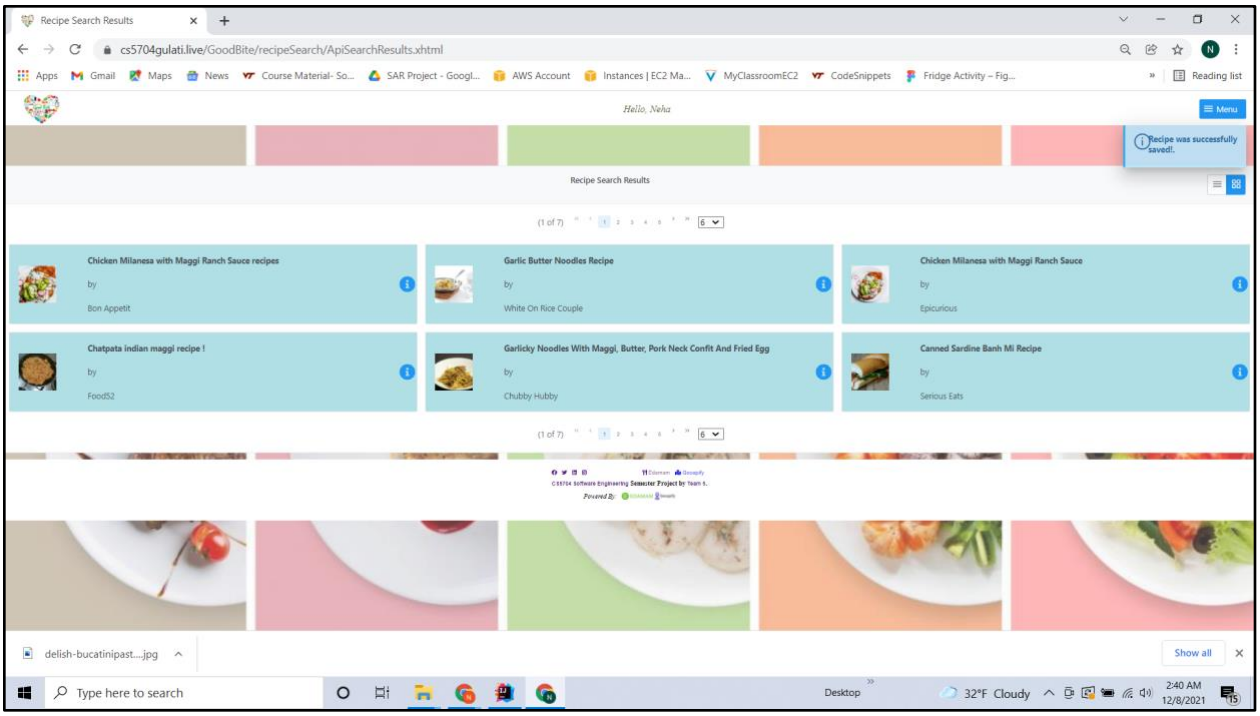


Fig. 36: Upon clicking save, a status message will appear informing the user if the recipe was successfully saved or not.

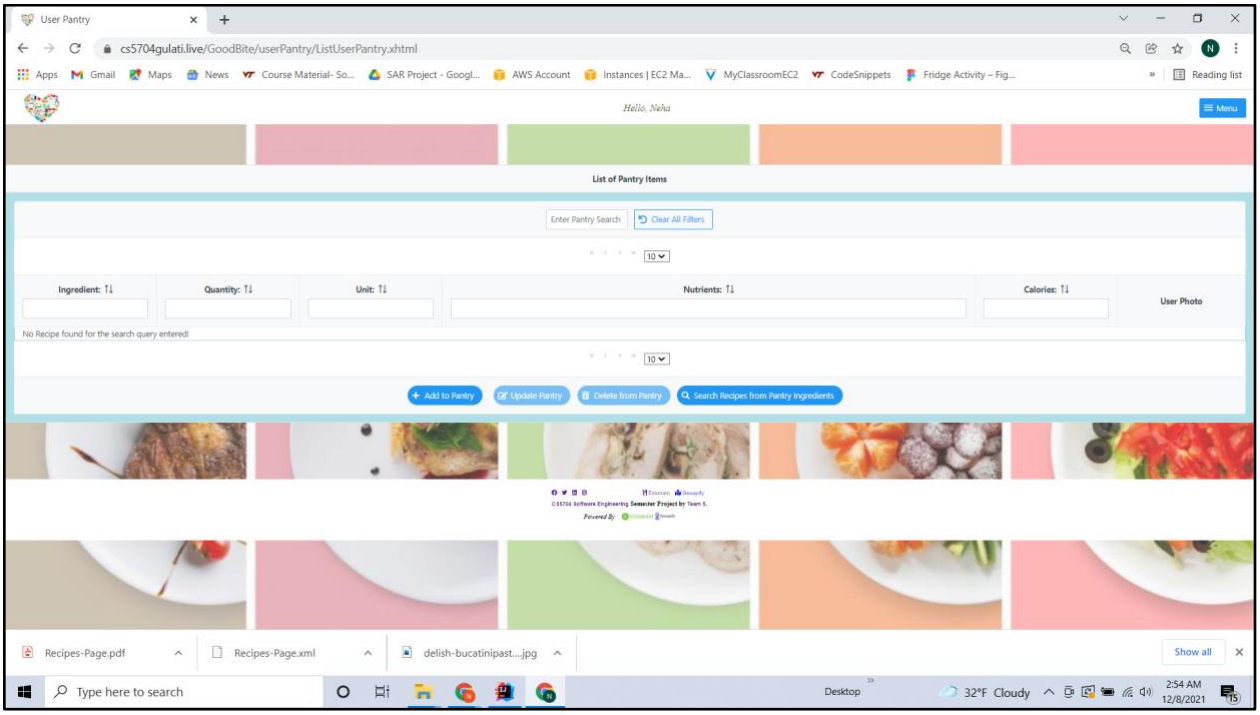


Fig. 37: The Virtual Pantry page is where the user can add their owned ingredients to the database in order to easily search for possible recipes. The page includes a button to add an ingredient to the pantry, update a pantry ingredient, delete an ingredient, or search recipes from pantry ingredients.

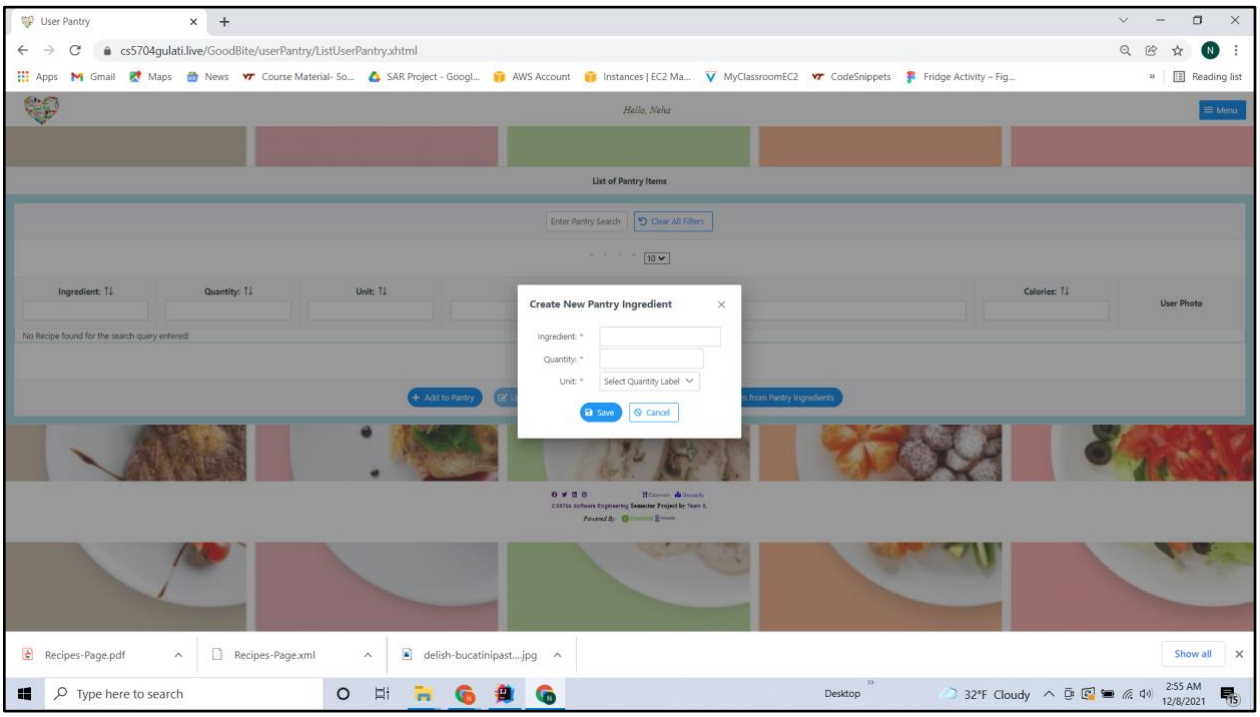


Fig. 38

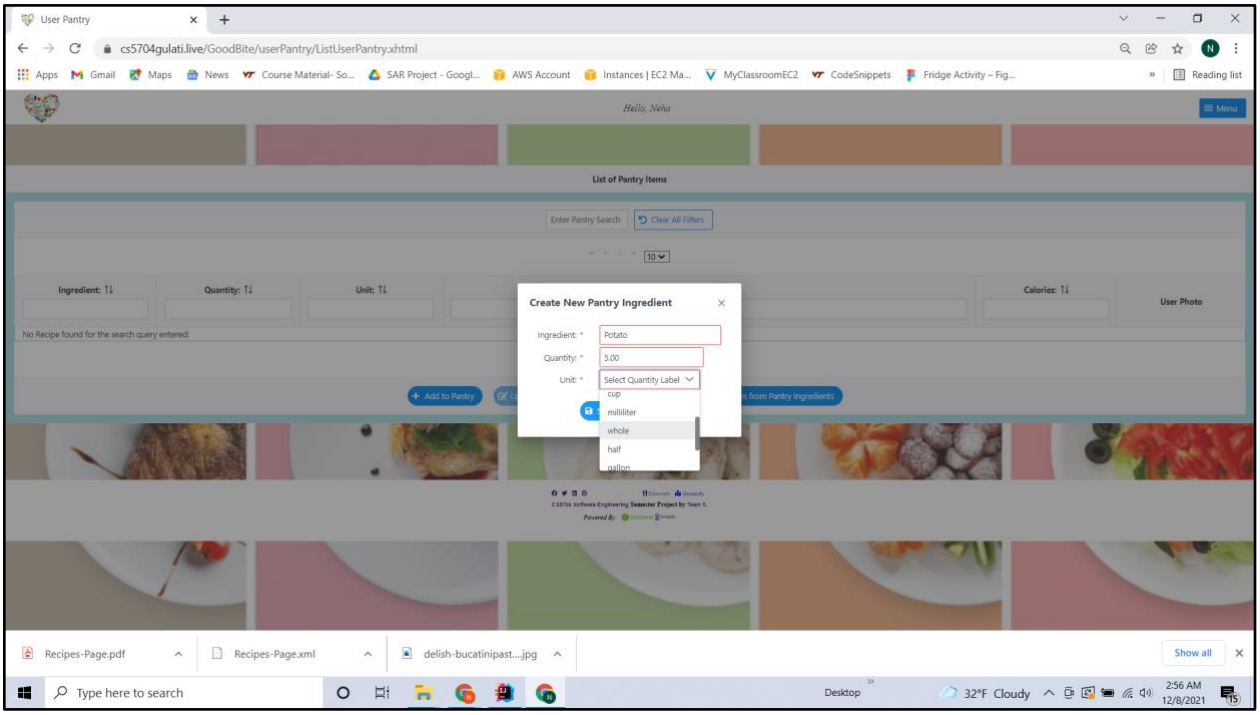


Fig. 39: The “add to pantry” button displays a dialog in which the user enters the ingredient name, quantity, and unit of measurement.

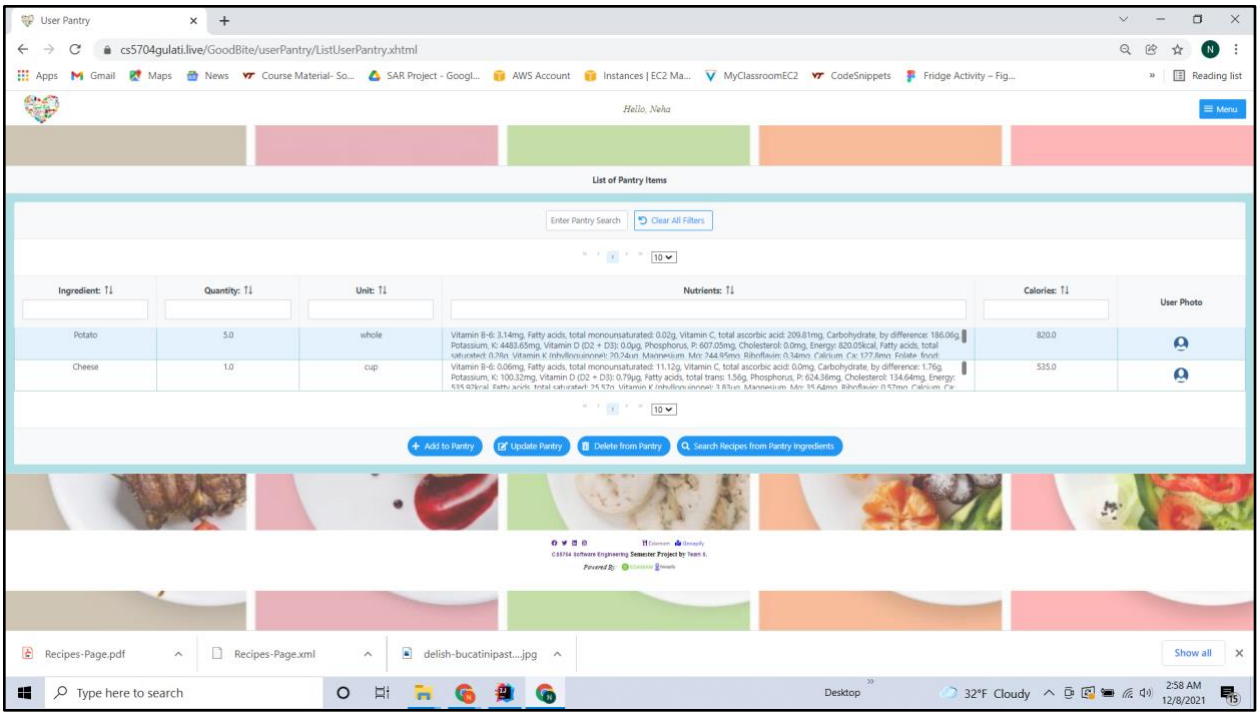


Fig. 40: Clicking save will add the ingredient to the virtual pantry list. The nutrition and calorie information are automatically filled in via a query to the nutrients API.

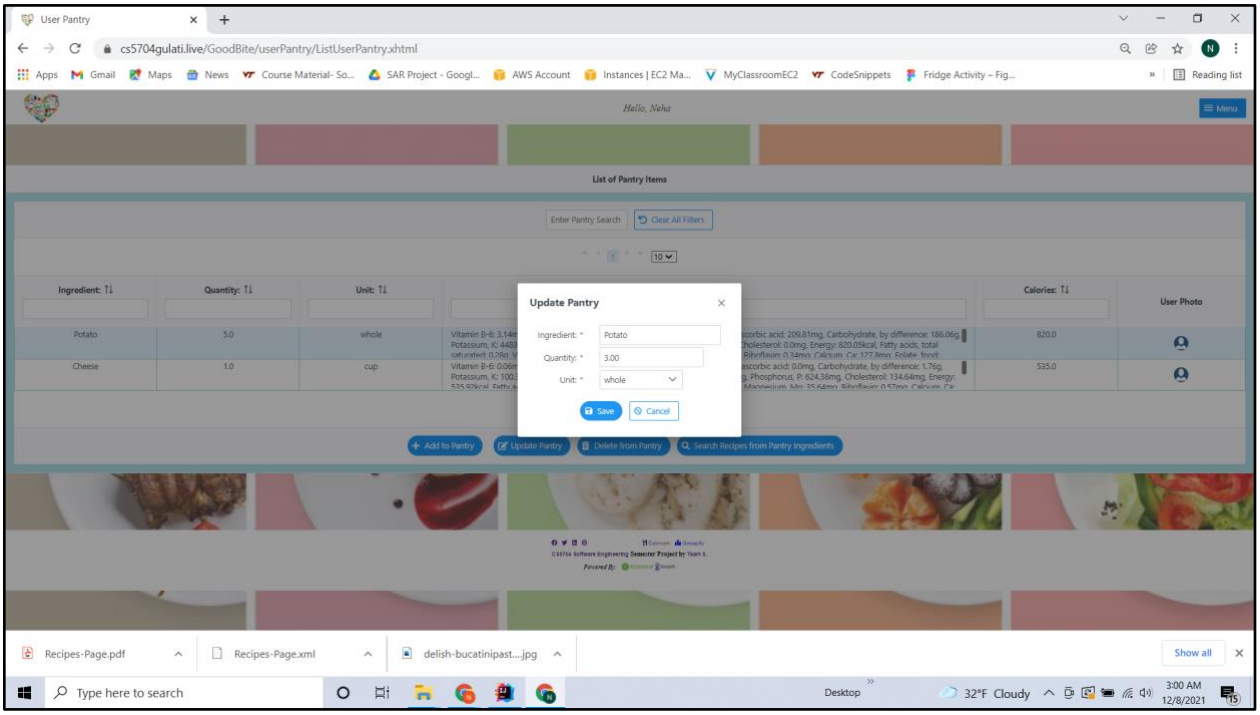


Fig. 41: The “update pantry” and “delete from pantry” button are only functional if an ingredient is selected from the table. The “update pantry” button allows the user to adjust a selected ingredient's name, quantity, and unit.

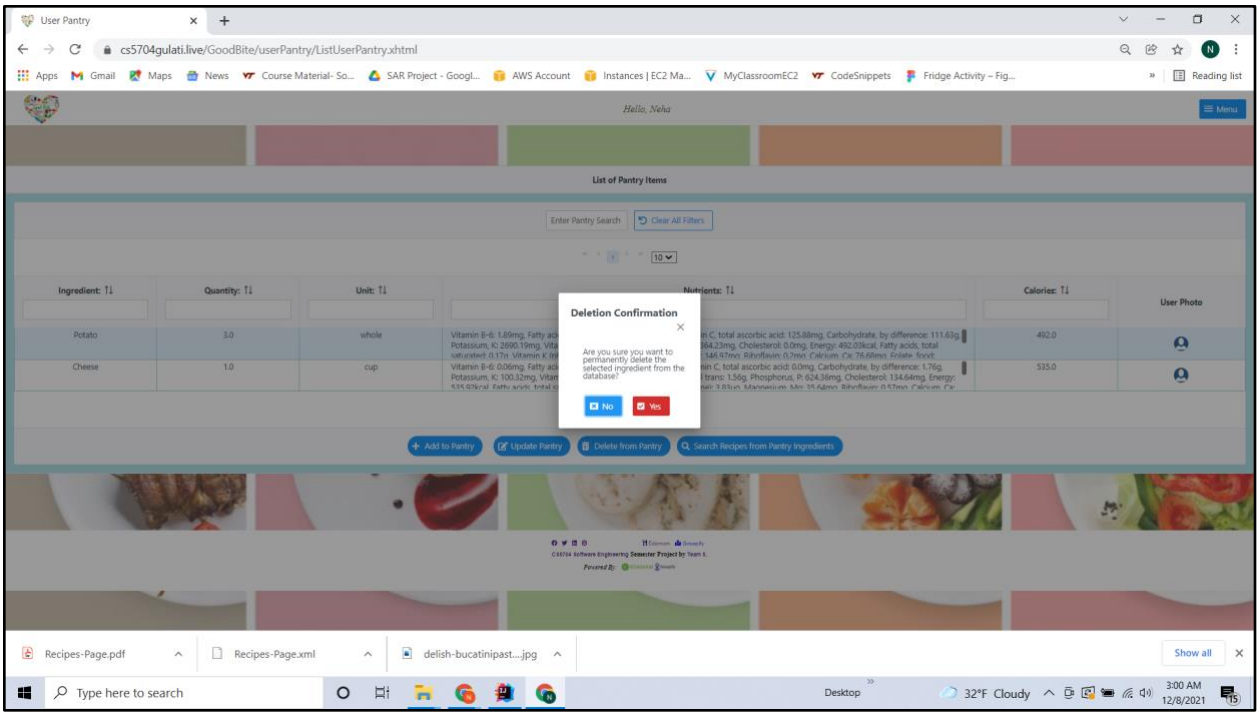


Fig. 42

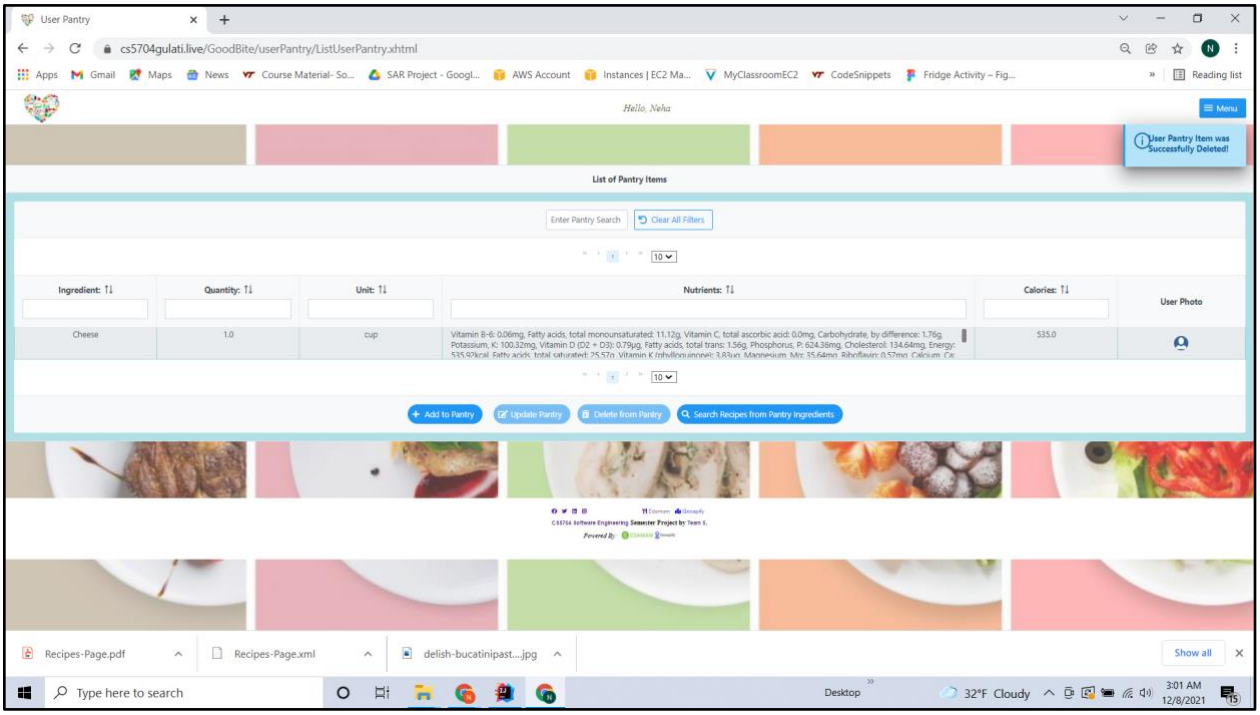


Fig. 43: Selecting an ingredient and clicking the "delete from pantry" button displays a confirmation dialog (Fig. 42) where the user can confirm or cancel the deletion. If confirmed, a status message will be displayed if the deletion succeeded or failed.

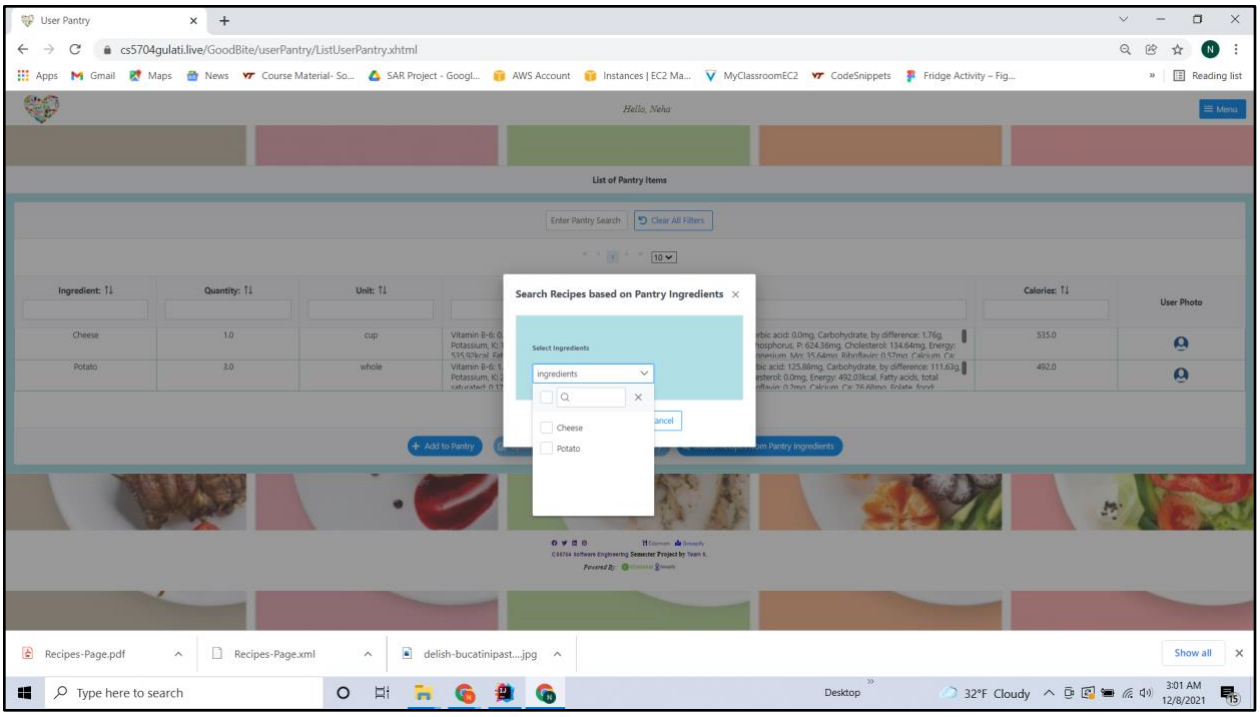


Fig. 44

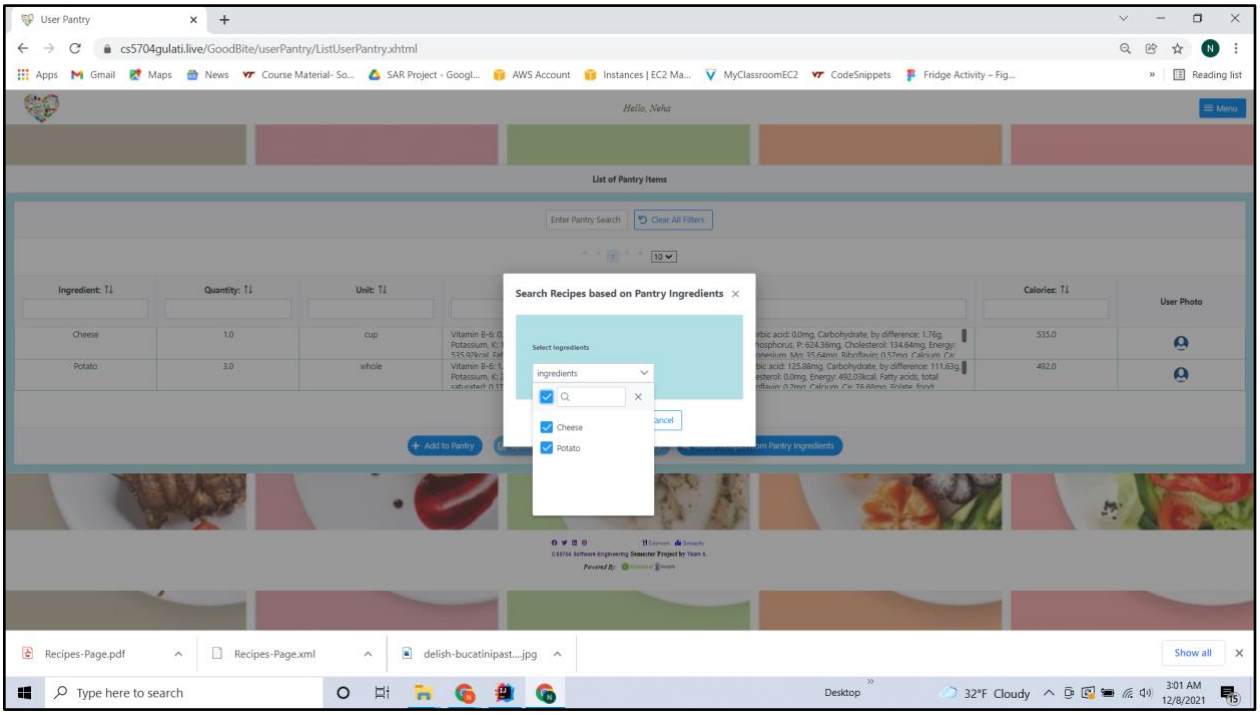


Fig. 45: Clicking the "search recipes from pantry ingredients" button displays a dialog with a dropdown menu containing the ingredients in the virtual pantry. Individual ingredients can be selected from the dropdown, or the top box can be checked to select all ingredients. Pressing search will query the recipes API using the selected ingredients and redirect the user to the Recipe Results page.

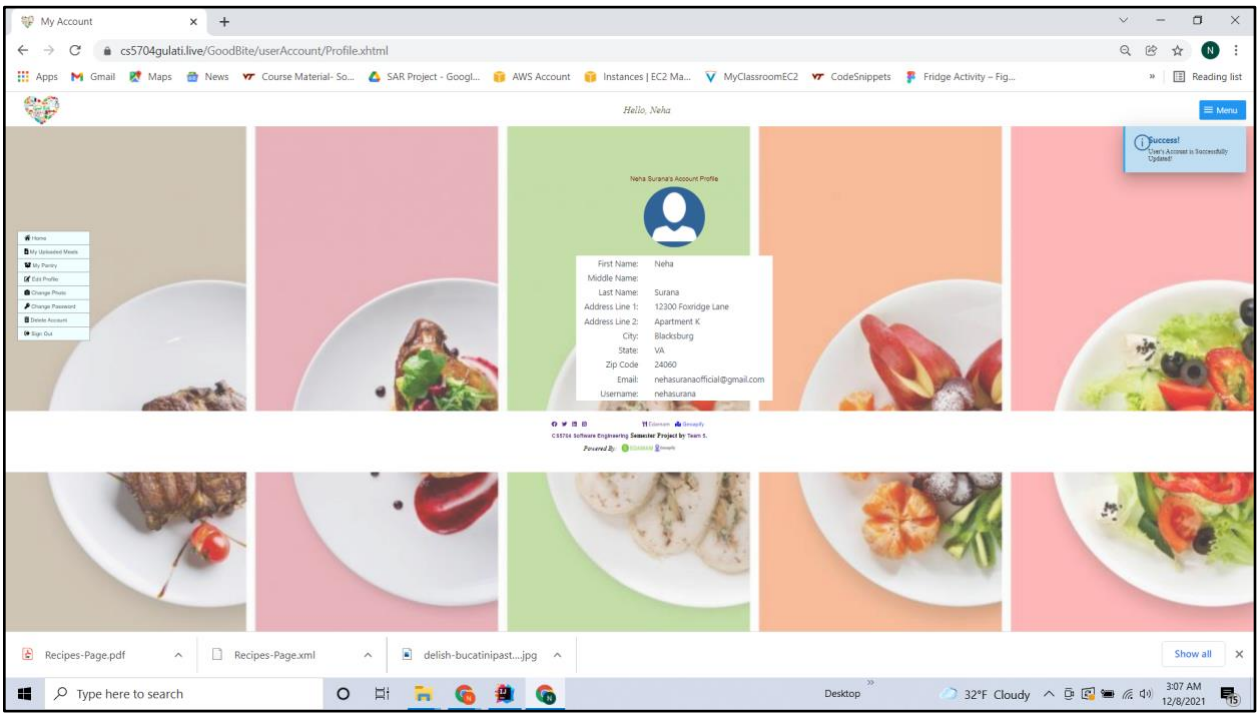


Fig. 46: The user can update their account information, mirroring the functionality of the CloudDrive tutorial.

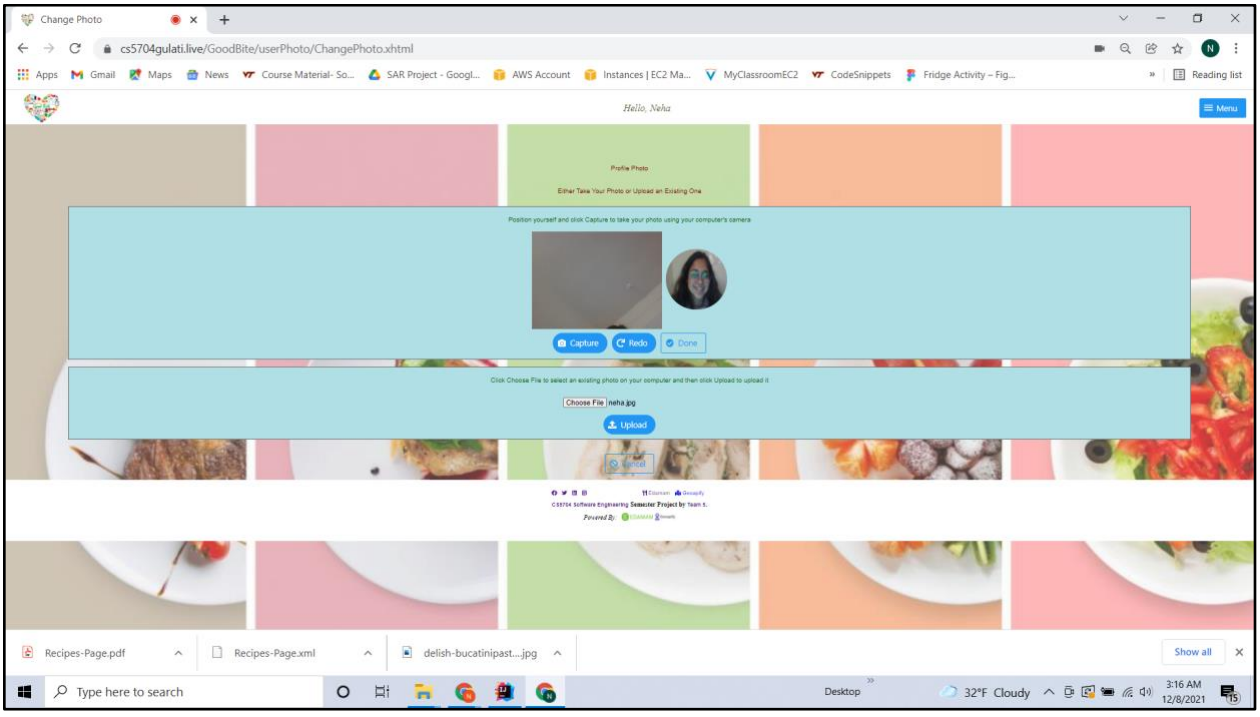


Fig. 47: The user can change their profile photo by either taking a picture with their webcam or uploading an image file. Since our application uses HTTPS, the webcam functionality works even in the deployed application.

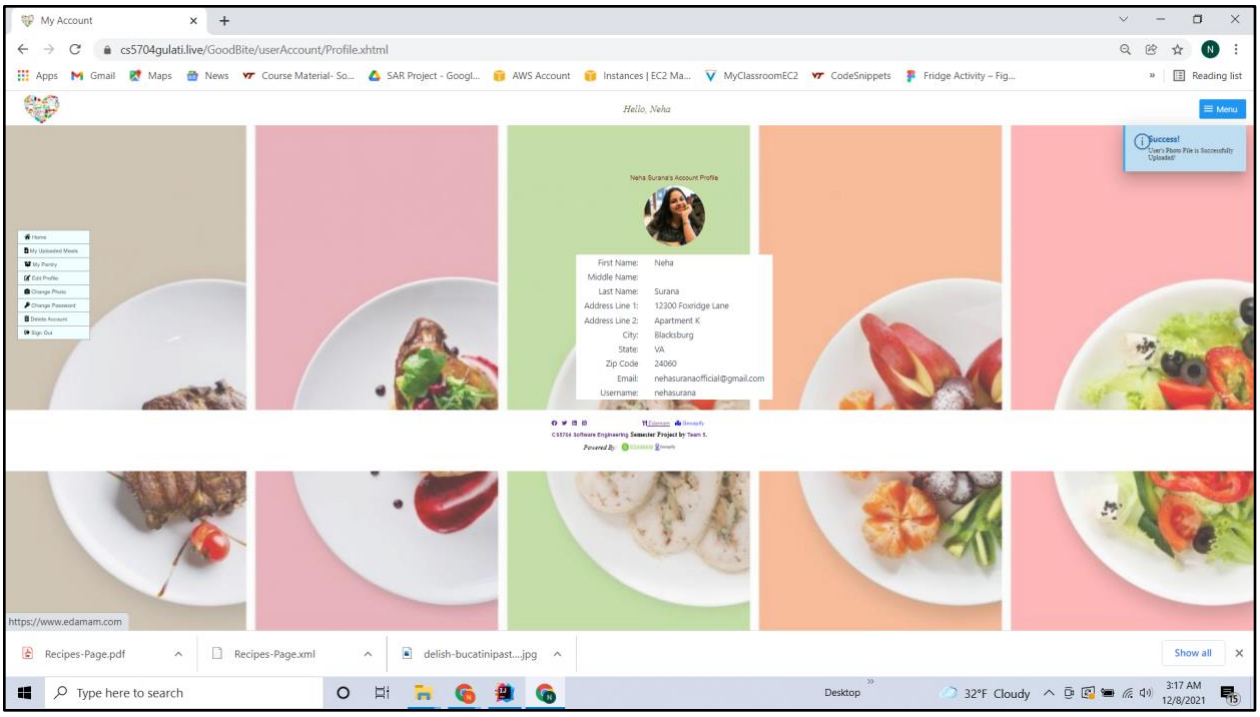


Fig. 48: Upon clicking the upload button, a status message is displayed informing the user whether the profile photo was successfully uploaded and changed.

8. CONCLUSIONS

This project was a huge learning experience for all team members involved. Through this project we were able to apply what we learned in previous tutorials and assignments in our course and apply it to something new and of a much larger scale. Additionally, we gained valuable experience in how to properly engineer a Jakarta/Java EE cloud software application to solve a complex problem. Throughout our development process, we also gained valuable experience in certain aspects of software engineering not explicitly focused on in class, including CSS design, relational table operations including table joins, HTTPS configuration, and software development integration practices using git and GitHub. Although this project is academically motivated and is not intended to compete with real world web applications of a similar focus on food and nutrition, we designed our application to be practically useful, addressing real-world problems surrounding planning and preparing meals, and to be fun to use.

9. SUBMISSION INSTRUCTIONS

Step 1: If your app uses a file storage directory, name it as CS5704-TeamN-FileStorage, where N is your team number.

Step 2: Deploy your cloud software application to your AWS EC2 virtual private server computer.

Step 3: Using the deployed software, each team member is required to create an account and generate *meaningful content* in the database for evaluation and testing. You will be penalized for not having sufficient content. In the case of two-factor authentication, your software is required to provide a Bypass Two-Factor Authentication option for grading. List the usernames and passwords under this option in the *User Accounts* section on page ii.

Step 4: Open your IntelliJ project and change the file paths for your app to run locally.

Step 5: Create a ZIP file containing the following:

1. This file: *CS5704 Team5 Semester Project Report.docx*
2. Your IntelliJ IDEA Ultimate project folder containing your entire documented application.
3. Your Database SQL file to populate the tables in your database.
4. Your CS5704-TeamN-FileStorage directory, original version with no data.

Step 6: Upload the ZIP file to your account in Google Drive, right click the uploaded file and select **Share** from the pop-up menu, enter balci@vt.edu, add a note as "CS5704 Team N Semester Project Submission – Your Name", and click Send.

10. PERCENTAGES OF CONTRIBUTION

We hereby certify that the list of contributions and the corresponding percentages of contribution specified below truly reflect the actual contributions of the team members.
(Write your name as your signature)

Student Name & List of Contributions	*% Contributed*	*Signature*
Jack Sloane's Contributions: 1. A shared meals page based on the concept of 'table joins. 2. Dynamic checking of user ownership of publicly shared files 3. Simple & Advanced Cards- to display information on various pages 4. My Uploaded Meals functionality 5. Shared Meals functionality 6. Database design 7. UI design of respective pages 8. **SvcV-1a**: SOA Conceptual Layers 9. **SvcV-1c**: Services Context Description 10. **SvcV-2**: Services Resource Flow Description 11. **SvcV-4**: Services Functionality Description 12. Wireframing	33.33%	Jack Taylor Sloane
Neha Surana's Contributions: 1. Carousel -to display recipes on the home page 2. Galleria with Thumbnail - to display food facts on home page 3. Virtual Pantry functionality 4. Home Page 5. User Account related pages 6. Search Meals based on pantry ingredients functionality 7. Database design 8. UI Design of our respective pages 9. **OV-1**: High-Level Operational Concept Graphic 10. **OV-2**: Operational Resource Flow Description 11. **OV-5b**: Operational Activity Model 12. Wireframing	33.33%	Neha Surana
Ishaan Gulati's Contributions: 1. Prime Faces : SelectCheckboxMenu (Basic) - to display and select the list of ingredients from a dropdown menu 2. Database search for geocodings 3. Recipe Search functionality. 4. Grocery store search functionality.	33.33%	Ishaan Gulati

5. Nutrition search API integration. 6. Database design 7. UI Design of our respective pages. 8. HTTPS configuration. 9. **SV-1**: Systems Interface Description 10. **SV-2**: Systems Resource Flow Description 11. **SV-4:** Systems Functionality Description 12. Wireframing		
Sum of Percentages:	100%	

11. CALCULATION OF GRADES BASED ON PERCENTAGES OF CONTRIBUTION

Grade Calculation.xls

	A	B	C	D	E	F
1		Grade Given for the Project	**Student 1** Percentage of Contribution	**Student 2** Percentage of Contribution	**Student 3** Percentage of Contribution	
2	Example:	86	40%	25%	35%	100%
3						
4	**Student 1**	91.73	= B2 + B2 * (C2 - 33.333%)			
5	**Student 2**	78.83	= B2 + B2 * (D2 - 33.333%)			
6	**Student 3**	87.43	= B2 + B2 * (E2 - 33.333%)			
7						

Additional percentage of contribution cannot exceed 10%.

If a situation arises where a student is doing more than 10% extra work, Dr. Balci must be informed immediately.

12. TEAM PROJECT REQUIREMENTS

1. Each team member is expected to contribute equally.
2. You shall submit percentages of contribution together with a list of each student's individual contributions with signatures of all team members. In case of disagreement, you shall submit it separately with your rationale. (Write your name as representing your signature.)
3. Grades shall be determined based on the percentages of contribution.
   a. If you contribute more than your equal share, then your grade shall be increased based on the extra percentage of contribution, which cannot be more than 10%.
   b. If you contribute less than your equal share, then your grade shall be decreased accordingly.

c. The extra percentage of contribution shall not be more than 10%. Dr. Balci shall be notified immediately if a situation arises where a student needs to contribute more than 10% extra.

4. A team member who does not cooperate with other team members for conducting the project with equal contribution shall be penalized. Doing more work than agreed upon by the team and claiming extra contribution shall not be acceptable. Cooperation is essential!

13. GRADING SHEET

CS5704 Semester Project Grading: Team N				
		Points	Earned	Notes
Requirements Specification		10		Minimum 18 correct Functional Requirements (6 per team member) Minimum 9 correct Non-Functional Requirements (3 per team member)
Architecture Specification		16		Provide DoDAF diagrams of the Client-Server + Service-Oriented architecture for your cloud software application.
Software Design Specification		16		UI design specification with graphics created with Balsamiq and Database design specification with entities, relationships, and ER diagram.
Delivered Software Functionality				Description of the functionalities (features) of your deployed cloud software application by using annotated screenshots of user interfaces.
	Quantity of Features	16		How many significant features each taking at least one hour to implement?
	Variety of Features	16		User accounts, database, and API are required.
	Complexity of Features	16		Difficulty of creating each feature? How many hours would it take for each?
Software Documentation (All Java and XHTML files)		5		Meaningful and informative documentation showing that you understand what the code is doing in all Java and XHTML pages.
Project Report Quality		5		(-5) Document instructions are not deleted in the final version of the report. Title page, table of contents, styles, formatting, hyperlinks, etc.
Documentation Penalty				
	Subtract 3 to 10 points			After copy-and-paste of code, you are required to update the associated documentation and change the names of variables, constants, arrays, functions, methods, and classes to be meaningful.
	Total Points	100	0.00	
		% Cont.	*Grade*	
	Team Members:			**Minimum** time required: 6 hrs/week x 4 weeks x 3 students = 72 hrs
1.	Student Full name 1	33.333%	0.00	
2.	Student Full name 2	33.333%	0.00	
3.	Student Full name 3	33.333%	0.00	

REFERENCES

1. Balci, O. (2021), "CS3754 Cloud Software Development Course Website," https://manta.cs.vt.edu/cs3754
2. Balsamiq Wireframes: https://balsamiq.com/
3. Eclipse Foundation (2021), "Jakarta EE," https://jakarta.ee/
4. Edamam, "Nutrition Api", https://developer.edamam.com/edamam-docs-nutrition-api
5. Edamam, "Recipes Api V2",https://developer.edamam.com/edamam-docs-recipe-api
6. GeoApify, "Geoapify Location Platform API Docs", https://apidocs.geoapify.com
7. Ishaan Gulati, "HTTPS Configuration for Apache web server", https://docs.google.com/document/d/1zqlEqL6PTCrre8IJWcubLredYmrmLuK3/edit?usp=sharing&ouid=108394463113325486234&rtpof=true&sd=true
8. Ishaan Gulati, "Reverse Proxy Setup Apache and WildFly", https://docs.google.com/document/d/1CFeXc08kjIQa1vfxxxSZZIwM7Hk-jrkYcxtieOepIVQ/edit
9. JetBrains (2021), "Intelligent Java Integrated Development Environment Advanced (IntelliJ IDEA)," https://www.jetbrains.com/idea/
10. Lucid Chart: https://www.lucidchart.com/pages/
11. MySQL (2021), "MySQL Open Source Relational Database Management System," http://www.mysql.com/
12. PrimeTek (2021), "PrimeFaces UI Component Library for JSF," https://www.primefaces.org/
13. Prime faces 11 : https://www.primefaces.org/primefaces-11-0-0-rc1-released/
14. WildFly (2021), "WildFly Jakarta / Java Application Server," https://www.wildfly.org/

APPENDIX A: MEETING MINUTES

Meeting Number 1

Date & Duration:	October 11, 4 hours.
Location:	Library
Members Present:	All
Members Absent:	None
Discussions:	API and topic selection
Decisions Made:	Edamam Recipes API, Edamam Nutrition API, GeopApify
Work Assignments:	Explore API usage
Minutes Prepared By:	Jack

Meeting Number 2

Date & Duration:	October 25, 3 hours.
Location:	Library
Members Present:	All
Members Absent:	None
Discussions:	UI Design and basic functionality
Decisions Made:	Overall UI layout, page descriptions, user scenarios
Work Assignments:	Implement home page, and sign-in/sign up functionality
Minutes Prepared By:	Neha

Meeting Number 3

Date & Duration:	November 1, 4 hours
Location:	Library
Members Present:	All
Members Absent:	None
Discussions:	Division of labor
Decisions Made:	Division of labor into three groups of related pages
Work Assignments:	Start development of allocated work
Minutes Prepared By:	Ishaan

Meeting Number 4

Date & Duration:	November 8, 2 hours
Location:	Library
Members Present:	All
Members Absent:	None
Discussions:	Reporting in on progress
Decisions Made:	Specific design decisions
Work Assignments:	Continued development
Minutes Prepared By:	Jack

Meeting Number 5

Date & Duration:	November 15, 4 hours
Location:	Library
Members Present:	All
Members Absent:	None
Discussions:	Reporting in on progress
Decisions Made:	Specific design decisions and documentation work allocations
Work Assignments:	Continued development
Minutes Prepared By:	Neha

Meeting Number 6

Date & Duration:	November 22, 2 hours
Location:	Library
Members Present:	All
Members Absent:	None
Discussions:	Reporting in on progress and Documentation work
Decisions Made:	Specific design decisions
Work Assignments:	Continued development
Minutes Prepared By:	Ishaan

Meeting Number 7

Date & Duration:	November 29, 2 hours
Location:	Library
Members Present:	All
Members Absent:	None
Discussions:	Integration
Decisions Made:	Identification of bugs and integration plans
Work Assignments:	Continued integration, UI polishing, debugging
Minutes Prepared By:	Jack

Meeting Number 8

Date & Duration:	Dec 3, 4 hours
Location:	Library
Members Present:	All
Members Absent:	None
Discussions:	Continued Integration and Documentation
Decisions Made:	Integration progress, Documentation completion
Work Assignments:	Testing
Minutes Prepared By:	Neha

Meeting Number 9

Date & Duration:	Dec 8, 4 hours
Location:	Library
Members Present:	All
Members Absent:	None
Discussions:	Deployment and Submission
Decisions Made:	Deployment and Submission
Work Assignments:	N/A
Minutes Prepared By:	Ishaan