Cookable

In general, this is a really good milestone. I can see that you improved a lot from M1 to M2. That is the point of this class, getting better from one milestone to the next.

Excellent mockups, and thank you for implementing the storyboards.

Database architecture needs more work. That is completly understandable because we only had a lecture about DB architecture and design. All teams are having the same problems with that section as well. Please, feel free to come to my office hours, and we can do that section together.

See detailed comments per sections below.

By:
Copyright Team 5 CSC 648-848 Spring 2019
Trevor Sampson (Team Lead, Front & Back-end) tsampson1@mail.sfsu.edu
Thanh Dip (Back-end Lead Developer, Database & Back-end)
Pyae Naing (Back-end Developer, Github Master)
Tony Rodriguez (Front-end Lead Developer)
Shivam Rai (Front-end Developer, UI/UX Developer)
Surabhi Chavan (Front-end Developer)

Milestone 2

April 3, 2019	Milestone 2 Version 1

Your table is missing the older milestone versions. It needs to more rows for M1 v1, and M2 v2. The newest version by date must be on the top of the table.

1. Data Definitions Version 2

Casual User - Casual visitor who browses the website shall be able to view and search for recipes.

- Casual users can look up recipes but unable to add, edit or save them.
- Casual users will not be able to save their ingredients and recipes to the database.

Registered User - Users who are registered can view and browse like a casual user but also has access to add recipes.

- Registered user will have their username and emails stored along with an encrypted password.

Admin - User who has elevated permissions, able to edit and delete recipes.

- Admins will be able to edit and change information across the site including but not limited to any recipes, ingredient, comments, etc.

Privacy Policy - Shall denote policies about user's data and why, what, and where the data is stored.

- The privacy policy shall be specified on the website at /privacy.
- Only basic user info will be recorded i.e username, email and any information voluntarily given up by the user.

Profile - User customization, preferences, and other data shall be saved here.

- Connected in the database with a registered user id.

Recipes - Recipes will be stored in a table with the name of the food as well as the ingredients to make the food.

- All recipes will have a unique id to match.
- All recipes will display the numbers of ingredients required.
- Cooking time and other recipes descriptions will be available.

Recommendations Tab - This is where popular/recommended recipes will be displayed.

- The recommendation will be based on the ingredients the user currently have.
- The recommendation will be based on popular recipes from other users.

Recipe Instructions - Instructions for the recipes step by step.

- All the recipe instructions will display the required ingredient amounts.
- Each recipe will have step by step instructions for the required ingredients.

Ingredients - A list of ingredients and other names that can be added.

- Users can look up recipes that have a specific ingredient in them.

Pantry - Ingredients the user has to use to cook any recipe.

Ingredient Type - Groups ingredients into certain types like vegetables, meats, dairy, dairy alternatives, etc.

Likes - Counts of likes for each recipe respectively.

- Registered users are able to like the recipes by others.

Favorite - Registered users can mark recipes and save them as favorites.

Comments - Each recipe can be commented on.

Cuisine - The type of food, i.e Chinese, Mongolian, American, Pies, Cookies, etc. A recipe can be multiple food types.

Calories- The calorie count of a recipe if available.

I really think that your priorities #1 are very light. You need to add more functional requirements from priority #2 to priority #1. I know that you are trying to play safe here, and I understand that. But, if you only implement the priorities #1 that you have right know, the project won't meet the quality standards expected in this class. So, please make it more interesting adding more complicated priorities to #1 from #2 or #3

2. Functional Requirements Version 2

Priority 1

Casual Users:

1. A casual user can find recipes without registering.
2. A casual user can search ingredients to find recipes.
3. A casual user can search for recipes by name.

Registered User:

1. A registered user can get recipe from another registered user.
2. A registered user will be able to manage their own profile.

Website and Admins:

#1 is not a functional requirement, it looks more like a story

Functional requirements should not be more than one or two lines per requirement. Anything longer than that, should be revised or splited into more requirements

1. The application will have pages for recipes, this page will feature steps, cuisine, meal type, general type, serving size, ingredients needed vs ingredients available, recipe rating (in terms of likes), calorie count and nutritional information. We will also have a comments section on each recipe page.
2. New users shall be able to register.
3. A privacy policy shall be displayed to new users upon registration, and it shall be available under user settings.
4. Registered users shall be given the option to upload their photo on their profile.
5. A free text search box shall be displayed to all users which shall accept search terms regarding recipes, a button would be placed on this search bar to open ingredient search.
6. The default display in the free text search box shall be “Search recipes”, with a button to switch to ingredients.
7. There will be a sidebar to search from ingredients (need to review search feature if this needs to be added), you can type, and it will return you suggestions or you can find them ordered categorically in dairy, vegetables, fruits, etc.
8. A help section shall be displayed to all users with details regarding how the application can be used.

Priority 2

Registered User:

1. A registered user can create a grocery (to buy) list.

Website and Admins:

1. Search parameters shall be search

Priority 3

Registered User:

1. A registered user can upload recipes.
2. A registered user can maintain pantry.

Website and Admins:

1. An administrator can update delete recipes.
2. An administrator can update delete users.
3. The application shall have pages for ingredients, type, and availability in the pantry, also the list of stores from where the order can be made, calorie count and nutritional information.

4. The application shall have a recommendations Tab where we will store recommended recipes.
5. Registered users shall be given the option to set default preferences such as their price range, pantry, and favorite recipes.
6. All (registered and casual) users shall be allowed to filter search results by type of prices ‘low to high’, ‘high to low ‘and/or user.
7. All users shall be able to view a list of stores for an ingredient to buy with store rating and distance from pin code.
8. An ‘add to favorites’ option shall be displayed so that registered users can save recipes or ingredients to this list.

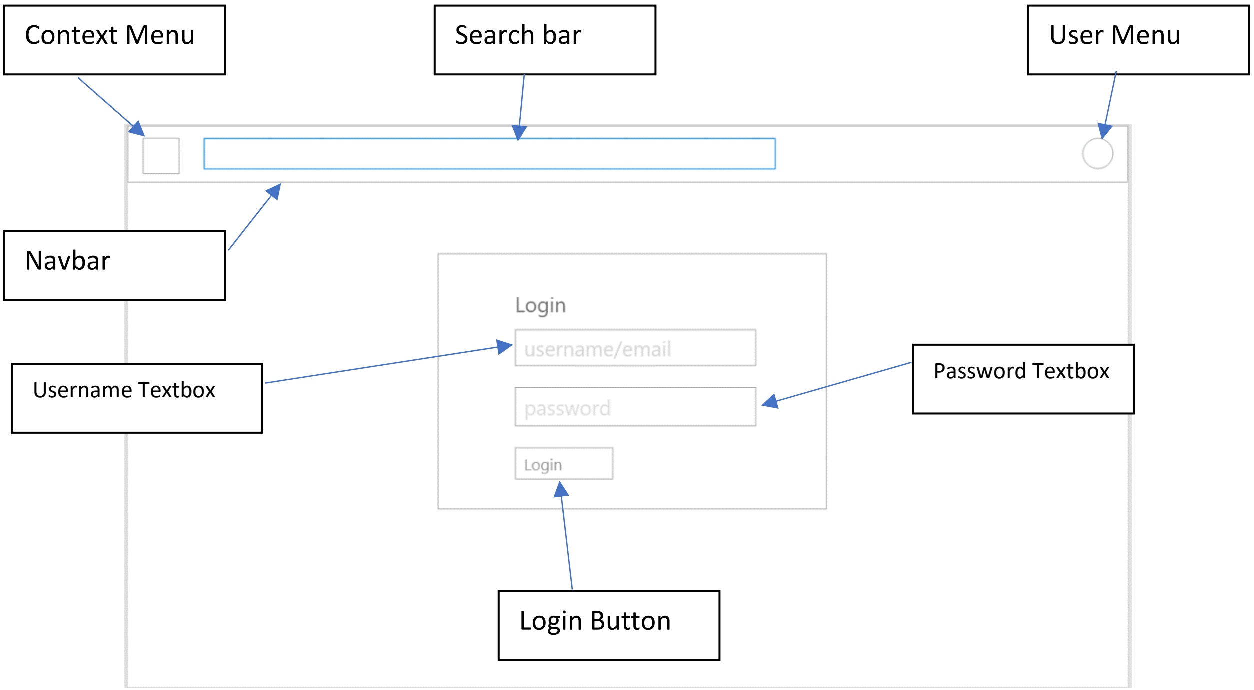
Excellent mockups. Really great job in this section.

3. UI Mockups and Storyboards

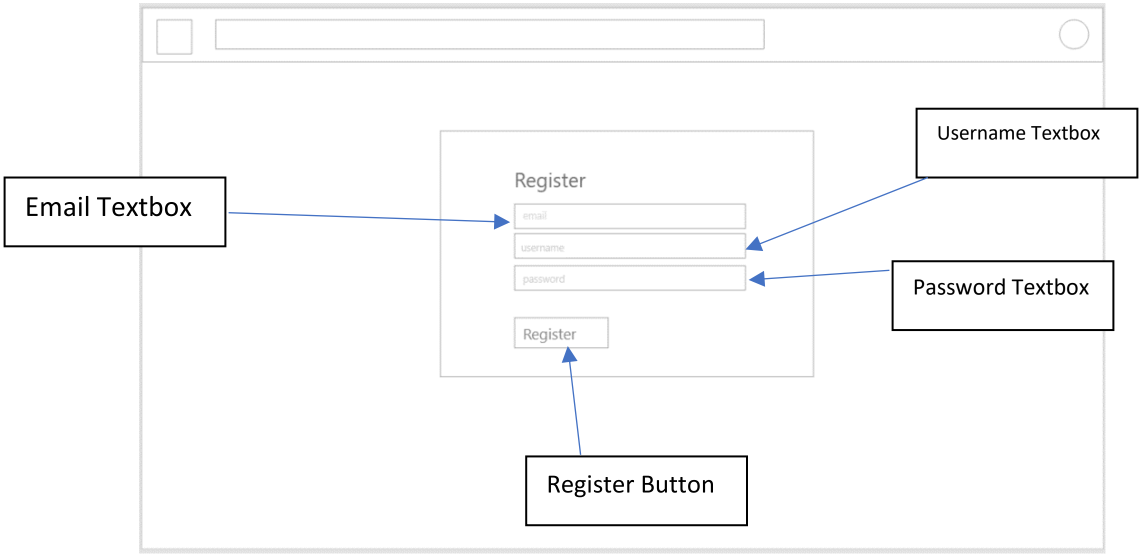
Mockups for UI

1. Login page should have a 'recover password link'.
2. It is more natural for user experience to have the login button aligned to the right

Login Page



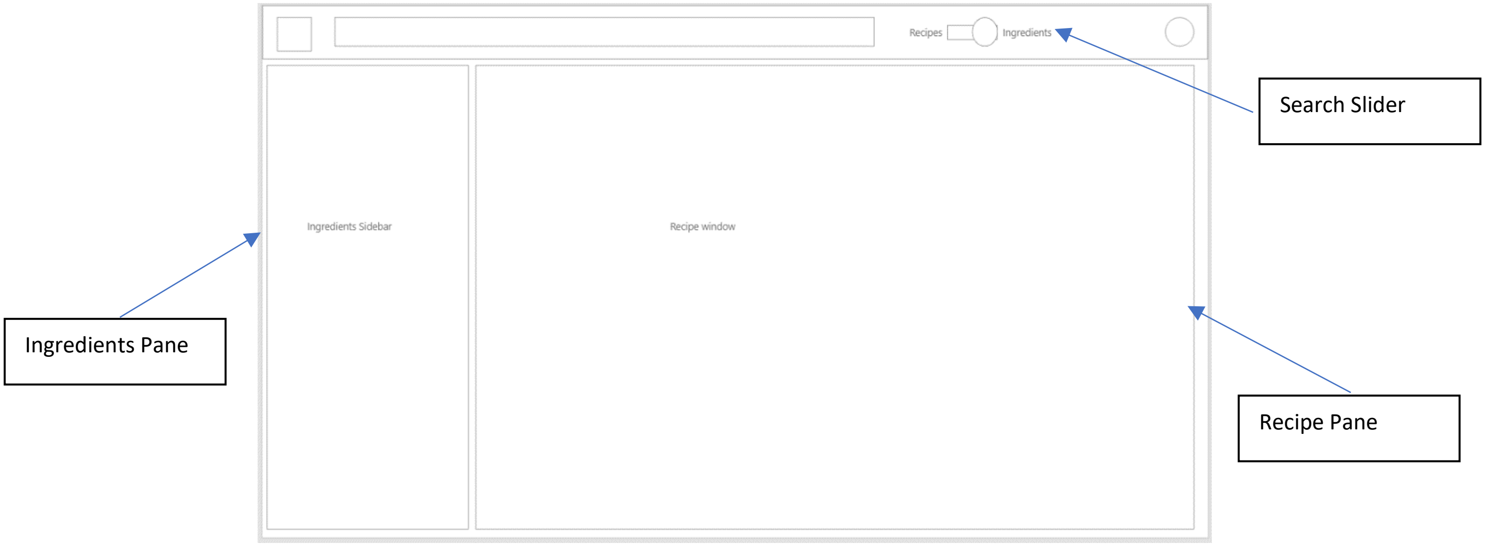
Register Page



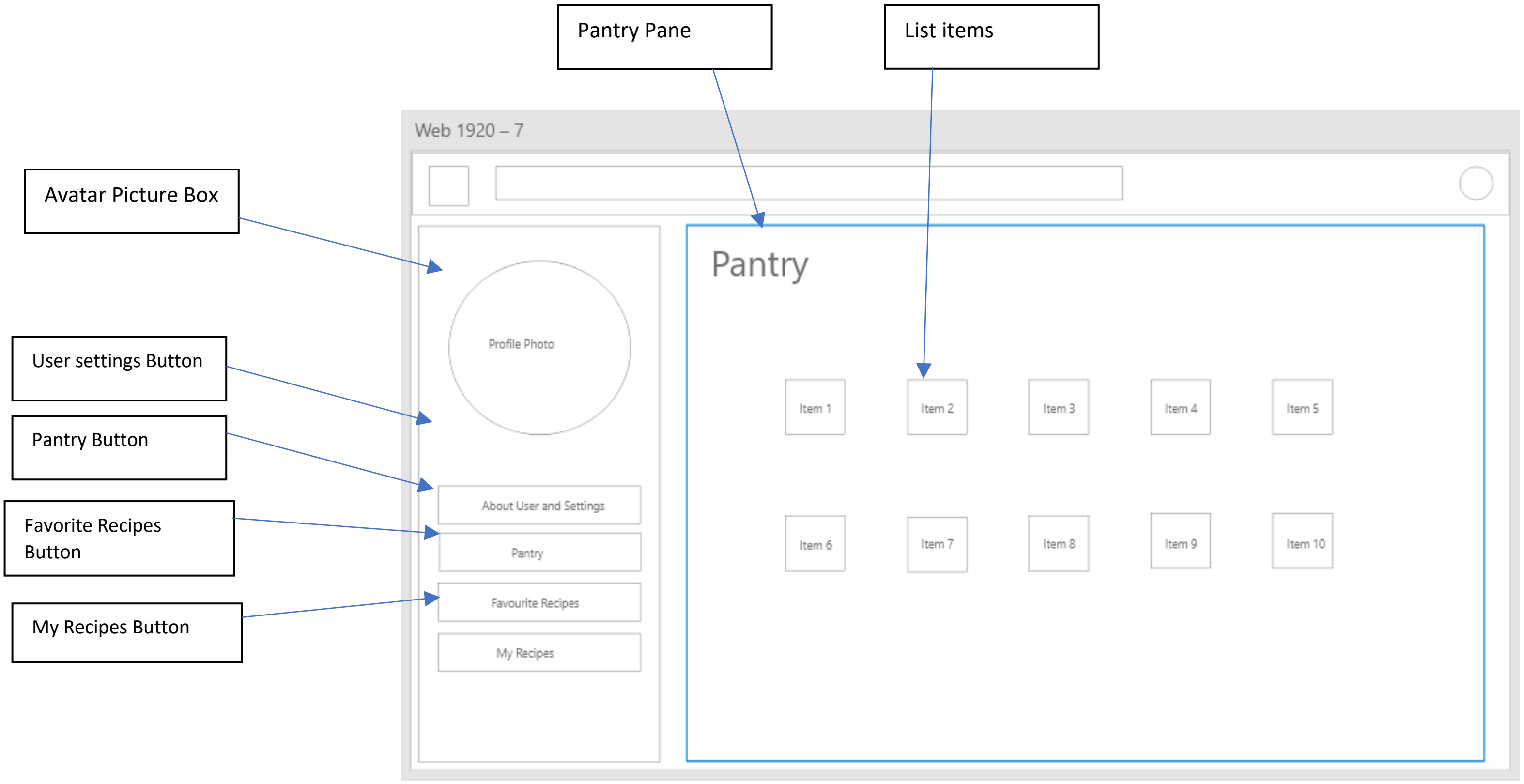
Register page should have a checkbox where user is informed that by registering, they accept the site policies. If they don't check the checkbox, then, they cannot register because they need to accept the site policies. Then, in your home page, add a link to your site policies.

I don't understand why you need a search slider

Main Page

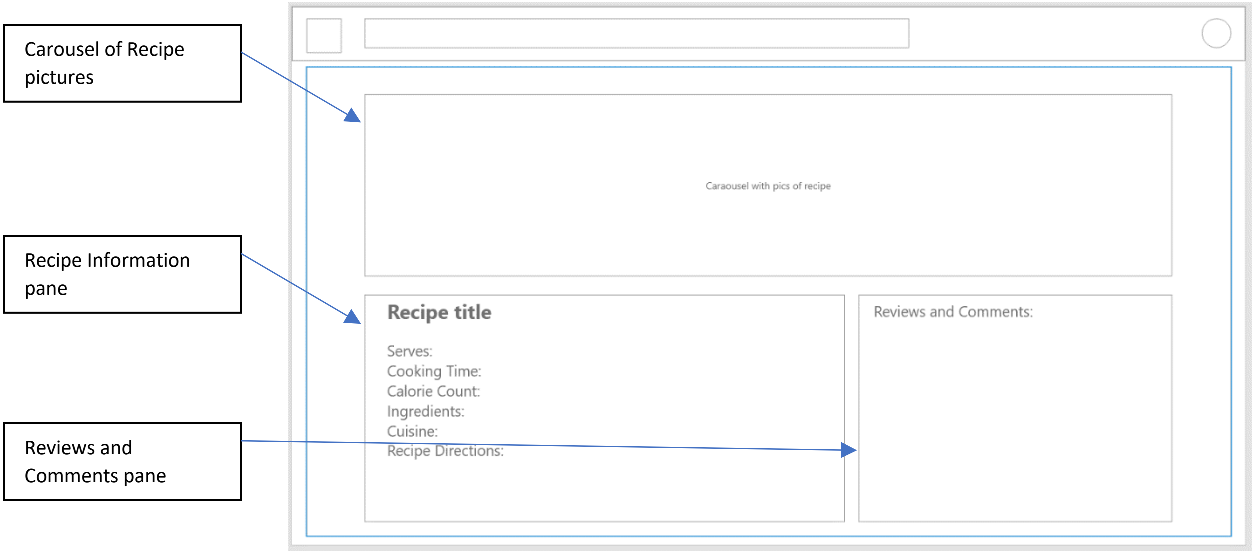


Pantry Page

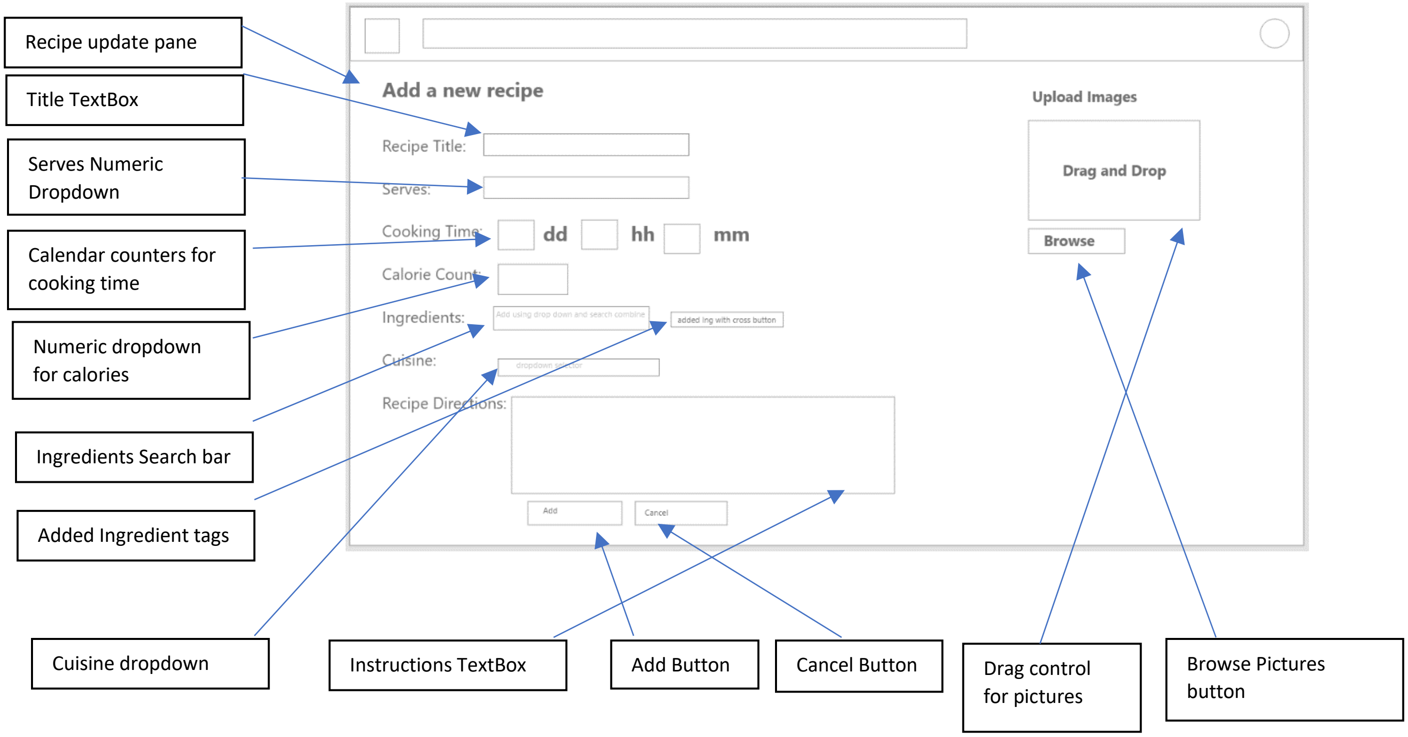


Good component alignment.

Recipe Page

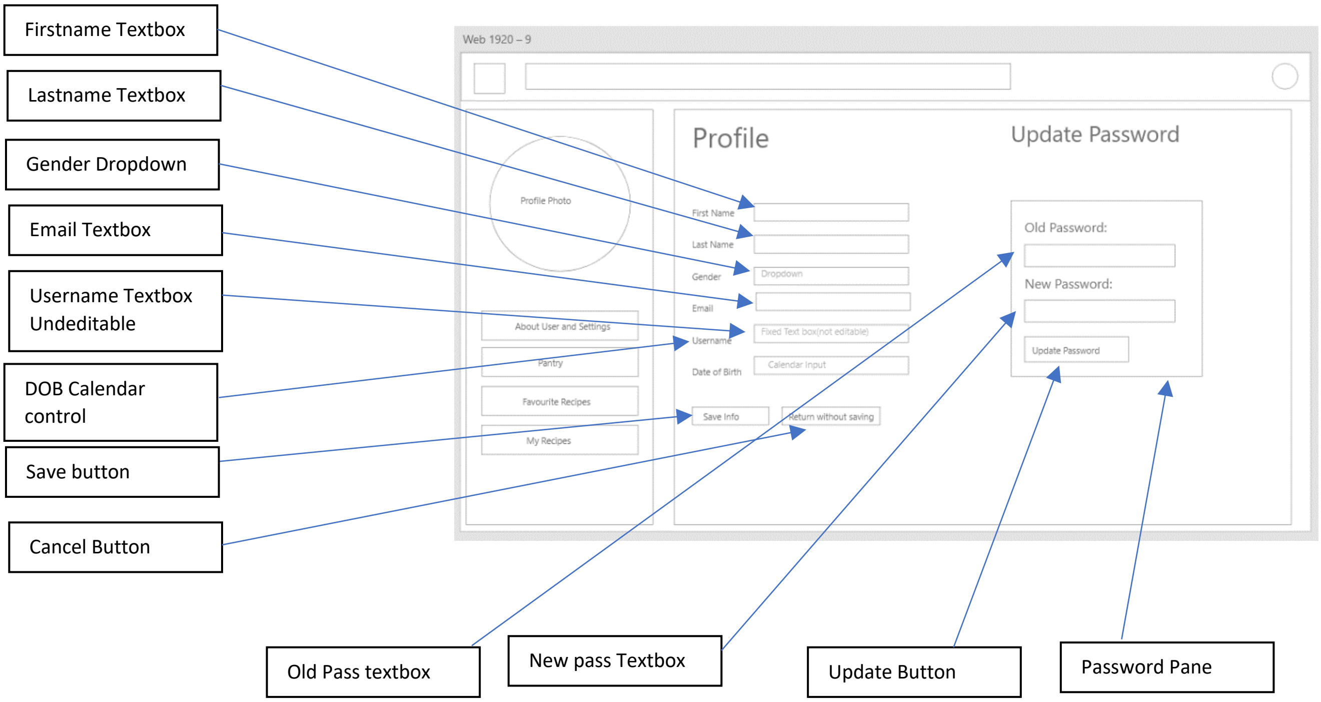


Recipe Creation Page (registered user only)

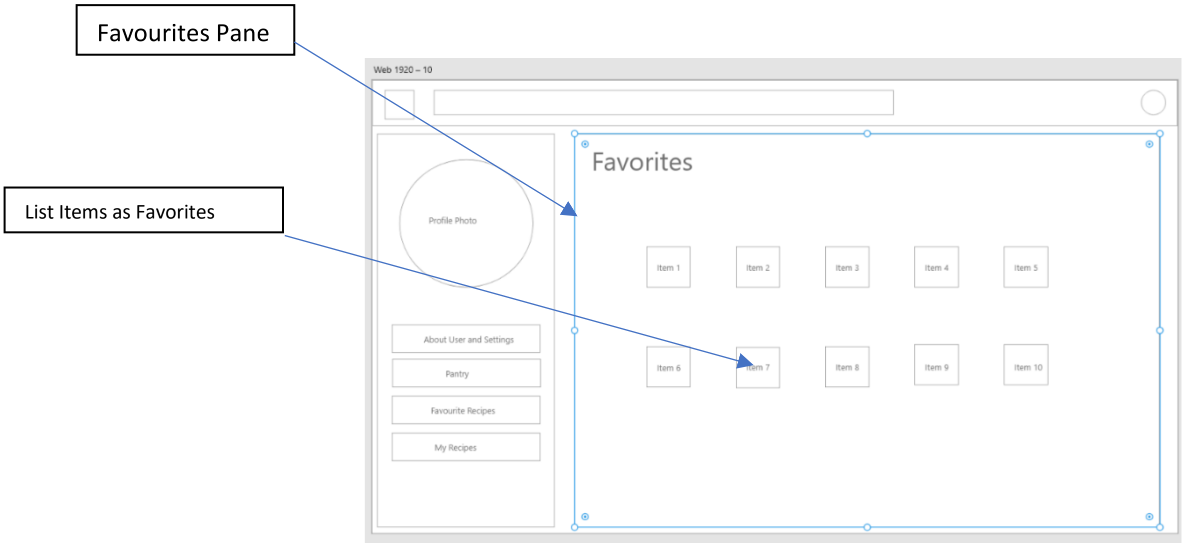


Really good

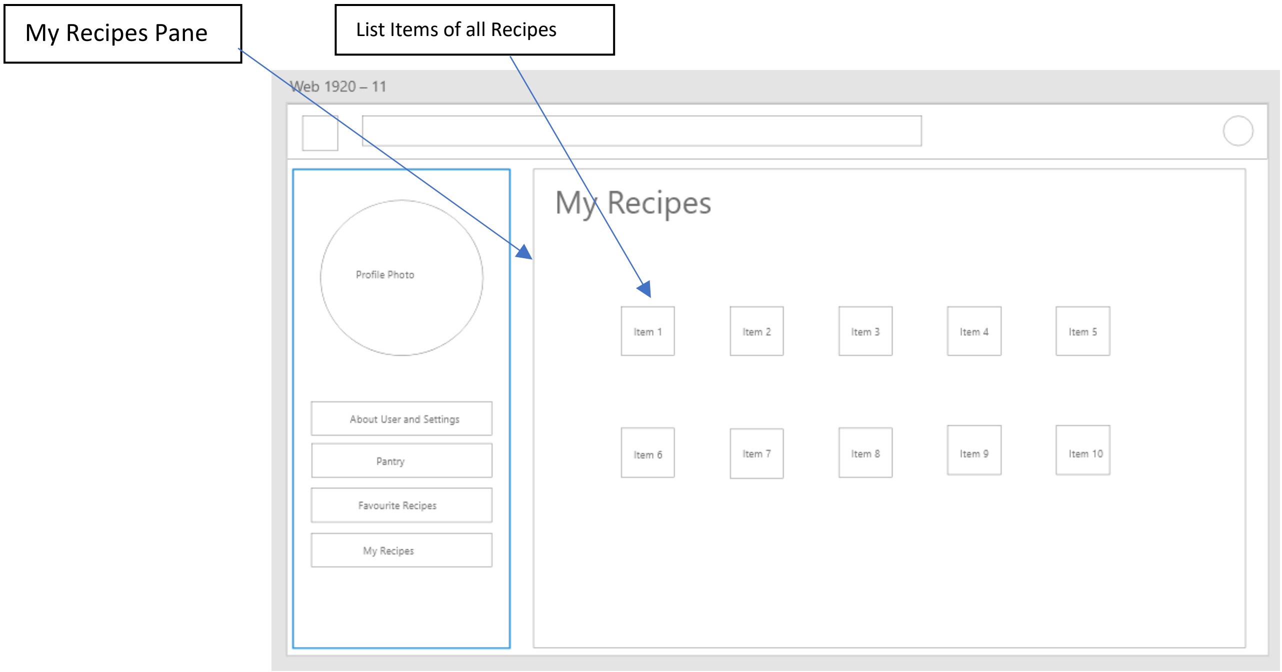
Profile Page



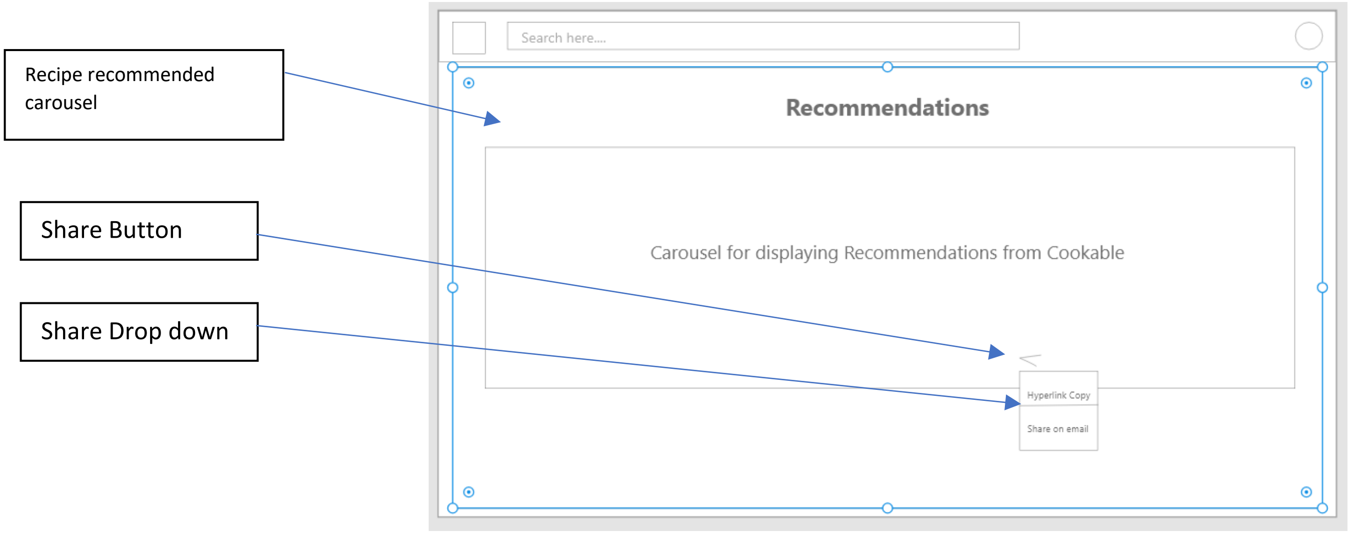
Favorite Recipes Page



My Recipes Page



Recommendations Page

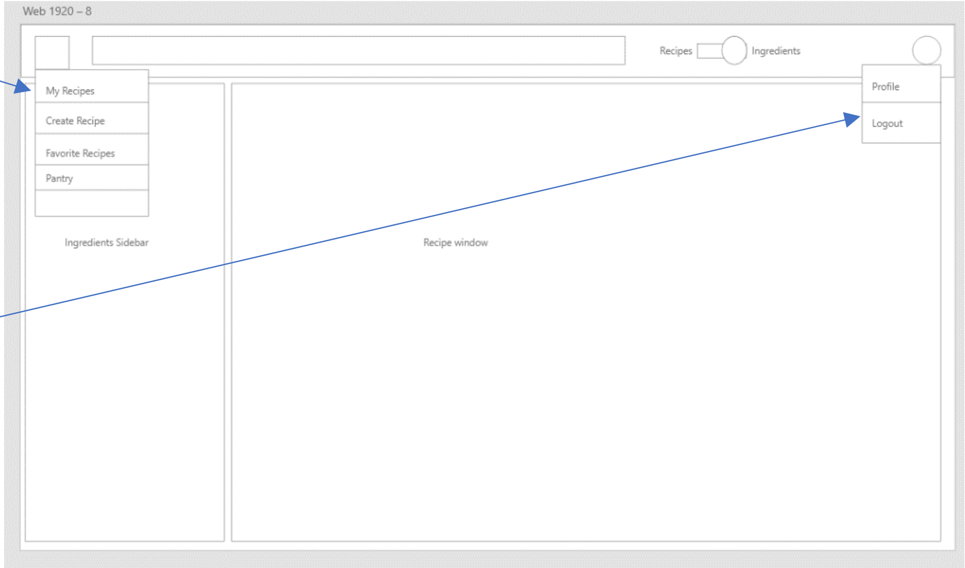


Menus on Nav Bar

Site Menu Dropdown

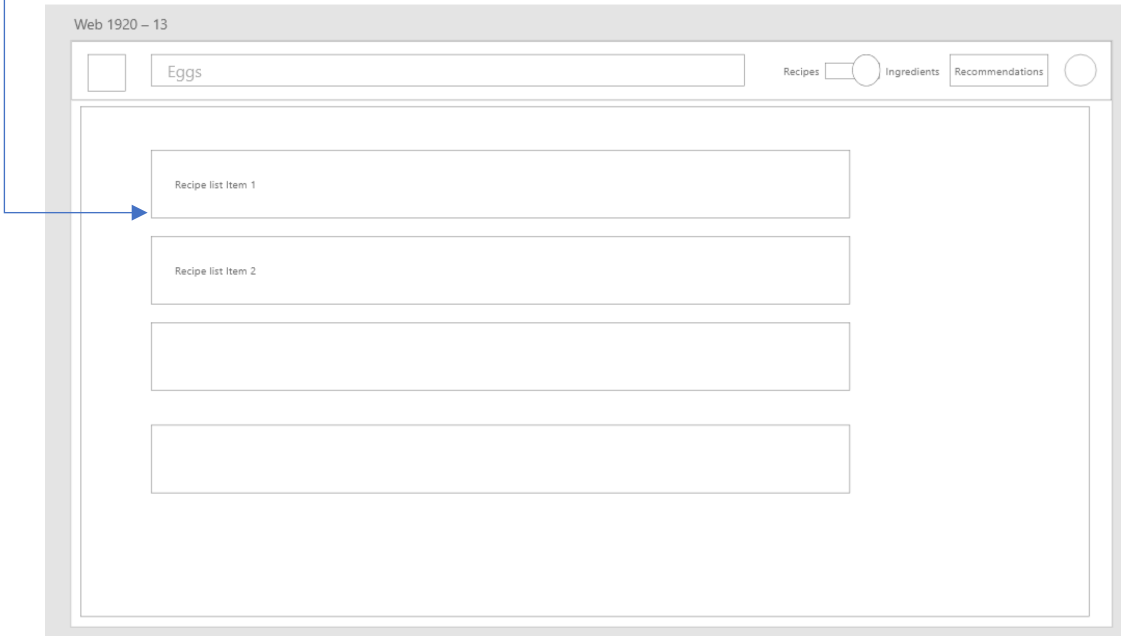
With Redirections

User Menu Dropdown
with redirections

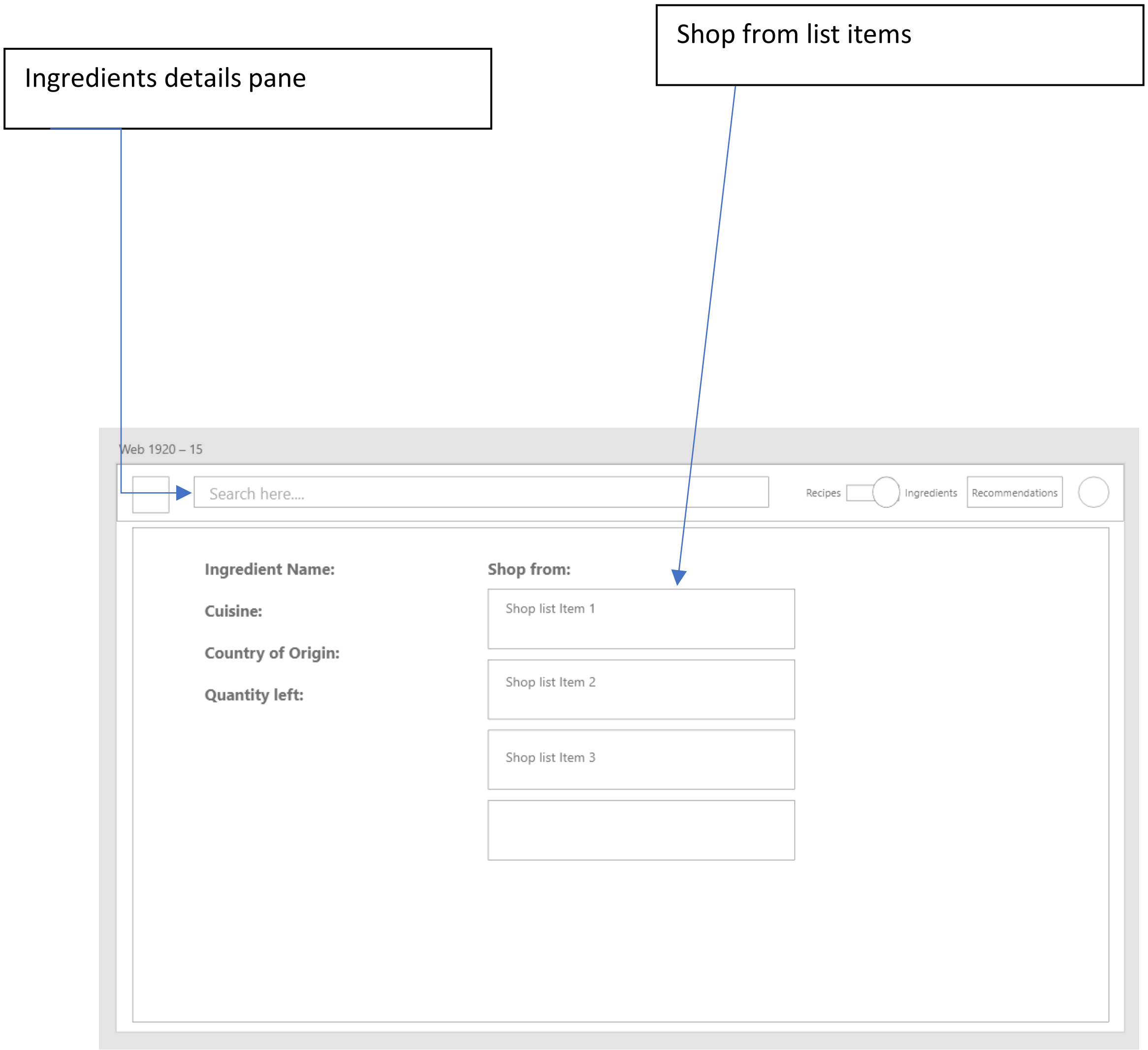


Search Window with results

Search results as list items



Ingredients Page:



This is excellent, you are the only team who implemented this part. This tell me how to test your prototype for M3 by following your transitions from page to page.

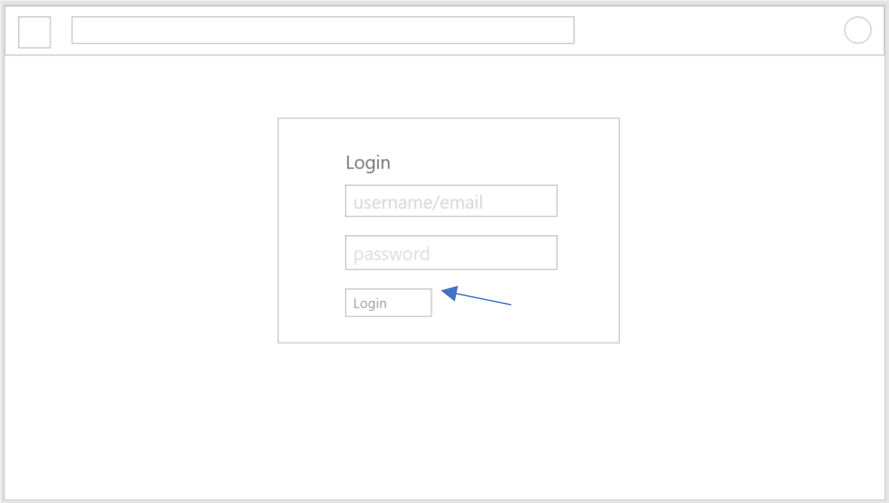
Storyboards

Casual:

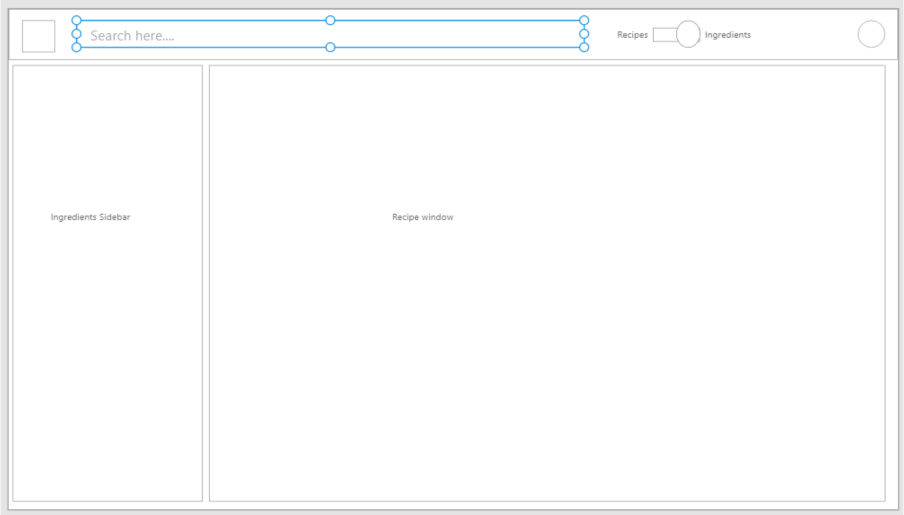
Emily:

1) A casual user, Emily, wants to cook something at home but she doesn't know what she can cook and is not familiar with recipes. She goes onto cookable to find possible recipes she can cook with the food only in her fridge. She finds that she can make a lot of different food with the simple ingredients in her fridge.

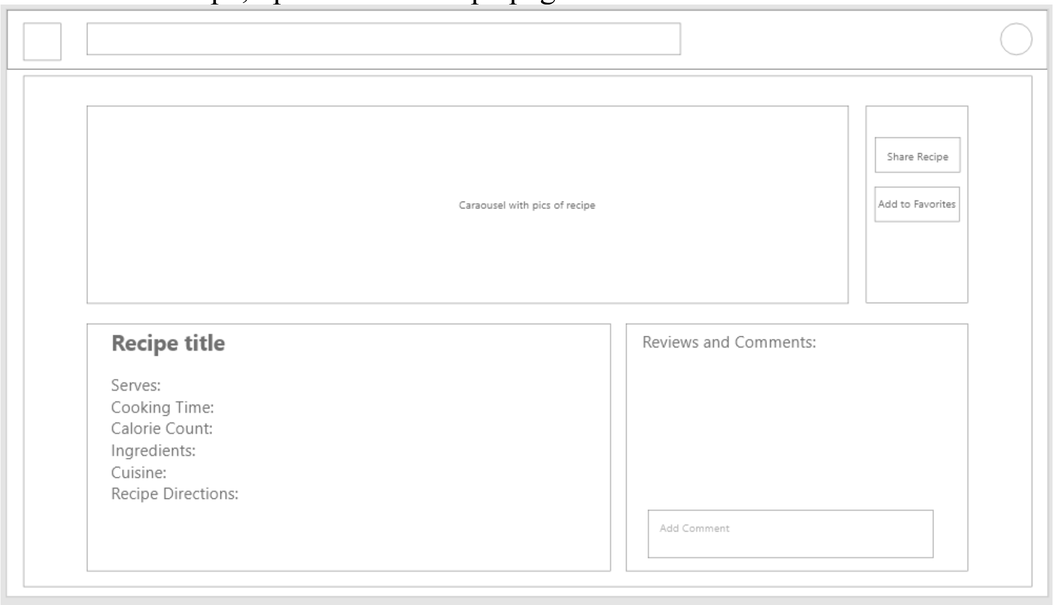
She Logs in to Cookable:



Searches for Recipes:



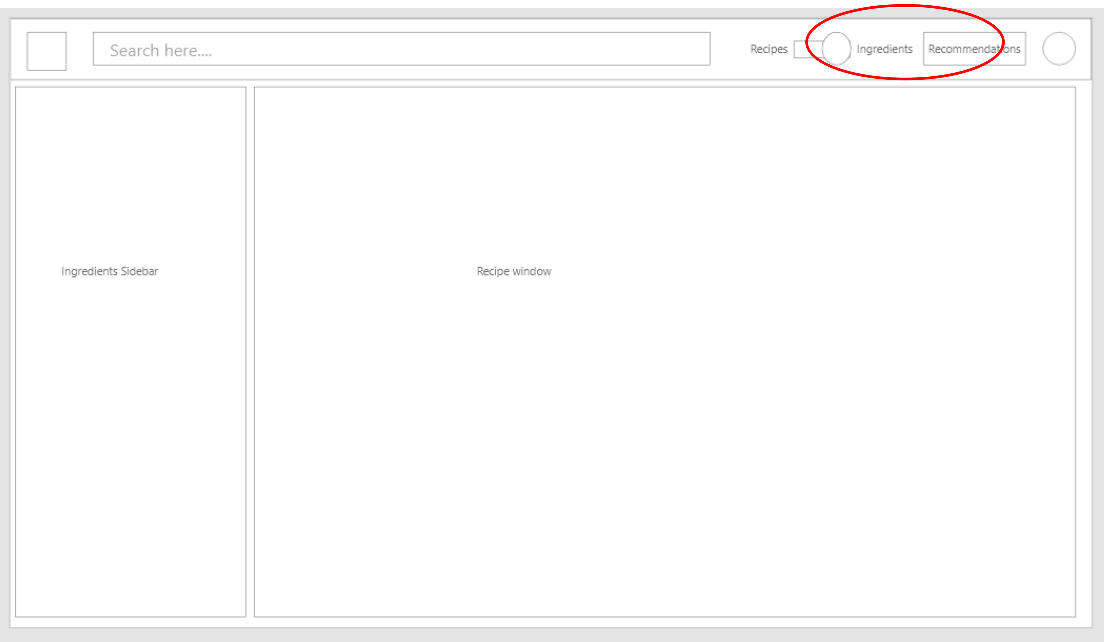
Clicks on a Recipe, opens a new recipe page

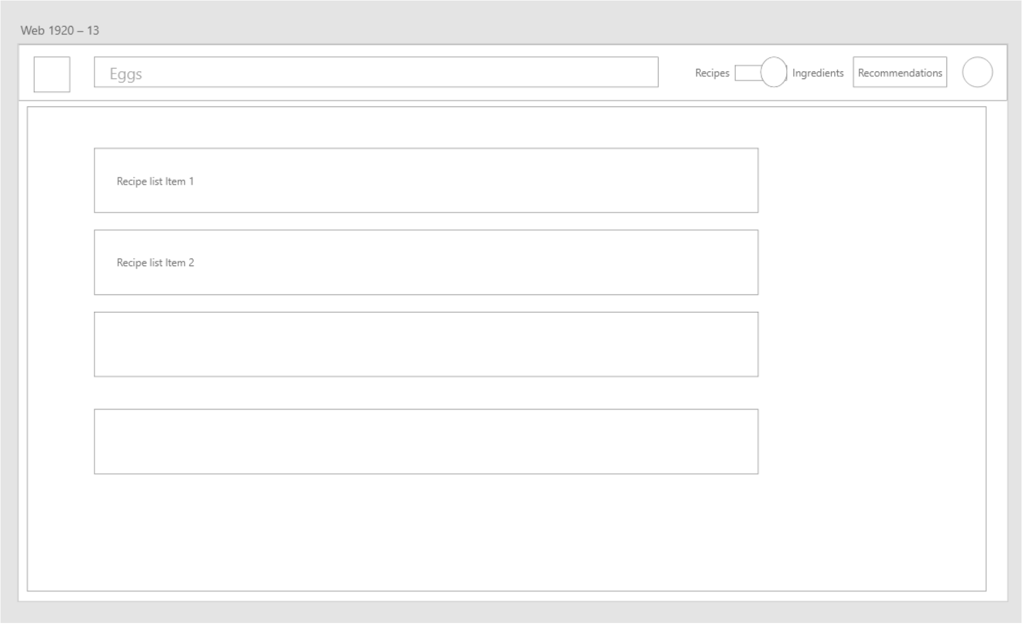


2) Emily is at the grocery store and she wants to cook something at home but doesn't know what to get. Her budget is limited as well so she can only buy cheap ingredients. She opens the website and enters in the ingredients she can buy, and it gives her all the recipes she can cook under a budget.

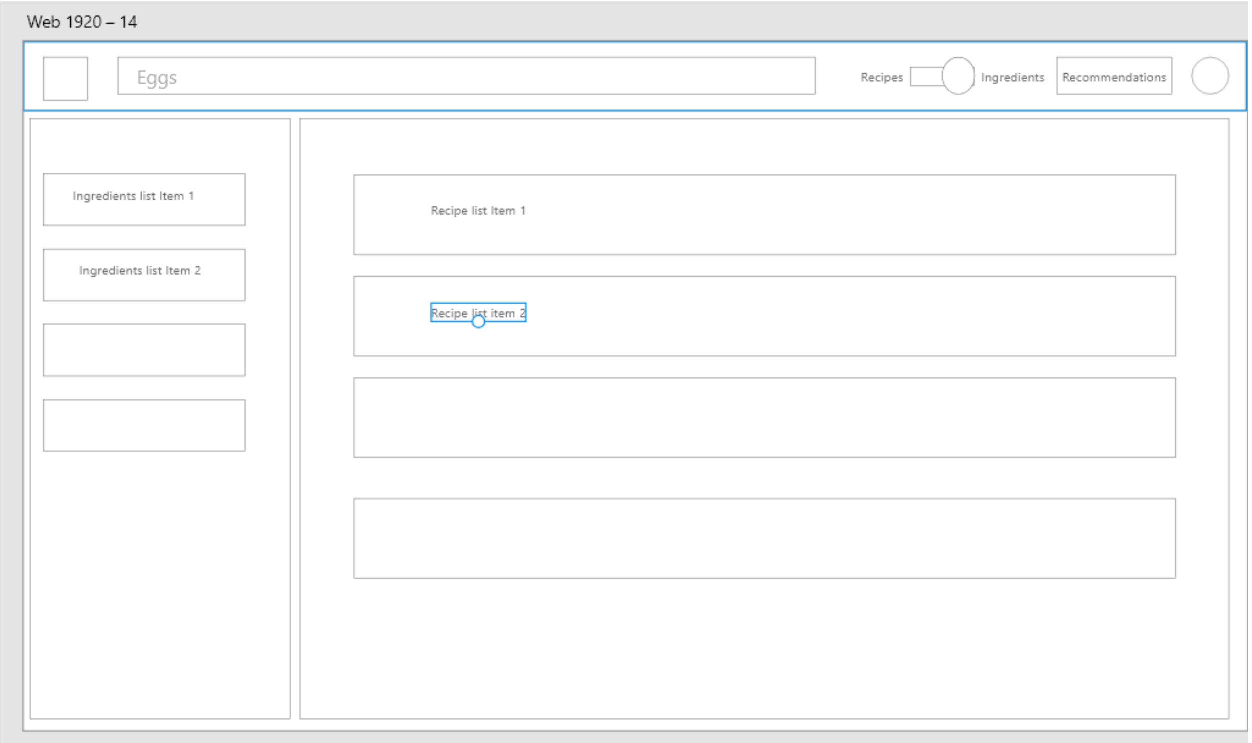
Slider Switched to Ingredients search

Search Results displayed as list items (All recipes for ingredients):





On clicking the ingredient, recipes appear

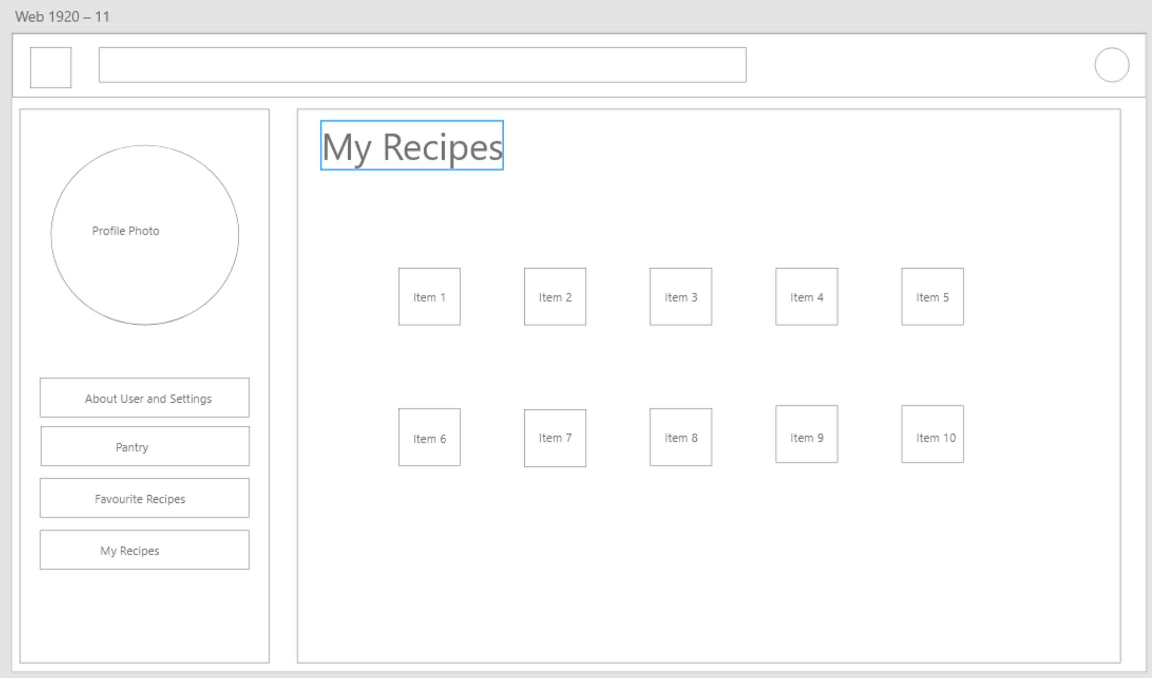


Registered:

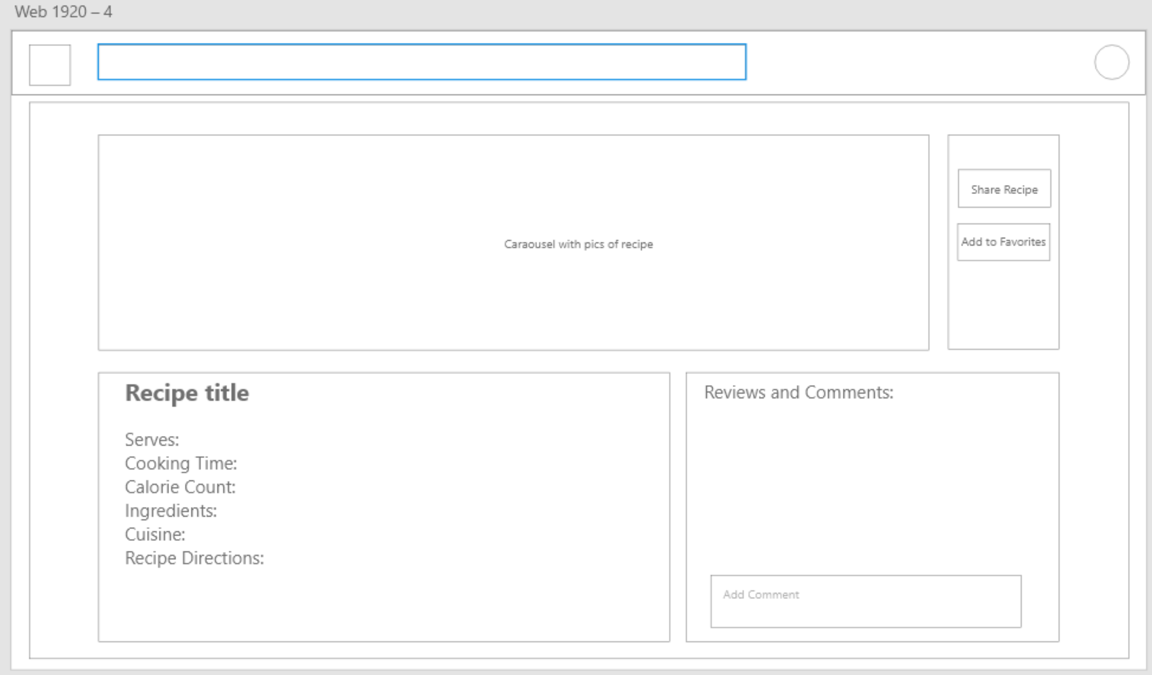
Emily:

1) Emily wishes to get a family recipe from Haley so she registered and asked Haley to do the same.

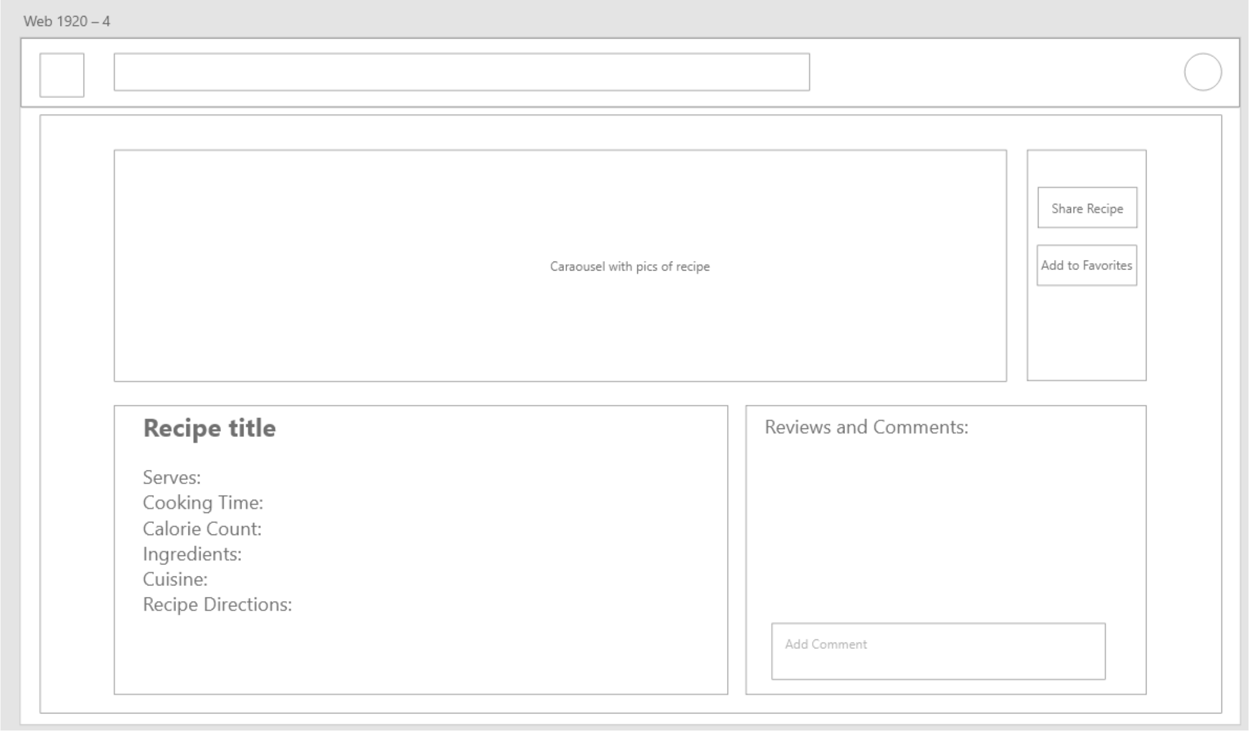
Haley opens her uploaded recipes page after logging in.



Selects a Recipe, shares it using Share Recipe button

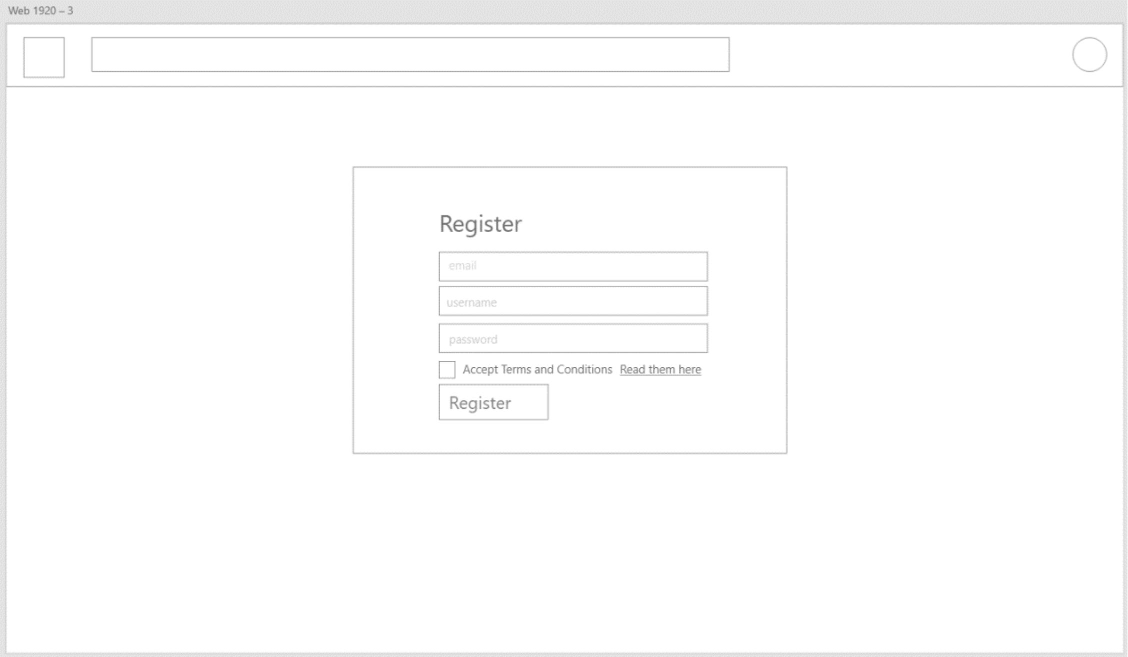


Emily opens the recipe.

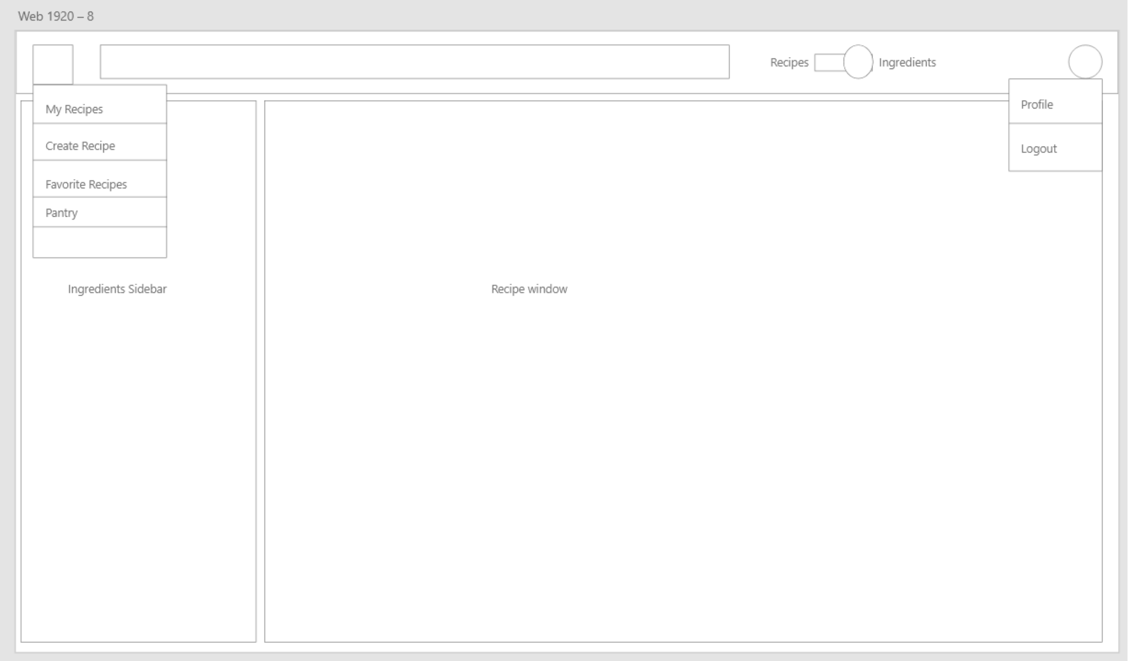


2) Haley decided to create an account with cookable because she wishes to share the family's recipes with her niece Emily to cook. She finds it easy to sign up and then uploads her recipe to cookable and sent it to Emily.

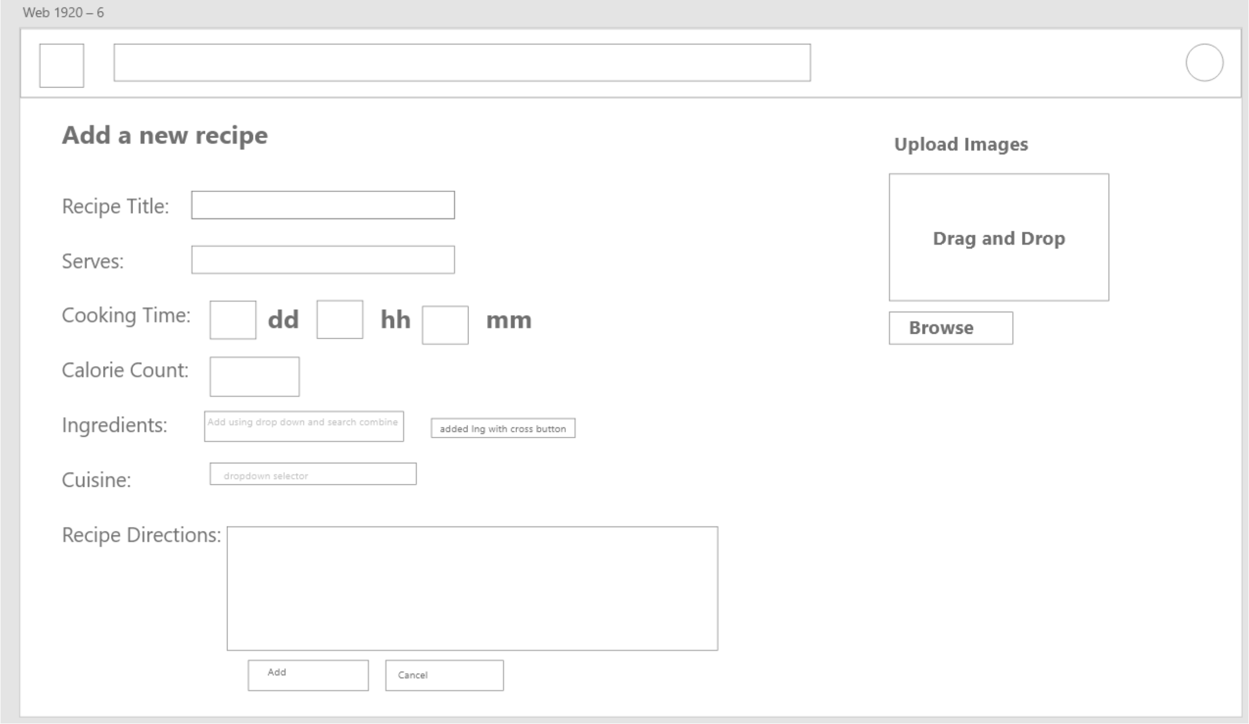
She opens cookable.com Sign up page.



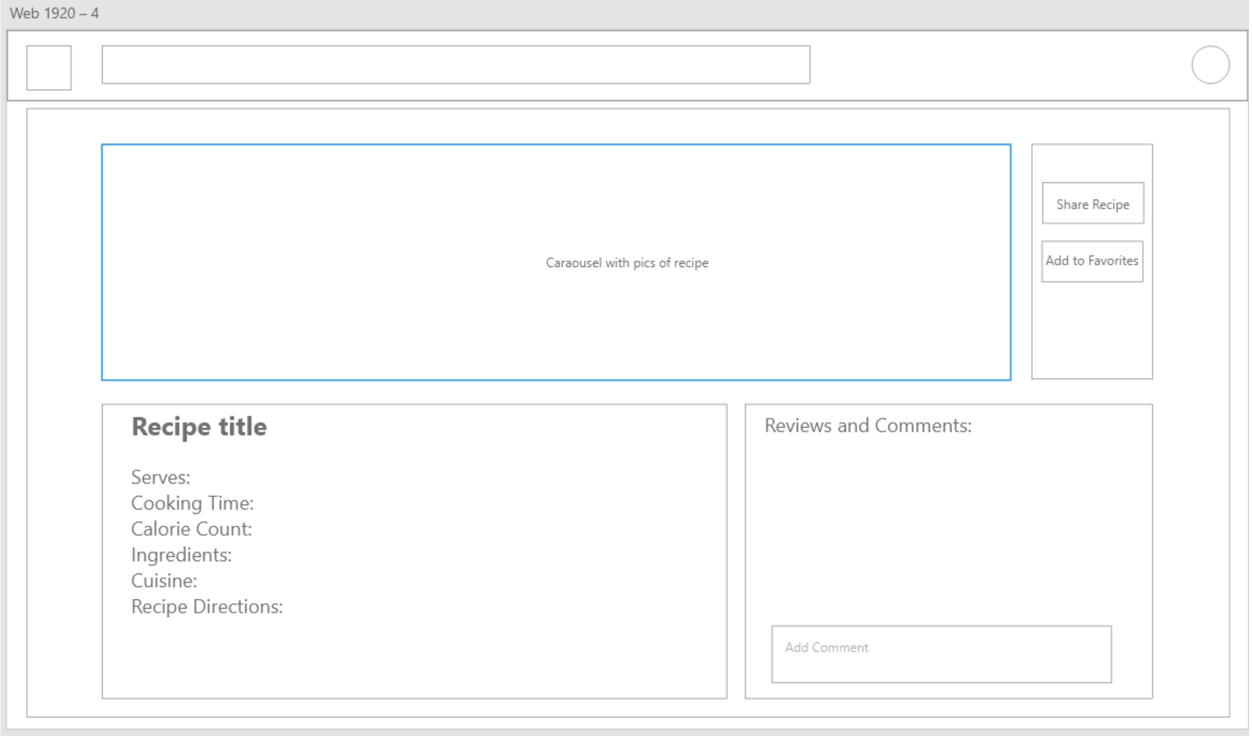
Registers herself and the site redirects to home page.



Uploads her recipe using Create Recipe option.

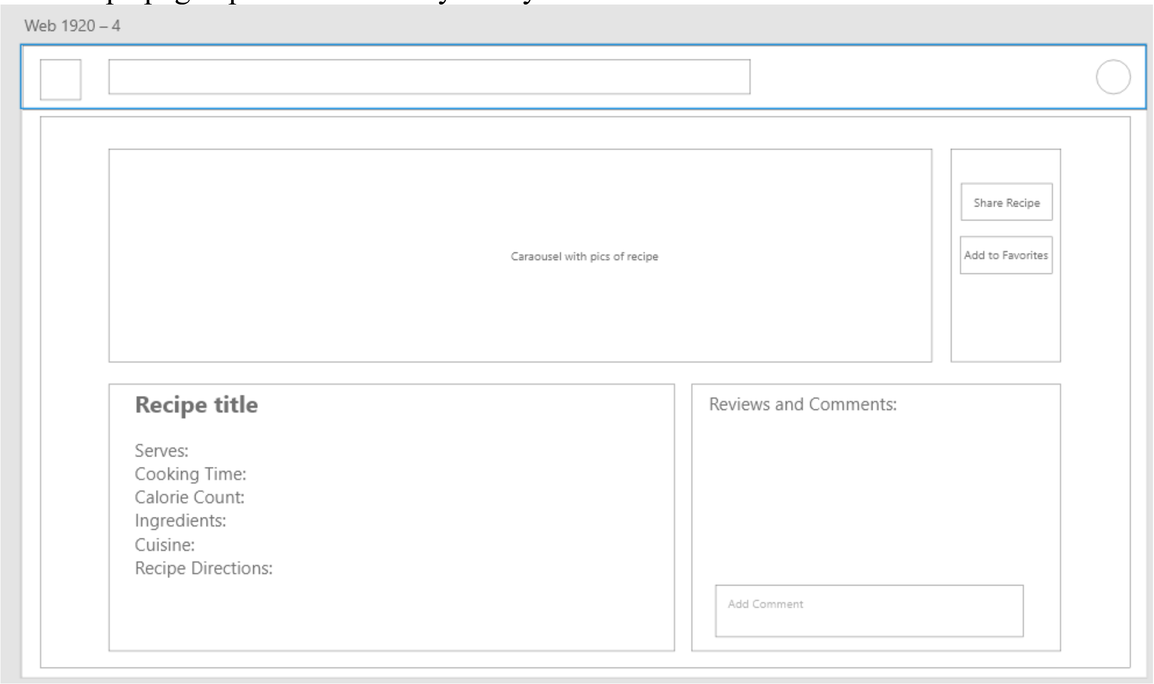


Shares her recipe from recipe page.

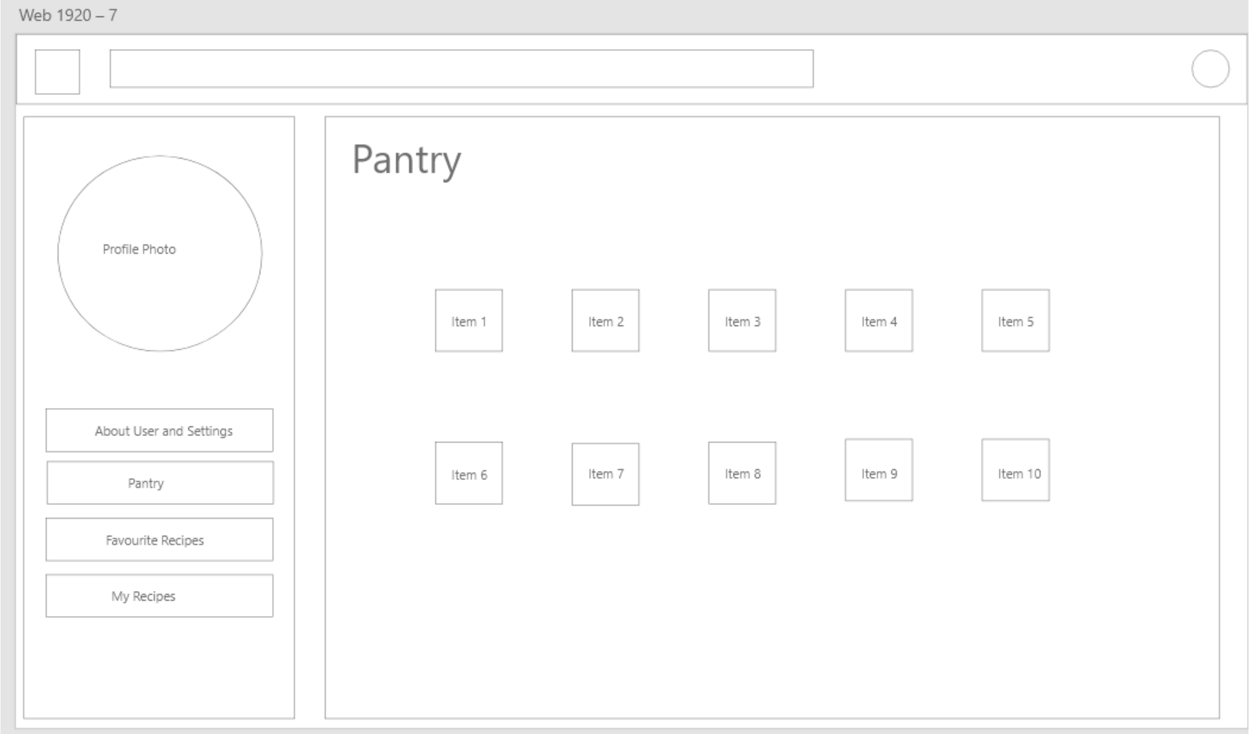


3) Emily receives the recipe with the instructions and amount of ingredient needed to cook her family’s recipe and she has an easy to access grocery list.

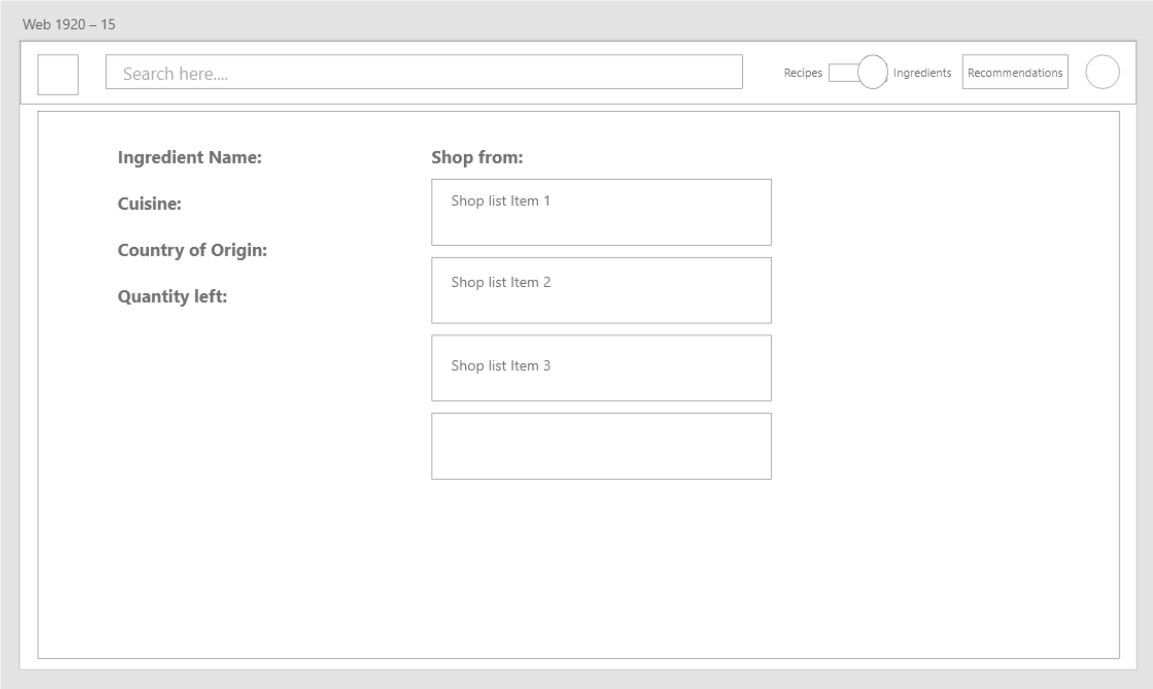
The recipe page opens as shared by Haley.



And the Pantry checkup using the pantry page.



Clicking on the ingredient will take you to ingredient grocery list/to-buy page.



4. High level Architecture, Database Organization

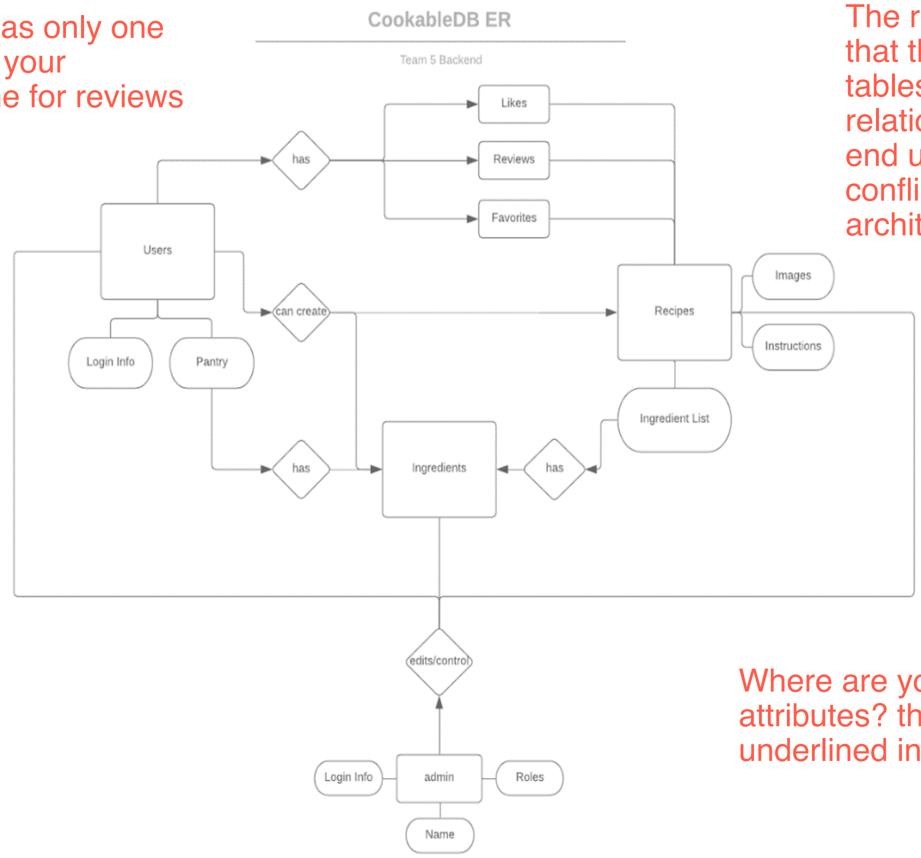
Business Rules:

1. Users shall be able to view and search recipes normally.
2. Users shall be able to search recipes based on ingredients.
3. Registered users shall be able to create and update recipes.
4. Registered users shall be able to comment, like, or favorite any recipe.
5. Registered users shall be able to add ingredients to their pantry representing what ingredients users currently have access.
6. Registered users shall be able to edit their profile which includes currently their password and username.
7. Admins shall be able to create, delete, or update any ingredient, user, or recipe.

You cannot have different relationships with the same name. They must be related in some way. The reason behind it is that they might become tables in M:N relationships and you'll end up with a lot of conflicts in your db architecture.

DB fig. 1 - Database ERD

according to this, user has only one review. That is not what your bussines rule says, same for reviews and favorites.



Where are your primary keys attributes? they need to be underlined in your ERD.

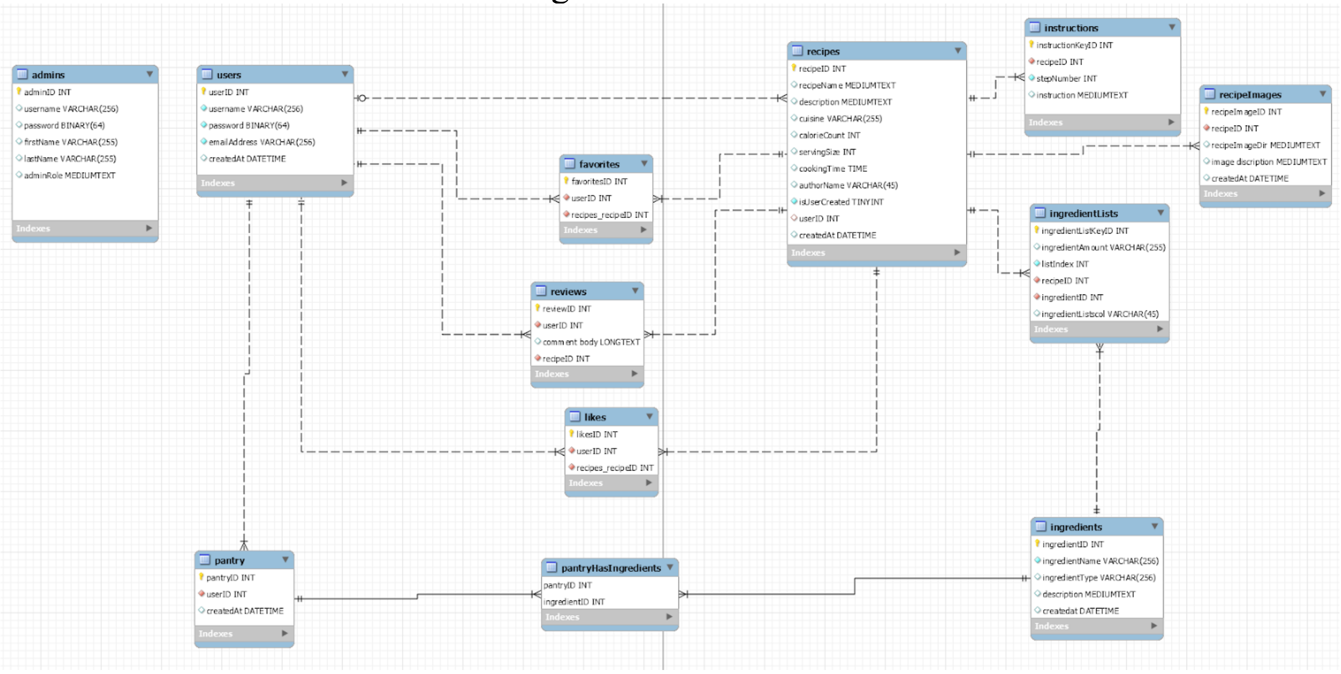
This section needs a lot of work. First of all, those are not bussines rules, they are functional requeriments. Bussines rules contain: one or more entities, cardinalities e.g only one, multiple, at least one, at most one....., and they also contain conditions which are optional. An exemple of bussines rule for your #4:

1. Registered users can add multiple comments to any recipe
2. Registered users can add multiple likes to any recipe
3. Registered users can add multiple updates to any recipe

So one rule per line. Your ERD has a lot of 1:1 relationships that I don't see in your bussines rules. If you need help with this section, come to my office hours and we'll do it together. So you can learn how to do it correctly

DB model needs to be modified to meet your new ERD

DB fig. 2 - Database model



DBMS: MySQL

We are using MySQL since it used very commonly in the industry and can easily translate our relational database needs and includes most features offered in other DBMSes like PostgreSQL or even NoSQL features like in MongoDB.

Media Storage:

Any media like images will be stored in an Amazon S3 instance with file paths specified in the database. Mostly images like PNGs and JPEGs will need to be stored and not much else.

Search/Filter Architecture and Implementation:

The search will be implemented using MySQL's "LIKE" function for most simple searches. More complicated search can also be done with regular expressions. Any search request will come in the form of a GET i.e website/api/search?s=curry. All recipes that match with curry will be returned. A more complicated search will be implemented where keywords will be matched with a recipes food type. In the example above multiple recipes can be categorized as "curries" and those can be returned.

NonTrivial Algorithms:

The search algorithm for recipes will be nontrivial. Search terms may include ingredients, recipe names, author name or even calorie count for ease of use. This will need a robust search algorithm that is fast and can predict in some sense what a user wants. The recipes returned will also need to be ranked and prioritized based on measures like the number of comments, favorites, or "likes".

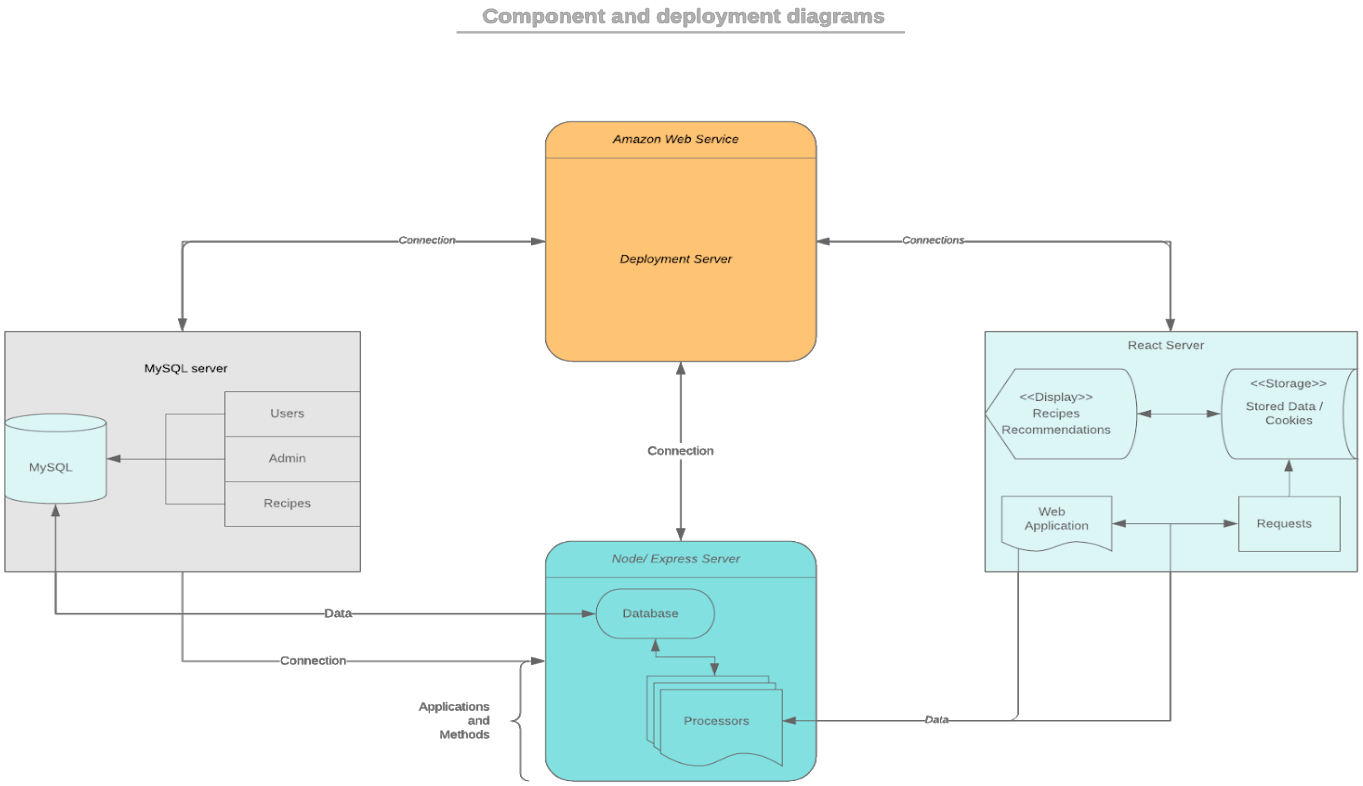
Additional Technologies:

- **SequelizeJS** to create models for tables within the database so the data can easily be accessed without the need for SQL queries to be written.
- **PassportJS** for login to create sessions and cookies so users do not have to log in every time they visit the site.
- **PBKDF2** for encryption of passwords and storage.
- **Amazon S3** for media storage and access.

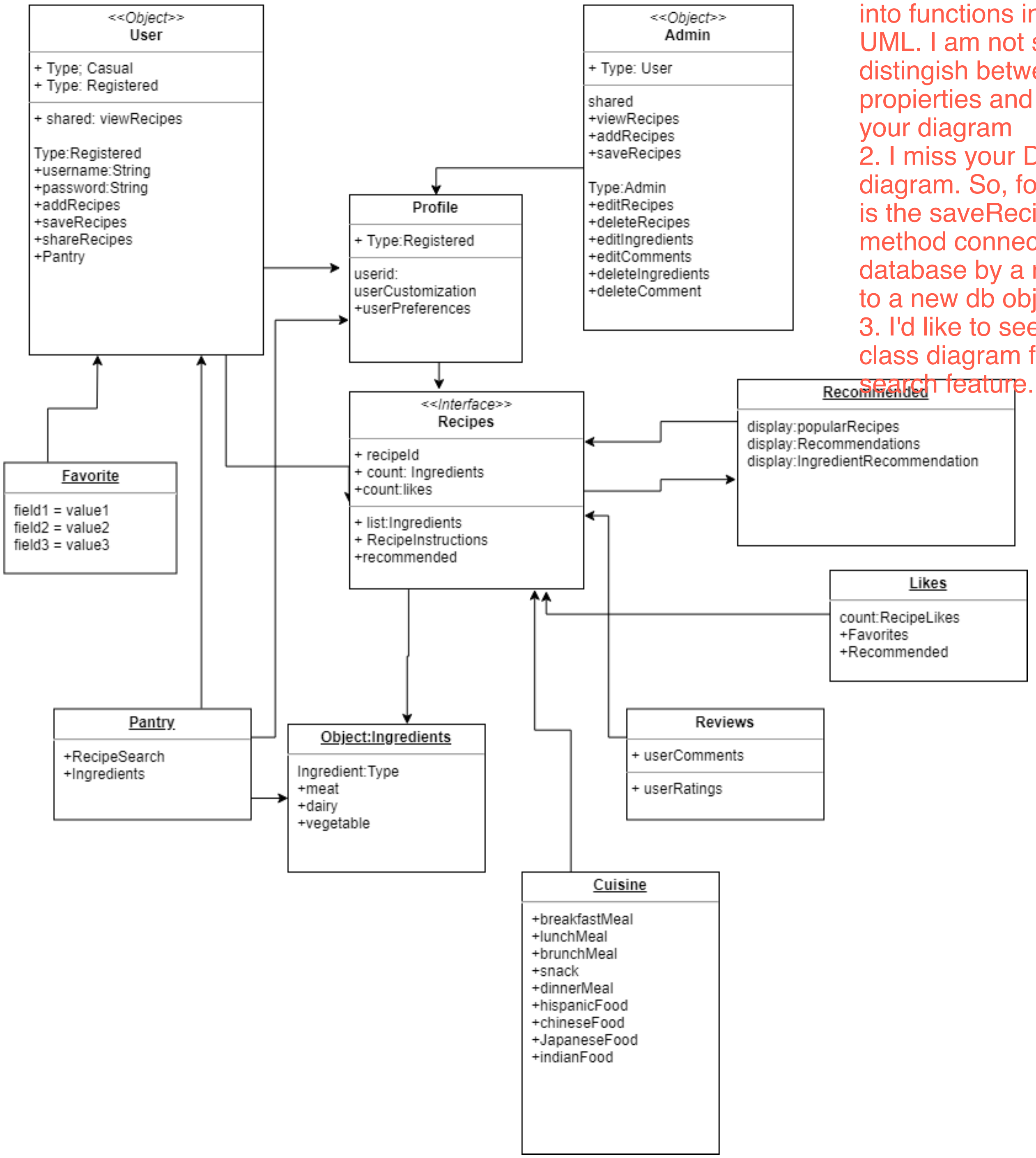
This is really good work.

5. High Level UML Diagrams

UML Component and deployment diagram



UML Class Diagram



These are good UML diagrams. But I am missing three main things here:
1. I need to know if your methods are returning anything, and some basic parameters that you pass into functions in your UML. I am not sure how to distingish between propierties and methods in your diagram
2. I miss your DB class diagram. So, for example, is the saveRecipes method connecting to the database by a reference to a new db object?
3. I'd like to see the UML class diagram for the search feature.

6. Key Risks for Project

Skills Risks: React

The main concern is lack of experience with React.js framework. All team members are currently putting in extra practice to prepare for Milestone 3. Our application is being designed with this concern in mind so that the implementation of the front-end will go smoothly.

Schedule Risks: Application Functionality

Due to the lack of experience with React.js framework, we may need to cut down on the functional requirements of the project. In order to stay on schedule, we will put a huge focus on functionality prioritization so that in the event that we need to cut functionality, we will still maintain the most important aspects of the application design.

Technical Risks: React state management, Image storage, Recipe Data

Two technical risks include how we are going to handle state management in the React front-end portion of the project, and how we are going to manage image storage in the MySQL backend. We plan on doing research in order to determine the best approaches to handling these two problems. We also need to solve the problem of how we are going to obtain all of our recipe data. We are currently looking into ways to avoid recipes into our database.

Teamwork Risks: Attendance

The main issue with teamwork has been attendance related. Aside from that team members have all been contributing and communicating well. All issues related to previous milestones have been brought up to the professor and will be handled accordingly.

Shedule Risks: Ok now, I perfectly understand why you were playing safe with priorities #1. I can see now that you were playing safe thinking on staying on schedule. I like that way of thinking. But at the same time, you need to challenge yourselves in this class. Why? because out there in the software industry, this happens all the time. You are assigned a project, and after discussing the non-functional requirements, the team lead or manager thinks that a specific technology that the team has no expertise with, is more suitable to solve a specific problem. Then, the whole team needs to learn about that technology, and at the same time, stay in schedule, and keep good quality standards in the project.

Tecnical Risk: For prototypes and final grading of your project, you only need to have some small sample data in your database, and a mechanism to insert data dynamicaly. You don't need to have a huge database with thousends of recipe data entries.

Good

7. Project Management

For milestone 2, we managed the distribution of work much like we did with milestone 1. We split the work up evenly based on front and back-end tasks. For example, Database management and Data entities were handled by the back-end team members, while UI Mockups were handled by the front-end team members. Our team has been using Trello for both milestone 1 and 2 to keep track of tasks that have been assigned to each team member. We currently have three boards set up on the website that will allow us to manage our code development in an organized and timely manner. We have our respective front-end and back-end boards which will allow us to keep track of tasks without the distraction of worrying about what the other team members are working on. Front-end members will focus on front-end, while back-end members will focus on back-end. We also have a 3rd board which is what the two teams will use to communicate their needs to one another. For example, if the front-end team needs to receive a piece of information from the database using a GET request, they can create a post explaining in great detail what they require from the back-end team. This way, both teams can continue to develop without needing to wait for the other team to finish their portion. Hopefully this structure will allow us to develop our application in a timely and structured manner. Regarding communication we will continue to use Discord as our message board seeing that it allows us to keep message logs as well as host group phone/video sessions.