“SW Engineering CSC 648/848 Fall 2019”

Milestone 2

Team 104

CERES

Member Name	Member Role
U Khyoi Nu (unu@mail.sfsu.edu)	Team Lead
Srushti Buddhadev	Backend Lead
David Fang	Frontend Lead
Darnell Kenebrew	Github Master/Backend
Brian Lopez	Scrum Master/Frontend
Jonathan Munoz	Frontend/Backend

History Table
October 25, 2019, version 1.0, First Draft
December 16, 2019, Version 2, Second Draft

1. Data Definitions V2

Users:

- This class of data definition includes all the users that have registered themselves on this application.
- The user represents all the roles that people play for different purposes of using this application.
- The user can make use of many functionalities other than grocery shopping. They can make their own group and request other known users to join the group. For example, a user can have different groups for the workplace, home, and other miscellaneous purposes.
- The user can also access a shopping list in whichever group he wants.
- In this application, whenever a user gets registered, a group is created by default where the user is the only member. Later, he can add other users, or get himself added into another group, in which case his group code will change.
- Also, if the inventory has the necessary ingredients, recipes matching those ingredients are suggested to the user, and the link to them is provided.
- The user has the flexibility to either add items in the inventory by uploading images or manually.
- Attributes:
  1. Id (primary key)
  2. email
  3. name
  4. password
  5. type

Fridges:

- When a user gets registered, he is assigned a group or also known as a Fridge by default with a unique group code and group name.
- Whenever the user has the need to communicate for groceries and other items in a group, he can send requests to link his group to other users of this application. Another case is where the user can request to join a particular group to the group maker.
- All the users in the group will have their own generated user Id, but also every group will have a unique group code and name.

- When a user creates a new fridge, the group code of the group will be the one assigned to that user, but when a user joins another fridge, his group code and group name will get changed to the one he joined.
- A fridge can have lists of their own which can be accessed only by the respective group members.
- Only the original maker can add or remove members from the fridge, no one else can send or accept the request.
- Attributes:
  1. Id (primary key)
  2. name
  3. code
  4. User_id (foreign key)

Inventory:

- The inventory here is analogous to the fridge. It stores all the current food items present along with their quantities, and expiration dates.
- The expiration date field, in this case, is calculated by adding the expiration age from the details of the food item's database, to addition-date, i.e. when the item was bought.
- There is one inventory per group and can be accessed only by the members of that group.
- Inventory details also play a role in suggesting recipes to the users, if more than a certain number of ingredients are available in the group's inventory for a recipe then that recipe will be suggested.
- Whenever an item expires, the status field shows that the item is no longer usable.
- The entries in the inventory table are entered using OCR on the images of the receipt or manually by the user. In the latter case, the image_id field is kept null, as it was not generated using any kind of image.
- Attributes:
  1. id (primary key)
  2. image_id (foreign key)
  3. food_id (foreign key)
  4. quantity
  5. group_is
  6. status
  7. addition_date
  8. expiration_date

Food details:

- This table has all the general and fixed details of an item, unlike inventory, where the data is a variable and frequently temporary.
- The inventory table is connected in various ways to this table, where the food Id is referred from this table.
- The inventory table uses the expiration age of the item and accordingly estimates, how long can the item be stored in inventory without expiring.
- Attributes:
    1. Id (primary key)
    2. name
    3. calories
    4. expiration_age

Recipes:

- The recipes table helps show or suggest recipes whose ingredients are available, and the user can click on the particular link to watch the whole recipe video.
- Attributes
    1. Id (primary key)
    2. name
    3. link
    4. ingredients

List:

- A list is where group members can jot down all the items they want to purchase as well as anything important to note down.
- A group can have multiple lists, but a particular list can only be accessed by the members of that group.
- There is also a field status, where if the item listed in the list is bought, it will be marked as done, and not pending.
- Attributes:
    1. Id (primary key)
    2. quantity
    3. group_id
    4. food_name
    5. status

Consumption Progress:

- Includes a target for calorie consumption, the current state of progress by a family(or individual), start date, end date, etc.

2. Functional Requirements V2

Priority 1:

General:

Users must sign in to use website services.

Registered user:

1. Users shall be able to
   1.1. Input information about adding an item to the refrigerator or removing an item from the refrigerator. (Read input from receipt)
   1.2. Ability to add items manually (Items not in the shopping list)
   1.3. Keep an inventory of what they have
   1.4. Search for recipes based on current items
   1.6. Create a shareable grocery list (Grocery List for a household)
   1.7. Be notified before items expire or are depleted
   1.8. Add into a shareable grocery list (Grocery List for a household/work)
   1.9. Able to purchase, new items independent of current items

Priority 2:

Registered user:

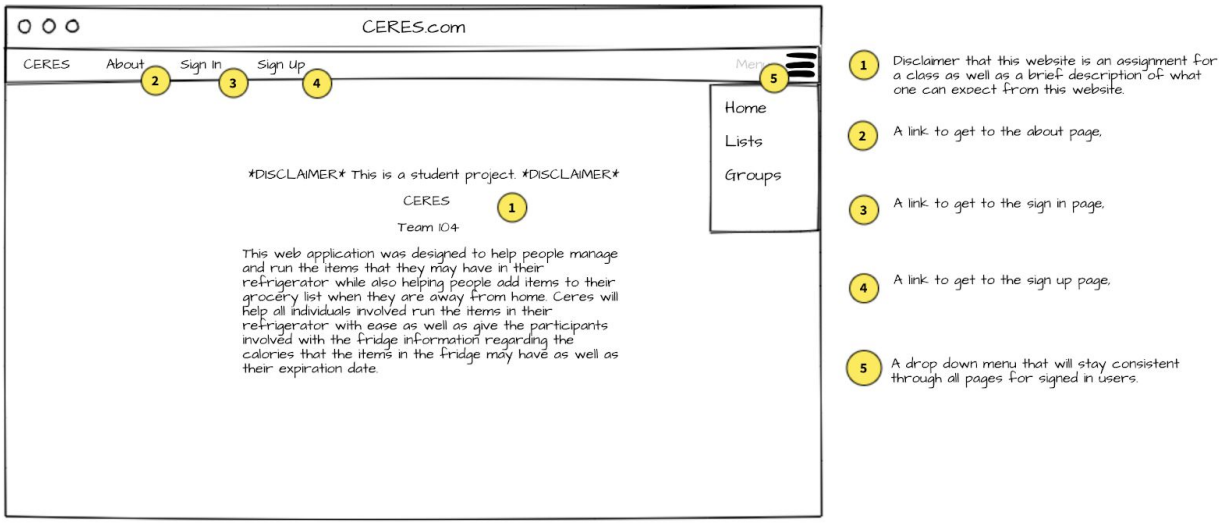
2. Users shall be:
   2.1. Able to set goals for food consumption (calorie - limit, target)
   2.2. Able to view consumption report regularly on their history of inventory (chart/graph)
   2.3. Notified when Refrigerator space is nearly full
   2.4. Able to create a chart/point meter to show how close they are to their target/limit (maybe some reward on completion)
   2.6. Make a shopping list based on recipes the user wants to try

3. UI Mockups and Storyboards (high level only)

a. Mockups

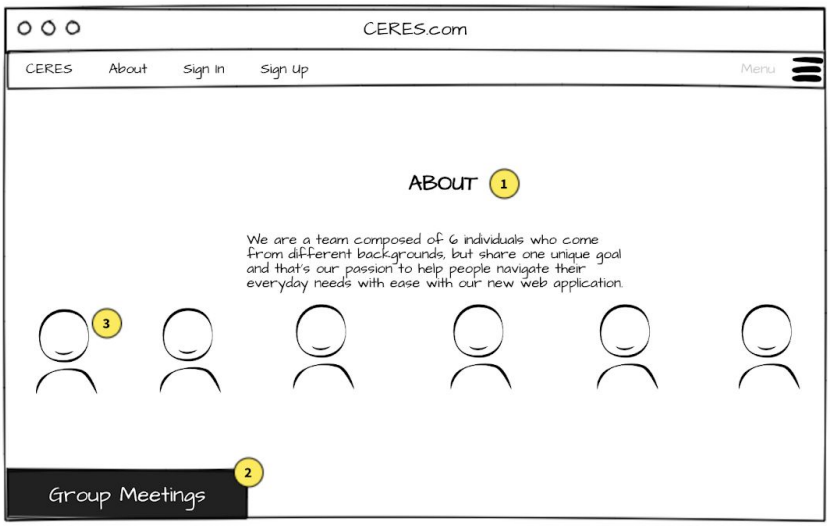
Landing Page:

This page will be displayed to all users when they first access the website. Clicking on the name ‘CERES’ on the navigation bar will also redirect to this page. From here a user will be able to easily access the About, Sign Up and Sign In pages. An unregistered user will not be able to see the menu bar.



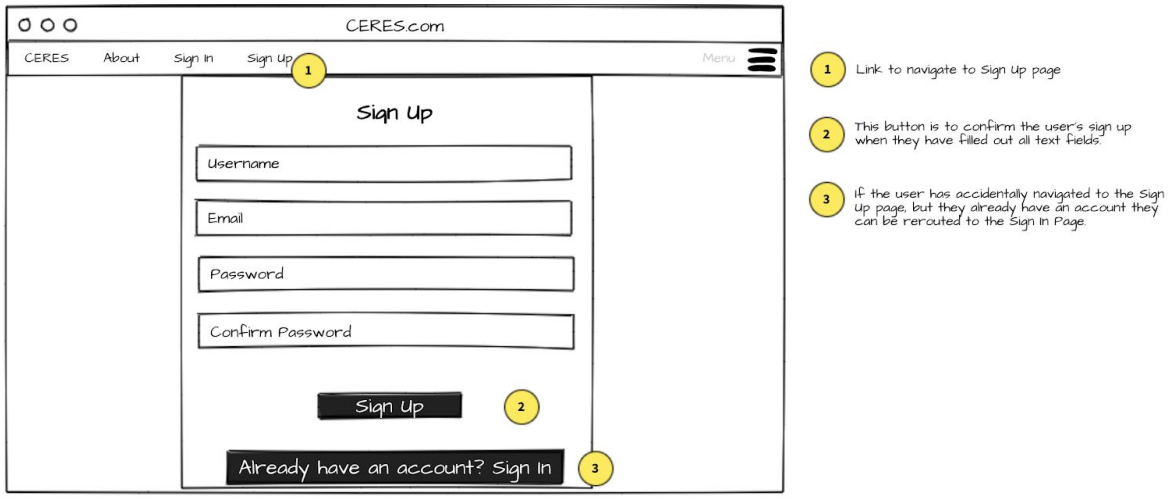
About Page:

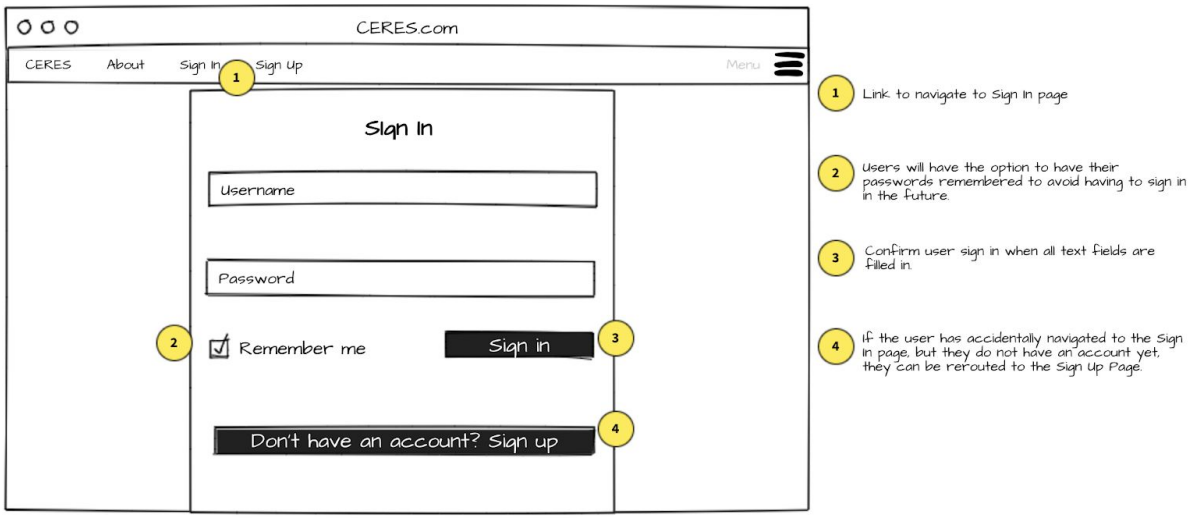
The About page is a brief description of our team and the passion that drives us to finish this application. It also provides visitors to the website with a bio of all the members on the team.



Sign Up and Sign In Pages:

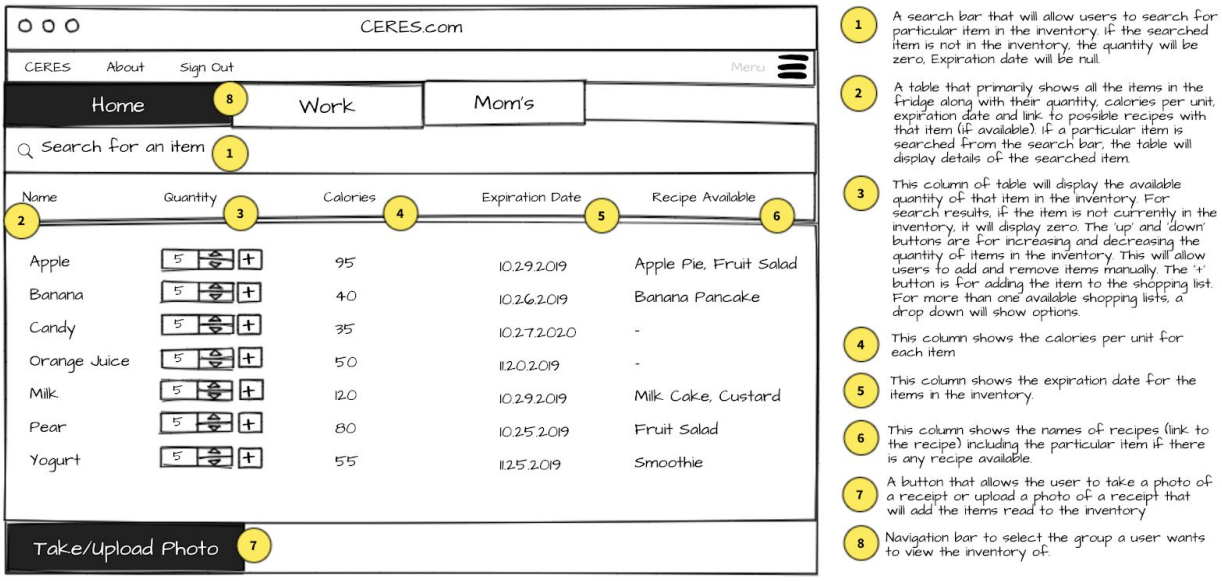
The Sign-Up and Sign-In forms can be accessed by all visitors of the site. A user will be able to access these pages through the links on the navigation bar when they are not logged in. When selected, the user will be redirected to the individual pages.





Home Page:

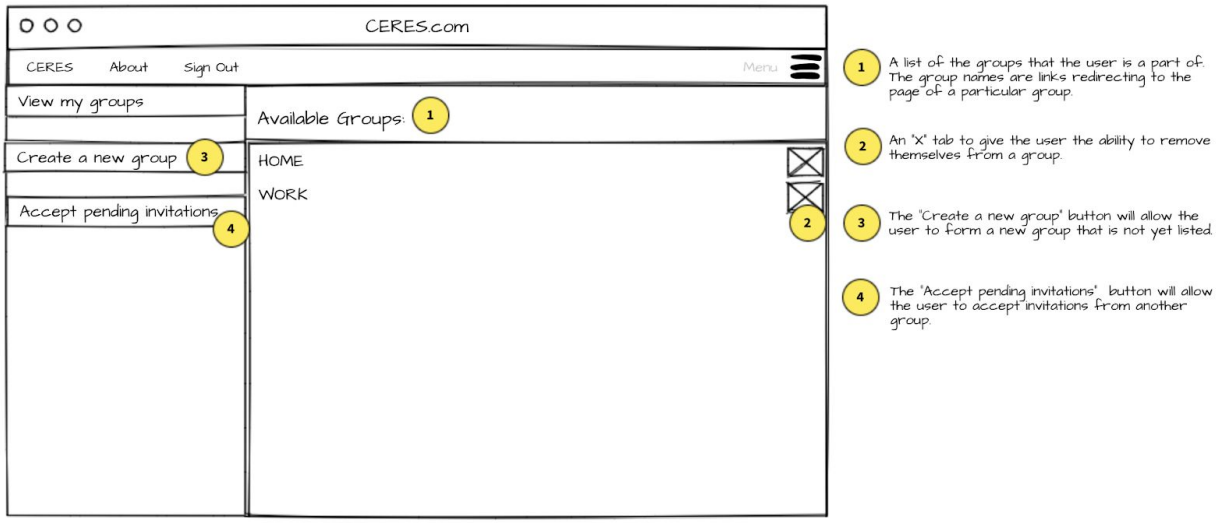
The Home page can only be accessed by a user once they have signed in to their account. This page will allow them to view their current fridge inventory as well as navigate to any other desired pages.



Fridges Page:

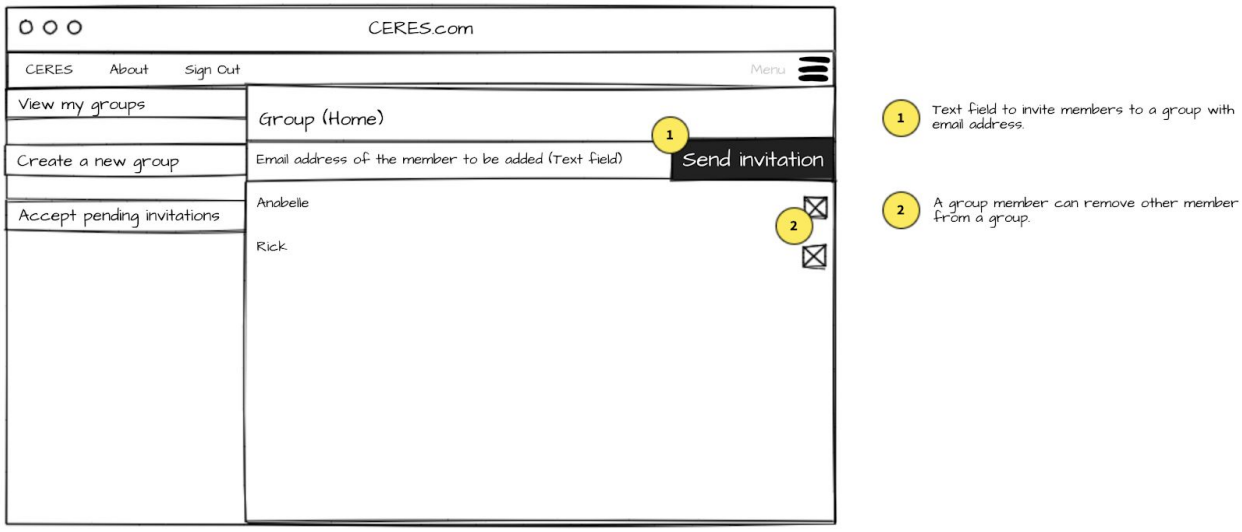
The Fridge page can be accessed by clicking on the 'Fridge' tab on the menu bar or the 'View my fridges' button on the sidebar of the page itself. This page will allow the user to view the fridges that they are currently a part of as well as remove themselves from a fridge. The options to create

a new fridge and to accept an invitation to a new fridge are available to the user from this page.



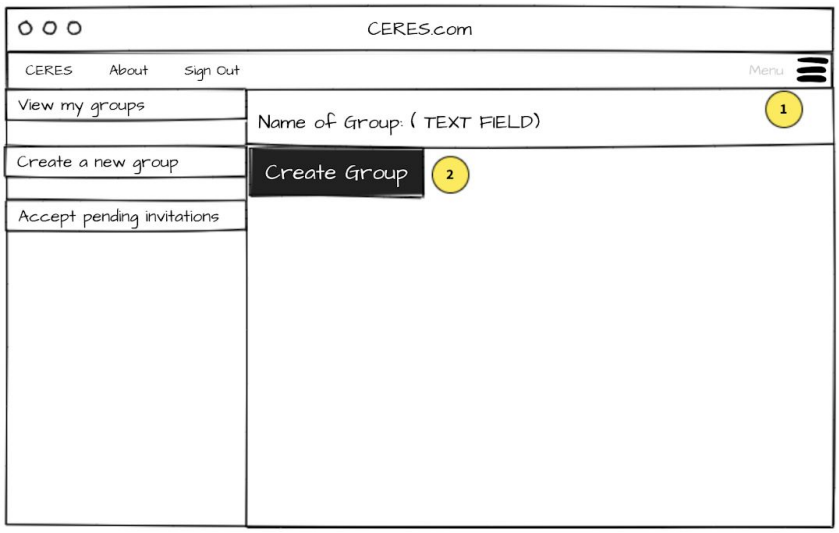
Manage Fridge Page:

The Manage Fridge page is meant to give the user more details about the group they are already a member of. This page can be accessed by clicking on the particular group from Fridge Page. This page allows the user to edit the members in the group such as inviting or removing members.



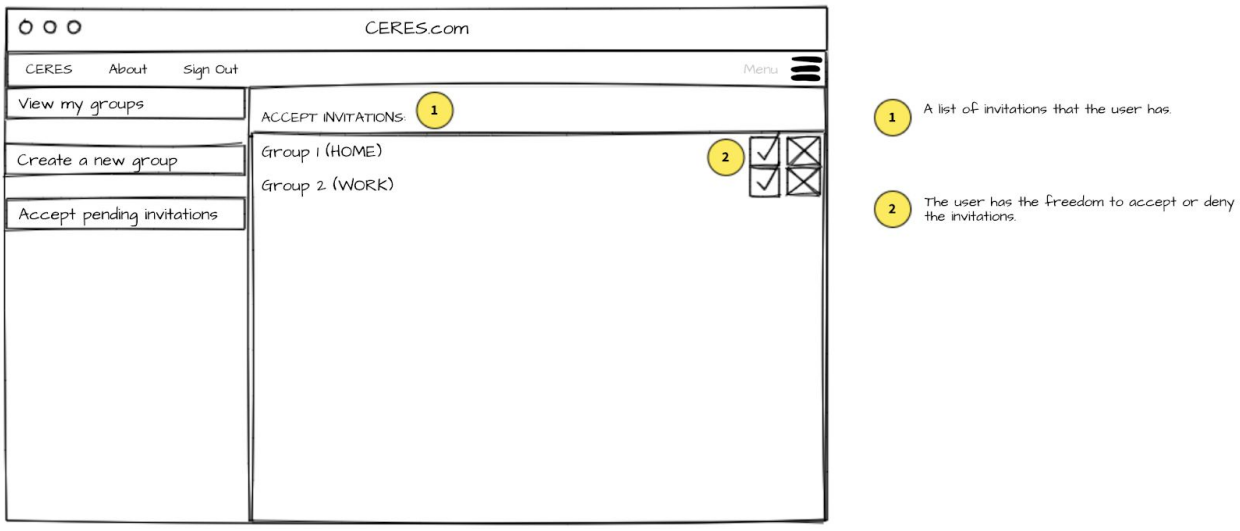
Create a Fridge Page:

The Create Fridge page can be accessed by clicking on the ‘Create a new fridge’ button on the sidebar. This will allow the user to create a new fridge.



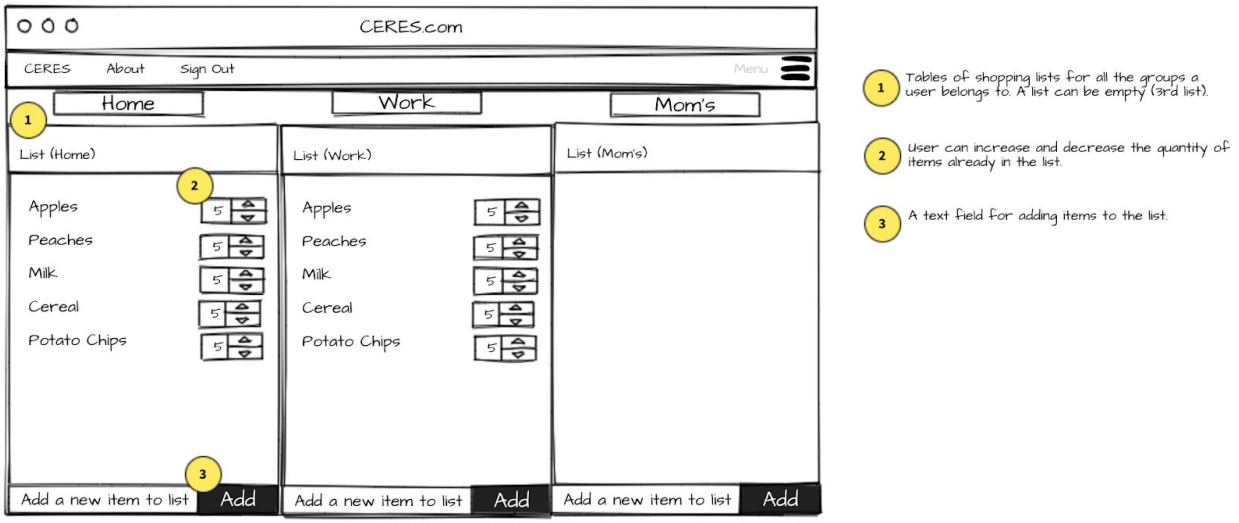
Accept Invitations Page:

The Accept Invitations page will give users the freedom to accept or deny pending invitations they have received from other users of the application. If a user accepts the invitation they will be added to the new fridge, and if the invitation is denied then it will simply disappear.



Lists Page:

The Lists page is meant to give users an easier way to view and edit shopping lists for all the fridges they belong to. This page can be accessed by clicking on the ‘Lists’ button on the menu bar.

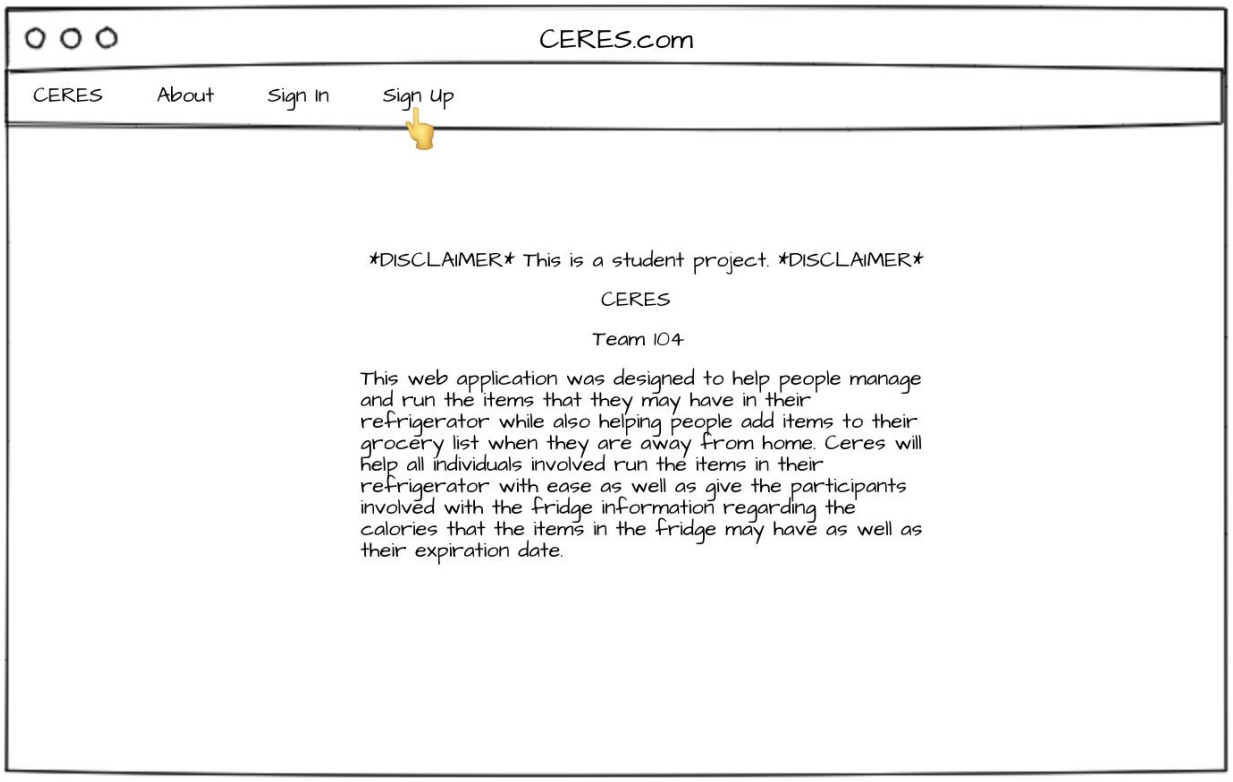


b. Storyboards

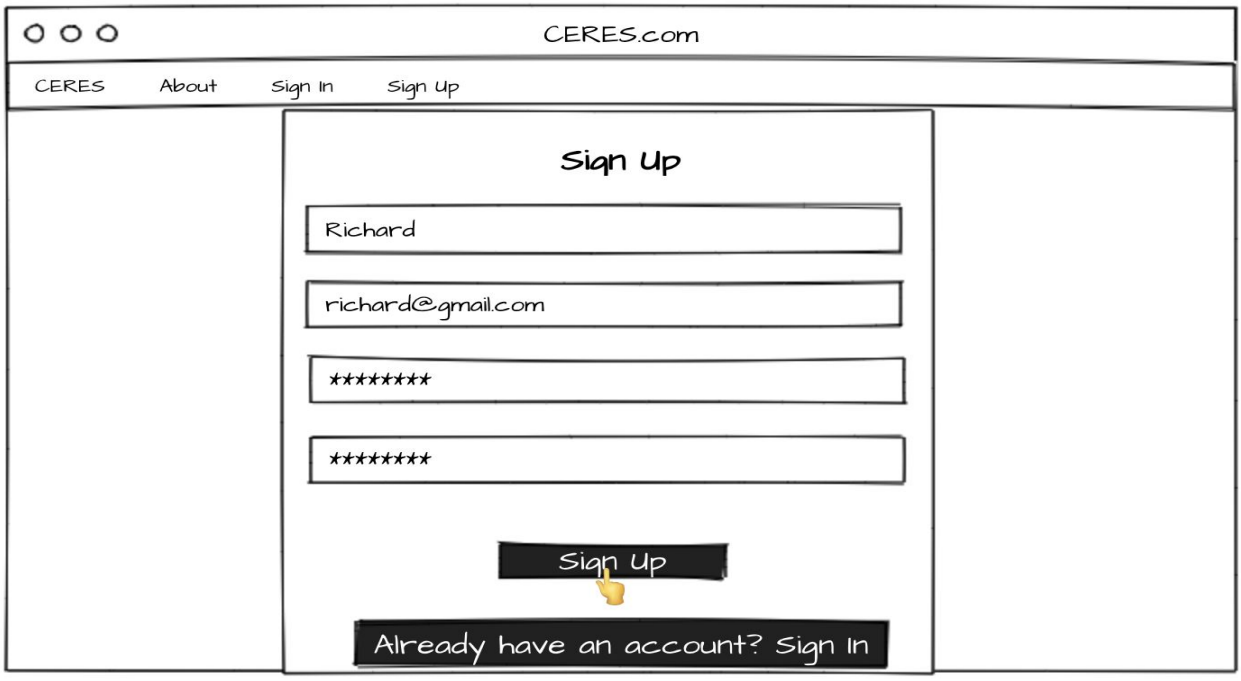
User Story 1: Richard

Richard, a college student, after hearing about the app, creates an account. He wants his roommate Adam to be in his group so that Adam can look at Richard’s shopping list when he(Adam) goes to the grocery store and buys the items for Richard.

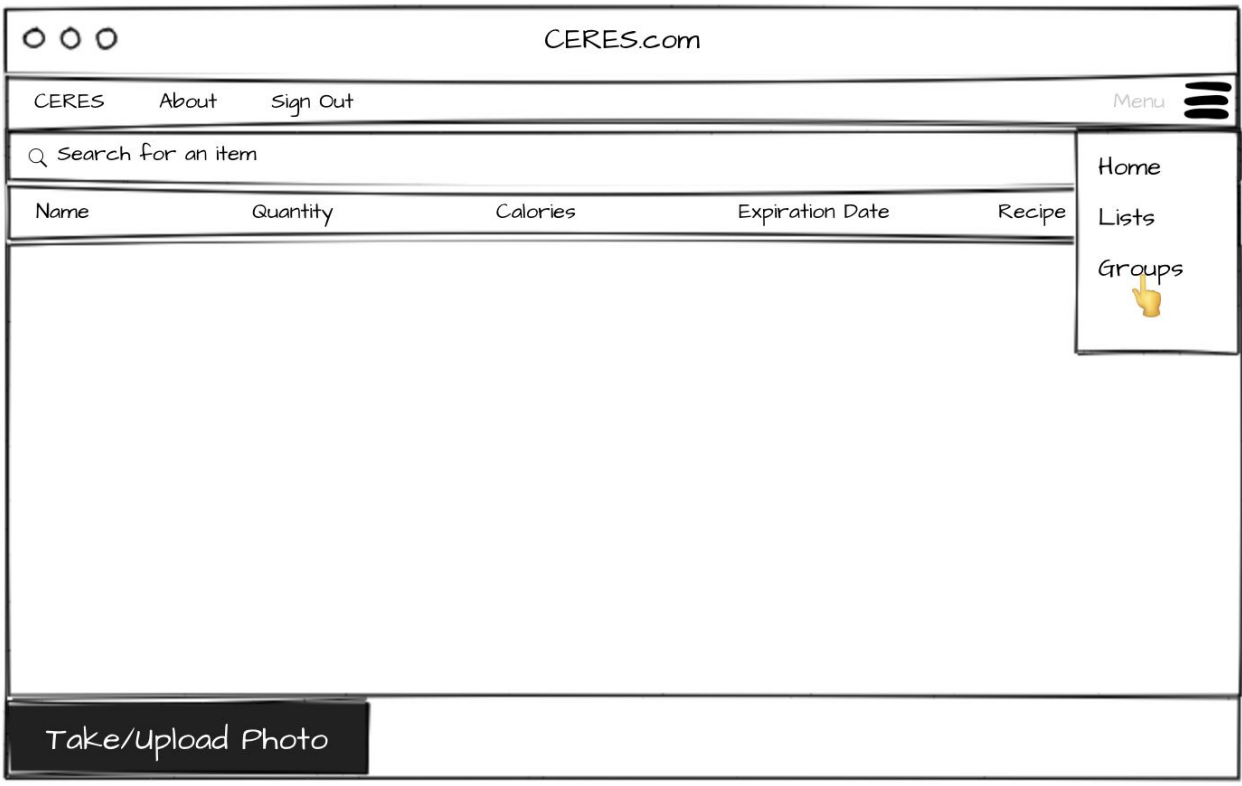
1. Richard visits the website for the first time and clicks Sign Up to create an account.



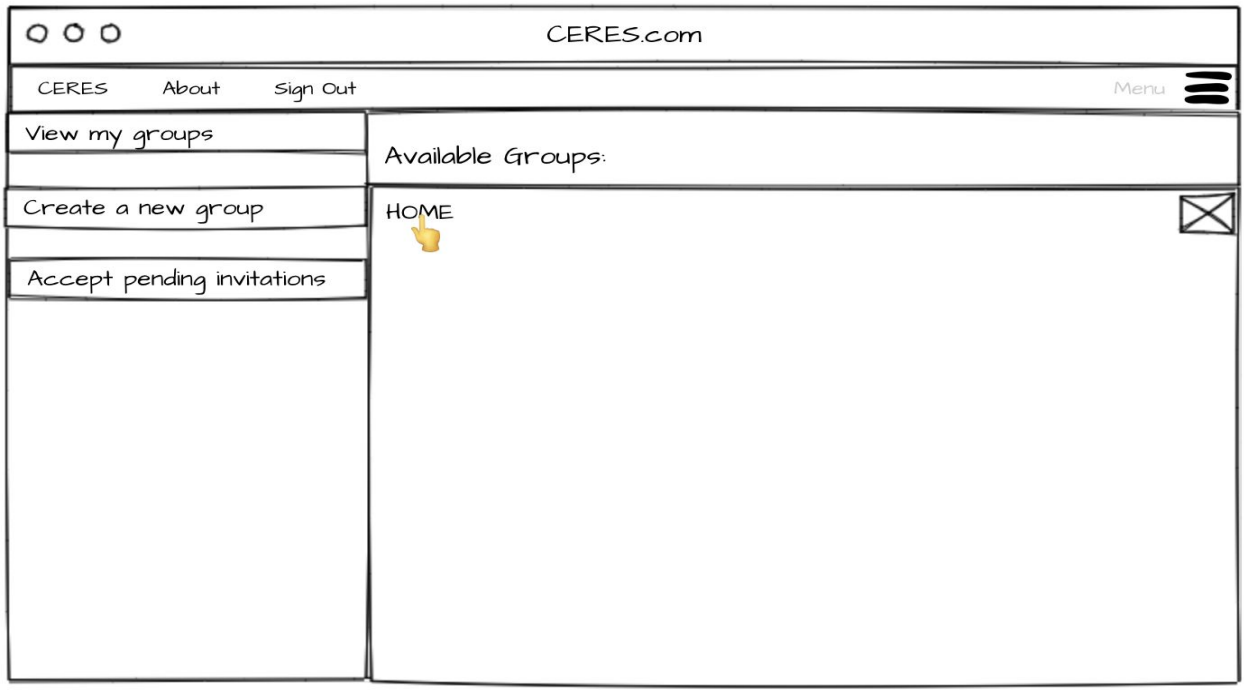
2. Redirected to the Sign Up page, Richard fills out the form and clicks on Sign Up to create the account. When he will visit the website again, he will be able to sign in by clicking on the Sign In link on the navbar or the button saying ‘Already have an account, Sign In’.



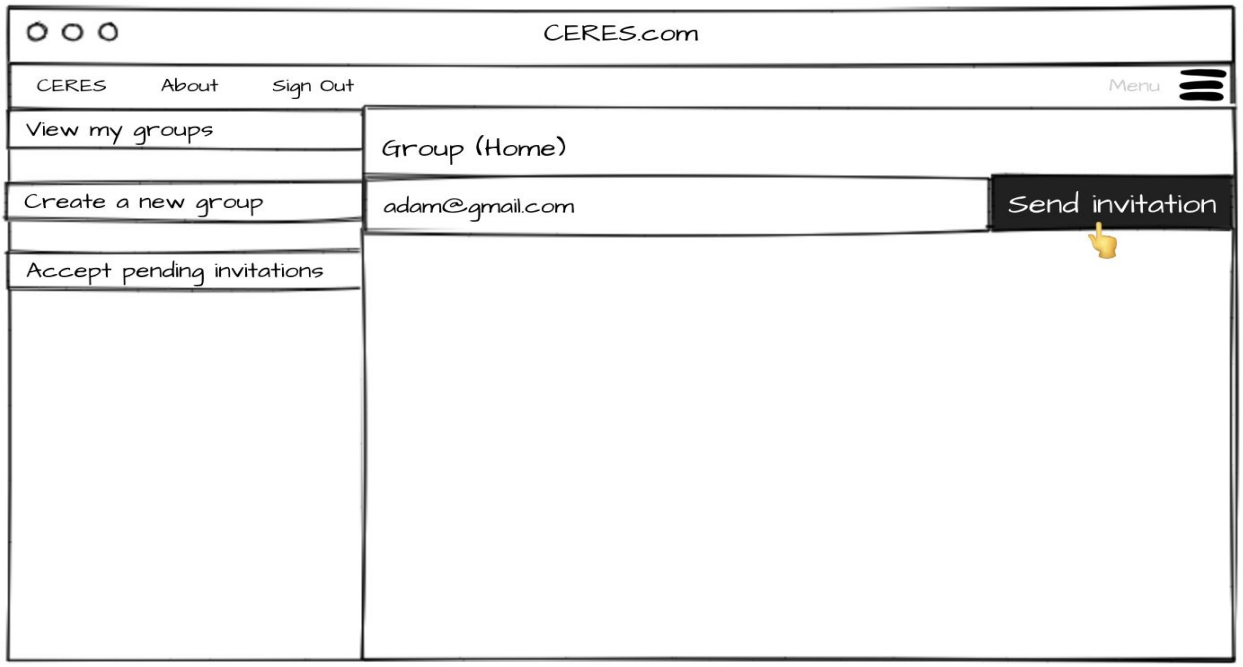
3. Redirected to the homepage, Richard can see that he currently has no items in his inventory as he has just created the account. Richard clicks on Fridges from the menu bar to view his fridges.



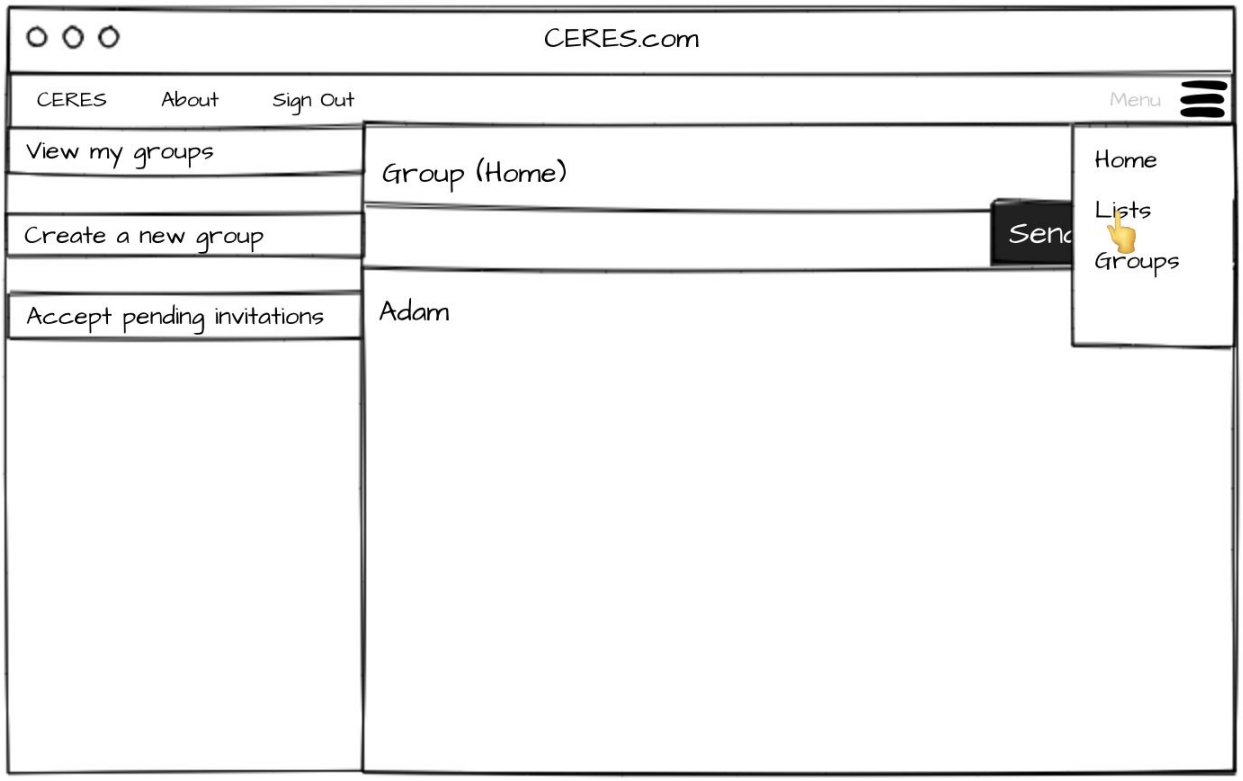
4. Richard is a member of his default fridge ‘Home’. He clicks on Home to add members to the fridge. If Richard were a member of other groups, he could see the names of all the groups here and access them by clicking on the fridge names.



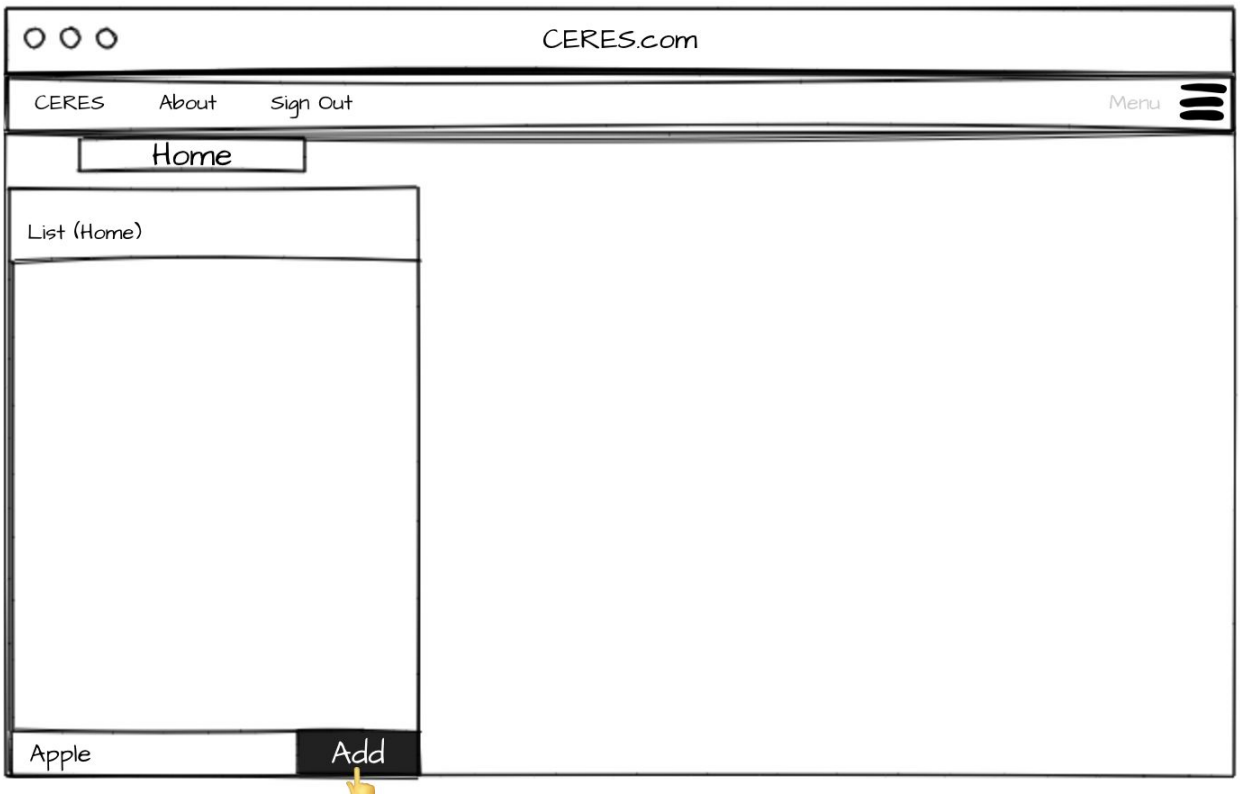
5. Richard sends an invitation to his roommate Adam, who already has an account on the website, using his(Adam) email address. Richard can send invitations to his other roommates in the same way.



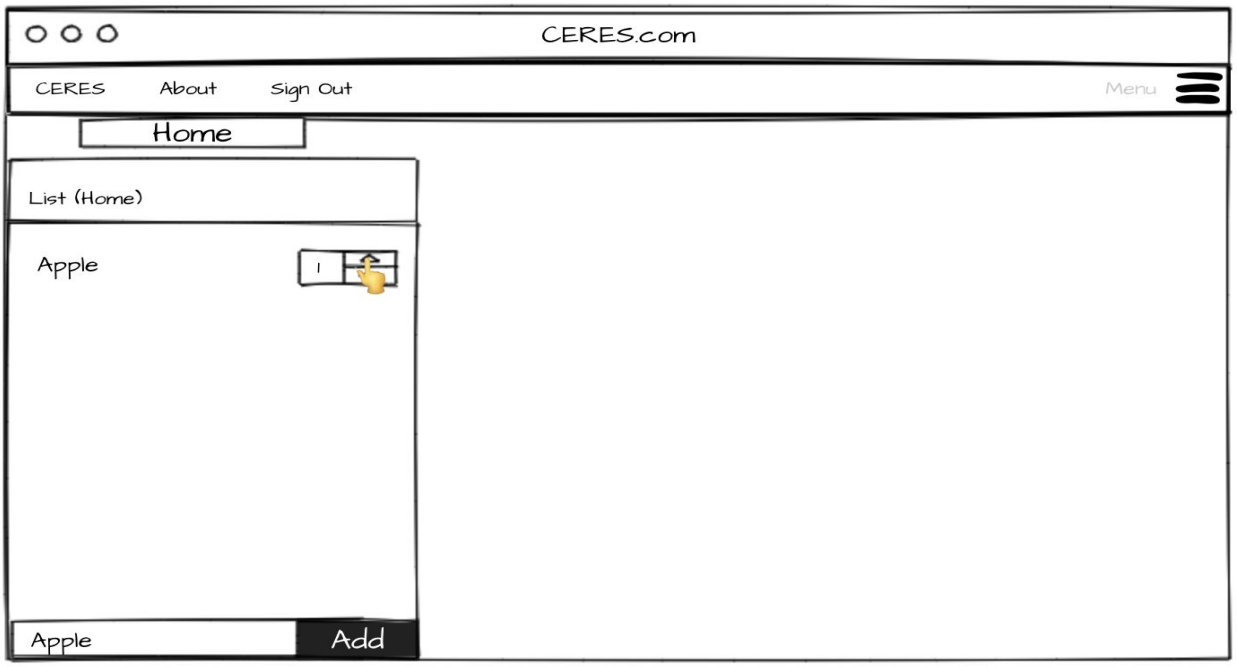
6. After Adam accepts Richard's invitation, his name appears on the Manage Group page of Richard's Homegroup. Then Richard clicks on Lists on the menu bar to add items to his shopping list.



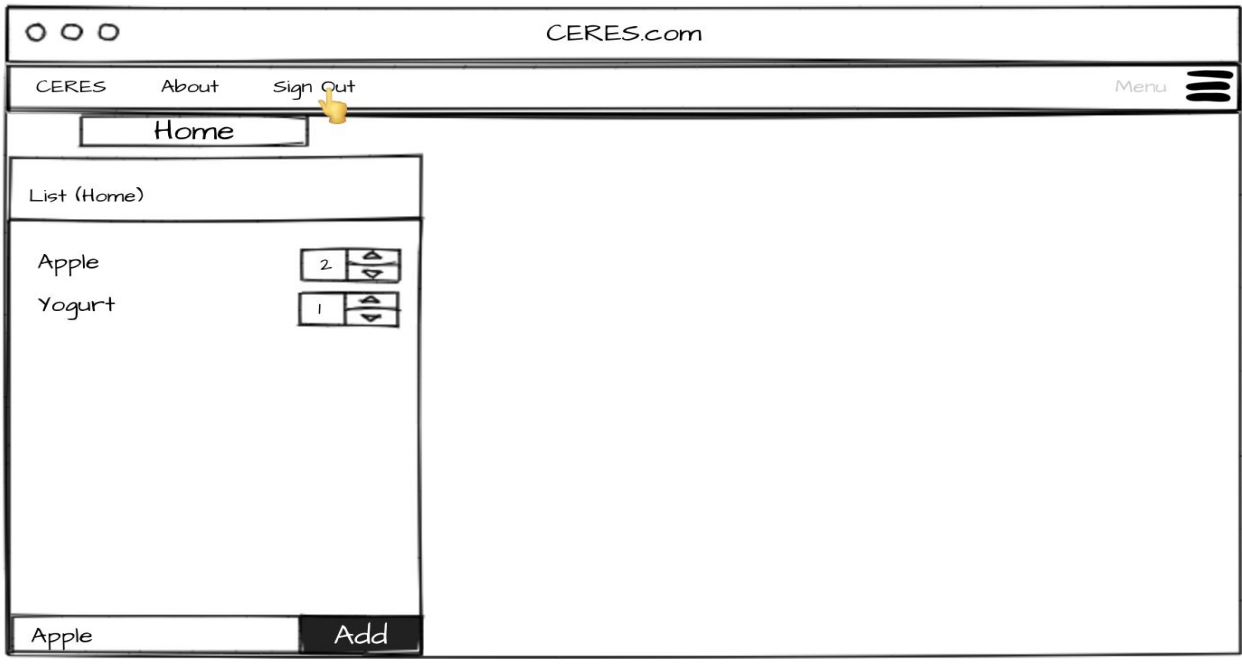
7. Richard needs to buy 2 apples. So, he enters Apple in the text field of the list of his Home fridge and clicks on Add. He can add as many items as he wants in the same way.



8. Apple is now listed. Richard clicks on the ‘up’ button to increase the quantity of apple. If he wants to decrease the quantity of any item in the list, he can click on the down button to do so. If the quantity goes below 1, the item is removed from the list.



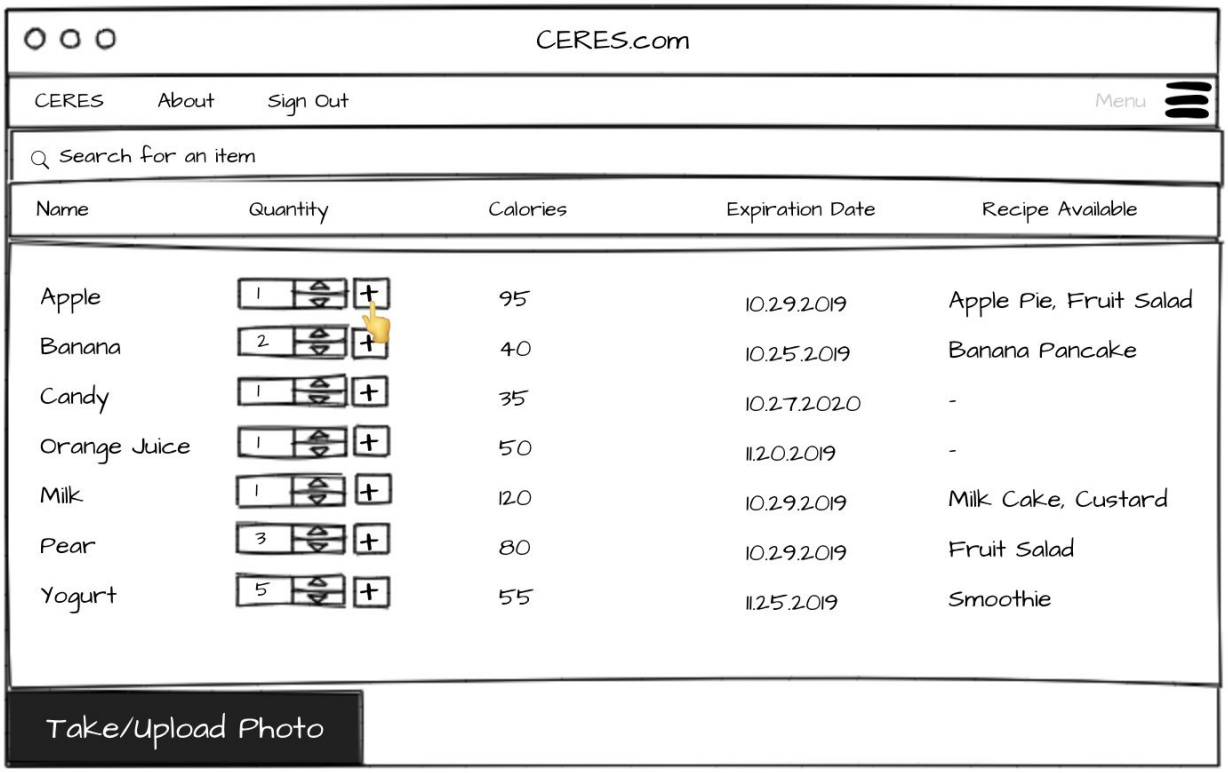
9. After adding all the items in the shopping list and adjusting the quantities, Richard is done and signs out by clicking on the Sign Out button on the navbar. Adam along with all other members of Richard's Home fridge can now view these items in Richard's shopping list.



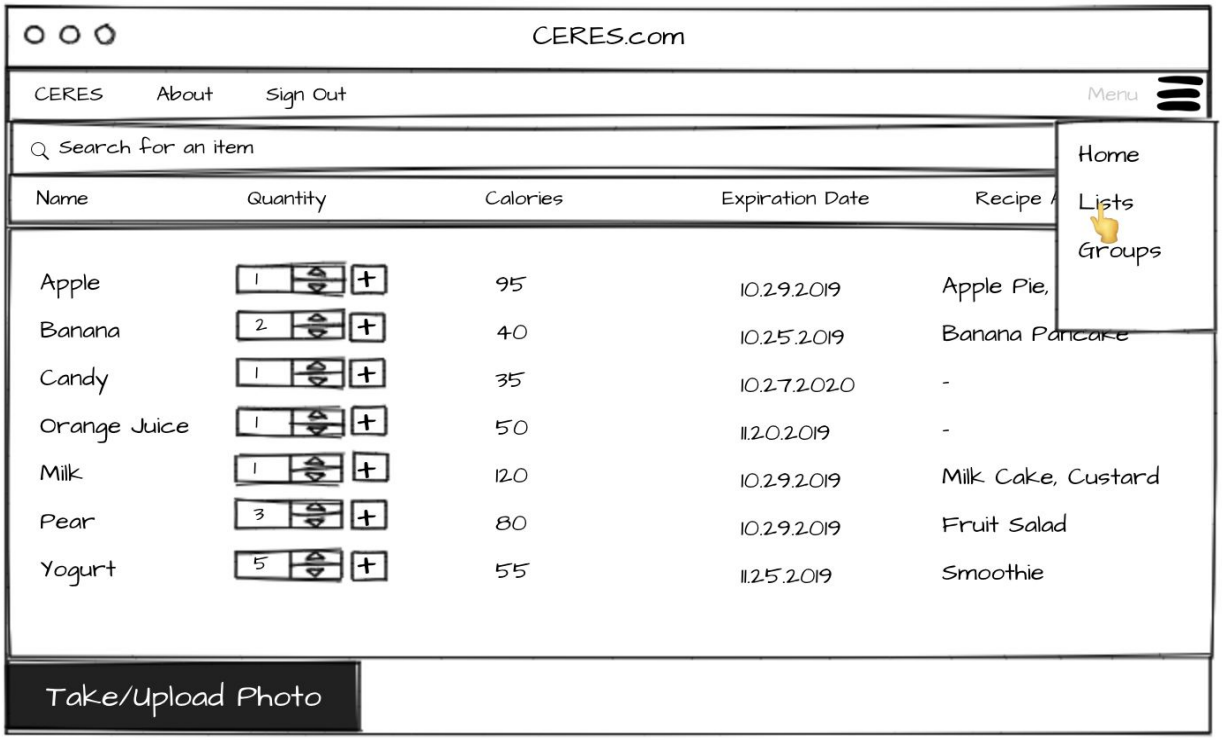
User Story 2: Jenny

Jenny, mother of Triston, is planning to go to the grocery store on her way home from work. She wants to know what items are in her fridge and what Triston has added to the shopping list.

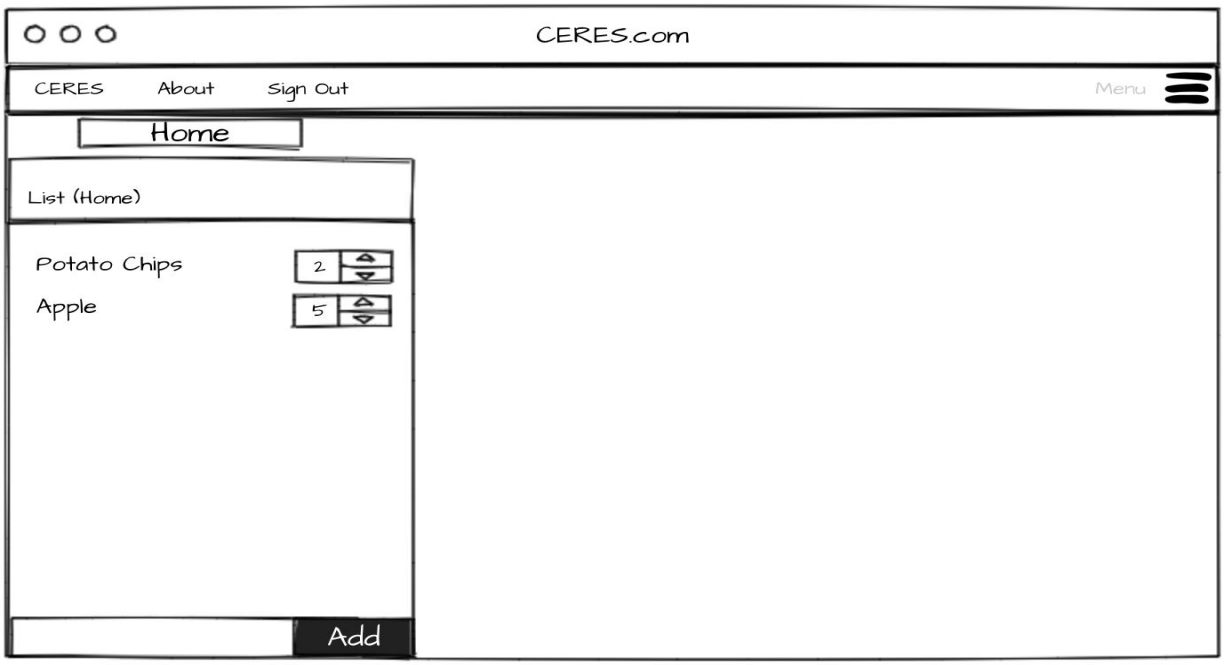
1. After signing in, Jenny can see the name, quantity, expiration date, etc. of items in her inventory. Based on her current inventory, she wants to buy more apples. So, she clicks on the '+' button for Apple to add them to the shopping list.



2. Jenny wants to know if Triston needs anything. So, she clicks on the Lists tab from the menu bar to view the list.



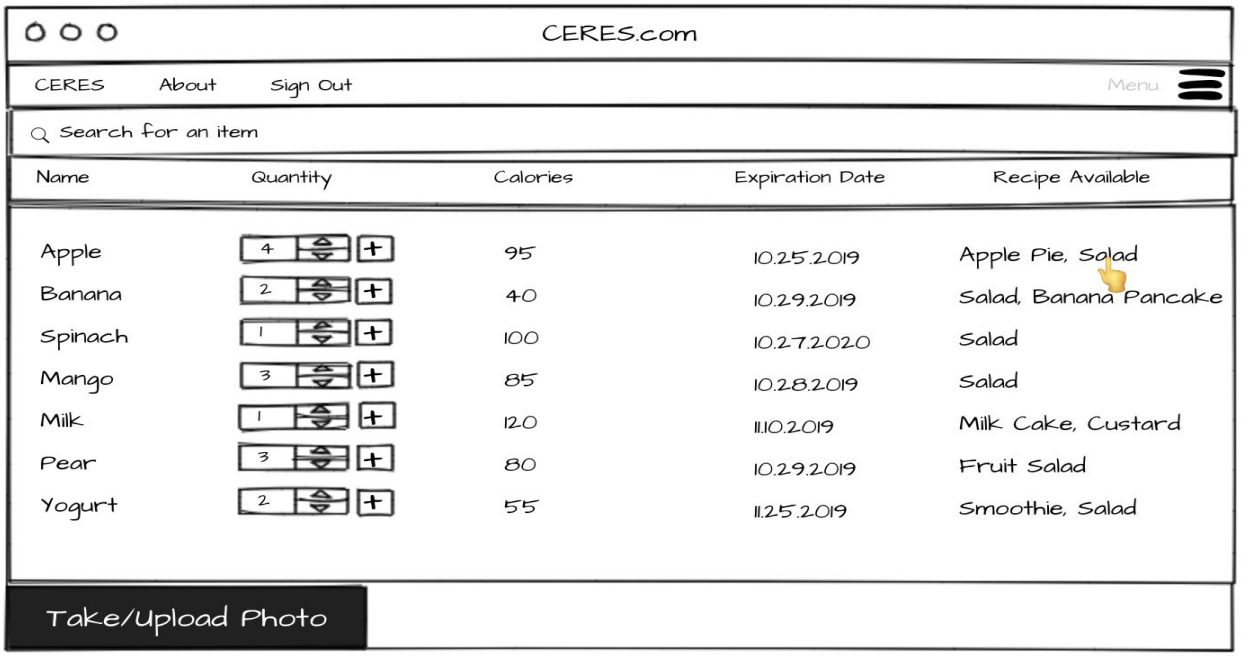
3. Jenny can see that Triston has added 2 Potato Chips to the list. Apple added by her from the home page is also reflected in the list. Taking note of that she can now leave the page and do the shopping accordingly.



User Story 3: John

After a long day at work, John goes home by bus. He wants to know what he can make for dinner with the items in his inventory.

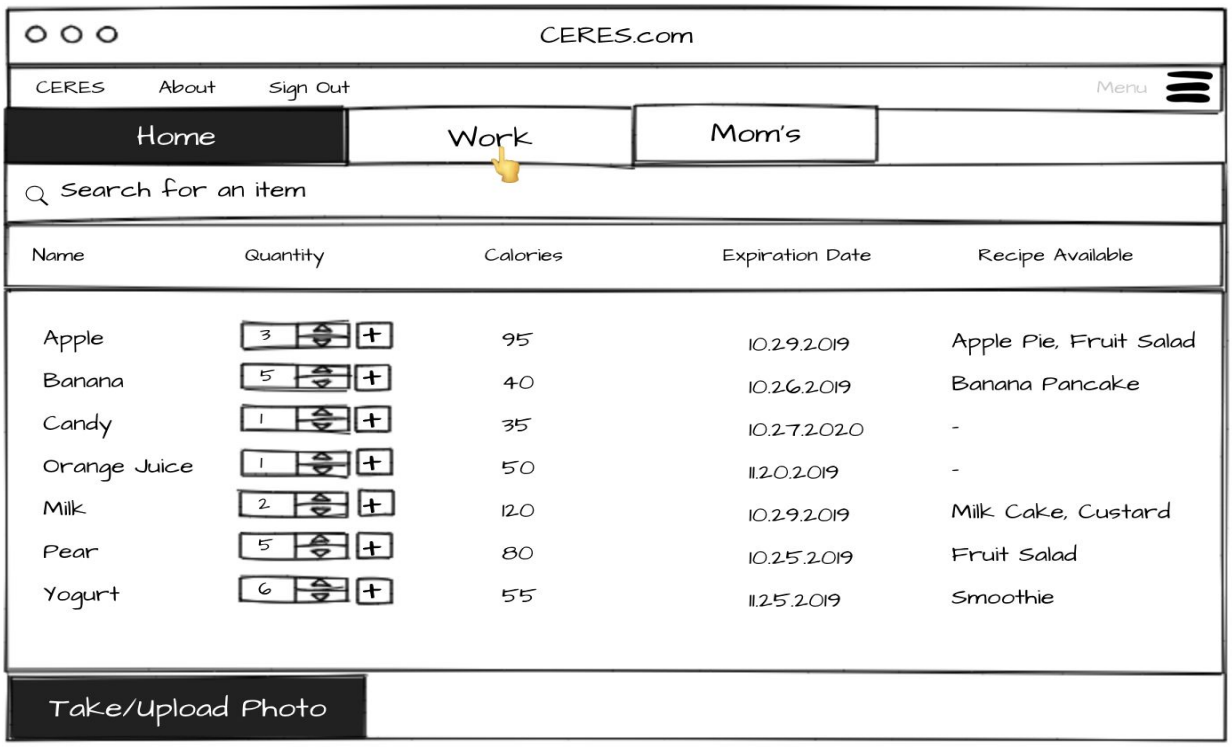
1. Signing into his account, John looks at his current inventory. He can see that he has a lot of fruit in the fridge and some of them are going to expire soon. And most of the items in his fridge have the common recipe ‘Salad’. So, he decides to eat Salad for dinner and clicks on ‘Salad’ to see the recipe. This will redirect him to another website that has the video or written instructions of the selected recipe.



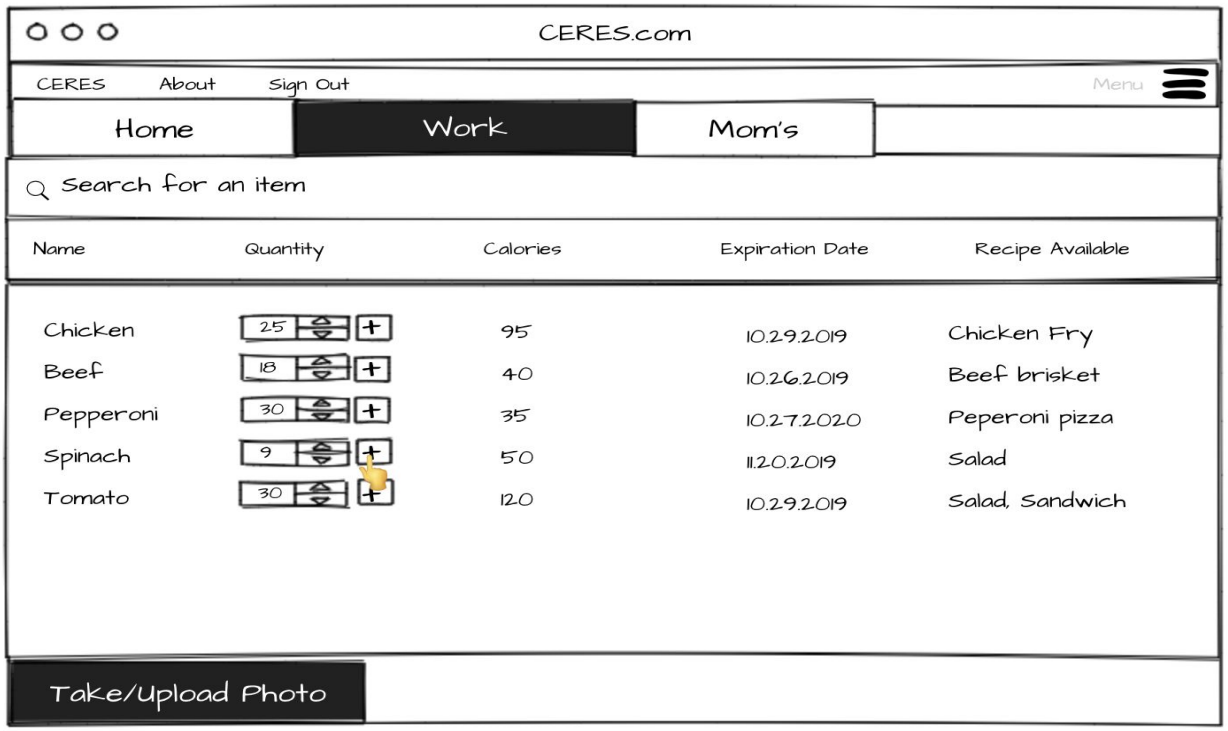
User Story 4: Jensen

Jensen, in charge of the food supplies of a restaurant, wants to take a look at the current inventory of the restaurant and buy items if necessary.

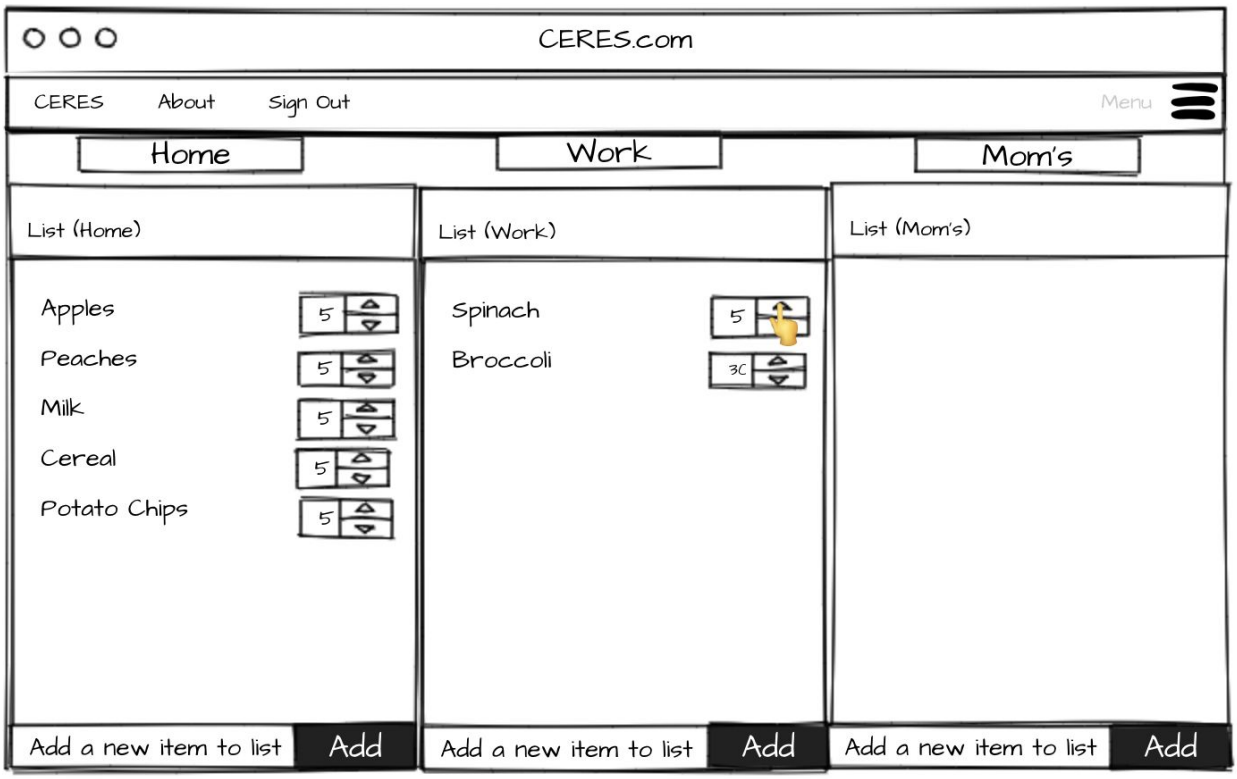
1. Signing into his account, Jensen can see the inventory of his house. So, he changes the group by clicking on the Work button on top. He can navigate to any fridge he wants by clicking on the appropriate button.



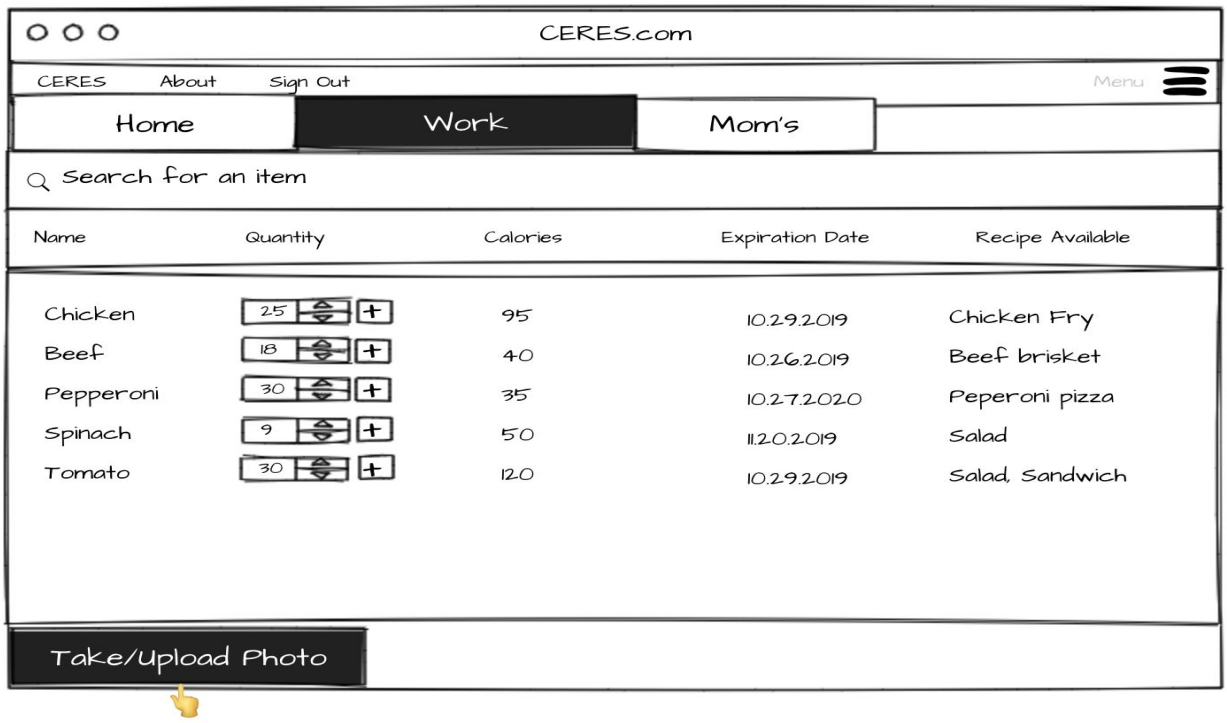
2. After navigating to the Work inventory, Jensen sees that they will need more spinach. So, clicking on the + button for spinach, he adds Spinach to the shopping list for his Work.



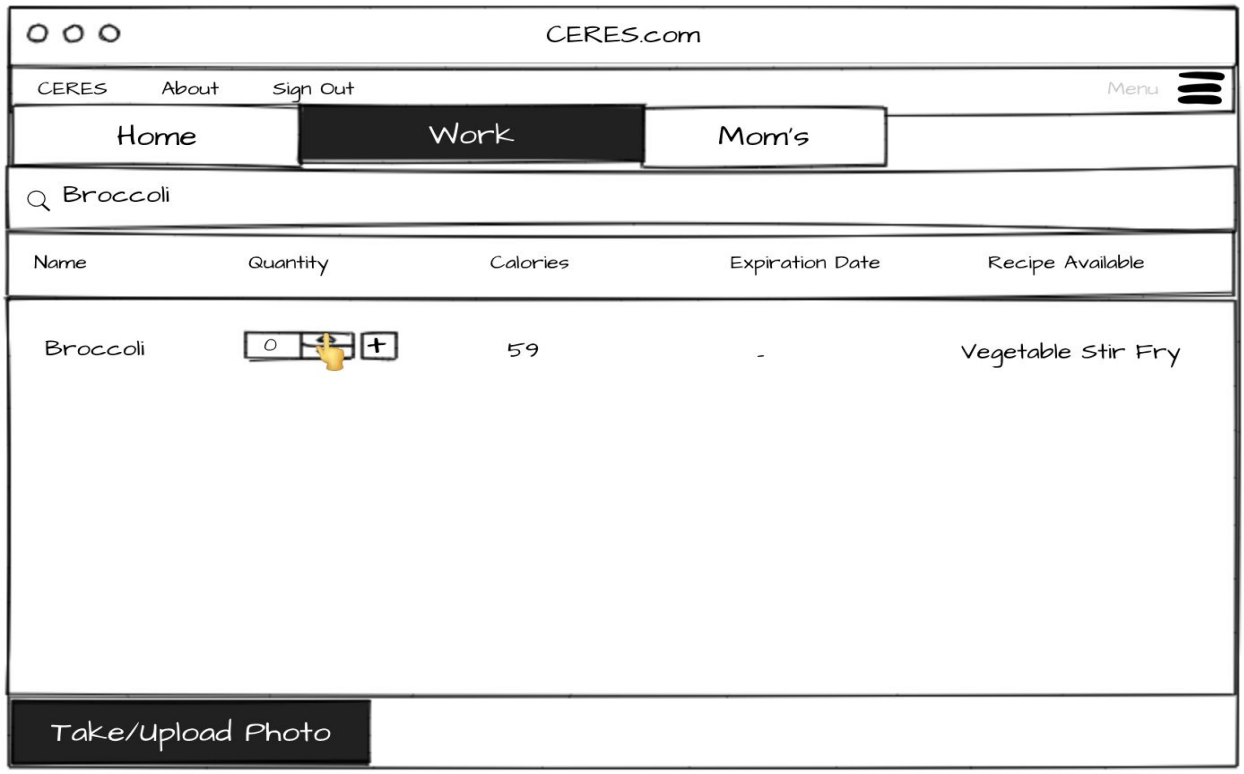
3. Navigating to the lists directory from the menu bar, Jensen reviews his shopping list. He adds new items to the respective shopping list by clicking on the Add button for the desired fridge. He can increase the quantity of items by clicking on the appropriate up button for a particular item in the desired fridge.



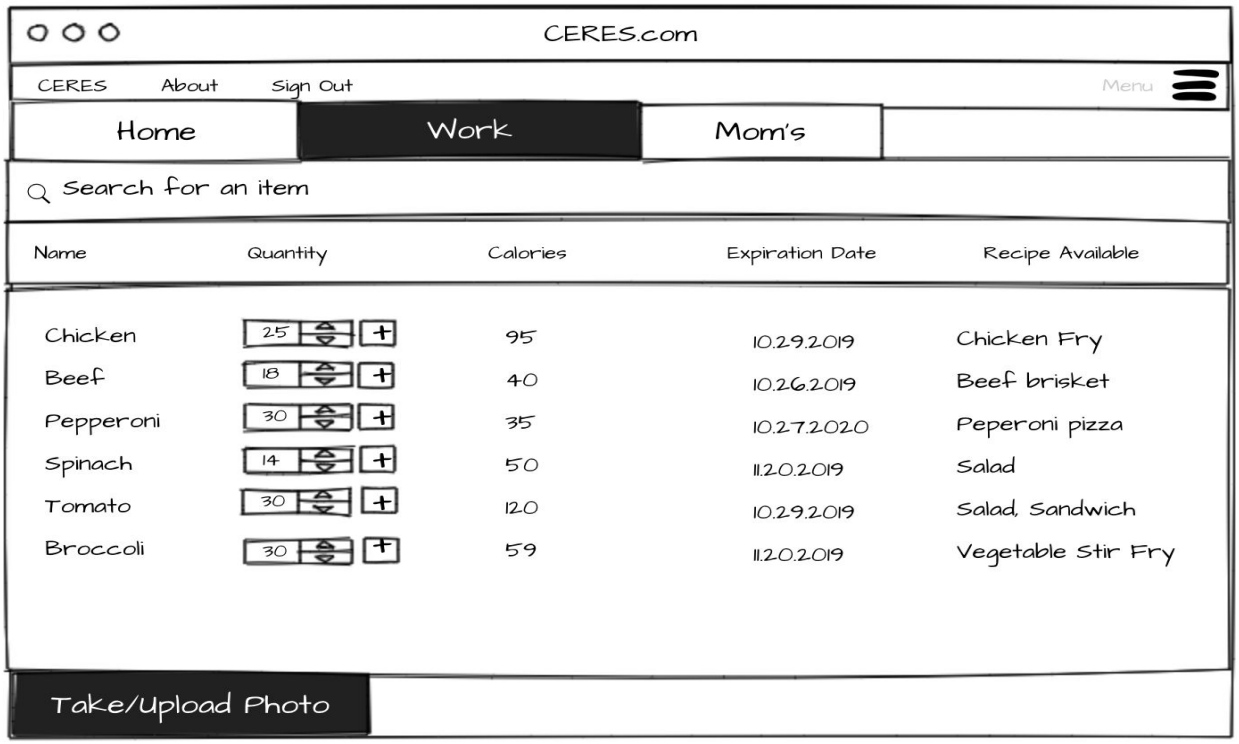
4. After buying the items in the shopping list, Jensen signs in again and clicks on ‘Take/Upload Photo’ to take a picture of the receipt for adding new items to the inventory.



5. Jensen left the receipt for buying Broccoli at home. So, he will add it to the inventory manually. He searches for Broccoli in the search bar. When the result is displayed, he clicks on the ‘up’ button to adjust the quantity.



6. Finally, he will be able to see the updated inventory by refreshing the page.



4. High-Level Architecture, Database Organization

a. Database Organization:

Our database is organized as such:

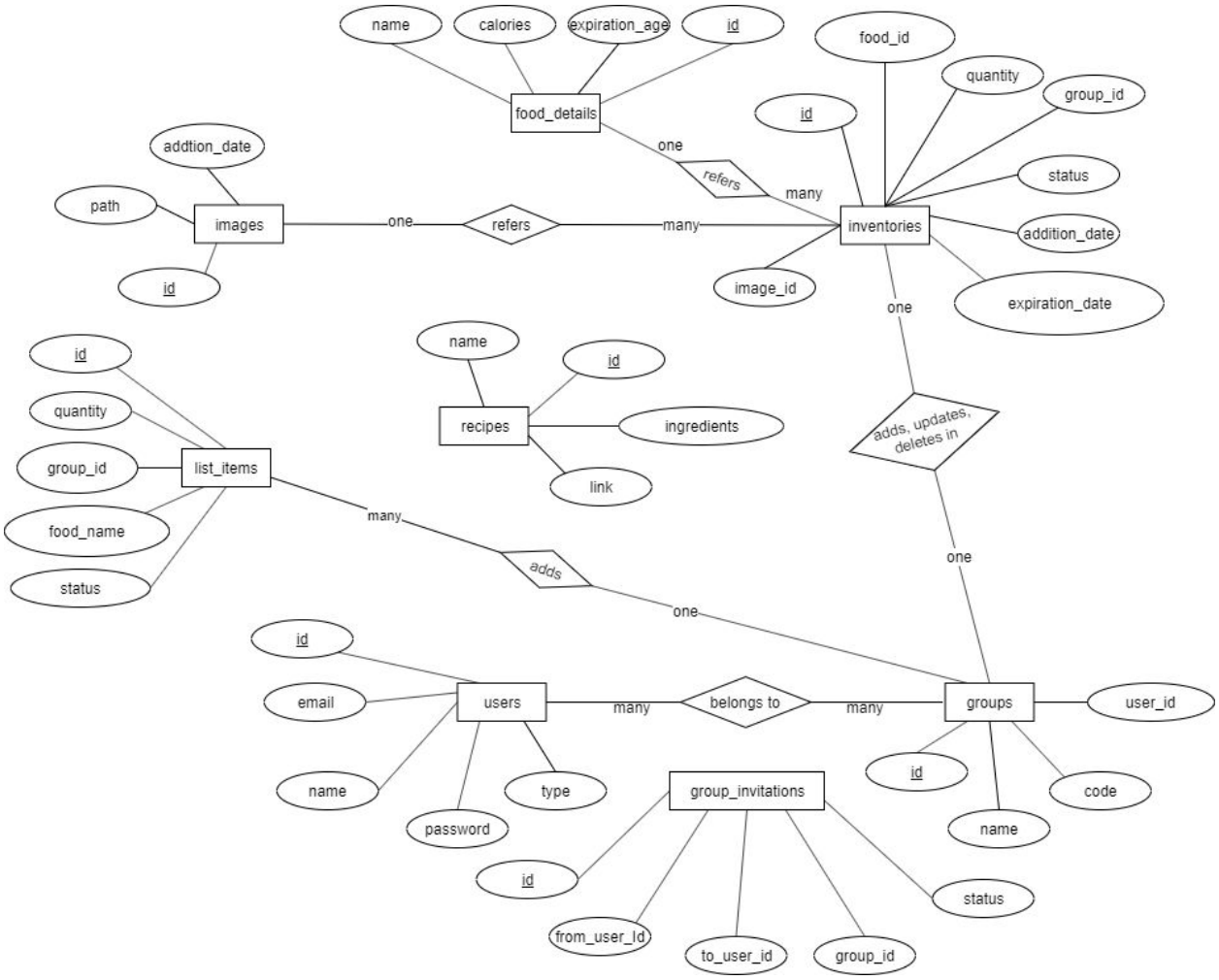
- Users table has all the registered users. The attributes are id, name, email, password and type. Id is the primary key of the table and it auto increments. Name, email and password attributes are required for each user to sign in. Email should be unique for all users. Type attribute is for future data analysis. It is not a required field. The attribute tells whether a user uses the website for personal or professional use.
- Groups table has all the connections of users with fridges. The attributes are id, user_id, name and code. Id is an auto incremented primary key. User_id comes from the id of users table. Name field is the name of a fridge given by the creator of the fridge. Code is unique for each individual group. This attribute distinguishes between different fridges (essentially a group) of the website.
    - Users and group tables have a many to many relationship. Each fridge can have one or more users and each user can be a member of multiple fridges. Each user must be a

member of at least one fridge. A user can be the only member of his fridge.

- Group_invitations table is for tracking the invitations sent for joining a fridge. The is attribute is an auto incremented primary key. From_user_id is the user_id of the sender and to_user_id is the user_id of the receiver of the invitation. Group_id is the code of the fridge for which the invitation has been sent. Status indicates whether the invitation has been accepted, declined, or still pending for response.
- List_items table contains each entries added to a list. The attributes are id, group_id, food_name, quantity, status. Id is an auto incremented primary key. Group_id comes from the code from groups table. Food_name and quantity are inputs given by the user while adding items to the list. Status is to track whether the list is active or inactive.
- The Images table stores all the images of receipts used to add items to the inventory. Id, path and addition_date are the three attributes of this table. Id is an auto incremented primary key. Path is the path of the image inside the project. Addition date is the date the image has been uploaded.
- Food_details table is a static table with the details of food items. Id is an auto incremented primary key. Other attributes are name, calories and expiration_age. This table is pre populated with food data and will not be changed by the website. This table will be used to fetch details of a food item in the inventory.
- The Inventories table contains all the items that go into the fridge. The attributes are id, group_id, food_id, image_id, quantity, addition_date, expiration_date, status. Id is an auto incremented primary key. Group_id, food_id and image_id are foreign keys pointing to code of groups table, id of food_details table and id of images table respectively. Quantity represents the quantity of the food item in the inventory. Addition date is the date the item has been entered into the table. Expiration_date is calculated from the addition_date and expiration_age of food_details table. Status indicated whether the item is still in the fridge currently or not.
- Recipe is a static table pre populated with data and never changed from the website. Id is an auto incremented primary key. Name is the name of the food the recipe is for. Ingredients is an array of

necessary items for preparing the food. Link is the link to the video or written instruction of the recipe of that food.

b. Entity-Relationship Diagram:



c. Media Storage:

Our website will be able to read an image from the user's storage or give the user the option to take a picture of a receipt. In the database the image will be saved as a path to the image that was uploaded by the user. Users will have the ability to upload images to the website. Image format can be stored as a .png, .jpg

d. Search/filter architecture and implementation:

- We decided that for our search algorithm we will be using MySQL to form our queries. The data will be pulled from our tables
- The shopping list page will be dynamic, based on user modifications regarding the list and when they add, or delete an item, this will be stored in a database called List and will be unique to a set of users, also referred to as a fridge.
- When searching for an item, users will be able to search for items based on key identifiers, like item name, and addition_date.
- If searching for an item and no items of said names, id, or category are found, a default “not-found” page is shown
- The list_page will be chronological and will show items in the order that they are added.
- Users will be allowed to search their entire inventory of items, using date_added, item_id, item name, owner of item, category of items
- Implementation of this is done using a Django Library called filters, and we look to build off our search query implementing building off library.

e. Non-Trivial Algorithms

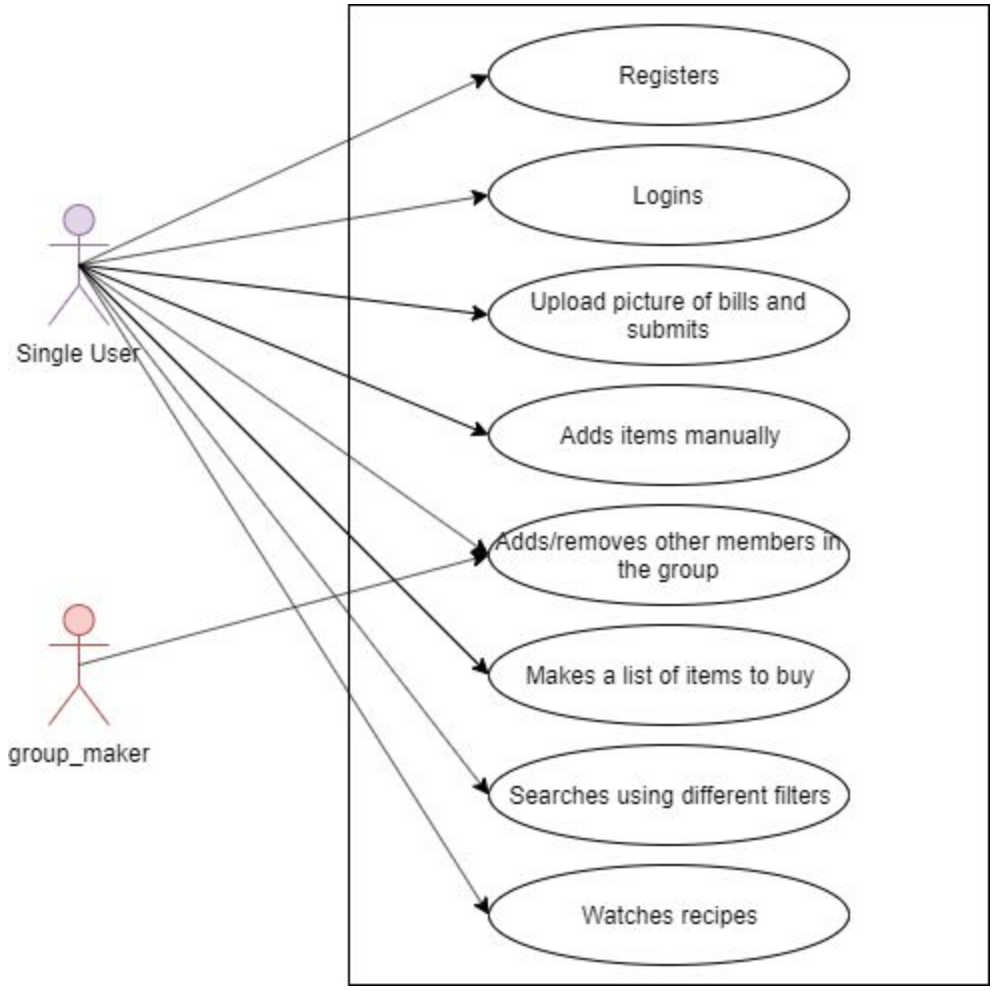
- Using Django as our framework the Django Filters Library that allows users to filter down a queryset based on a model’s fields, displaying the form to let them do this. This function will exist in a python file called filters.py.
- With recipes we also plan to implement a system that shows a user their most compatible recipes unique to their current owned inventory items, since items can be in an inventory but owned by specific users, this feature can be changed in settings.

f. Changes in software

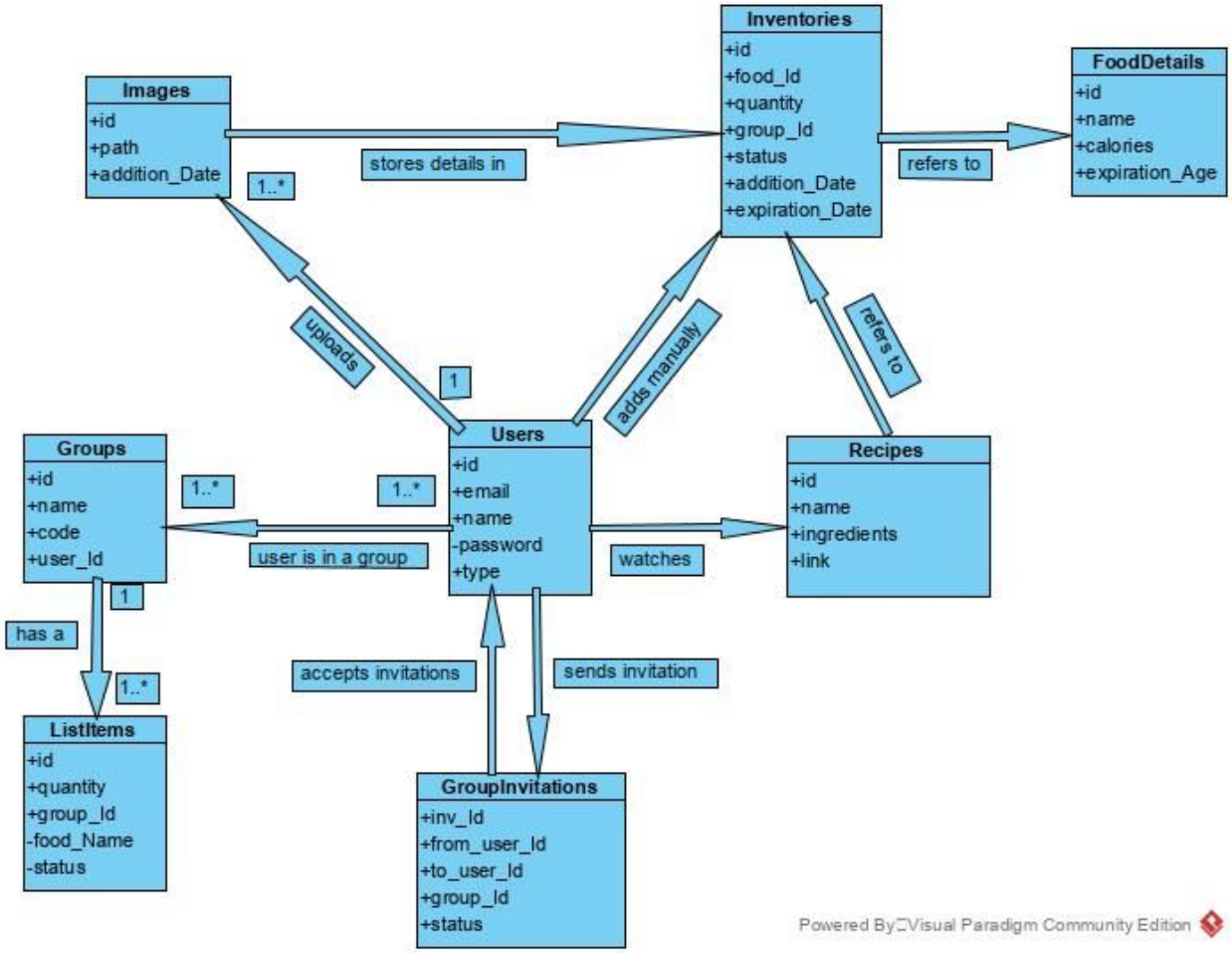
- Django contains many libraries and so far we have not made any changes, but have tried to utilize these libraries.

5. High-Level UML Diagrams

a. Use Case Diagram:



b. Class Diagram:



6. Key Risks for the Project

The current risks our team is facing:

- **Scheduling conflicts:** Time has proved to be an issue for our group as our schedules only allow us to meet at very specific times during the week; we currently have a date where we meet outside of class, however, outside of our allotted meeting time the possibility of meeting at any other time is very limited due to other obligations such as family obligations, work, and other priorities that are sometimes more pressing. To solve this issue we have set up a slack channel that the team must check throughout the day to ensure that questions or concerns regarding the project and milestones can be answered in a timely manner.
- **Technical limitations:** As we familiarize ourselves with the Django framework and the pytesseract API. We are learning more, but also learning how much we don't know; the solution we've come up with is an accountability log that we have to submit our studying hours into. For example, we have dedicated x amount of time into learning about our framework API, or any outside resource that can benefit the project as a whole.
- **Knowledge/Familiarity with Github**: For most of the team, this is the first time that they have worked on a project of this scale and with this many people. In order to tackle this we have a GitHub master that facilitates or relays information regarding issues; plus, we have a group effort approach on fixing issues regarding GitHub and follow the git-flow model.

7. Project Management

Currently, in the team we have divided up the workload to specific groups of people, we have a backend lead and frontend lead as well as helpers of said leads. During any given milestone the work that is placed or requested by a specific person, correlates to the role that they are given, however, we understand the team sometimes has other obligations that overlap so we try to help each other where we can since ultimately, we're a team first. In addition, as far as time management, we have promised a specific amount of hours in a week towards a specific part of the project or in the milestone. In addition, Trello is the platform we use to manage tasks and to keep track of tasks assigned by the Team Lead. Progress made is communicated through Slack, or if an error is

faced a team member can hop in and help out by answering questions through Slack. Our UI mockups were done with Mockflow which proved to be an easy program to use. Lastly, we don't like to overload any specific team members, so we try to divide the work as fairly as possible by asking who wants what tasks and we assign these tasks accordingly on Trello as to who volunteered for what. The set-up we have for managing our project and tasks within Trello has been useful for our group so we plan on keeping this the same because we have not run into any issues thus far. Using Slack has also helped the group communicate with one another. Overall our project management plans have worked for us, therefore, we plan on keeping everything the same.