

CSC 648-03 Fall 2019 Software Engineering

Milestone 2:

a) Architecture, UI Mock-Ups & GUI Design

b) Vertical SW Prototype

Team 103 - *Fridge Tracker 9000*

Due: October 25, 2019

Joshua Lizak Project Lead <i>jlizak@mail.sfsu.edu</i>	Nina Mir Scrum Master / Front End Developer	Abhishek Mane Back End Lead
Wilson Xie Full Stack Developer	Douglas Hebel Front End Lead	Gangzhaorige Li GitHub Master / Back End Developer

Date Submitted For Review:	Date Revised After Feedback:
October 25, 2019	~

1. Data Definitions V2

Users

All application access requires a registered user to be logged in. Any visitor to the website who is not logged in will be presented with the main login page. Once signed in, a user gains access to all application functionality. A user can be an **Owner** of a fridge and/or a **Friend** of other fridges that were created by users. We have no separate users for owners and friends thus allowing a single account to be applied to multiple fridges no matter as a friend.

Functionality:

- Any user can create a refrigerator
- Can delete refrigerators that they created (owned refrigerators)
- Add other users as friends of refrigerators that they own
- Can add items to their own fridge(s) or fridges they're friends-of by uploading a picture of their receipt and submitting the OCR discovered items
- View the inventory of their fridge(s) as well as fridge(s) they are friends of
- The creator of a fridge (the owner) can edit the items that are being tracked by the fridge
- Any owner or friend of a fridge can view the grocery list associated with that fridge.
 - This includes the auto generated list based on missing tracked items
 - Any user can add items into the grocery list manually
- Add and remove personal notes viewable only by the single user
- Can view possible recipes based upon the contents of a selected fridge
- The owner of a fridge can add and remove friends to/from the fridge

Attributes:

- Name (String)
- Username (String)
- Password (String)
- Email (String)
- Owned Fridges (List)
- Friended Fridges (List)
- Saved Recipes (List)
- Personal Notes (String)
- Primary Fridge (Integer - Fridge ID)

Refrigerator [Details] (Dynamic)

Refrigerators have a single owner (the user who created the fridge) and a list friend users that is maintained by the owner. Each fridge contains a list of the items that it holds.

Functionality:

- Items can be added via receipt by the **owner or friends** of the fridge
- Maintains a list of the items that have been added and removed by its users
- Maintains a list of tracked items set by the owner
- Organizes its contents by their date of expiration
- Modifying the fridge name, adding and removing friends, and changing tracked items is only accessible to the owner of the fridge

- Items are displayed using **different colors** depending on its age relative to its expiration date. **Red is expired and yellow is close to expiring.**

Attributes:

- Fridge Name (String)
- Owner (Integer - User ID)
- Friends (List)
- Items (List)
- Date of Creation
- Date of Deletion
- Grocery List (List)
- Tracked Items (List)

Item (Static)

Items are the groceries that users keep track of, add to the fridge, and consume. A static database of a set number of items will be maintained by us. Only these specific items will be available for users to track and add to their refrigerators.

Functionality:

- Items have a set name, caloric value, age, date added, and perishability.
- When added into a refrigerator, the item has its Date Added and user who added it applied to it
- Upon deletion from the fridge, a date of deletion will be applied to the item and it subsequently will not be shown in the fridge

Attributes:

- Item Name (String)
 - Size (If Applicable)
 - Expiration Age (time from input)
 - Perishable (Boolean)
 - Calories (Integer)
 - Date Added
 - Date Deleted
 - Added By (String/Integer-User ID)
-

2. Functional Requirements V2

Priority 1:

- Create and log into user accounts
- Users can own multiple fridges
- Users can be friends of multiple fridges
- Fridges have one owner and multiple possible “friends”
- Owners can add and remove friends to/from fridge
- Users can upload photos of receipts, items on that receipt will be recognized and matched to the static item database then added to fridge upon user confirmation
- Fridges store items. Items have expirations which are made easily visible to the user and will be notified close to their expiration date
- Items can be added and removed at any time by any user who has access to the fridge
- Grocery lists automatically generated based on tracked items
- Recipes will be recommended to users based upon selected contents of the fridge
- Owners can edit tracked items, change fridge name, add/remove friends, and delete the fridge

Priority 2:

- Adding/removing items to the grocery list manually
- Users each have personal notes accessible only by them
- Users can save recipes they’d like to try

Priority 3:

- Manual addition of individual items into the refrigerator
 - Automatic recipe suggestions based on contents of fridge
 - Monthly Food Consumption Report to show user’s food and nutrition consumption status.
-

3. UI Mockups and Storyboards (high level only)

Mockups

Homepage

This is the page all users see upon accessing the website.

The image displays two hand-drawn UI mockups side-by-side. The left mockup is titled 'Main Landing Page (not signed-in)' and features a 'LOGO' at the top, the text 'Let's Manage Your Fridge!' with a refrigerator icon below it, and two buttons at the bottom: 'Login' (marked with a green circle and the number 1) and 'Sign Up' (marked with a green circle and the number 2). The right mockup is titled 'Signup Page' and features a 'LOGO' at the top, a small square icon, and the text 'Create an Account'. Below this are four input fields labeled 'Your Name', 'Username', 'Password', and 'Email', followed by a 'Submit' button.

(1) Login button to login to our app [Landing Page]

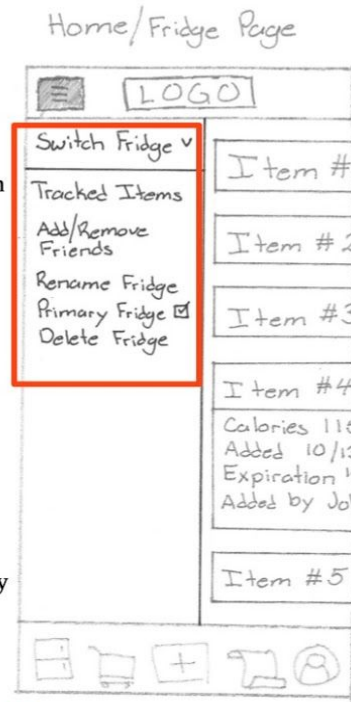
(2) Sign Up button directs the user to a dialogue box to create an account as seen above right.



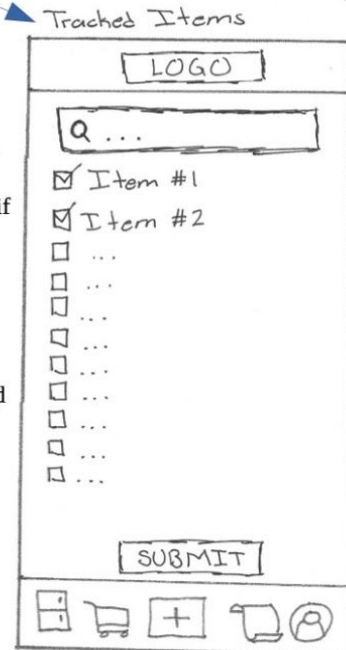
- (3) Hamburger button when untoggled allows the user, depending on their relationship to a fridge, (owner or friend), to perform certain actions.

The figure on the right is an example of the untoggled hamburger button options owner of a fridge sees [in a red frame for emphasis]:

- Add/remove Friends
- Rename fridge
- Delete Fridge
- Select fridge as a primary fridge
- To edit tracked items



- (4) Logo placeholder of our app
- (5) If a user has access to no fridge (owned/friended), they can click on this button to create a fridge. A dialogue box will appear asking for the fridge name and friend's user names, if any.
- (6) Fridge button (aka Home button); clicking on this button takes the user to the fridge page.
- (7) Grocery Cart button: clicking on this button results in viewing the auto-generated grocery list based on the tracked grocery item list. More info in Grocery Page sub-section.
- (8) Plus button allows the user to upload an image of their grocery shopping receipt and submit it.
- (9) Scroll button: clicking this button takes the user to Food Recipe landing page.
- (10) Dummy figure button takes the user to their profile page.



Home/Fridge Page

LOGO

Create Refrigerator

Name

Find Friends

Q

Next

Icons: Home, Cart, Add, Remove, Profile

Create a fridge dialogue box as explained in number (5) in the previous page.

Home/Fridge Page

LOGO

Your refrigerator is empty!
Click the + to add some.

Icons: Home, Cart, Add, Remove, Profile

A fridge that contains no grocery items!

Home/Fridge Page

LOGO

Item #1 ✓

Item #2 ✓

Item #3 ✓

Item #4 ✓

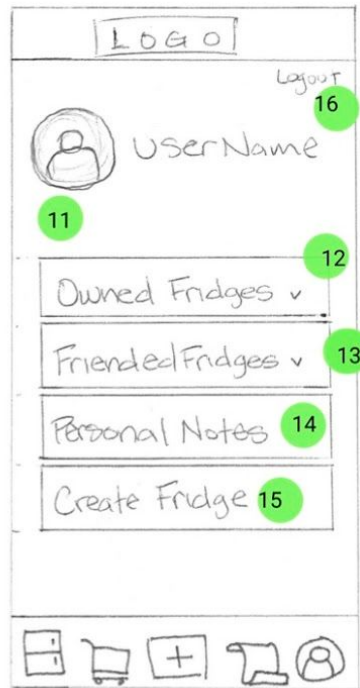
Calories 115
Added 10/13
Expiration 10/23
Added by John

Item #5 ✓

Icons: Home, Cart, Add, Remove, Profile

A full fridge containing multiple grocery items. Upon clicking on any Item card, a drop down menu is displayed containing information about the item nutritional value, quantity, expiration date, added by which user, and the ability to delete the item from the fridge by pressing on the Trash Bin button.

Profile Page

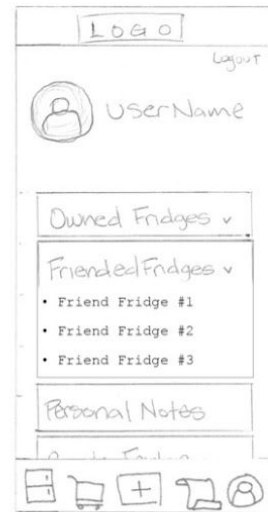


Profile Page (nav bar profile picture button)

(11) User's picture

(12) Clicking owned Fridges button rolls out the list of fridges a user has created/owns.

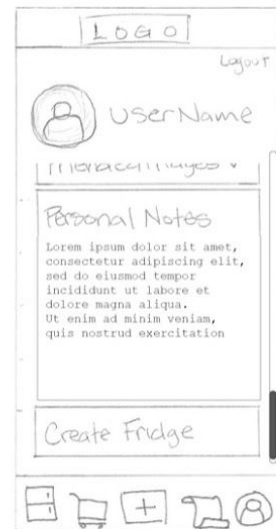
(13) Clicking Friended Fridges button rolls out the list of fridges as shown below [similar to (12) above]:



(14) Clicking Personal Notes button results in a drop down notepad as shown below:


(15) Create Fridge button does exactly as explained in bullet number (5) earlier in this section.

(16) Logout button.



Grocery Page (nav bar grocery cart button)

Grocery Page

 LOGO

Missing These Items:






- Item #1
- Item #2

Manual Items: + 17

• Item #1 ✓


• Item #2
Added by John
Added on 10/13/19
Remove 18

• Item #3

(17) Add button to manually add grocery items to the Grocery list:

Grocery List Manual Item Add

 Logo

Missing These Items

☐ Item 1

☐ Item 2

☐ Item 3

☐ ...

☐ ...

☐ ...

☐ ...

☐ ...

☐ ...

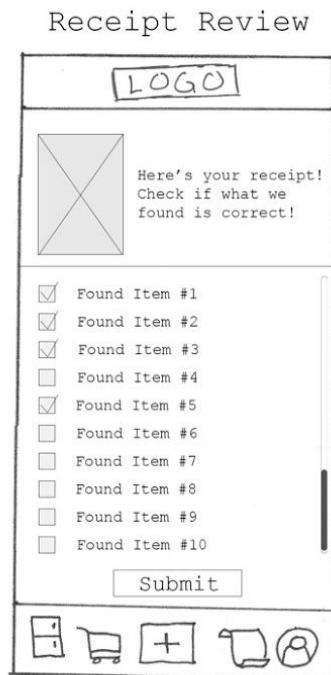
Submit

(18) Remove button to delete a manually added item.

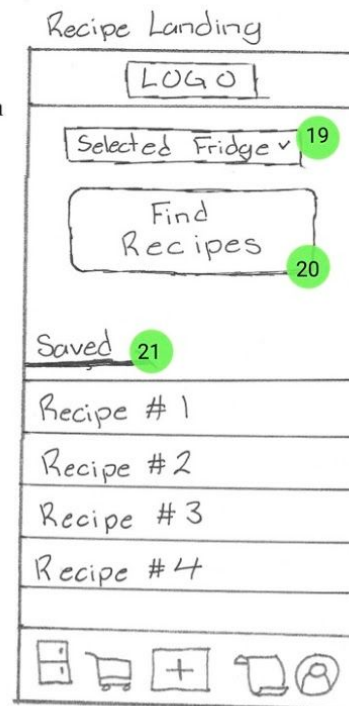
Receipt Review (nav bar + button)

Upon uploading a grocery receipt to our app, the user will be asked to verify the contents of the grocery receipt that our OCR engine returns as displayed in here.








Recipe Page (nav bar's scroll icon)






- (19) Selected Fridges button allows the user to choose which fridge is the focus of the Recipe finding process
- (20) This button allows the user to select certain ingredients based on the content of the current fridge for the recipe finding process. The sequence of Finding recipes is shown in the next page.
- (21) The list of already saved recipes by the user.



Find Recipes 1

LOGO
Select the items you wish to use
<input type="text" value="a ..."/>
<input type="checkbox"/> Item 1
<input checked="" type="checkbox"/> Item 2
<input type="checkbox"/> Item 3
<input type="checkbox"/> Item 4
<input checked="" type="checkbox"/> ...
<input type="checkbox"/> ...
<input type="checkbox"/> ...
<input type="checkbox"/> ...
<input type="checkbox"/> ...
<input type="checkbox"/> ...
<input type="checkbox"/> ...
Next+→
    

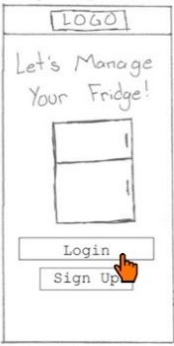



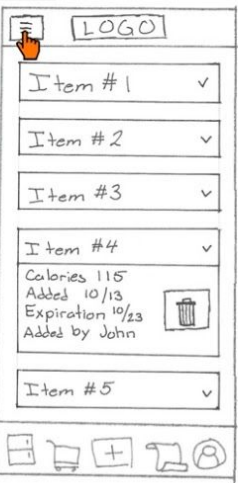
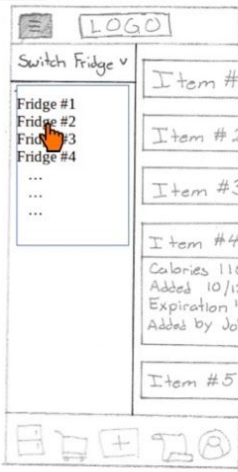
Find Recipes 2

LOGO
Suggested Recipes
Recipe 1
Recipe 2
Recipe 3
Recipe 4
http://www.url-recipe-treasure.com /link-to-recipe-#57482
...
...
...
    

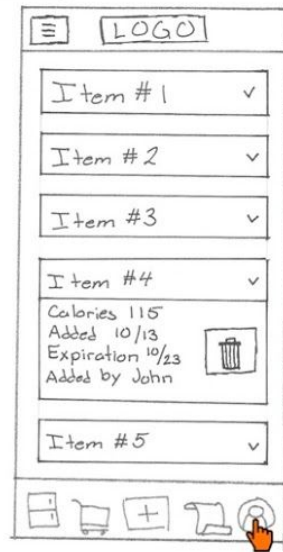
Storyboards

User Story 1: Nyla (Viewing Friends' Fridges and Adding Personal Notes)

Nyla Mir is a high school student who works a part-time job as a Subway fast-food sandwich artist in the suburbs of Austin, and lives with her dad, but she visits her mom's house on the weekends. She is a vegetarian and needs to check what is inside both her parents' fridges to figure out what will need to buy for her day.

<p>1</p>  <p>Landing Page Nyla goes on the app using her phone.</p>	<p>2</p>  <p>Login She types her username and password and then clicks submit.</p>	<p>3</p>  <p>Profile Page She then goes into her profile, where she can access her friended fridges.</p>
<p>4</p>  <p>Friended Fridges Nyla finds her mom's fridge among her friended fridges.</p>	<p>5</p>  <p>Viewing Her Mom's fridge content She clicks on her mom's fridge and looks for strawberry in her mom's fridge -- there's no strawberry sadly!</p>	<p>6</p>  <p>Looking at her dad's fridge She clicks on the hamburger button and switches to her dad's fridge to see if her dad has strawberry or not.</p>

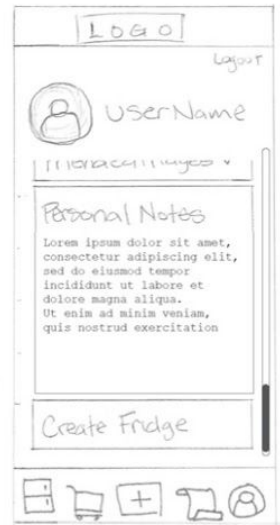
7



Dad's Fridge

Nyla's Dad's fridge does not have strawberry as well! Bummer!

8

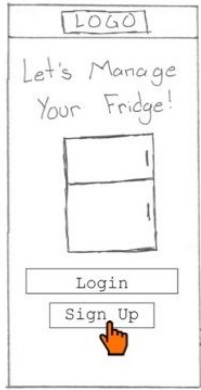

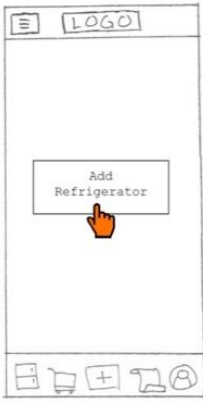


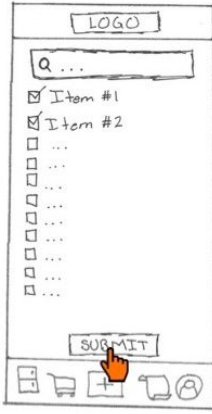


Personal Notepad Fridge

After not finding it there either, she goes into the profile and into her personal notes, where she makes a note to buy strawberries.

User Story 2: Jennifer (Creating a fridge and adding friends)

Just discovered this web application and wants to start using it to track items in her brand new refrigerator. She will need to create the fridge within the app and add her family onto it as friends.

<p>1</p>  <p>Landing Page</p> <p>This is the first time Jennifer is visiting the website therefore she would like to create an account to start managing her fridge. She clicks on "Sign Up" to begin the process.</p>	<p>2</p>  <p>Create Account</p> <p>Jennifer fills out her name, username, password, and email to complete the account sign up process and then clicks submit.</p>	<p>3</p>  <p>No Refrigerators</p> <p>Because this is the first time Jennifer has logged into her account, she doesn't have any associated refrigerators. She is presented with the option to add her own refrigerator.</p>
<p>4</p>  <p>Create New Refrigerator</p> <p>The "Add Refrigerator" button took her to the "Create Refrigerator" page which asks her for a name for the fridge as well as allows her to add friends.</p>	<p>5</p>  <p>Add Friends</p> <p>Jennifer would like to add her husband and kids to the fridge. They already have accounts which allows her to add the friends by clicking on "Find Friends" and then searching for them by username.</p>	<p>6</p>  <p>Edit Tracked Items</p> <p>After clicking on the Next button, Jennifer is brought to add her tracked items. She checkmarks all the items she would like to always have in the refrigerator and clicks the submit button.</p>

7


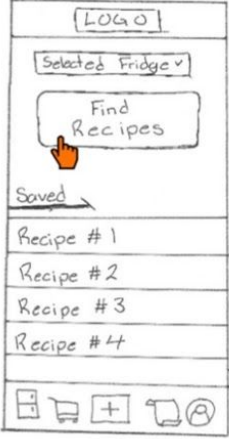
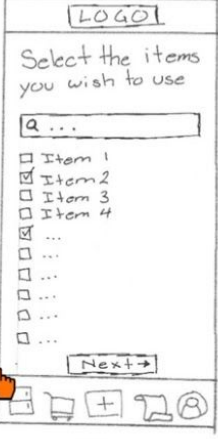



Home Fridge

Jennifer's refrigerator is now fully set up. She has no items in it, so she is prompted to add some by clicking on the "+" button on the toolbar.

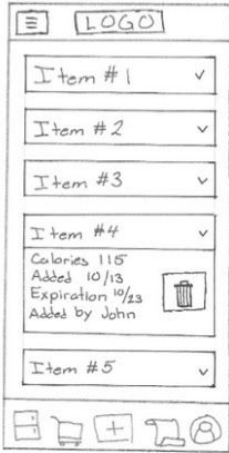
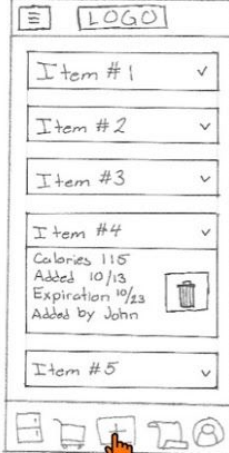
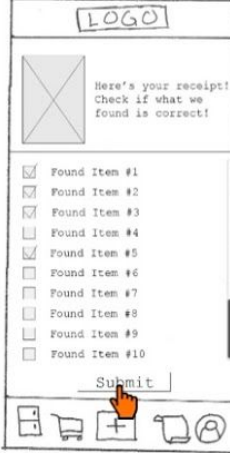
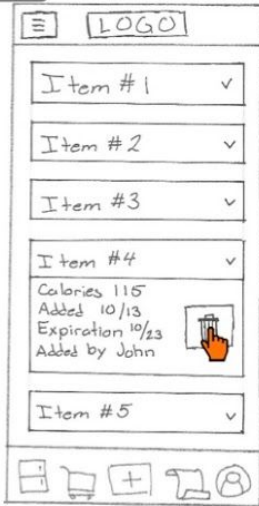
User Story 3: Josh (Finding Recipes and making a meal & Adding Manual Grocery Items to Friend's Grocery List)

Josh needs photos for his food instagram page and wants to check what items are in the fridge for his next meal/shoot. Josh will take items from the fridge and make a beautiful meal from them.

<div>1</div>  <p>Landing Page Go to webpage (already logged in and can view fridge as a friend) - scrolls through fridge landing page - sees all the items he wants to use for his shoot - uses the items in the green section to make sure he only uses the freshest ingredients.</p>	<div>2</div>  <p>Create Account Goes to the recipes tab, selects "Find Recipes".</p>	<div>3</div>  <p>No Refrigerators Selects the ingredients he wants to use.</p>
<div>4</div>  <p>Create New Refrigerator Scrolls through returned recipes and selects one based upon the prettiest image - Views the recipe.</p>		

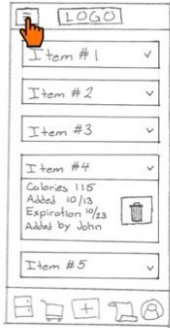


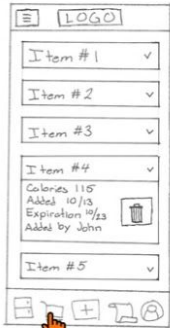
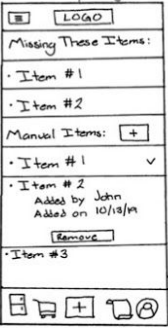
User Story 4: John (Receipt Capture and Review, also deleting items)

John is working at a startup in downtown San Francisco, and is about to clock out to pick up his 10 year old daughter at school. He picks up his daughter and goes grocery shopping at 5 o'clock.

<div>1</div>  <p>View Items</p> <p>While shopping, John opens up the app on his phone, and since he is always logged in, the first thing he sees is the contents of his fridge. He now knows what he has in his fridge for shopping.</p>	<div>2</div>  <p>Upload Receipt</p> <p>After he is done shopping, he takes a picture of his receipt and presses the big plus button on the nav bar to upload that picture to the app.</p>	<div>3</div>  <p>Receipt Review</p> <p>After uploading the picture, the app then props him with what will be added to the inventory of the fridge.</p>
<div>4</div>  <p>Delete Items</p> <p>He goes home and cooks dinner for him and his daughter, then removes the items that he uses for that meal.</p>		

User Story 5: Joseph (Grocery List & Tracked Item Editing)

Joseph rarely goes shopping for groceries. When he does go shopping, he needs to be sure he's getting only what he absolutely requires. Today, Joseph is going to the market again after two weeks.

<div>1</div>  <p>Fridge Home</p> <p>Joseph opens his Fridge App 9000 application and take a quick look at what is currently inside his fridge. He would like to add strawberries to his list of tracked items that way he always knows if he has them in his refrigerator.</p>	<div>2</div>  <p>Hamburger Access</p> <p>Joseph clicks on the hamburger on the home screen and goes into the contextual menu to check his tracked items. From there he clicks on the</p>	<div>3</div>  <p>Edit Tracked Items</p> <p>He clicks on the search bar and starts typing in "straw" the strawberry result populates and he selects the item's check mark. Once finished, he clicks "Submit".</p>
<div>4</div>  <p>Checking Groceries</p> <p>Joseph now drives over to the market to purchase the items. He opens up the app and clicks on the shopping cart to check the automatically generated grocery list.</p>	<div>5</div>  <p>Viewing The Grocery List</p> <p>At the market, he opens the application and clicks on the shopping cart button to bring up the grocery list. Joseph purchases all the items on his list and takes the receipt home with him.</p>	

4. High level Architecture, Database Organization

Database Organization

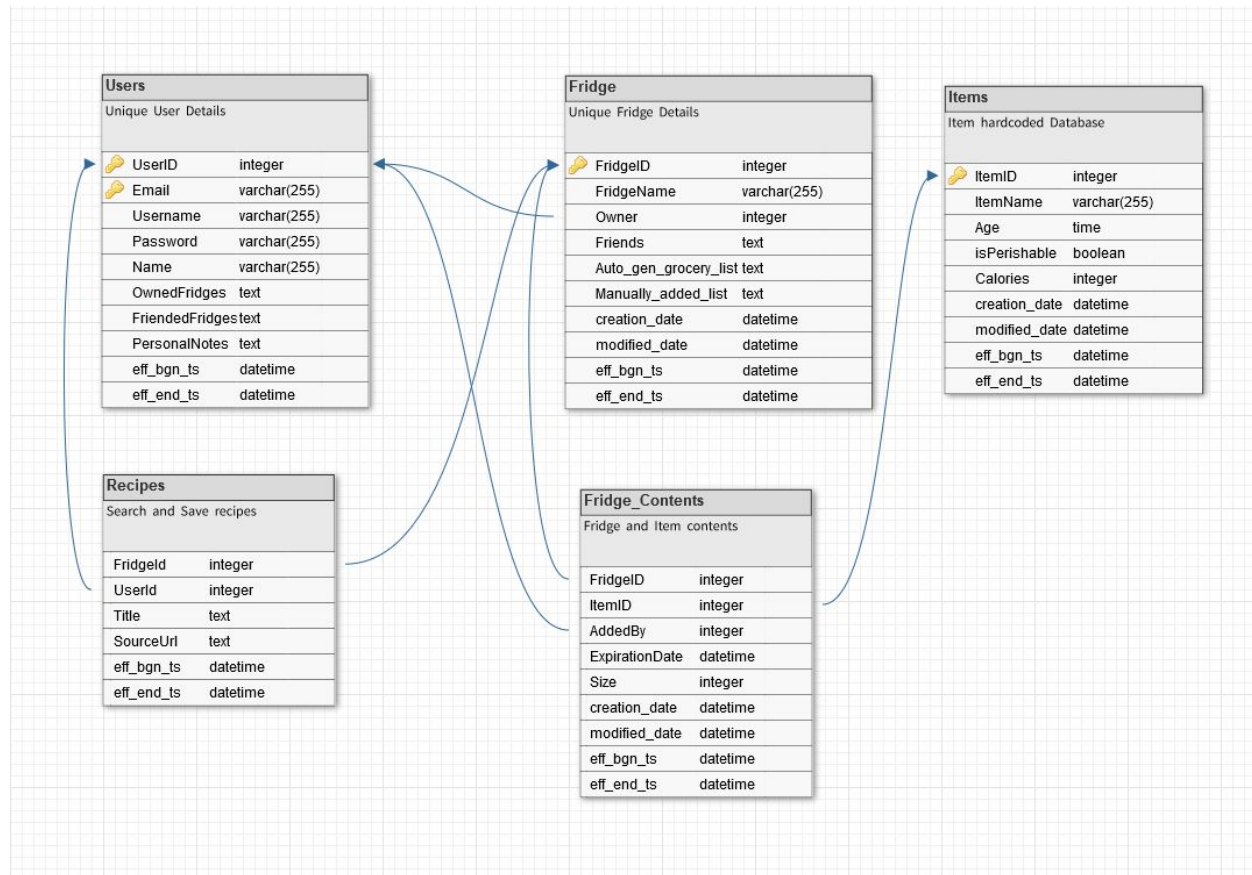
On a high level overview of database we have -

- Users - Stores all user information.
- Fridge - Which has all the fridge information.
- Items - Which has all the item information.
- Fridge_Contents - Which records all items stored in a fridge.
- Recipes - All the recipes a user has searched and saved for a fridge.

The relation between these entities is -

- The owner has a relationship 'creates a fridge' with the fridge table such that no fridge can be without an owner.
- The user can be differentiated as an owner or friend based on it's relationship with the Fridge.
- The Fridge_content entity has a relationship with items, Fridge and Users where no entry will be invalid without an item, fridge or an owner of an item.
- The recipes entity has a relationship of 'Save/view recipes' with the Users and fridge entities.
- The owner and friends have an 'add/remove/modify items' relation with Fridge_Contents database.

ER Model Diagram of Our Database



Media Storage

Our website will take in any type of images with capacity of less than 500 KB for the best performance even though the limit of an individual image capped at 20MB. The image files will be saved in the folder called “MEDIA” on the server. Any user who has a fridge has the ability to capture the receipt and upload the image to the website.

Search/Filter

- User can search for a friend using the email address to add into the refrigerator.
- System searches for an item which are tracked in the tracked item list to check if the item’s quantity is running low.
- User can search for a tracked item to check or check off an item from the tracked list manually.

Non-Trivial Algorithms

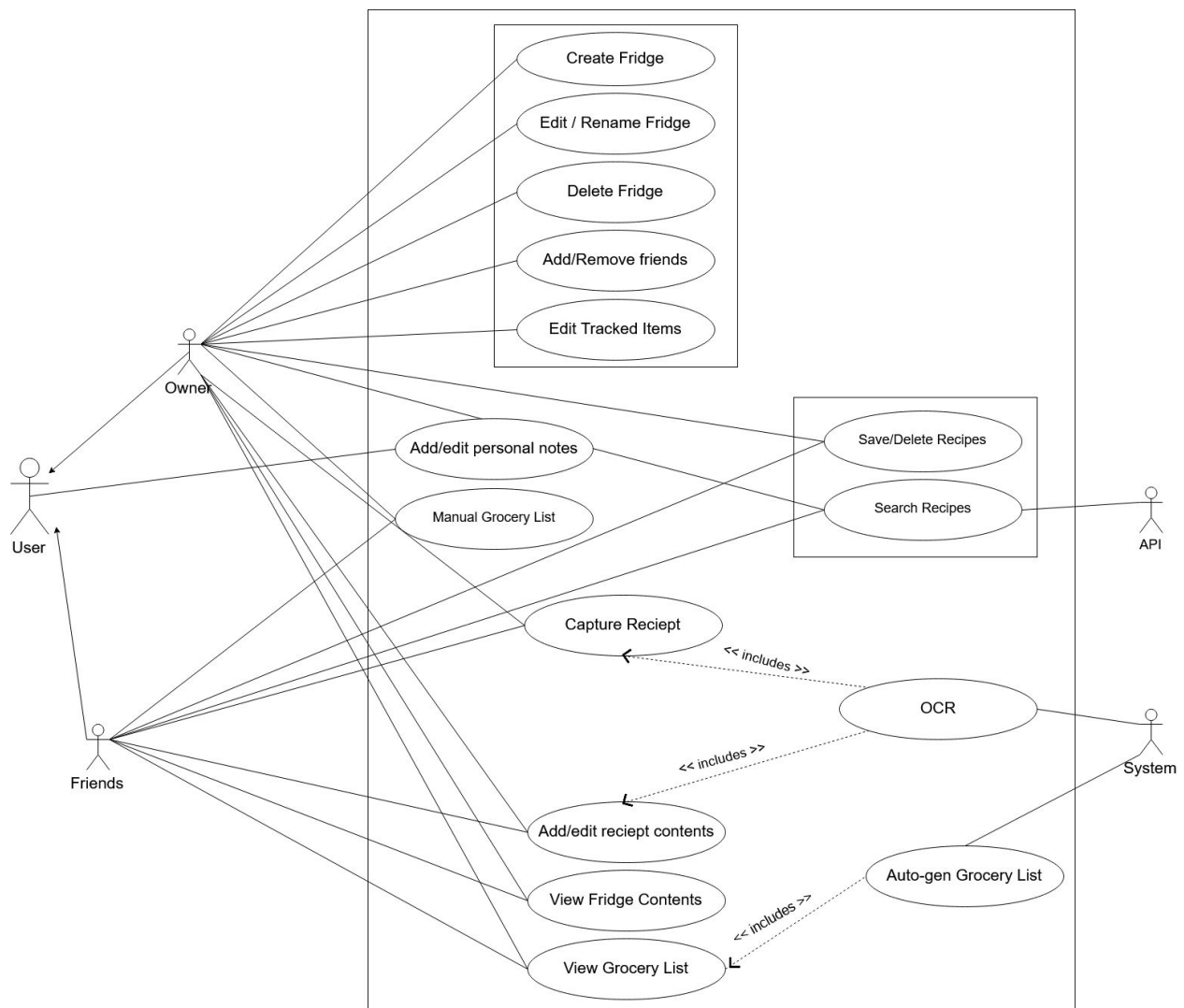
Possible monthly statistic calculation of calorie consumption.

Any changes in Software

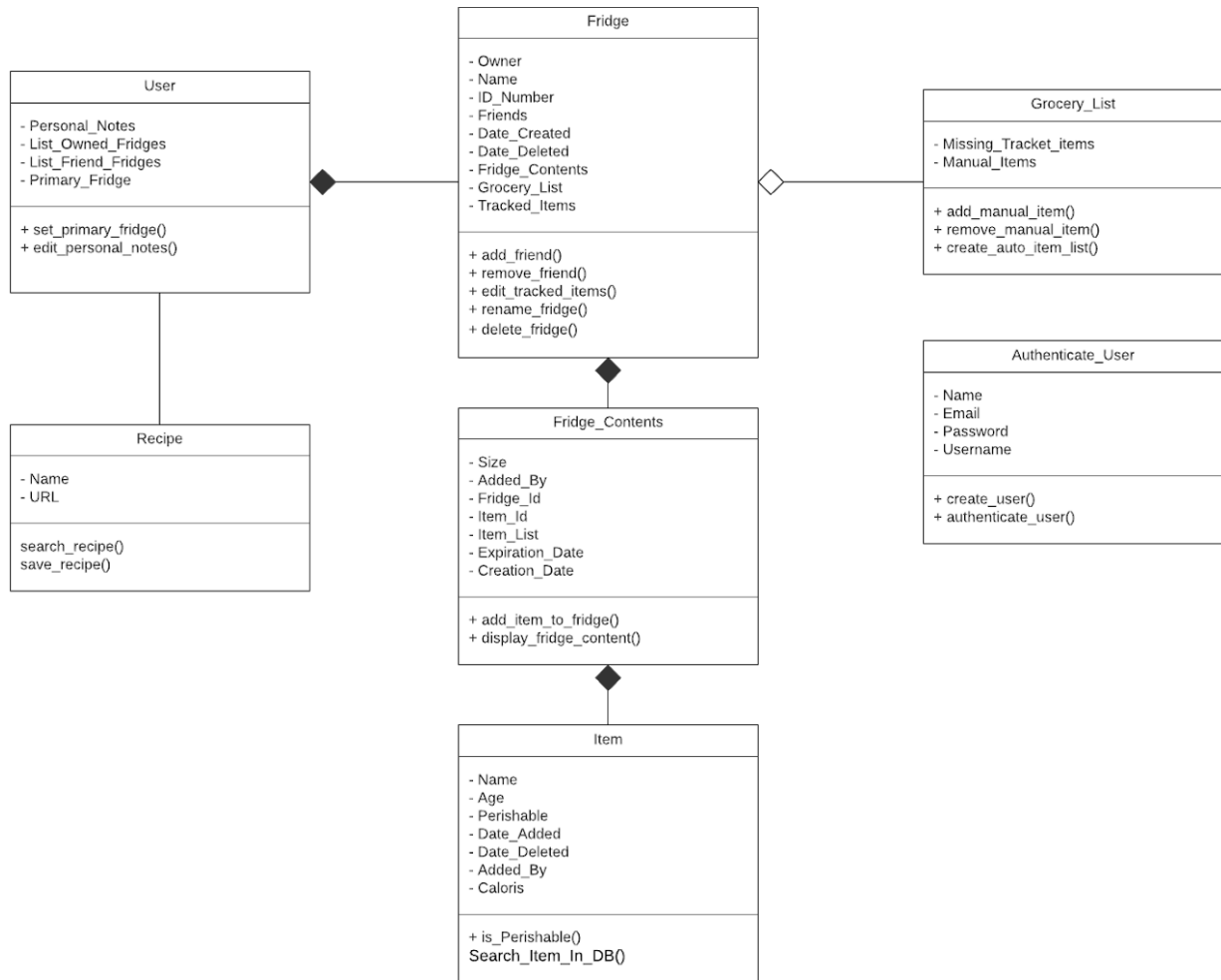
So far there is no significant change to the core software at this time.

5. High Level UML Diagrams

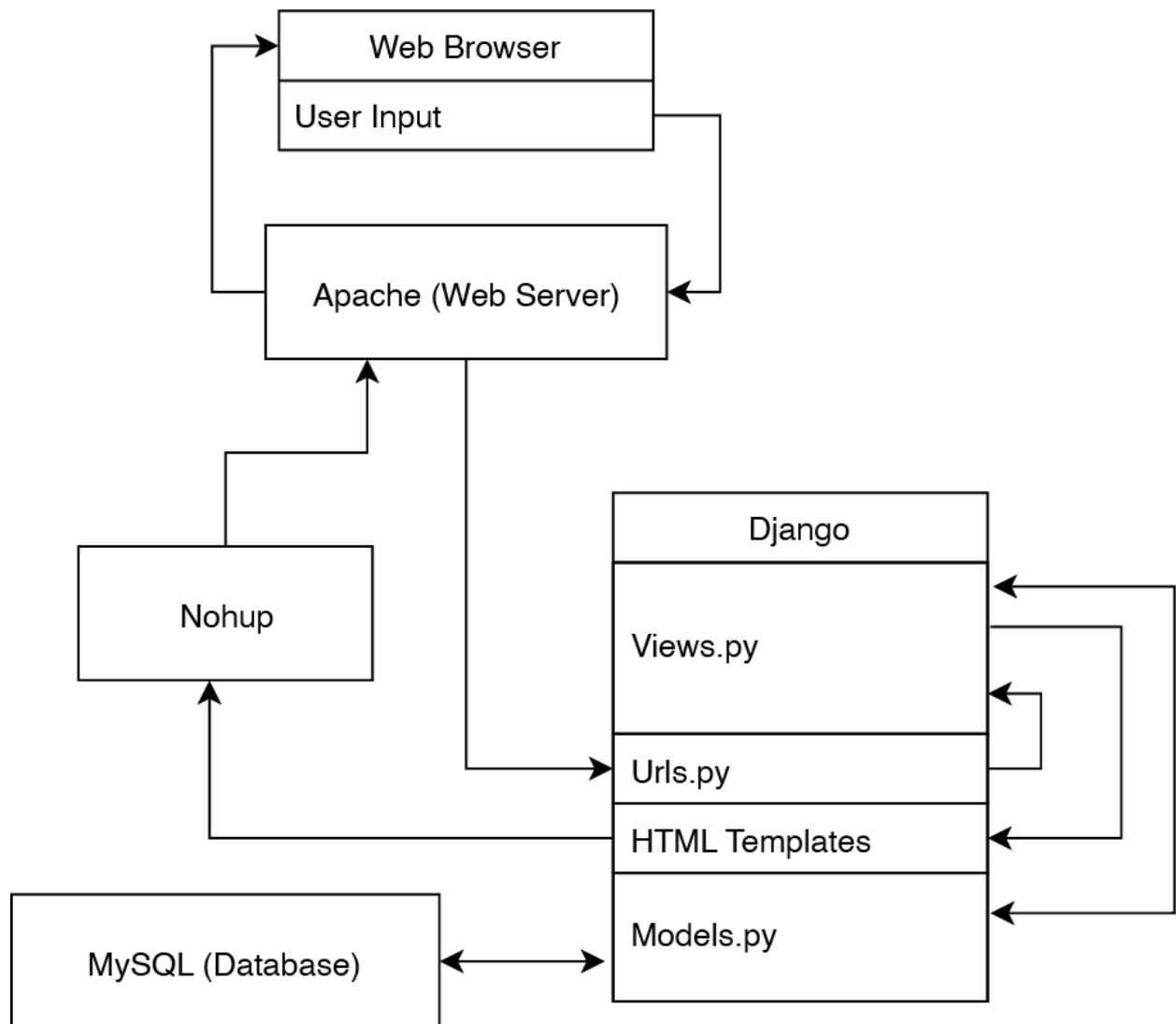
Use Case Diagram



UML Class Diagram



UML Deployment Diagram



6. Identify actual key risks for your project at this time

- **Skills risks (do you have the right skills)**

All of us are new to the Django Framework, and some of us are not completely comfortable with python, so we need to spend time learning the basics before we can dig into coding. We have been looking at tutorial videos and documentation to study how to build certain features for our app.

- **Schedule risks (can you make it given what you committed and the resources)**

One of our teammates has trouble meeting up on our big Saturday meetings. So we try to have some smaller meetups throughout the week to get everyone up to speed.

- **Technical risks (any technical unknowns to solve)**

The APIs we use and Django itself may prove to be difficult as we get further along in development, especially when we need to pull everything together at the very end. Learning where to adjust certain variables in Django and hooking everything up will need to be achieved quickly, so we can mitigate the chance that not every feature will be implemented in time. We plan on doing features that are feasible and not too complex for us to implement, allowing us the time to do research and finish on time.

- **Teamwork risks (any issues related to teamwork)**

Our team have not committed to GitHub that much and we haven't coordinated much on what tasks will be divided up when we start coding. Our team usually does a lot of the work in our group meetings, but we will start to assign specific coding tasks once we have a proper Django project directory setup on GitHub. Our knowledge of Django at this point varies between teammates, but the people with stronger grasp of Django can assist the others to ensure everyone participates in the coding process.

- **Legal/Content risks (can you obtain content/SW you need legally with proper licensing, copyright)**

There are no legal risks currently being tracked with our project. However, we are using third-party icons which may have copyright protections. On many of these non royalty-free icons the license requests that we display the author of said icons. To keep up with legal policy we plan to implement this disclaimer later.

7. Project Management

In the three week span this project accumulated, as a team, we have accomplished much of this project in person during our meetings. Our team originally planned to meet as a whole each Saturdays for fast-paced group discussion and prototyping. Unfortunately this meeting time did not end up being practical for every member of our team which led to us meeting on Tuesdays and Thursdays as well.

During class we were introduced to the Trello from another group. We've applied this task tracking application to our strategy and so far it is going very well for us. As far as documentation, we have been developing our milestone in a Google Document which allows for collaborative addition do the document. It has been very helpful having all six of us working on the document at the same time.

Creating the wireframes for this milestone, while initially intended to be done using Figma, were eventually done by hand and then modified using image editing applications. Due to our complications working with and learning Figma's interface and working collaboratively we decided against using it and moved forward with hand drawn prototypes. We made drawn prototypes of most of the screens and functions of our application and finished the rest of the wireframes using Photoshop to modify the wireframes that were already made.

Our backend team took on the development of the UML class and deployment diagrams based upon the functionality we are planning for the application. Throughout this milestone the backend team got Django up and running on our server and connected it to our MySQL database to create a basic demonstration page to be used along with this milestone.

Going forward we will start the development of the application itself. All of our team is working on learning the front to backend relationship within Django. While this learning curve may be steeper for some in our team, we do plan on getting everyone on track for successful development of a connected frontend and backend.