Sprint 2 Plan
Product Name: Recipe Retriever
Team Name: RexFetch
Sprint Completion Date: Tuesday–May 09, 2023
Revision Number:
Revision Date:

Goal: Connect frontend and backend, and actually fetch recipes.

Task Lists:

User Story 1: As a user, I want to look up a recipe based on the ingredients in my kitchen. (MVP)
1. Set up recipe API calls and parsing the recipe data into the outgoing JSON format that is described in the communication_docs.md or backend_doc.md
2. Make frontend fragment that displays the received backend recipe data
3. Database communication for recipes (testing)
4. Communication documentation (describes the json objects being sent, request format, etc.)
5. Backend to frontend communication (implements http requests from server to app)
6. Frontend to backend communication (implements http requests from app to server)

User Story 2: As a user, I want my login information to be stored in a profile, so I can save some information about myself (favorite recipe, my fridge content, etc.)
1. Set up user database and integrate with server
2. Front end changes
   a. Grocery Page
   b. Dynamically Create Recipe Cards
   c. Spinner
   d. Shared View Model for Data Sharing
   e. Settings - (Substitutions & allergens)
3. Synchronize data with live front-end (authentication)
4. Make a new user functionality versus returning user authentication.

To Do

Communication stack
≡

Parse and format incoming back-end data

Recipes coming from various sources

Parse incoming front-end requests

Parse incoming recipe data

Set up user database and integrate with server

Front end changes

Synchronize data with front-end

Storing fridge inventory
☑ 0/5

Front end design
≡

Store data locally

➕ Add a card

Doing

Edit ingredients data in front end
KY

Add ingredients data in front end
KY

Remove ingredients from front end
KY

Server design
JK

Login
☑ 2/3   PA VN

➕ Add a card

Code Review

Recipe caching // database
⬗ 2 👁 ≡ 💬 4   KH VN

➕ Add a card

Done

Github

➕ Add a card

➕ Add another list