Test Plan And Report

Product Name: Recipe Retriever

Team Name: RexFetch

Members: Kim Ho, Vyankatesh Nandapurkar, Jackson Kohls, Kevin Yosifov, Prithvi Arunshankar

Testing approaches used:

Backend - Automated tests that our group members created.

Test can be found in reciperetriever/backend/testing/ and reciperetriever/backend/src/testing/

They test 1. That the server is operational, 2. That clients can be created and stored in a client database, and 3. That the server can handle clients requesting/updating recipes in the recipe database. These tests were run every week just to make sure that any new code didn't mess with the existing code. These tests were made in the beginning of the project and were used all the way until the end.

The tests created tested mainly that the databases worked as expected and were black box tests, i.e. they tested the functionality rather than the structure of the functions.

In addition to automated tests, the backend code was tested anytime that the code base was changed but those tests were manual and generally more localized than the automatic tests.

Frontend - Fully manual tests. Whenever a task in the frontend was completed and pushed, the code was tested to make sure that it was functionally sound as well as tested that it didn't interfere with any other code (cause bugs). The entire frontend was then manually fully-tested once a week (generally before a TA meeting) to make sure that *everything* both old and new functions were fully functional and operational (Full Path Coverage). At the end of the project the frontend was then peer-tested by other members of the group to fix any bugs missed due to programmars' bias. Just for fun, we gave the app to actual users (roommates and friends) and told them to find any errors/bugs if they can. These tests seemed even more thorough than the Full Path Coverage tests (that's a joke, but in all honesty those tests seemed to include a lot more stress testing than the ones performed by the programmers as the users were excited to destroy the application).

Summary: The Backend utilized automated tests since week 2 in order to prevent any errors from arising in the server or databases. The Frontend, on the other hand, used manual tests and tested the code both locally (regularly) and globally (once a week).

Scenarios:
Frontend:
Scenario One - “Check Registration Page & User Login is functional”

1. Start Recipe Retriever App
2. Click “New User”
3. Enter “sampleUsername” for Username
4. Enter “myPassword” for Password
5. Enter “myPassword” for Re-Enter Password
6. Optional: Type “peanuts” for Allergens and click the “Add” Button
7. Optional Verification: “peanuts” should pop up as an item in the scrollable list
8. Click the “Register” Button
9. Enter “sampleUsername” for Username
10. Enter “myPassword” for Password
11. Click Login
12. Verification - The User should be in the home page and there should be a message on the bottom of the screen which reads “Welcome, sampleUsername”

Scenario Two - “Makes sure that the Fridge & Grocery page work and can communicate with each other (Also tests no-input & duplicates)”

1. Start Recipe Retriever App.
2. Complete Scenario One (i.e. Create New User and Log In)
3. Click “My Fridge”
4. Type “eggs” into the search bar and click “Add to Fridge” Button
5. Click “Add to Fridge” Button
6. Verification - A message “Null Error: I think you forgot to type a food item” should pop up at the bottom of the screen.
7. Type “eggs” into the search bar and click “Add to Fridge” Button
8. Verification - A message “Duplicate Error: Item is already in the fridge.” should pop up at the bottom of the screen.
9. Type “milk” into the search bar and click “Add to Fridge” Button
10. Type “eggs” into the search bar and click “Move To Grocery” Button
11. Click “Move to Grocery” Button
12. Verification - A message “Null Error: Nothing was typed in the text input” should pop up at the bottom of the screen.
13. Type “potato” into the search bar and click “Move To Grocery” Button
14. Verification - A message “Move Error: Item to move not found” should pop up at the bottom of the screen.
15. Click “Back To Home Page” Button
16. Click “My Grocery List” Button

17. Verification - “eggs” should be an item in the scrollable list under the text “My Grocery List”. (Verifies that the fridge can send ingredients to the grocery list. This note was added because this wasn’t self-explanatory).
18. Type “sausage” into the search bar and click “Add to Grocery” Button
19. Click “Add to Grocery” Button
20. Verification - A message “Null Error: I think you forgot to type a food item” should pop up at the bottom of the screen.
21. Type “eggs” into the search bar and click “Add to Grocery” Button
22. Verification - A message “Duplicate Error: Item is already in the grocery.” should pop up at the bottom of the screen.
23. Type “milk” into the search bar and click “Add to Grocery” Button
24. Type “milk” into the search bar and click “Move To Fridge” Button (Note: milk should disappear from the grocery page but when going to the Fridge Page milk should only appear once not twice)
25. Click “Move to Grocery” Button
26. Verification - A message “Null Error: Nothing was typed in the text input” should pop up at the bottom of the screen.
27. Type “potato” into the search bar and click “Move To Grocery” Button
28. Verification - A message “Move Error: Item to move not found” should pop up at the bottom of the screen.
29. Click “Back To Home Page” Button
30. Click “My Fridge” Button
31. Verification - “milk” should be an item in the scrollable list under the text “My Fridge”. (Verifies that the grocery page can send ingredients to the fridge page & makes sure that the fridge page doesn’t contain duplicate “milk” entries. This note was added because this wasn’t self-explanatory).
32. Verification - Feel satisfied with the work you’ve done (Optional)

Scenario Three - Make sure the “Menu” works as intended

1. Start Recipe Retriever App.
2. Complete Scenario One (i.e. Create New User and Log In)
3. Click the menu button (three dots on the top right corner) and Click “Profile”
4. Verification - This should navigate to the Profile Page
5. Click “Back To Home Page”
6. Navigate to another page and repeat steps 3 through 5.
7. Repeat step 6 until all pages are exhausted (Except Login and New User Page as they are inaccessible from the Home Page).
8. Click the menu button (three dots on the top right corner) and Click “Log Out”
9. [Now on Login page] Click “Profile”

10. Verification - A message “Please log in before going to the Profile Page” should pop up at the bottom of the screen.
11. Click “New User” Button
12. Click “Profile”
13. Verification - A message “Please log in before going to the Profile Page” should pop up at the bottom of the screen.
14. Click the menu button (three dots on the top right corner) and Click “Log Out”
15. Verification - User should be on the Login Page
16. Enter your account credentials (Scenario One: Steps 8-10) (i.e. Log In)
17. Navigate to a page in the app
18. Repeat steps 14 through 17 until all pages have been exhausted.
19. Verification - Feel Satisfied and Tired from doing all these tests. Note: This tests that the Menu is operational from any page (that it should be operational from).

Scenario Four - Make sure the “Profile” Page works as intended

1. Start Recipe Retriever App.
2. Complete Scenario One (i.e. Create New User and Log In)
3. Click “My Profile” from the Home Page or click the menu button (three dots on the top right corner) and Click “Profile”
4. Type “John Smith” in the Name field and click the “Save” Button
5. Verification - The text at the top of the profile page should change from “sampleUsername’s Profile” to “John Smith’s Profile”.
6. Click the “-” Red Button next to substitutions
7. Verification - “Substitutions: 2” should change to “Substitutions: 1”
8. Repeat step 6, 2 more times
9. Verification - A message “You can’t have less than 0 substitutions” should pop up at the bottom of the screen.
10. Click the “+” Green Button next to substitutions.
11. Verification - “Substitutions: 0” should change to “Substitutions: 1”
12. Repeat step 10, 99 more times
13. Verification - A message “You can't have more than 99 substitutions” should pop up at the bottom of the screen.
14. Click the “Save” Button Next to the Name Field.
15. Verification - A message “Null Error: You didn't input a valid name” should pop up at the bottom of the screen.
16. Optional: Type “peanuts” for Allergens and click the “Add” Button
17. Optional Verification: “peanuts” should pop up as an item in the scrollable list
18. Click the “Back to Home Page”
19. Click “My Profile” from the Home Page or click the menu button (three dots on the top right corner) and Click “Profile”

20. Verification - Make sure that all the data is consistent with the data from right before step 18.

Scenario Five - Testing Scrollable Views

1. Start Recipe Retriever App.
2. Complete Scenario One (i.e. Create New User and Log In)
3. Click “My Fridge”
4. Type “egg” into the search bar and click “Add to Fridge” Button
5. Type “milk” into the search bar and click “Add to Fridge” Button
6. Type “cucumber” into the search bar and click “Add to Fridge” Button
7. Type “potato” into the search bar and click “Add to Fridge” Button
8. Type “carrot” into the search bar and click “Add to Fridge” Button
9. Type “peanut” into the search bar and click “Add to Fridge” Button
10. Type “sausage” into the search bar and click “Add to Fridge” Button
11. Type “zucchini” into the search bar and click “Add to Fridge” Button
12. Type “cabbage” into the search bar and click “Add to Fridge” Button
13. Hold and drag from the “milk” item that is created and drag up.
14. Verification - The Scrollable List should show you all the items you put in your Fridge as you scroll.
15. Click the “Remove” Button next to an item and repeat steps 13 and 14 again for assurances that the list updates itself correctly.

Scenario Six - Testing Recipes & Specific Recipe

1. Start Recipe Retriever App.
2. Complete Scenario One (i.e. Create New User and Log In)
3. Click “My Fridge”
4. Type “egg” into the search bar and click “Add to Fridge” Button
5. Click the “Back to Home Page” Button
6. Click the “Choose a Meal” Button.
7. Click “Select” on the first Recipe
8. Verification - Button text should switch from “Select” to “Unselect”
9. Click “Cook” Button on first Recipe
10. [Should be taken to a Specific Recipe Page] Note: As the backend is not currently integrated at the time of this test I hard coded an exampleRecipe to test if moving between fragments is possible as well as displaying data on the new fragment.
11. Verification - The data on the Specific Recipe Page should be consistent with the Recipe whose “Cook” Button was Clicked (First Recipe Data)
12. Click “Back To Recipe Page”
13. Verification - [Should Go back to the Recipe Page]

14. ———————————————————————————————————————
    a. Scenario Seven - Filtering Function
15. Uncheck “milk” from the filtering Scrollable List and click “Filter”
16. Verification - [NOTE: NOT YET IMPLEMENTED] displays a different (smaller) set of recipes (namely those that don’t include milk as an ingredient (unless substitutions is > 0 in which case they may or may not.)) (Note; not implemented because as of the time the test was created the backend and frontend wasn’t integrated. Should work by the time the MVP is done)
17. ———————————————————————————————————————
    a. Scenario Eight - Randomizer Function
        i. Sub-Scenario 8A - Randomizer function with no selected recipes should select a random Recipe from the list
18. Click “Pick My Meal” Button (Assuming no buttons say “Unselect”)
19. Verification - The text “Here are the recipes you can make” should be replaced with the name of a random Recipe on the scrollable list
        i. Sub-Scenario 8B - Randomizer function with some selected recipes should select a random Recipe from the list of selected Recipes
20. Click the “Select” Button on one or more recipes
21. Click “Pick My Meal” Button
22. Verification - The text “Here are the recipes you can make” should be replaced with the name of a random Recipe from the recipes that you clicked the “Select” Button for.

That should be most if not all of the Frontend Tests that were used.

Unit Testing:
As mentioned on page one of this document, unit tests can be found in reciperetriever/backend/testing/
If you look at page one it should have a better explanation of what each one of the tests in this directory does.
For instance, client-db-testing/db_test.go is a test that simply makes sure you can write to and read from the client database without corruption of data using get_client()