Test Plan: BitOHealth

The 6 Bits:

Angel Cueva

Dante Secundino

Emily Sahyoun

Jacob Munoz

Rami Iskender

Sebastian Vasquez (Team Leader)

12/04/2021

Table of Contents

Document Purpose

The Test Plan for the BitOHealth progressive web application ensures that essential testing of every feature is defined and specified clearly. This document is to be developed based on the requirements the developer team created and the client's feedback. This document is intended to include all relevant information regarding the testing of the web application's features and should be referred to in relation with the project plan and network diagram.

a. Unit Testing
    - This is the most basic form of testing. We will isolate each feature and subfeature of the product to test. We will test for the correct output of all functionalities. It is essential to test like this before the product as a whole because a system needs all working parts before working together as a system. We will be able to automate this type of testing.
b. Negative Testing
    - The use of negative testing will guarantee that failure scenarios are handled properly within each feature of our code. Essentially, invalid user input will be entered into the system. The expected outcome during this testing is for all appropriate error messages and warnings to appear. However, if error handling was not implemented properly, it could potentially cause the entire system to crash. The use of this testing will identify these issues, allowing us to fix them prior to launch.
c. API Testing
    - Throughout our product, various APIs will be used to access information vital to feature success. We will be able to validate that information provided by the APIs is secure, reliant, performant, and functional. If an API function fails our tests, it can be addressed appropriately
d. UI Testing
    - Performing user interface testing is important to the overall experience of our users. It will test any controls the user can interact with to ensure all are working properly and that visual elements are appearing as implemented. This testing will guarantee that the UI is optimal for user experience and product flow.
e. Integration Testing
    - As mentioned in unit testing, a system needs all of its parts to work together to make a whole. Integration testing would test the system on how it works as a whole rather than individual parts. The expected

outcome is that all parts of the product communicate accurately and efficiently, from user input to being stored in the database. If a particular area of our product fails this testing, we can identify where the communication between parts failed so it can be fixed and addressed.

f. Regression Testing
    - This type of testing is necessary for our product because of its scalability support. It ensures that features are working as expected despite feature changes or additions. It will allow us to have a consistent product while still expanding.
g. Security Testing
    - Security testing entails assessing sufficient product protection of sensitive data like user passwords. Throughout testing, we can identify system vulnerabilities and set various methods to eliminate them.
h. Stress Testing
    - Various stress tests will allow us to confidently specify the highest load our product can sustain. Essentially, our product features will be pushed to their limits until they fail. We want to be able to guarantee that the product can still operate in some sort of capacity like supplying error messages despite being pushed to the limit.
i. Performance Testing
    - A performance test will test our system's speed, stability, and scalability. We want to ensure that the system executes and outputs within our specified time constraints. Also, we must identify the maximum load each unique feature can handle. Lastly, that the system is consistent under various loads within our maximum load.
j. System/End-to-End Testing
    - This type of testing will test the flow of our product from beginning to end. It is the replication of user behaviors and common scenarios to make sure the product works as a whole. If the test results in a failure, the specific component that caused the failure can be fixed. It is also important to check user flow to guarantee that each feature is easy to get to.

Test Scope

- The testing mentioned above will be very beneficial to the success of our product. Realistically, all of this testing cannot be completed due to resource constraints. Many of our limited available resources should be focused on testing set functional and nonfunctional requirements to ensure the best user experience. In-scope testing refers to a test that will be completed prior to the launch of our product. Out-of-scope testing is the opposite, so testing that is not planned to be completed. Most of the testing that was defined will be in-scope,

but regression testing is classified as out-of-scope. Scalability is an important aspect that our product holds, but our focus is to prioritize pushing core requirements that make BitOHealth unique before focussing on different features. This is summarized below.

- In-Scope
    - Unit Testing
    - Negative Testing
    - API Testing
    - UI Testing
    - Integration Testing
    - Security Testing
    - Stress Testing
    - Performance Testing
    - End-to-End Testing
- Out-of-Scope
    - Regression Testing

Key Dates for Testing Periods

The purpose of Key Dates for Testing Periods is to show when the testing period for each feature and subfeature begins and ends. The Start Date indicates the start of the testing period. The End Dates indicates the end of the testing period.

Milestone	Start Date	End Date	Features
1	01/20/2022 02/03/2022	02/03/2022 02/17/2022	**Login:** 2 Factor Authentication, Authorization. **Core Components:** Logging, Archiving, Account Deletion, Account Recovery.
2	02/17/2022 03/03/2022	03/03/2022 03/17/2022	**Medication/Health Recorder:** Create Record, Input Data/Edit Record, Save Record, View Record, Delete Record. **Track Records:** Track Records, Search Records, Categorize Records, View Folder **Weight Management/Calorie Counter:** Input Weight Goal Data,Edit Weight Goal Data, Search Food Item Add Food Item, Custom Food Item, Delete Food Item, Counting Calories, Export Food Log.
3	03/17/2022	03/31/2022	**Reminders:**

	03/31/2022	04/14/2022	Create Reminder, Edit Reminder, View Reminder, Delete Reminder. **Medication Lookup:** Start Search, View Medication, Favorite Medication. **Hot Topic:** View Hot Topic. **Core Component:** User Management **Remainders:** Export Reminder, Enable Notifications. **Medication Search:** View Favorite List, Refill Medication **Hot Topic:** Opt-in Notifications **Core Component:** User Analysis Dashboard
4	04/14/2022 04/28/2022	04/2820/22 05/05/2022	**Diet Recommendations:** Create Diet Recommendations, View Details of Recommendations Add Meal to Meal List, View Meal List, Delete Meal from Meal List **Health Locator:** Start Search **UI/UX Upgrades:** Improve UI

Features:

Note:

For a success scenario to occur, all its cases must pass.

For a success scenario to fail, at least one of its cases must fail.

1. Medication/Health Recorder

Type of Testing:

- Unit, Negative, UI, Integration, Security, Stress,Performance, and End-to-End Testing.

Assumptions:

- User is logged in.

- User is at the Medication/Health Recorder View.
- User information is retrieved from the Profile.

1.1 User selects Create Record Option

User selects a category and inputs a **valid** name for the record.

Success Scenarios:

- Create Record submission request is successful, if the user has not made a record in the last 30 seconds or is not on the record limit.
- Record is created and saved onto the Record View.
- Record is saved under the user selected category.
- Timing of record creation is saved into the data store and associated with the record.
- Medication/Health Recorder View is updated to display newly created record.
- User is redirected to the Record View.
- Database is updated.

Failure Scenarios:

- Create Record submission request is unsuccessful.
- Create Record submission request is successful if the user made a record in last 30 seconds
- Record is not created and saved onto the Record View.
- Record is not saved under the selected category.
- Medication/Health Recorder View is not updated to display the newly created record.
- User is not redirected to the Record View.
- Database is not updated.

User selects a category and inputs an **invalid** name for the record.

Success Scenarios:

- Create Record submission is unsuccessful.
- An error message is displayed stating to follow the proper guidelines of the specific invalid input.
- Record is not created.
- Database is not updated.

Failure Scenarios:

- Create record submission is successful.
- An error message is not displayed.
- Record is created and saved onto the Record View.
- Medication/Health Recorder View is updated to display the newly created record.
- User is redirected to the Record View.
- Database is updated.

Testing Plan:

- Execution Steps:
    - User selects Create Record option.
- Testing Plan:
    - Test 1: Input valid name and select category.
    - Unit Testing:
        - Input:
            - “Record 1” is inputted for name.
            - Category A is selected.
            - Success Case:
                - Record 1 is created and saved under the correct category.
                - Record 1 is saved into the data store.
        - Failure Cases:
            - Record 1 is not created.
            - Record 1 is created and saved under Category B
            - Record 1 is made against the record limit.
        - Execution Time: 1 minute.
    - Test 2: Test 30 second upload limit
        - Unit Testing:
        - Input:
            - “Record 1” is inputted for name.
            - Category A is selected.
            - “Record 2” is created within 30 seconds of “record 1”

        - Success Case:
            - Record 1 is created and saved under Category A.
            - Record 1 saved into the data store.
            - Record 2 is not created and the user is notified a record was made in the last 30 seconds.
        - Failure Cases:
            - Record 1 is created.
            - Record 2 is created as well against the 30 second block limit.
        - Execution Time: 2 minutes.
    - Test 3: Test Record Limit
        - Unit testing
        - Input:
            - “Record max” is inputted for name while at record limit
        - Success:
            - Record max isn’t made and user is informed they’re on limit
        - Failure:
            - Record max is made
        - Execution Time: 1 minute

- Test 4: Testing Database storage
- Integration testing
    - Input:
        - “Record 1” is inputted for name.
        - Category A is selected for the category.
    - Success Case
        - A record is created under the selected category and is stored in the database including the date the record was made.

        - Failure Case
            - A record is not stored in the database.
        - Execution Time: 1 minute
    - Test 5: Testing invalid input
    - Negative Testing
        - Input
            - “Record1Record1Record1Record1Record1Record1Record1Record1Record1Record1Record1Record1Record1Record1Record1Record1”
            - Category A is selected for the category.
        - Success Case
            - An error is given to user stating the character limit is too long
        - Failure Case
            - Record is made even though the name exceeds the character limit.
        - Execution Time: 1 minute.

1.2 User selects Input Data on Record Option:

Assumptions:

- User must be logged in.
- Use must be in the Record View.

  Success Scenarios:
    - Input Data on Record request is successful.
    - User is able to enter text to the record.
    - User is able to upload a maximum of two files onto a single record per day.
    - User is able to set the date and time for the record.
    - User is able to save changes made to the record.
    - User is notified to confirm saving changes made to the record.
    - Record is updated in the data store.

  Failure Scenario:
    - Input Data on Record request is unsuccessful.

- User is unable to enter text to the record.
- User is unable to upload files to the record.
- User is unable to set the date and time.
- User is unable to save changes made to the record.
- User is not notified to confirm saving changes made to the record.

Testing plan:

Execution Steps:

- User selects a record.
- User inputs data onto record.
- User sets Date and Time of record.

Testing Plan

Test 1: Input valid data onto record.

- Unit Testing:
    - Input Data 1:
      HellooHelloHELLOHellooHelloHELLOHellooHelloHELLOHellooHell
      oHELLOHellooHelloHELLOHellooHelloHELLOHellooHelloHELLO
      HellooHelloHELLO…….etc…
    - Input Data 2: (blank)
    - Select save option.
- Success Case:
    - Message is displayed to the user: “Are you sure you want to save changes made to the record?”
    - Input is saved.
    - Database is updated.
- Failure Case:
    - No confirmation message is displayed to the user.
    - Input is not saved.
    - Database is not updated.
- Execution Time: 1 minute.

Test 2: Set Date and Time.

- Unit Testing:
    - Input Data:

        12/1/2021 2:47 PM
- Success Case:
    - “12/1/2021 2:47 PM.” will be displayed at the top of the record.
    - Automatically saved to reflect this date and time.
    - Record will display this date and time.
- Failure Case:
    - “11/29/1999 2:47 PM” is displayed at the top of the record. (incorrect data).
    - (blank) is displayed at the top of the record.
- Execution Time: 1 minute.

User uploads **valid** File:

Success Scenarios:

- User’s file is successfully uploaded to the Record View.
- User’s file is stored in the data store.
- User may select and view contents of the file.
- A confirmation message is displayed to the user.

Failure Scenarios:

- User’s selected file is not uploaded.
- User’s file is not stored in the data store.
- User can't select and display the file.
- No confirmation message is displayed to the user.

User uploads **invalid** File:

Success Scenarios:

- File upload request is unsuccessful.
- An error message is displayed stating to follow the proper guidelines.
- User is given the option to select a different file to upload.

Failure Scenarios:

- File upload request is successful.
- No error message is displayed to the user.
- Incorrect error message is displayed to the user.

- User is not given the option to select a different file to upload.

User reaches maximum file uploads for the day:

Success Scenarios:

- Once the last file is successfully uploaded and stored, a message is displayed to the user stating the limit has been reached.
- Upload file option will no longer be visible until the next day.

Failure Scenarios:

- No message is displayed to the user regarding reaching the maximum file upload limit.
- Upload file option is still visible to the User.
- User is able to successfully upload more files exceeding the file upload limit.
- File is stored and is displayed in the Weight Management View.

Testing Plan:

Execution Steps:

- User selects a record.
- User selects Input File option.
- User selects a File.
- User selects Insert File option.

- Test 1: User inputs Valid File
  - Integration Testing
  - Success Case:
    - Valid.jpg is uploaded onto the record after the database is checked to make sure the limit isn't breached.
    - Valid.jpg is also visible to user and may interact with it.
    - Message displayed to user "File successfully uploaded".
  - Failure Case:
    - Valid.jpg is not uploaded onto the record despite no limit breach.
    - Valid.jpg is not visible to the user.

        - XYZ.jpg is uploaded onto the record and visible to the user. (wrong file).
        - No/wrong message is displayed to the user.
    - Execution time: 2 minutes
- Test 2: User inputs Invalid File
    - Negative Testing:
    - Success:
        - Invalid.ppt is not uploaded onto the record.
        - Message is displayed to the user: “File Type not allowed. Please select a JPG or PDF”.
        - User is given the option to reselect a file.
    - Failure:
        - Invalid.ppt is accepted and uploaded onto the record.
        - No/wrong error message is displayed to the user.
        - User is not given the option to reselect a file.
    - Execution time: 2 minutes.
- Test 3: User inputs file exceeding Size Limit.
    - Unit Testing:
    - Success Case:
        - SizeTooBig.pdf is not uploaded onto the record.
        - Message is displayed to the user: “File too big. Please select a file that does not exceed 16 MB”.
        - User is given the option to reselect a file.
    - Failure Case:
        - SizeTooBig.pdf is uploaded onto the record.
        - No/wrong message is displayed to the user.
        - User is not given the option to reselect a file.
- Test 4: User inputs a file under the minimum size.
    - Unit Testing:
    - Success Case:
        - SizeTooSmall.pdf is not uploaded onto the record.

        - Message is displayed to the user: “File too small. Please select a file that is at least 0.5 MB”.
        - User is given the option to reselect a file.
    - Failure Case:
        - SizeTooSmall.pdf is uploaded onto the record.
        - No/wrong message is displayed to the user.
        - User is not given the option to reselect a file.
    - Execution Time: 2 minutes.
- Test 5: Test 2 files per day limit.
    - Negative Testing
        - Input:
            - “inp1.jpg” , “inp2.jpg”, “inp3.jpg”
        - Success:
            - inp1.jpg, inp2.jpg are uploaded onto record, inp3.jpg is rejected.
            - Message is displayed to the user. ‘“Two File Limit is reached. You may upload again in 24 hours”.
        - Failure
            - Inp1.jpg,inp2.jpg, and inp3.jpg are uploaded onto the record.
            - No/wrong message is displayed to the user.
        - Execution time: 3 minutes.

1.3 User selects the Edit Record Option:

Assumptions:

- User is logged in.
- User is on the Medication/Health Recorder View.
- User selects a record.

Success Scenarios:

- User is able to edit the title and/or category of the record.
- User is able to edit the text, files, and date/time of the record.

- A confirmation message to save changes is displayed to the user.
- Record is updated with the new content and saved in the data store.

Failure Scenarios:

- User is unable to edit the title and/or category of the record.
- User is unable to edit the text, files, and date/time of the record.
- No confirmation message is displayed to the user.
- Record is not updated with the new content.
- Record changes are not saved.
- User successfully saves changes despite exceeding storage capacity.

Testing Plan:

Execution Steps:

- User selects a record.
- User selects inputs data onto record.
- User selects the Save option.

- Test 1: Save changes made to record.
  - Integration test
  - Input:
    - Data: “HELLOOOO”, “em.pdf” “12/12/2012 4:44 AM”
    - Save.
  - Success Case:
    - User is displayed a confirmation message: “Are you sure to save changes made to the record?”
    - New Data is saved onto the record and previous data is overwritten.
    - Changes are saved to the datastore.
    - User is displayed a message: “Changes have been successfully saved.”
  - Failure Case:
    - User is not displayed a confirmation message.

        - Inaccurate data is saved onto the record.
        - Changes are not saved onto the datastore.
        - No message is displayed to the user regarding the changes being saved.
    - Execution Time: 1 minute.
- Test 2: Database Storage
    - Integration Testing:
        - Input:
            - Data: “HELLOOOO”, “em.pdf” “12/12/2012 4:44 AM”
            - Save.
        - Success Case:
            - The changes made are saved on the datastore.
        - Fail Case:
            - The changes made are saved on the datastore.
            - Outdated version remains in the datastore.
        - Execution Time: 1 minute.
- Test 3: Capacity Testing
    - Negative Test:
        - Input
            - Data: “HELLOOOO”, “sixbits.pdf” “12/12/2012 3:44 AM”
            - Save.
        - Success Case:
            - Message is displayed to the user. “Record Data is too big, unable to store information”.
            - Changes are not saved the system
            - Changes are not saved onto the datastore
            - User may edit and submit save again.
        - Failure Case:
            - No/Wrong message is displayed to the user.

- Changes are saved and user exceeds storage capacity.

- Execution Time: 2 minutes.

1.4 User selects the View Record Option:

Assumptions:

- User is logged on.
- User is at the Record List View.
- At least one record exists.

User selects the record to be viewed.

Successful Scenarios:

- View Record request is successful.
- User is redirected from the Records List View to the Record View.
- User is able to view the contents of the Record.
- User is able to edit the contents of the Record.

Failure Scenarios:

- View Record request is unsuccessful.
- User is redirected to the wrong Records View.
- User is unable to view the contents of the Record.
- User is unable to edit the contents of the Record.

Testing Plan:

Execution Steps:

- User selects a record.

- Test 1: Records and data inside it are visible:
  - Integration testing
  - Input:
    - No input required.
  - Success Case:
    - User is able to view “Record 1” and “Record 2”.
    - User is able to view files uploaded onto record.
  - Fail Case:
    - User is unable to view “Record 1” or “Record 2”.

        - User is unable to view files uploaded onto record.
    - Execution Time: 1 minute.

1.5 User selects the Delete Record Option:

- User is logged in.
- User is at the Medication/Health Recorder View.
- At least one record must exist.
- User selects the Delete Record.

    Success Scenarios:

        - Delete Record request is successful.
        - A confirmation message is displayed to the user verifying whether they want to delete the record or not.
        - Record is deleted.
        - Medication/Health Recorder View is updated and reflects the removal of the meal item.
        - A message is displayed to the user stating that the record has been successfully deleted.

    Failure Scenarios:

        - Delete Record request is unsuccessful.
        - No confirmation message is displayed to the user.
        - Record is not deleted.
        - The incorrect record is deleted.
        - Medical/Health Recorder View does not reflect the removal of the record.
        - No message is displayed to the user.

Testing Plan:

    Execution Steps:

        - User selects a record.
        - User selects Delete option.

- Test 1 Delete a Record:
    - Unit Testing:

- Success Case:
    - A message is displayed to the user to confirm: “Are you sure you want to delete the record?”
    - Record is deleted and no longer visible.
    - Datastore is updated.
    - A message is displayed to the user: “Record has been successfully deleted.”
- Failure Case:
    - No/wrong confirmation message is displayed to the user.
    - Record is not deleted.
    - Record is still visible.
    - Datastore is not updated to reflect deletion of a record.
    - A message is not displayed to the user.
- Time Execution: 1 minute.

2. Reminders

Type of Testing:

- Unit, UI, Negative, Integration, and Regression Testing.

Assumptions:

- User is logged in.
- User is at the Reminder List View.
- The user’s current Reminder List has less than 99 reminders.

2.1 User selects the Create a Reminder Option.

User inputs a **valid** Name, Description, and selects a Date/Time for the reminder.

Success Scenarios:

- Create Reminder submission request is successful.
- Reminder is created and saved to the Reminder List.
- Reminder List is updated to display the newly created reminder.
- Datastore is updated.

Failure Scenarios:

- Create Reminder submission request is unsuccessful.

- Reminder is not created.
- Reminder is not saved to the Reminder List.
- Reminder List does not display the newly created reminder.
- Datastore is not updated.

User inputs an **invalid** Name, Description, and selects a Date/Time for the reminder, as well as frequency.

Success Scenarios:

- Create Reminder submission is unsuccessful.
- An error message is displayed stating to follow the proper guidelines of the specific invalid input
- Reminder is not created.

Failure Scenarios:

- Create reminder submission is successful.
- An error message is not displayed.
- Wrong error message is displayed.
- Reminder is created.
- Reminder List View is updated to display the newly created reminder.
- Datestore is updated.

Testing Plan:

Execution Steps:

- User selects Create Reminder Option.
- Test 1Input valid required data :
    - Integration Testing:
        - Input:
            - Name: ”WATER”
            - Description: ”GO DRINK YOUR WATER!”
            - Date: 12/4/2021 8:59 AM.
            - Frequency: Once.
        - Success Case
            - Reminder is created, and saved with the accurate, valid input.

            - Reminder can be viewed by the user.
            - Datastore is updated.
        - Failure Case
            - Reminder is not created and/or not saved.
            - Reminder is not viewable by the user.
            - Datestore is not updated.
        - Execution time : 2 minutes
- Test 2 Input invalid required data:
    - Negative Testing:
        - Input:
            - Name="name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1name1"
            - Description = "name1" 200 times
            - Date= 12/4/2012 8:59 AM.
            - Frequency: Once.
        - Success Case:
            - A message is displayed to the user: "Please follow the proper guidelines. Select an appropriate date."
            - Reminder is not created or saved into the system.
            - No changes are made to the datastore.
            - User is given the option to reinput the invalid fields.
        - Failure Case:
            - No/wrong message is displayed to the user.
            - Reminder is created and saved into the system.
            - Datastore is updated.
            - User is not given the option to reinput the invalid fields.
        - Execution time: 2 minutes.

2.2 User selects Edit Reminder Option

Assumptions:

- User must be logged in.
- User must in the Reminders List View.
- At least one reminder must exist.
- At least one record must exist.

Success Scenarios:

- User is able to edit the title, description, and select a different date/time for the reminder.
- A confirmation message to save changes is displayed to the user.
- Reminder is updated with the new content and saved.
- User is notified of the reminder by the newly selected date/time.

Failure Scenarios:

- User is unable to edit the title, description, and select a different date/time for the reminder.
- No confirmation message is displayed to the user.
- Reminder changes are not saved.
- User is still notified of the reminder by the outdated date/time.

Testing Plan:

Execution Steps:

- User selects the reminder.
- User edits the required fields of the reminder.
- User selects the Save option.

- Integration Testing:
    - Test 1: Changes made to the reminder are saved.
        - Integration Testing:
            - “Name1” changed to “name2”
        - Success Case:
            - A confirmation message is displayed to the user. “Are you sure of the changes made to the reminder?”
            - New reminder is saved to the system.
            - Datastore is updated.

- Failure Case:
    - No/wrong confirmation message.
    - Reminder changes are not saved.
    - Datastore is not updated to reflect new changes.
- Execution Time: 1 minute

2.3 User selects the View Reminder Option:

Assumptions:

- User is logged on.
- User is at the Reminders List View.
- At least one reminder exists.

User selects the reminder to be viewed.

Successful Scenarios:

- View Reminder request is successful.
- User is redirected from the Reminders List View to the Reminder View.
- User is able to view the contents of the Reminder.
- User is able to edit the contents of the Reminder.

Failure Scenarios:

- View Reminder request is unsuccessful.
- User is redirected to the wrong Reminder View.
- User is unable to view the contents of the Reminder
- User is unable to edit the contents of the Reminder.

Testing Plan:

Execution Steps:

- User selects a reminder.

- Test 1 View Reminder:
    - Integration Test
        - Input:
            - “WATER” reminder.
            - “MEDICINE” reminder.
        - Success Case:

- User is able to view both reminders.
- “WATER” displays:
    - Name: ”WATER”
    - Description: ”GO DRINK YOUR WATER!”
    - Date: 12/4/2021 8:59 AM.
    - Frequency: Once.
- “MEDICINE” displays:
    - Name: ”MEDICINE”
    - Description: ”GO TAKE YOUR MEDICINE!”
    - Date: 4:00 PM.
    - Frequency: Everyday.

- Fail Case:
    - At least one reminder isn’t visible.
    - At least one reminder displays inaccurate information.
    - WATER displays:
        - Name: ”MEDICINE”
        - Description: ”GO DRINK YOUR WATER!”
        - Date: 12/4/2021 8:59 AM.
        - Frequency: Once.
- Execution time: 1 minute.

- Test 2 Select Reminder
    - Integration Test:
        - Input:
            - “WATER” is selected.
        - Success Case
            - WATER is retrieved from database and opened.
            - User is viewing WATER.
        - Failure Case
            - WATER isn’t selected from database or opened.
            - MEDICINE is selected instead.

- Execution time: 1 minute.

2.4 User selects the Delete Reminder Option:

- User is logged in.
- User is at the Reminder List View.
- At least one reminder must exist.

Success Scenarios:

- Delete Reminder request is successful.
- A confirmation message is displayed to the user verifying whether they want to delete the reminder or not.
- Reminder is deleted.
- Reminder List View is updated and reflects the removal of the reminder.

Failure Scenarios:

- Delete Reminder request is unsuccessful.
- No confirmation message is displayed to the user.
- Reminder is not deleted.
- The incorrect reminder is deleted.
- Reminder List View does not reflect the removal of the reminder.

Testing Plan:

Execution Steps:

- User selects a reminder.
- User selects Delete option.

- Test 1 Delete a Reminder:
  - Unit Testing:
  - Success Case:
    - A message is displayed to the user to confirm: “Are you sure you want to delete the reminder?”
    - Reminder is deleted and no longer visible.
    - Datastore is updated.

        - A message is displayed to the user: “Reminder has been successfully deleted.”
    - Failure Case:
        - No/wrong confirmation message is displayed to the user.
        - Reminder is not deleted.
        - Reminder is still visible.
        - Datastore is not updated to reflect deletion of the reminder.
        - A message is not displayed to the user.
    - Time Execution: 1 minute.

2.5 User selects the Export Reminder Option

Assumptions:

- User is logged in.
- The reminder must exist.
- User is at the Reminder View.

  Successful Scenarios:
    - Export Reminder request is successful.
    - A confirmation message is displayed to the user verifying whether they want to export the reminder or not.
    - When exporting from a mobile device, the reminder is exported as an iCal and saved to the user’s Files to manually be added to 3rd party services.
    - When exporting from a desktop, the reminder is exported as a iCal and the user chooses where to save it.
    - Reminder is successfully exported to their selected destination.
    - Exporting the selected reminder does not affect the Reminder List.
    - A message is displayed to the user stating that the reminder has been exported.

  Failure Scenarios:
    - Export Reminder request is unsuccessful.
    - No confirmation message is displayed to the user verifying whether they want to export the reminder or not.

- No export of reminder occurs.
- Correct reminder is exported to the incorrect destination.
- Incorrect reminder is exported to the user's selected destination.
- Exporting a reminder affected the Reminder List.
- No message is displayed to the user stating the status of the export.

Testing Plan:

Execution Steps:

- User selects a reminder.
- User selects export reminder.
- User selects destination.

- Test 1: Export Reminder is selected:
  - Unit testing
    - Input:
      - Reminder: "WATER".
      - Destination: Desktop.
    - Success Case:
      - "WATER" is successfully exported to the user's desktop.
    - Failure Case:
      - "MEDICINE" is exported to the user's desktop. (incorrect reminder)
      - "WATER" is exported to the user's files. (wrong destination)
    - Execution Time: 1 minute.

2.6 Opt-in Reminder Notifications Option

Assumptions:

- User is logged in.
- User is in the Reminder List View.
- A reminder exists

User selects the reminder preference (email or calendar).

Success Scenarios:

- The Opt-in reminder notification request is successful.
- A message is displayed to the user to confirm their selected preference.
- The user is notified of the reminder at their selected timing via their reminder preference.

Failure Scenarios:

- The Opt-in reminder notification request is not successful.
- No message is displayed to the user.
- The user is not notified at the correct timing or/and via their reminder preference.

Testing Plan:

Execution Steps:

- User selects a reminder.
- User selects the opt-in reminder notification option.

Input:

- User selects “Water” Reminder which is set to occur in 5 minutes.
- User selects opt-in notifications.

- Test Case 1: User is notified of reminder:
  - Unit Testing:
  - Success case:
    - A reminder message is displayed at the exact set date and time: “DRINK YOUR WATER!” (occurs after 5 minutes of setting date and time)
    - User is successfully notified.
  - Failure case:
    - User is not notified of the selected reminder.
    - A reminder message is displayed at an inaccurate date and time: “DRINK YOUR WATER!” (occurs after 10 minutes)
    - An inaccurate reminder is displayed: “GO TAKE YOUR MEDICINE!” (occurs after 5 minutes)
  - Execution Time: 1 minute.

3- Health Locator

Types of Testing:

- Unit, Negative, API, UI, Integration, Security, Stress, and End-to-End Testing.

Assumptions:

- User is logged in
- User is in Health Locator View

3.1 User searches by Zip Code

User inputs **valid** Zip Code and selects a Category for health type.:

Success Scenarios:

- Search request submission is successful.
- API request is made and returns correct results to the system.
- Health Locator View is updated to display the search results.
- User is notified when no results are generated.
- User is able to select a location item from the search list to view its details.

Failure Scenarios:

- Search request submission is unsuccessful.
- Search is not executed and results are not generated.
- No API request is made.
- API request is made and returns incorrect results to the system.
- Search displays wrong location items that do not correspond to the user's input.
- User is unable to select a location item from the search list to view it.

User inputs **invalid** Zip Code and selects a Category for health type:

Success Scenarios:

- User is unable to submit the Health Location Search request.
- No API request is made.
- An error message is displayed stating to follow the proper guidelines of the specific invalid input
- User is given the option to re-input the incorrect fields.

Failure Scenarios:

- User is able to submit the Health Location Search request.
- API request is made.
- Search is executed and results are generated.
- No error message is displayed to the user.
- Wrong error message is displayed to the user.
- User is not given the option to re-input the incorrect fields.

Search Results:

Success Scenarios

- Locations are sorted by proximity.
- User can interact with list of locations.
- User can select a location item to view additional information.
- User is notified when no locations match their request.

Failure Scenarios

- Locations are not sorted appropriately
- User is unable to interact with the results generated by the search.
- User is unable to view additional information on the selected item.
- User is not notified when no locations match their request.

User chooses to view a certain locations information:

Success Scenarios:

- The following information is displayed:
  - Location hours
  - Website
  - Phone number

Failure Scenarios:

- No information is displayed to the user.
- Incorrect information is displayed to the user.

Test Case

Execution Steps:

- User selects the Search Health Locator Option.
- User selects a category for health type.
- User inputs valid Zip Code.

Test 1: User inputs valid data.

Integration Testing:

Input:

- 90840 is inputted for Zip Code.

Success Case:

- Search is conducted and displays results that correspond to the user's input.

Failure Case:

- Search is not conducted and does not display results that correspond to the user's input.

Execution Time: 1 minute.

Test 2:User inputs invalid Zip Code.
Unit Testing

Input Possibilities:
- 90342232 is inputted for Zip Code.
- 9x321 is inputted for Zip Code.
- 91 is inputted for Zip Code.
- 99999 is inputted for Zip Code.

Success Case:
- An error is displayed to the user stating the zip code is invalid: “Please enter a valid Zip Code.”
- Search is not conducted and displayed to the user.

Failure Case:
- Search is conducted and displays results.
- No/incorrect error message is displayed to the user.

Execution Time: 1 minute.

4- Medication Lookup

Types of Testing:
- Unit, Negative, API, UI, Integration, Security, Stress, and End-to-End testing.

Assumptions:
- User is logged in
- User is at the Medication Lookup View

4.1 User searches for a Medication:

User enters a **valid** medication name using the search bar, zip code, and mile radius.

Successful Scenarios:
- Search request is successful
- API request is made and returns correct results to the system
- Medication Lookup view is updated to display the search results
- User is notified when no results are generated
- User is able to select the medication item from the search list to view its details

Failure Scenarios (at least one case must fail):
- Search request submission is unsuccessful
- Search is not executed and results are not generated
- No API request is made.
- API request is made and returns incorrect results to the system.

- Search displays wrong medication items that do not correspond to the user's input
- User is unable to selection a medication form the search list to view it

User enters a **invalid** medication name using the search bar, zip code, and mile radius

Success Scenarios:

- User is unable to submit the Medication Lookup Search request
- No API request is made
- An error message is displayed stating to follow the proper guidelines of the specific inclaid input
- User is given the option to re-input the incorrect fields.

Failure Scenarios:

- User is able to submit the Medication Lookup request
- API request is made.
- Search is executed and results are generated.
- No error message is displayed to the user.
- Wrong error message is displayed to the user.User is not given the option to re-input the incorrect fields.

Test Case:

- Execution Steps:
  - Select the search bar.
  - Input a valid Medication name.
  - Select the magnifying glass icon.
- Testing Plan
  - Test 1: Input valid data:
  - Unit Testing:
    - Input:
      - User inputs on the search bar “Advil”
    - Success Case:
      - Search Request successful
      - User is notified if when no results are generated
      - User is able to select the Medication.
    - Failure Case:
      - Search request unsuccessful
    - Execution Time: 1 minutes
  - Test 2: Testing invalid data:
  - Negative Testing:

- Input:
    - User inputs “12345abcd” or (blank)
- Success Case:
    - An error is displayed to the user stating that the medication must be at least 1-100 characters long or the medication is spelled incorrectly.
      “Input a valid Medication Name.”
- Failure Case:
    - The search request is successful.
    - No/wrong error message is displayed to the user.
- Execution Time: 1 minute.

- Test 3: The results displays the Pharmacy providing the medication.
    - API Testing:
    - Input:
        - “Advil is inputted for Medication name
    - Success Case:
        - A user is displayed the different vendors within the radius of the user that offer the medication.
          “CVS 5 miles away”
          “Wallgreens 15 miles away”
        - If the user selects the medication, the API displays the list of prices associated with the medication.
          “$7.99 at CVS”
          “$8.50 at Walgreens”
            - Note: Prices will be in US dollars
        - Search can return 0 results.
          “No results found.”
        - A maximum of 50 results will be displayed at a time.
    - Failure Case:
        - The input displays a incorrect set of results
        - If the user selects the medication, the API displays the wrong prices associated with the medication.
    - Execution Time: 2 minutes

4.2 User selects the View Medication on Medication Lookup.
Assumptions:
- User must be logged in
- User must be in the Medication Lookup view

- User successfully searched for a medication
- Search displays at least one medication

User selected the medication to be viewed

Success Scenarios:

- Medication view request is successful
- User is able to view the contents of Medication

Failure Scenarios:

- Medication view requisition is unsuccessful
- User is directed to the wrong Medication view
- User is unable to view the contents of Medication

Testing Plan:

- Execution Steps:
  - Go to the Medication Lookup View.
  - Click the search bar option
  - Input a valid Medication name
  - Click on the magnifying glass icon
- Testing Plan
  - Test 1: Viewing a medication
  - Unit Testing:
    - Input:
      - User inputs on the search bar “Advil”
    - Success Case:
      - Search Request successful.
      - User is notified if no results are generated: “No results found.”
      - User is able to select the Medication.
    - Failure Case:
      - Search request unsuccessful
    - Execution Time: 1 minute.
  - Test 2: The results displays the Pharmacy providing the medication.
  - API Testing:
    - Input:
      - User enters Advil on the search Bar
    - Success Case:

        - A confirmation message notifies the user which displays the different vendors within the raidosu of the user that offer the medication.
        - If the user selects the medication, the API displays the list of prices associated with the medication
            - Note: Prices will be in US dollars
        - Search can return 0 results
        - A maximum of 50 results will be displayed at a time.
    - Failure Case:
        - The input displays a incorrect set of results
        - If the user selects the medication, the API displays the wrong prices associated with the medication.
    - Execution Time: 1 minute.

4.3 User selects an Medication Item to put to Favorites List
Assumptions:

- User must be logged in
- User must be in the Medication View
- User successfully search for a medication
- User has less than 10 favorites in the favorite list

The user favorites the medication

Success Scenarios:

- The medication is added to the user's favorite List
- Medication vendor is favorited
- Favorite List incremented by 1.
- The action is updated on the database.
- Favorite must be added to the Favorite List in a maximum of 2 seconds.

Failure Scenarios:

- The medication is not added to the user's favorite list
- Medication vendor is not favorited
- Favorite List does not increment by 1.
- The action is not updated on the database.

The user unfavorites the medication

Success Scenarios:

- The medication is removed from the user's favorite list
- Medication vendor is unfavorited
- Favorite List decrements by 1.
- The action is updated on the database.

- Unfavoriting must be removed from the Favorite List in a maximum of 2 seconds.

Failure Scenarios:
- The medication is not removed from the user's favorite list
- Medication vendor is still favorited
- Favorite List does not decrease by 1.
- The action is not updated on database

Testing Plan:
- Execution Steps:
  - Go to the Medication Lookup View.
  - Click the search bar option
  - Input a valid Medication name
  - Click on the magnifying glass icon
  - User has already selected a valid medication.
    - For example, user types in Advil in the search bar.
- Testing Plan
  - Test 1: Viewing a medication
  - Unit Testing:
    - Input:
      - User inputs on the search bar "Advil"
    - Success Case:
      - Search Request successful
      - User is notified if when no results are generated
      - User is able to select the Medication
    - Failure Case:
      - Search request unsuccessful
    - Execution Time: 1 minute.
  - Test 2: The results displays the Pharmacy providing the medication.
  - API Testing:
    - Input:
      - User enters Advil on the search Bar
    - Success Case:
      - A confirmation message notifies the user which displays the different vendors within the raidosu of the user that offer the medication.
      - If the user selects the medication, the API displays the list of prices associated with the medication
        - Note: Prices will be in US dollars

        - Search can return 0 results
        - A maximum of 50 results will be displayed at a time.
    - Failure Case:
        - The input displays a incorrect set of results
        - If the user selects the medication, the API displays the wrong prices associated with the medication.
    - Execution Time: 1 minutes
- Test 3: Testing the Database
- Integration Test
    - Input: User favorites Advil to their Favorite List
    - Success Case:
        - Medication is added to Favorite List and updated on the database including the date the favorite was made.
    - Failure Case:
        - Medication favorite is not stored in the database
    - Execution Time: 1 minute.
- Test 4: Favorite in Favorite List Limit
    - Unit Testing
    - Input:
        - The user favorites another medication while at the favorite limit.
    - Success Case:
        - The favorite item is not added and informed they have exceeded the limit of favorite per list.
    - Failure Case:
        - User is able to favorite medication to their favorite list.
    - Execution Time: 1 minute.
- Test 5: Testing Database storage (within limit)
- Unit Testing:
    - Input:
        - “Advil” is favorited.
    - Success Case:
        - The Favorite List is updated onto the system and datastore.
    - Failure Case:
        - User is notified: “Changes cant be saved.”
        - The Favorite List is not updated onto the system and datastore.
    - Execution Time: 2 minutes.
- Test 6: Testing Database storage (exceeding limit)

- Unit Testing:
    - Input:
        - “Advil” is favorited.
    - Success Case:
        - User is notified: “Changes cant be saved.”
        - The Favorite List is not updated onto the system and datastore.
    - Failure Case:
        - The Favorite List is updated onto the system and datastore.
    - Execution Time: 2 minutes.

4.4 User selects the View Favorite List Option:
Assumptions:

- User must be logged in
- User must be in the Medication Lookup View
- User successfully search for a medication
- User has less than 10 favorites in the favorite list

User selects the favorite list to be viewed

Success Scenarios:

- View Favorite List is successful
- User is redirected from the View Favorite List to the Favorite view
- User is able to view the contents of the Favorite List
- User is able to edit the contents of Favorite List

Failure Scenarios:

- View Favorite List is unsuccessful
- User is redirected to the wrong Favorite List View.
- User is unable to view the contents of the Favorite List.
- User is unable to edit the contents of Favorite List.

Test Case:

- Execution Steps:
    - User selects the Display Favorite List.
    - User selects an item off the Favorite list.
- Testing Plan
    - Test 1: Viewing a Favorite Medication
    - Input Testing:
        - Input:

        - User selects “Advil”.
    - Success Case:
        - User is taken to the Medication View which displays its different locations and information:
          “Advil is a drug that helps with…”
          “Can be found near you at Walgreens, CVS,..”
          “Distance: 5 miles, 15 miles…”
    - Failure Case:
        - Search request unsuccessful.
    - Execution Time: 1 minute.

- Test 2: Testing the Database
- Integration Test
    - Input: User favorites “Panadol” to their Favorite List
        - Success Case:
            - Panadol is added to Favorite List.
            - Medication favorite is saved to the system and datastore is updated.

        - Failure Scenario:
            - Panadol is not added to Favorite List.
            - Medication favorite is not saved to the system and datastore is not updated.
        - Execution Time: 1 minute.
- Test 3: Test Favorite in Favorite List Limit
    - Unit Testing
    - Input:
        - The user favorites another medication while at the favorite limit.
    - Success Case:
        - The favorite item is not favorited and use is informed they have exceeded the limit of favorite per list.
    - Failure Case:
        - User is able to favorite item to their favorite list.
    - Execution Time: 1 minute
- Test 4: Access Favorite in Favorite List Limit
    - Unit Test
    - Input:
        - User chooses “favorite medication in favorite list 1”
    - Success Case:

        - System retrieves Favorite List 1 and shows it to use with option to unfavorite/favorite
    - Fail Case:
        - System fails to retrieve favorite list 1
    - Execution Time: 1 minute

4.5 User selects the Refill Medication option.
Assumptions:

- User is logged in
- User must be in Medication View
- User must have at least one medication.

User inputs a **valid** refill date, time and recurring time.

Success Scenarios:

- Refill Medication submission request is successful
- Refill Medication is created and saved under Medication view
- Medication view is updated to display newly created Refill Medication
- User is redirected to Medication view
- Adding the refill reminder to the Reminder List takes less than 2 seconds.

Failure Scenarios:

- Refill Medication submission request is unsuccessful
- Refill Medication is not created and not saved under Medication view
- Medication view is not updated to display newly created Refill Medication
- User is not redirected to Medication view

User inputs a **invalid** refill date, time, and/or recurring time

Success Scenarios:

- Create Refill Medication submission is unsuccessful
- Ab error message is displayed stating to follow the proper guidelines of the specific invlaid input
- Refill Medication is not created

Failure Scenarios:

- Create Refill Medication submission is successful
- An error message is not displayed
- Refill Medication is created
- Adding the refill reminder to the Reminder List takes less than 2 seconds.

Test Case:

- Execution Steps:
    - Go to the Medication View.
    - User has a valid Medication
    - User must select Refill Reminder Option
- Testing Plan
    - Test 1: Input valid date and time
    - Unit Testing:
        - Input:
            - User inputs the date and time to be reminded of the refill.
            - User inputs the recurring time. For example, every week, bi weekly, monthly, or yearly.
        - Success Case:
            - A refill reminder is added to the Reminder List
            - User is notified that the reminder was created successfully via a confirmation message.
        - Failure Case:
            - The refill reminder is not created
            - User receives an error message indicating that the date/time and/or recurring time was filled out incorrectly
        - Execution Time: 2 minutes
    - Test 2: Testing invalid date and time
    - Negative Testing:
        - Input:
            - Date: ab/cd/ef32 or Date: ^45/12/3232 or Date” “/” “/” “
        - Success Case:
            - An error message is given to the user saying that the date must follow the correct format (MM/DD/YYYY) and use numerical values only.
            - The reminder is not created.
        - Failure Case:
            - The remainder is created.
        - Execution: 1 minute.
    - Test 3: Testing
    - Negative Testing:
        - Input:
            - User attempts to pick a past date and time to set the refill reminder.
              For example, the System indicates the date and time to be 10/29/2021 6:18 pm
              The user attempts to choose 10/15/2021 6:18 pm

        - Success Case:
            - An error message is given to the user saying that the user cannot select the past date from their current date.
        - Failure Case:
            - A confirmation message is given to the user that the refill reminder is created.
            - Execution Time: 1 minute.

- Test 4:
- Unit Testing
    - Input:
        - “Favorite List” is inputted as name
    - Success Case:
        - The Favorite List is made and is stored successfully on the database including the date of the favorite list was made.
    - Failure Case:
        - The Favorite List is not stored in the database.
    - Execution Time: 1 minute.
- Test 5: Access Refill Reminder in Reminder List
- Unit Test
    - Input:
        - User chooses “Reminder List”
    - Success Case:
        - System retrieves Reminder List and shows the user all of the user’s created refill reminders
    - Fail Case:
        - System fails to retrieve Reminder List
    - Execution Time: 1 minute

5- Weight Management/Calorie Counter

Types of Testing:

- Unit, Negative, API, UI, Integration, Security, Stress, and End-to-End testing.

Assumptions:

- User is logged in.
- User is at the Weight Management View.
- User information is retrieved from Profile.

5.1 User selects the Create Goal Option:

User inputs **valid** Target Weight, and selects Exercise Schedule and Time Frame.

Success Scenarios:

- Create Goal request submission is successful.
- User's input is used to calculate the target weight by the given time frame.
- Weight Management View is updated to display the correct calculation results.

Failure Scenarios:

- Create Goal request submission is unsuccessful.
- No input or wrong values are used to calculate the user's weight goal.
- No or wrong calculation results are displayed to the user in the Weight Management View.

User inputs **invalid** Target Weight, and selects Exercise Schedule and Time Frame:

Success Scenarios:

- Create Goal request submission is unsuccessful.
- An error message is displayed stating to follow the proper guidelines of the specific invalid input.
- Inputs are not stored in the system and no calculations are made.
- User is given the option to re-input the wrong fields.

Failure Scenarios:

- User is able to submit "Create Goal Option" request.
- No error message is displayed to the user.
- Incorrect error message is displayed to the user.
- User is not given the option to re-input the wrong fields.
- Invalid Inputs are stored and calculations are made.
- Weight Management View is updated to display results of invalid input calculations.

Testing Plan:

- Execution Steps:
  - User selects the Create Goal option.

        - Select Exercise Schedule.
        - Select Time Frame.
- Testing Plan
    - Test 1: Input valid Target Weight.
    - Integration Test
        - Input:
            - 100 is inputted for target weight.
        - Success Cases:
            - Correct calculation results are displayed to the user and stored in the system.
        - Failure Cases:
            - No or wrong calculation results are displayed to the user in the Weight Management View.
        - Execution Time: 1 minute.
    - Test 2: Input invalid Target Weight.
    - Negative Test
        - Input:
            - S100 is inputted for target weight.
        - Success Cases:
            - An error message is displayed to the user to only enter integers.
            - No calculations are displayed to the user.
        - Failure Cases:
            - No or wrong error message is displayed to the user.
            - Incorrect calculations are displayed to the user.
        - Execution Time: 1 minute.
    - Test 3: Testing Database Storage
    - Integration Testing:
        - Input:
            - 100 is inputted as Target Weight.
        - Success Case:

        - Input is stored successfully on the database.
    - Failure Case:
        - Input is not stored in the database.
    - Execution Time: 1 minute.

5.2 User selects the upload File Option:

Assumptions:

- User must be logged in.
- User must be at the Weight Management View.

User uploads **valid** File:

Success Scenarios:

- User’s file is successfully uploaded.
- User’s file is stored in the Weight Management View.
- User can select and view contents of the file.
- A confirmation message is displayed to the user.

Failure Scenarios:

- User’s file is not uploaded.
- User’s file is not stored in the Weight Management View.
- User can't select and display the file.
- No confirmation message is displayed to the user.

User uploads **invalid** File:

Success Scenarios:

- File upload request is unsuccessful.
- An error message is displayed stating to follow the proper guidelines.
- User is given the option to select a different file to upload.

Failure Scenarios:

- File upload request is successful.
- No error message is displayed to the user.
- Incorrect error message is displayed to the user.
- User is not given the option to select a different file to upload.

User reaches maximum file uploads for the day:

Success Scenarios:

- Once the last file is successfully uploaded and stored, a message is displayed to the user stating the limit has been reached.
- Upload file option will no longer be visible until the next day.

Failure Scenarios:

- No message is displayed to the user regarding reaching the maximum file upload limit.
- Upload file option is still visible to the User.
- User is able to successfully upload more files exceeding the file upload limit.
- File is stored and displayed in the Weight Management View.

Testing Plan:

Execution Steps:

- User selects a record.
- User selects Input File option.
- User selects a File.
- User selects Insert File option.

- Test 1: User inputs Valid File
  - Integration Testing
  - Success Case:
    - Valid.jpg is uploaded onto the record after the database is checked to make sure the limit isn't breached.
    - Valid.jpg is also visible to user and may interact with it.
    - Message displayed to user "File successfully uploaded".
  - Failure Case:
    - Valid.jpg is not uploaded onto the record despite no limit breach.
    - Valid.jpg is not visible to the user.
    - XYZ.jpg is uploaded onto the record and visible to the user. (wrong file).

        - No/wrong message is displayed to the user.
    - Execution time: 2 minutes
- Test 2: User inputs Invalid File
    - Negative Testing:
    - Success:
        - Invalid.ppt is not uploaded onto the record.
        - Message is displayed to the user: “File Type not allowed. Please select a JPG or PDF”.
        - User is given the option to reselect a file.
    - Failure:
        - Invalid.ppt is accepted and uploaded onto the record.
        - No/wrong error message is displayed to the user.
        - User is not given the option to reselect a file.
    - Execution time: 2 minutes.
- Test 3: User inputs file exceeding Size Limit.
    - Unit Testing:
    - Success Case:
        - SizeTooBig.pdf is not uploaded onto the record.
        - Message is displayed to the user: “File too big. Please select a file that does not exceed 16 MB”.
        - User is given the option to reselect a file.
    - Failure Case:
        - SizeTooBig.pdf is uploaded onto the record.
        - No/wrong message is displayed to the user.
        - User is not given the option to reselect a file.
- Test 4: User inputs a file under the minimum size.
    - Unit Testing:
    - Success Case:
        - SizeTooSmall.pdf is not uploaded onto the record.
        - Message is displayed to the user: “File too small. Please select a file that is at least 0.5 MB”.

        - User is given the option to reselect a file.
    - Failure Case:
        - SizeTooSmall.pdf is uploaded onto the record.
        - No/wrong message is displayed to the user.
        - User is not given the option to reselect a file.
    - Execution Time: 2 minutes.
- Test 5: Test 2 files per day limit.
    - Negative Testing
        - Input:
            - “inp1.jpg” , “inp2.jpg”, “inp3.jpg”
        - Success:
            - inp1.jpg, inp2.jpg are uploaded onto record, inp3.jpg is rejected.
            - Message is displayed to the user. ‘“Two File Limit is reached. You may upload again in 24 hours”.
        - Failure
            - Inp1.jpg,inp2.jpg, and inp3.jpg are uploaded onto the record.
            - No/wrong message is displayed to the user.
        - Execution time: 3 minutes.

5.3 User selects the Edit Weight Goal Option:

Assumptions:

- User is logged in.
- User is at the Weight Management View.

Type of Testing:

Success Scenarios:

- Fields are editable.
- User is able to refill the target weight, exercise schedule, and date.

- Recalculations are made with the new user input.
- Updated results are reflected in the Weight Management View.

Failure Scenarios:

- Fields are not editable.
- User is unable to refill the target weight, exercise schedule, and date.
- Recalculations are made with the wrong values (old input or garbage values)
- Previous calculation results are reflected in the Weight Management View.

Testing Plan:

- Execution:
  - User selects the Edit Weight Goal option.
  - User re-selects Target Weight, Exercise Schedule.
  - User re-inputs Time Frame.

Testing Plan:

- Test 1: Save
  - Integration test
  - Input:
    - “120” and submit save.
  - Success Case:
    - User is informed that the input is saved.
    - Recalculations are made and correct results are displayed to the user.
  - Failure Case:
    - User is not informed that the input is saved.
    - No recalculations are made.
    - No correct results are displayed to the user.
  - Execution Time: 1 minute.

- Test 2: Database storage
    - Integration Testing
        - Input:
            - “120” and save
        - Success Case:
            - The input is saved to the database.
        - Fail Case:
            - The input is not saved to the database.
        - Execution Time: 1 minute.

5.4 User selects the Search Food Item Option:

Assumptions:

- User is logged in.
- User is at the Food Log View.

User selects the category to add food items under in their Food Log.

User inputs **valid** food item name in search bar

Successful Scenarios:

- Search request submission is successful.
- API request is made and returns correct results to the system.
- Food Log View is updated to display the search results.
- Search displays correct food items and calories that correspond to the user’s input.
- User is notified when no results are generated.

Failure Scenarios:

- Search request submission is unsuccessful.
- Search is not executed and results are not generated.
- No API request is made.
- API request is made and returns incorrect results to the system.

- Search displays wrong food items and calories that do not correspond to the user's input.

User inputs **invalid** food item name in search bar

Successful Scenarios:

- Search request submission is unsuccessful.
- An error message is displayed to the user stating to follow the proper guidelines.
- User is given the option to re-input food item name in search bar.

Failure Scenarios:

- Search request submission is successful.
- Search is executed and wrong results are generated and displayed.
- No error message is displayed to the user.
- Wrong error message is displayed to the user.

Testing Plan:

- Execution:
  - User selects Search Food Item option.
  - Select a Food Category.
  - Click on search bar.
- Testing Plan:
  - Test 1: Input valid Food Item Name.
  - Unit Testing
    - Input:
      - "Apples" is inputted for the food item name.
    - Success Case:
      - Search results that correspond to the user's input are displayed.
      - User is notified if no results are generated.
    - Fail Cases:

        - Search results that do not correspond to the user's input are displayed.
        - User is not notified if no results are generated.
    - Execution Time: 1 minute.

- Test 2: Input invalid Food Item Name.
  Note: Invalid Food Item Name are item names that do not exist, contain special characters or numbers, or exceed more than 15 characters.
    - Input:
        - "Enkmwd" is inputted for the food item name.
        - "2232" is inputted for the food item name.
        - "&apple" is inputed for the food item name.
        - "strawberries and cream cake" is inputted for the food item name.
    - Success Case:
        - Error message is displayed to the user.
        - Search is not conducted and displays results.

    - Fail Case:
        - No error message is displayed to the user.
        - Search is conducted and displays results.
    - Execution Time: 1 minute.
- Test 3: Validate API works properly
    - Input: "Apple"
        - Success Case:
            - System is returned information from the API that corresponds to the input. "Apple", "Apple Pie", etc.
            - System is returned 0 results from the API if no matches are found.
        - Fail Case:

- System is returned incorrect information that does not correspond to the input. ‘Yogurt” “Bananas” etc.
- Execution Time: 1 minute.

5.5 User selects the Add Food Item Option

Assumptions:

- User is logged in.
- User is at the Food Log View.
- User must have successfully searched for an item.
- User has less than 20 food items added to the Food Log in the 24 hours.

User selects the Add option near the food item.

Successful Scenarios:

- Food Log is updated to reflect the newly added food item.
- Added Food item is the correct user selected item.
- Food item is added to the correct user selected category.
- A message is displayed to the user stating the food item has been successfully added to their Food Log.

Failure Scenarios:

- The user is unable to save a Food item to their Food Log.
- Wrong food item is added to the Food Log.
- Food item is added under the wrong user selected category.
- No message is displayed to the user.

Testing Plan:

- Execution Steps:
  - User successfully searched for an item
  - User selected the Add Food Item Option.
  - User selects a Food Item.
- Testing Plan:
  - Test 1:

- Unit Testing
    - Input:
        - Add button near food item “Apple” is clicked.
    - Success Case:
        - Food item “Apple” is added to the Food Log.
        - Food item “Apple” is displayed to the user when viewing Food Log.
        - A confirmation message that the Item has been added to the Food Log is displayed.
    - Failure Case:
        - Food item “Apple” is not added to the Food Log.
        - A different food item is added to the Food Log.
        - Food item “Apple” is not displayed to the user when viewing Food Log.
    - Execution Time: 1 minute.
- Test 2: Testing Database storage.
- Integration testing.
    - Input:
        - Add button near food item “Apple” is clicked.
    - Success Case:
        - Food item “Apple” is successfully saved onto the Food Log.
        - Updated Food Log is successfully stored into the database.
    - Failure Case:
        - Food item “Apple” is not saved onto the Food Log.
        - Updated Food Log is not stored in the database. Test 3: Test Record Limit.
    - Execution Time: 1 minute.
- Tesk 3: Test Food Log Max Limit
    - Negative Testing
    - Input:

- “Apple” food item is added when Food is at its maximum limit.
- Success Case:
    - “Apple” is not added to Food Log and user is informed they’re on limit
- Failure Case:
    - “Apple” is added to Food Log and maximum limit is disregarded.
    - Execution Time: 1 minute

5.6 User selects the Add Custom Food Option:

User selects category to add food items under in their Food Log.

Assumptions:

- User must be logged in.
- User must be in the Food Log View.

User inputs **valid** custom food item Name, Description, and Calories:

Success Scenarios:

- Add Custom Food Item request is successful.
- Food Log is updated to reflect the newly added custom food item under the user selected category.
- A message is displayed to the user stating the food item has been successfully added to their Food Log.

Failure Scenarios:

- Add Custom Food Item request is unsuccessful.
- Custom food item is not added to the Food Log.
- Wrong food item is added to the Food Log.
- Custom food item is added under the wrong category.
- No message is displayed to the user stating the food item has been successfully added to their Food Log.

User inputs **invalid** custom food item Name, Description, and Calories:

Success Scenarios:

- Add Custom Food Item request is unsuccessful.
- An error message is displayed stating to follow the proper guidelines of the specific invalid input.
- User is given the option to re-input the wrong fields.

Failure Scenarios:

- “Add Custom Food Item” request is successful.
- Food Log is updated to reflect the newly added invalid custom food item.
- No error message is displayed to the user.
- Wrong error message is displayed to the user.
- User is not given the option to re-input the wrong fields.

Testing Plan:

- Execution Steps:
  - User selects Add Custom Food.
  - User inputs required data.
  - User submits.
- Testing Plan:
  - Test 1: Input valid Food Name, Description, and Calories.
    - Unit Testing:
      - Input:
        - Food Name: “LACNOR Banana Milk”
        - Food Description: “it is 180 ml, has 3 grams of fat, 0.1 grams of cholesterol, 177 mg of Calcium, etc.”
        - Calories: 150 kcal.
      - Success Cases:
        - Food item is created and added to the Food Log.
        - Food item “Banana Milk” is displayed to the user when viewing Food Log.

        - A confirmation message that the Item has been added to the Food Log is displayed.
        - Able to select the food item and view its details.
    - Failure Cases:
        - Food item is not created and added to the Food Log.
        - Food item “Banana Milk” is not displayed to the user when viewing Food Log.
        - A different Food item is displayed to the user when viewing Food Log.
        - Unable to select the food item and view its details.
    - Execution Time: 1 minute.

- Test 2: Input invalid Food Name, Description, and Calories.
  Note: Invalid Food Item Name are item names that exceed 15 characters, contain special characters or numbers. Invalid Descriptions exceed the 1000 character limit. Invalid Calories include nonintegers.
    - Unit Testing:
    - Input:
        - Food Name: “LACNOR Banana Milk LACNOR Banana Milk LACNOR Banana Milk LACNOR Banana Milk LACNOR Banana Milk LACNOR Banana Milk LACNOR Banana Milk LACNOR Banana Milk”
        - Food Description: exceeds 1000 characters.
        - Calories: “frfdsa” “%231”
    - Success Cases:
        - Food item is not created and added to the Food Log.
    - Failure Cases:
        - Food item is created and added to the Food Log.
        - Food item is displayed to the user when viewing Food Log.

        - A confirmation message that the Item has been added to the Food Log is displayed.
        - Able to select the food item and view its details.
    - Execution Time: 1 minutes.
- Test 3: Test every other combination of invalid/valid cases for Test Name, Description, and calories.
    - Unit Testing.
    - Execution Time: 5 minutes.

- Test 4: Test Food Log Max Limit
    - Negative testing
    - Input:
        - "LACNOR Banana Milk" food item is added when Food is at its maximum limit.
        - Success Case:
            - "Apple" is not added to Food Log and user is informed they're on limit
        - Failure Case:
            - "Apple" is added to Food Log and maximum limit is disregarded.
        - Execution Time: 1 minute

5.7 User selects the Delete Food Item Option.

Assumptions:

- User must be logged in.
- User must be in the Food Log View.
- At least one food item must be in the Food Log.

User selects the Food Item to be deleted.

Success Scenarios:

- A confirmation message is displayed to the user verifying whether they want to delete the food item or not.

- The selected food item is deleted.
- Food Log is updated to reflect the changes.
- A message is displayed to the user once the food item is deleted.

Failure Scenarios:

- No confirmation message is displayed to the user.
- Wrong food item is deleted.
- No food item is deleted.
- Food Log view is not updated to reflect the deletion of the food item.
- No message is displayed to the user once the food item is deleted.

Testing Plan:

Execution:

- User selects the Food Item Delete option.
- User selects the Food Item.
- User selects confirm.

- Tests Cases
    - Test 1- Delete a Food Item :
        - Unit Test
            - Input:
                - Delete Food Item “Apple” from Food Log.
            - Success Cases:
                - “Apple” is deleted.
                - A message is displayed to the user that “Apple” has been deleted.
                - “Apple” is no longer visible on Food Log.
            - Failure Cases:
                - “Apple” has not been deleted.
                - “LACNOR Banana Milk” has been deleted instead.
                - A message is not displayed to the user that “Apple has been deleted”
            - Execution Time: 1 minute.

- Test 2: Testing Database storage.
  - Integration testing.
    - Input:
      - Delete Food Item “Apple” from Food Log.

      - Success Case:
        - Food Item “Apple” is removed from Food Log.
        - Updated Food Log is successfully stored into the database.

      - Failure Case:
        - Food item “Apple” is not removed from the Food Log.
        - Updated Food Log is not stored in the database.
      - Execution TIme: 1 minute.

5.8 User selects the Counting Calories Option:

Assumptions:

- User is logged in.
- User is in the Calorie Counter View.
- System retrieves user inputted data from Weight Management and Food Log. System retrieves the user's personal data from their profile.

  Successful Scenarios:
  - The following calculations are successfully made:
    - User’s recommended daily calorie intake.
    - User’s total calories eaten.
    - User’s remaining available calories.
    - User’s surplus or deficit in calories.
    - User’s average amount of calories accross the week.

- Calculation results are correct.
- Calorie Counter View is updated to display these calculations.

Failure Scenarios:

- No calculations were made.
- System retrieves incorrect data to make the calculations.
- System retrieves no data to make the calculations.
- Incorrect results from calculations.
- Calorie Counter View does not display all calculation results to the user.

Testing Plan:

- Execution:
  - User selects count Calories Option.
- Testing Plan:
  - Test 1:
  - Input:

    -System retrieves information from Weight Management and Food Log.

  Success Cases:
  - Correct information corresponding to the user is retrieved from the Weight Management and Food Log.
  - Correct calculations are made.
  - Correct results are displayed to the User.

  Failure Cases:
  - Incorrect information is retrieved from Weight Management and Food Log.
  - Incorrect calculations are made.
  - Incorrect results are displayed to the User.

  Execution Time: 1 minute.

5.9 User selects the Export Food Log Option:

Assumptions:

- User is logged in.
- User is at the Food Log View.

User selects a destination to export Food Log to.

Successful Scenarios:

- Export food log request is successful.
- Correct Food Log is successfully exported to the user's selected destination.
- User can access the Food Log from their selected destination.
- Exporting the Food Log has no effect on the Food Log View in the application.
- A message is displayed to the user stating that the export was successful.

Failure Scenarios:

- Export food log request is successful.
- Correct Food Log is not exported.
- Correct Food Log is exported to the wrong destination.
- Wrong (old version or empty) Food Log is exported to the user’s selected destination.
- User is unable to access the exported Food Log.
- Exporting the Food Log affects the Food Log View in the application.
- No message is displayed to the user.

Testing Plan:

- Execution:
    - User selects the Export Food Log Option.
    - User selects the Food Log.
    - User selects the export destination.
    - Test 1: Export Food Log is selected:
        - Unit testing
            - Input:
                - Food Log: “AAAAA”.
                - Destination: Desktop.
            - Success Case:
                - “AAAAA” is successfully exported to the user’s desktop.

- Failure Case:
    - “BBBBB” is exported to the user’s desktop. (incorrect reminder)
    - “AAAAA” is exported to the user’s files. (wrong destination)
- Execution Time: 1 minute.

6. Track Records

Types of Testing:

- Unit, Negative, UI, Integration, Security, Stress, Performance, and End-to-End testing.

Assumptions:

- User is logged in.
- User is is at the Track Records View.
- At least one record exists.

6.1 User selects the Search Records Option.

User inputs a **valid** (existing) name or date of a previously created record:

Successful Scenarios:

- Track Records request is successful.
- Track Records View is updated to display the results.
- Search displays correct record(s) that correspond to the user’s input.
- User is notified when 0 results are generated.
- User is able to select a searched record and view its contents.

Failure Scenarios:

- Track Records request is unsuccessful.
- Search is not executed and results are not generated.
- Search is executed but Track Records View does not update to display the results.
- Search is executed and displays wrong records that do not correspond to the user’s input.

- User is not notified when 0 results are generated.
- User is unable to select a record to view its contents.

User inputs an **invalid** name or date for a previously created record:

Successful Scenarios:

- Track Records request is unsuccessful.
- An error message is displayed to the user stating to follow the proper guidelines.
- User will be given the option to re-input name or date in the search bar.

Failure Scenarios:

- Track Records request is successful.
- No error message is displayed to the user.
- Wrong error message is displayed to the user.
- Search is executed and displays wrong records.

Testing plan:

- Execution Steps:
  - User selects Search Records.
  - User inputs Name/Date of an existing record.
  - Test 1: Input valid data.
  - Input Testing:
    - Input:
      - “Record 1” is inputted for name
      - “10/01/21” is inputted for date
    - Success Case:
      “Record 1” is retrieved and can be viewed/categorized by user.
      “10/01/21” is retrieved and can be viewed/categorized by user.
    - Failure Case:
      - No record is displayed that matches either the user inputted name or date:
        “No records found”
      - Incorrect record is displayed to the user:

“Record 2” is retrieved and can be viewed/categorized by user.

- Execution time: 1 minute.

- Test 2: User chooses to view a specific record
- Unit Testing:
- Input:
  - “Record 1” is inputted for name
  - Success Case:
    - Contents of “Record 1” is viewable by the user.
    - Accurate information is displayed to the user.
  - Failure Cases:
    - Contents of “Record 2” is viewable by the user.
    - Inaccurate information is displayed to the user.
    - No record is shown
  - Execution time: 1 minute

6.2 User selects the Categorize Records Option:

Assumptions:

- User is logged in.
- User is in the Track Records View.
- User successfully performed a record search.

User selects the Create Folder Option.

User inputs **valid** Folder Name:

Successful Scenarios:

- Create Folder request is successful.
- Folder is created with the correct user inputted name.
- Track Records View is updated to display the newly created folder.
- User is able to select and view the contents of the folder.

Failure Scenarios:

- User is unable to submit the Create Folder Request.
- Folder is not created.

- Track Records View is not updated to display the folder.
- User is unable to select and view the folder.

User inputs **invalid** Folder Name:

Successful Scenarios:

- Create Folder request is unsuccessful..
- An error message is displayed stating to follow the proper guidelines of the specific invalid input.
- User is given the option to re-input the name of the folder.

Failure Scenarios:

- User is able to submit Create Folder request.
- Folder is created with the invalid inputted name.
- No error message is displayed to the user.
- Incorrect error message is displayed to the user.
- User is not given the option to re-input the folder name field.

Testing Plan:

- Execution Steps:
  - User selects Create Folder.
  - User creates a valid folder.
  - User inputs required fields.
- Testing Plan:
  - Test 1: User inputs valid folder name
  - Unit Testing:
    - Input:
      - “Folder a”
      - Success Cases:
        - Folder is created
      - Failure Cases:
        - Folder is not created
      - Execution time: 1 minute
    - Test 2: User inputs invalid folder name
    - Negative Testing:

- Input:
    - “Aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa aaaaaaaaaaaaaaaaaaaaaaaaaaaa…..”
- Success Cases:
    - Folder is not created
    - Error given that folder name is too large
- Failure Cases:
    - Folder is created
- Execution time: 1 minute

6.3 View Folder

Assumptions:

- User is logged in.
- User is in the Track Records View.
- Folders must exist.

User selects the folder to be viewed.

Successful Scenarios:

- View Folder request is successful.
- User is able to view the contents of the folder.
- User is able to search for a record that was saved to the folder.
- User is able to select and view the records saved into the folder.

Failure Scenarios:

- View Folder request is unsuccessful.
- User is redirected to the wrong Folder View.
- User is unable to search for a record in the folder.
- User is unable to select and view the records in the folder.

Testing plan:

- Execution Steps:
- User selects a folder.
- Testing Plan:

- Test 1: User selects a folder
- Unit Testing:
    - Success Case:
        - Folder contents are available to view
    - Failure Case:
        - User is directed to wrong folder
        - User is unable to view folder
        - User can not view specific contents of a folder
    - Execution time: 1 minute.

7- Diet Recommendations

Types of Testing:

- Unit, Negative, API, UI, Integration, Security, and End-to-End Testing.

Assumptions:

- User is logged in.
- User information is retrieved from Profile.
- User is in the Diet Recommendations View.
- 10 dietary questions are displayed to the User.

7.1 User selects the Create Diet Recommendation option

User selects answers to all the questions.

Successful Scenarios:

- Create Diet Recommendation request is successful.
- API request is made and returns correct, personalized results to the system.
- Diet Recommendations View is updated to display the results to the user.
- User is able to select meal items from the meal list.

Failure Scenarios:

        - Create Diet Recommendation request is unsuccessful.
        - Search is not executed and results are not generated.
        - No API request is made.
        - API request is made and returns incorrect results to the system.
        - Diet Recommendations View is not updated to display results.
        - User is unable to select items from the list.

Testing plan:

- Execution:
    - User selects Create Diet Recommendation.
    - User inputs:
        - Height
        - Weight
        - Weight Goals
- Testing Plan:
    - Test 1: User answers questionnaire
    - Unit Testing:
        - Input:
            - “5’8” inputted for height
            - “150” inputted for weight
            - “170” inputted for weight goals
        - Success Case:
            - Diet recommendation request is successful
            - API returns correct personalized data
            - User is able to view data and recommendations
        - Failure Case:
            - API makes no or wrong request
            - Diet recommendation is not displayed
            - Errors given
        - Time Execution: 2 minutes.

7.2 View Details of Recommended Meals.

User selects a meal from the generated list in the Diet Recommendations View.

Successful Scenarios:

- User is redirected to the Meal View.
- Correct details of the selected meal are displayed to the user.

Failure Scenarios:

- User is not redirected to the Meal View.
- User is redirected to the incorrect Meal View.
- Incorrect information is displayed to the user.
- No information is displayed to the user in the Meal View.

Testing Plan:

- Execution Steps:
  - User selects a meal.
- Testing Plan:
  - Test 1: User chooses a specific meal from view
  - API Testing:
    - Input:
      - API filters through user questionnaire answers
    - Success Cases:
      - Accurate meal view is displayed
    - Failure Cases:
      - User is given wrong meal view
      - User is not shown anything
    - Execution Time: 1 minute.

7.3 Add Meal to Meal List

Assumptions:

- User must be logged in.
- User must be in the Diet Recommendations View.
- A list of meals has been successfully generated.
- Meal list contains less than 5 meals in 24 hours.

User selects the Add Meal Option

Successful Scenarios:

- Add Meal request is successful.

- A message is displayed to the user to confirm adding the meal item to the Meal List.
- Meal List View is updated to reflect the newly added meal item.
- Added meal item is the correct user selected item.

Failure Scenarios:

- Add Meal request is unsuccessful.
- No confirmation message is displayed to the user.
- Selected meal item is not added to the Meal List.
- Incorrect meal item is added to the Meal List.

Testing Plan:

- Execution:
  - User selects add meal to meal plan.
  - User searches for a meal.
  - User selects a meal from the generated search list.
- Testing Plan:
  - Test 1: User chooses Add Meal
  - Unit Testing:
    - Success Cases:
      - Meal is added to meal list
    - Failure Cases:
      - Meal is not added to meal list
      - Incorrect meal is added to list
    - Time Execution: 1 minute.

7.4 View Meal Item from Meal List.

User selects a meal from the meal list.

Assumptions:

- User is logged in.
- User must be in the Meal List View.

Successful Scenarios:

- User is redirected to the Meal View.

- Meal View displays meal item's details.
- Correct meal item details are displayed to the user.

Failure Scenarios:

- User is not redirected to the Meal View.
- User is redirected to the wrong Meal View.
- Meal View does not display any information regarding the selected meal item.
- Meal View displays incorrect information regarding the selected meal item.

Testing Plan:

- Execution:
- User selects meal from meal list.
- Testing Plan:
    - Test 1: User chooses a specific meal from list
        - "Banana Bread"
    - Unit Testing:
        - Success Cases:
            - Meal details are displayed.
              "Banana Bread is……"
        - Failure Cases:
            - No meal details are displayed
            - Wrong meal details displayed.
              "Walnut Bread is……"
        - Execution Time: 1 minute.

7.5 Delete Meal Item from Meal List.

User selects the delete option of a meal item from the Meal View.

Assumptions:

- User must be logged in.
- User must be in the Meal List View.
- Meal List has at least one meal added to it.

Successful Scenarios:

- Delete Meal request is successful.
- A confirmation message is displayed to the user verifying whether they want to delete the meal item or not.
- Meal is deleted from the Meal List.
- Meal List View is updated and reflects the removal of the meal item.

Failure Scenarios:

- Delete Meal request is unsuccessful.
- No confirmation message is displayed to the user.
- Meal item is not deleted from the Meal List.
- The wrong meal item is deleted from the Meal List.
- Meal List View does not reflect the removal of the meal item.

Testing Plan:

- Execution:
- User selects meal from meal list.
- User selects delete meal.
- User confirms delete.
- Testing Plan:
    - Test 1: Uses chooses delete option when viewing a meal
    - Unit Testing:
        - User Input:

            Meal: “Banana Bread.”
        - Success Cases:
            - “Banana Bread” is no longer a part of meal list.
            - System and datastore is updated to reflect this change.
        - Failure Cases:
            - Meal is still a part of meal list
            - Wrong meal is deleted.
            - System and datastore is not updated to reflect this change.

- Execution Time: 1 minute.

8- Hot Topics

Types of Testing:

- Unit, Negative, UI, API, Integration, Security, and End-to-End testing.

Assumptions:

- User must be logged in.
- User must be in the Hot Topic View.

8.1 User selects the Hot Topic Search option.

User inputs **valid** Zip Code:

Successful Scenarios:

- Hot Topics Search request is successful.
- API request is made and returns correct results to the system.
- Hot Topics View is updated to display the results.
- User is notified when no results are generated.
- User is able to select an item from the search list to view its contents.

Unsuccessful Scenarios:

- Hot Topics Search request is unsuccessful.
- Search is not executed and results are not generated.
- No API request is made.
- API request is made and returns incorrect results to the system.
- Search displays incorrect information that does not correspond to the user's inputted zip code.
- User is user is unable to select an item and view its contents.

User inputs an **invalid** zip code input:

Successful Scenarios:

- Hot Topics Search request is unsuccessful.
- No API request is made.
- An error message is displayed to the user stating to follow the proper guidelines.

- User is not given the option to re-input the zip code in the search bar.

Failure Scenarios:

- Hot Topics Search request is successful.
- API request is made.
- Search is executed and results are generated.
- No error message is displayed to the user.
- Wrong error message is displayed to the user.
- User is not given the option to re-input the incorrect fields.

Testing Plan:

- Execution Plan:
    - User makes a Hot Topics Search request.
    - User inputs a zip code.
    - User submits search.
- Test 1: Check user input with valid data.
    - API Testing:
        - Input:
            - “90840” for zipcode
        - Success Cases:
            - Hot topics from the area displayed
        - Failure Cases:
            - Hot topics from the wrong area displayed
        - Execution Time: 1 minute.
    - Test 2: Check user input with invalid data
    - Negative Testing:
        - Input:
            - “908a0” for zipcode
        - Success Cases:
            - User is told to re-enter zipcode
        - Failure Cases:
            - Search is made and hot topics displayed
        - Execution Time: 1 minute.

8.2 Opt-In Notifications:

Assumptions:

- User is logged in.
- User is at the Hot Topics View.

User confirms zip code:

Success Scenarios:

- User receives emails on hot topics around them

Failure Scenarios:

- User does not receive any emails
- User receives emails on hot topics from a different area

Testing Plan:

- Execution: User verifies information
- Testing Plan:

  Assumptions:

  - Test 1:Check user input with valid data.
  - Unit Testing:
    - Success Cases:
      - User starts receiving relevant hot topics through email
    - Failure Cases:
      - User receives no emails regarding hot topics
      - User receives the wrong hot topics
    - Execution Time: 1 minute.

  Testing Plan:

  Execution Steps:

  - User enters a zip code.
  - User selects the opt-in reminder notification option.
  - User confirms to receive notifications of news happening around their area.
  - Test Case 1: User is notified of the News:
    - Unit Testing:
    - Input:

      Zip Code: 92832

Notifications: Email.

- Success case:
    - An email is sent to the user at the appropriate timing regarding the news arround them.
    - “Thunderstrom near Long Beach….”
    - User is successfully notified.
- Failure case:
    - User is not notified of the news.
    - User is notified of inaccurate news.

    Execution Time: 1 minute.

9. Registration

Assumptions:

- User is on registration page
- User does not have an existing account
- User is not logged in to existing account

9.1 User attempts to register and create an account.

User inputs valid email and password:

Success Scenarios:

- User’s data is store
- User is taken back to homepage already logged in
- Welcome email is sent

Failure Scenarios:

- Data is not saved
- Data is seen as invalid and prompts user to enter valid data
- Incorrect data is saved
- User is not logged in once registering
- No email is sent

User inputs invalid email or password:

Success Scenarios:

- User is given one or more of the following errors:
    - Invalid username
    - Invalid email format
    - Invallid password

        - Missing special character
        - Too short
        - Repetitive
    - Data is in a saved state so user does not have to re-enter valid data

Failure scenarios:

- No error is show
- Incorrect error shown
- Account created
- Valid data is not saved

User enters information already belonging to another account:

Success Scenarios:

- User is told an account with matching information already exists.
- User is give option to be taken to login page
- Data is in a saved state so user does not have to re-enter valid data

Failure Scenarios:

- Duplicate account is created
- No error is given
- Wrong error given
- Valid data is not saved
- Wrong error displayed

9.1 Test Plan:

- Example Input: “bobby@gmail.com:hks!ksmAA”
- Example Username: “DaBaby2”
- Execution steps:
    - Make sure you are logged out
    - Click Register
    - Enter valid email,username, and password in respective text boxes
- Unit Testing:
    - Success Case:

        Display ‘Successfully Registered’

        Redirected to homepage
    - Failure Case:

        Display ‘Registration Failed’
    - Time Execution: 1 minute.
- Integration Testing:
    - Success Case:

Confirms that account is stored in database

- Failure Case:

  Account is not stored in database

- Time Execution: 1 minute.

- Negative Testing:
  - Example Input: “bobby!!gmail.com:hks!ksmAA”

    Make sure you are logged out

    Click Register

    Enter invalid email,username, or password in respective text boxes
  - Success Case:

    Displays ‘Registration Failed’
  - Failure Case

    Displays ’‘Successfully Registered’
  - Time Execution: 1 minute.

9.2 Registering account is provided a username.

Assumptions:

- System sends a confirmation email to the user that is only valid for 24 hours.
- A message is displayed to the user notifying them of the confirmation email.

Required Input:

- User most select the link provided in the email sent to them.

Success Scenarios:

- User is provided with a unique username that is associated with his account.
  - Username will consist of:
  - At most 15 characters.
  - a-z,A-Z,0-9, .,@!
- A message is displayed to the user confirming that the account has been successfully created.

- User is able to login with the provided username.
- Changes are made to the database.

Failure Scenarios:

- User is not provided a username.
- User is not provided a unique username.
- User is unable to login with the provided username.
- No changes are made to the database.

Testing Plan:

User is in Registration View.

- Execution Steps:
  - User inputs email.
  - User inputs password.
  - User clicks on the link provided in the email that was sent to them by the system.
- Testing Plan:
  - Test 1: Creation of Username
  - Unit Testing:
    - Success Case:
      “Your username is xyz123”
    - Failure Case:
    - “Your username is emememem” (username already taken)
    - “Your username is “toolonogtoolongtoolong” (doesnt follow proper guidelines)
    - (blank)

  Execution Time: 1 minute.

10 Login

Assumptions:

- User is on any view logged out

- User selects login button and
- User information is retrieved from the Profile.

10.1 User attempts to log into their account

User provides valid login Email and Password:

Success Scenarios:

- User is logged in and taken to homepage

Failure Scenarios:

- User is told they entered wrong username/password
- User is not logged in but taken to homepage
- User remains in login page with no update

User inputs invalid login information:

Success Scenarios:

- User is told they entered wrong username/password
- User remains in login page
- User is told they have five opportunities before being locked out

Failure Scenarios:

- User is logged into an account
- User is sent back to homepage
- No error is displayed
- Wrong error displayed

10.1 Test Plan:

- Execution steps:
  - Create account if you do not already have one
  - User clicks Login .
  - Enters Email and Password in respective text fields
- Example Input: “bobby@gmail.com:hks!ksmAA”
- Test 1 Unit Testing:
  - Success Case:

    Successfully Logged in
  - Failure Case:

    Displayed ‘Invalid Login’
  - Time Execution: 1 minute.
- Test 2 Integration Testing:
  - Success Case:

Backend confirms that it is the correct information being input.

- Failure Case:

  403 Error from backend

- Time Execution: 1 minute.

- Test 3 Negative Testing:
  - Steps:

    Create account if you do not already have one

    User clicks Login

    Enters incorrect Email and Password in respective text fields

  - Success Case:

    Displays ‘Invalid Login’

  - Failure Case

    Successfully logs into account with invalid credentials

  - Time Execution: 1 minute.

10.2 User inputs five wrong username/password combinations:

Success Scenarios:

- User is temporarily locked out
- User is taken back to homepage

Failure Scenarios:

- User is logged in
- Incorrect attempt is not saved
- User is not shown error
- Wrong error shown

10.2 Test Plan:

- Example Input: “bobby@gmail.com:hks!ksmAA”
- Execution steps:
  - Create account if you do not already have one
  - User clicks Login .
  - Enters an incorrect Email and Password in respective text fields and click ‘Login’
  - Repeat previous step 5 times
- Test 1: Unit Testing:

- Success Case:

    Displays ‘Your account is locked! Try again later or reset password.’
- Failure Case:

    Does not display text saying account is locked
- Time Execution: 2 minute.

- Test 2: Integration Testing:
    - Success Case:

        Account is flagged as locked in the database.

        403 Error
    - Failure Case:

        Database does not have the account flagged as locked.
    - Time Execution: 2 minute.

10.3 User has two step authentication

Assumptions:

- User has an account
- 2FA is enabled to you
- User has correct username and password
- User is in login page

User enters username/password information

Success Scenarios:

- Email verification is sent in a timely manner
- User is taken to passcode page

Failure Scenarios:

- Email is never sent
- Email arrives late
- User is logged in without needing to verify
- User is not taken to passcode page
- User is taken to homepage
- User is taken back to login page

User enters correct passcode:

Success Scenarios:

- User is verified and logged in
- User is taken back to homepage

Failure Scenarios:

- User is not logged in
- User is taken to homepage
- User is taken back to login page

- No error displayed
- Wrong error displayed

User enters invalid passcode:

Success Scenarios:

- User is given error
- User remains in passcode page
- User is told they have 5 tries before being locked out

Failure Scenarios:

- User is logged in
- User is sent back to homepage
- No error is displayed
- Wrong error displayed

10.3 Test Plan:

- Example Login: bobby@gmail.com:hks!ksmAA
- Auth code must be retrieved from authenticator

- Execution steps:
  - Create account if you do not already have one
  - Ensure 2FA is enabled and sign out.
  - User clicks Login .
  - Enters Email and Password in respective text fields and click ‘Login’
- Test 1: Unit Testing:
  - Success Case:

    Account is logged in, and redirected to home page.
  - Failure Case:

    Account is not logged in. Displays Internal Error.
  - Time Execution: 2 minute.
- Test 2: Integration Testing:
  - Success Case:

    Logs in user from back end comparing generated auth code successfully.
  - Failure Case:

    Auth code is successful but Log In fails. Internal Error
  - Time Execution: 2 minute.

- Test 3: Negative Test:
    - Example Login: bobby@gmail.com:hks!ksmAA
    - Auth code: “abcd”
    - Success Case:

        Account is not logged in.
    - Failure Case:

        Account is logged in. Authentication Error.
    - Time Execution: 2 minute.

11 Core Components

Note:
In the document provided in Core Requirements, we have used some contents in there to aid us in our Test Plan: BitOHealth.

11.1 Authentication

Assumptions:
- User must be logged out.
- User must be in Login View.
- User selects Login Option.
- User information is retrieved from the Profile.

11.1a User attempts to log into their account.
User enters the following **valid** information in the designated fields:

- Email address or username.
- Password.
- OTP.

Success Scenarios:
- User is logged in to their account.
- User is redirected to the user’s Home Page.
- User is able to access any pages they are authorized to.

Failure Scenarios:
- User is unable to login.
- User logs in but not redirected to Home Page.

- User remains in login page with no update.
- User is notified they entered wrong username/password.

User inputs **invalid** login information:

Success Scenarios:

- User is notified they entered wrong username/password
- User remains in Login Page and not redirected to Home Page.
- User is notified of the last attempt before account is disabled and locked.
- If user's account is disabled, user is notified to contact System Administrator for account recovery.

Failure Scenarios:

- User is redirected to the Home Page of an account.
- No error is displayed to the user.
- Wrong error displayed to the user.

11.1a Test Plan:

- Execution steps:
  - Create account if you do not already have one
  - User clicks Login .
  - Enters Email, Password, OTP in respective text fields.
- Example Input of valid credentials:
  - Email: bobby@gmail.com
  - Password: Password123!
  - OTP: 18732135
- Test 1 Unit Testing:
  - Success Case:

    Successfully Logged in.

    Redirected to Home Page.
  - Failure Case:

    System displays 'Invalid Login'

    User may retry up to 5 times.
  - Time Execution: 1 minute.
- Test 2 Integration Testing:
  - Success Case:

    Backend confirms that it is the correct information being input.

- Failure Case:

    403 Error from backend.
- Time Execution: 1 minute.

- Test 3 Negative Testing:
    - Steps:

        Create account if you do not already have one

        User clicks Login

        Enters incorrect Email, Password, OTP in respective text fields:

        - Email: bobby1.com
        - Password: password123
        - OTP: 12345
    - Success Case:

        Displays ‘Invalid Login’

        User may retry up to 5 times.
    - Failure Case

        Successfully logs into account with invalid credentials.
    - Time Execution: 1 minute.

11.1b User inputs Five invalid username/password combinations:

Success Scenarios:

- User is temporarily locked out.
- User is notified to contact System Administrator.

Failure Scenarios:

- User is logged in.
- User is not notified of error.
- Wrong error is displayed to the user.
- User is able to input credentials again before contacting system admin.

11.1b Test Plan:

- Example Input:
    - Enters incorrect Email, Password, OTP in respective text fields:
        - Email: bobby1.com
        - Password: password123
        - OTP: 12345

- Execution steps:
    - Create account if you do not already have one
    - User selects Login .
    - Enters an incorrect Email, Password, and OTP in respective text fields and selects ‘Login’ option.
    - Repeat previous step 5 times.
- Test 1: Unit Testing:
    - Success Case:

        Displays: “Your account is locked. Please contact System Administrator”
    - Failure Case:

        No message is displayed to the user.
    - Time Execution: 2 minute.
- Test 2: Integration Testing:
    - Success Case:

        Account is flagged as locked in the database.

        403 Error.
    - Failure Case:

        Database does not have the account flagged as locked.
    - Time Execution: 2 minutes.

11.2 Logging

Assumptions:

- User must have an active authenticated session on the device.
- User must be in Logging view.
- User must be a System Administrator.
- Persistent data store must be active.
- Persistent data store must be accessible by the system.
- Persistent data store must have storage capacity for log entry.
- Log entries include:
    - System and user success events.
        - System and user failure events.

        - Login/Logout events.
        - Pages accessed
        - Operation performed during every page access.

Success Scenarios:

- Every active session, system logs all required information.
- Information regarding every log entry will be recorded:
    - Timestamp in which it occurred.
    - Log Level.
    - Operation performed.
    - Category.
    - Description of log entry.
- All logs will be saved to the datastore.
- Logging continuously occurs as long as the web application is active.
- Logging does not block anyone from using the web application.

Failure Scenarios:

- System does not make accurate logs.
- No information/ missing information regarding logged entries.
- Logs are not saved to the database.
- Logging stops occurring despite web application being active.
- Logging blocks users from using the web app

11.2a Test Plan:

- Execution Steps
    - System Administrator selects View Logging.
    - System Administrator waits for a log entry to appear on Logging View or views previous entries.
- Test 1 End to End Testing:
    - Success Case:
        - An action or data is accurately logged to the database.
        - System admin is viewing logged entries.
        - System admin may modify logged entries.
        - Users are still able to interact with the system.
    - Failure Case:
        - An action or data is not logged to the database.
        - An action or data is not accurately logged to the database.
        - System admin is unable to view logged entries.
        - System admin is unable to modify logged entries.
        - Users are not able to interact with the system.

- Time Execution: 3 minutes.

11.3 Archiving

Assumptions:
- User must have an active authenticated session on the device.
- User must be on Archiving View.
- User must be a System Administrator.
- Persistent data store must be active.
- Persistent data store must accessible by the system.
- Archival destination location must have storage capacity.

Success Scenarios:

- Archiving takes place on the 1st of every month at exactly midnight.
- Log entries older than 1 month will be archived.
    - Log entries include:
        - System and user success events.
        - System and user failure events.
        - Login/Logout events.
        - Pages accessed.
        - Operation performed during every page access.
- System administration may access and view archived log entries.
- Archived logs are deleted from the system.
- Database is updated.

Failure Scenarios:
- Archiving takes place on a date other than the 1st of every month at exactly midnight.
- Incorrect logs are being archived.
- Incomplete logs are being archived.
- System administration unable to access and/or view archived log entries.
- Archived logs are still saved in the system.
- No updates to the database occur.

11.3a Test Plan:
- Execution Steps
    - System administrator is in an active authenticated session.
    - System administrator waits for an action or data to appear on Archiving View

- Test 1 End to End Testing:
    - Success Case:

        - Logs that are older than 30 days are successfully archived.
        - Logs are archived at a maximum of 1 minute upon invoking.

- Failure Case:
    - Logs that are older than 30 days are not archived.
    - Inaccurate logs are archived.
    - Logs are archived to the wrong destination.
    - Logs taking longer than 1 minute to archive.

Execution Time: 3 minutes.

11.4 User Management

Assumptions:
- User must have an active authenticated session.
- User must be in User Management View
- User must be a System Administrator.

Success Scenarios:
- System administrator will be given the option to perform any of the following operations on existing user accounts:
    - Create, Update, Delete, Disable, and Enable Accounts.
- System administrator will be able to perform bulk operations which should affect all users in the system.
- A message will be displayed to the System Administrator upon successfully completing an operation.
- An error message will be displayed to the System Administrator in case of any failure to complete an operation.

Failure Scenarios:
- System Administrator is unable to perform operations.
- System Administrator successfully performs operations, but the system does not update to reflect changes.
- No message is displayed to the System Administrator after completing an operation.
- No error message/wrong error message is displayed to the System Administrator when failing to complete an operation.
- Operations are not performed in a timely manner.

11.4a Test Plan:

- Execution Steps:
  - Administrator is in User Management View.
- Test 1 Integration Test
  - An Administrator attempts to Create User:

    Success Case:
    - A user account record is stored in datastore which saves the user's email and a hash value of their password.
    - A user account already existing in the data store cannot be duplicated.

    Failure Case:
    - A record is not stored in the data store.
    - Duplicate user records are stored into the datastore.
  - An administrator attempts to Delete Account:

    Success Case:
    - The selected user account record is deleted from the data store.

    Failure Case
    - The selected user record is not deleted from the datastore.
    - The incorrect user is deleted and not the one intended for deletion.
      i.e Username is not case-checked, therefore a user with a similar name but not identical is deleted instead.
  - An administrator attempts to Update Account:

    Success Case:
    - The selected user account record contents are updated such as email or password from the data store.

    Failure Case
    - The selected user record is not updated from the datastore.
    - The incorrect user is updated and not the one intended for updating.
      i.e Username is not case-checked, therefore a user with a similar name but not identical is deleted instead.
    - The selected user's password is not updated to the data store correctly.
    - The selected user's email is not updated to the data store correctly.
    - The administrator uses an existing email and the data store updates it.

- An administrator attempts to Enable a Disabled Account:
    Success Case:
        - The selected user account is able to login and updated from the data store.
    Failure Case
        - The selected user (account holder) is not able to login after going through the enable account process.
        - The incorrect user is enabled and not the one intended for the process.
          i.e Username is not case-checked, therefore a user with a similar name but not identical is enabled instead.

- An administrator attempts to Disable a Active Account:
    Success Case:
        - The selected user account is disabled/locked out and the system message displays “Account is disabled.”
        - An administrator cannot disable an disabled account and system message displays “Cannot disable a disabled account”.
    Failure Case
        - The selected user account is not disable and user can log in.
        - The incorrect user is deleted and not the one intended for deletion.
          i.e Username is not case-checked, therefore a user with a similar name but not identical is deleted instead.
        - The data store does not update correctly.

- An account holder attempts to delete the account
    Success Case:
        - The account holder record is deleted from the data store and is simultaneously logged out.
    Failure Case
        - The account holder record remains in the datastore.
        - The user remains logged in after successful account deletion
    Execution Time: 10 minutes.

- Test 2 End to End Test
    - An Administrator attempts to reset a user’s (an account holder) password

Success Case:

- A password reset link is sent to the preferred account holder's email.
- The password reset link sent should be a valid link.
- The password reset link should expire within 15 minutes.
- The new password associated with the account should not be the same as the old password,
- Upon completed password reset, the new password should be updated in the account holder's records.

Failure Case:

- No password reset link is sent to the account holder's preferred email
- The password reset link is sent to the account holder's preferred email.
- The new password is not stored in the datastore
- The new password does not update the old password
- Duplicate user records are stored into the datastore

Execution Time: 5 minutes.

11.5 User Analysis Dashboard

Assumptions:

- Persistent data store must be active.
- Persistent data store must accessible by the system.
- User must have an active authenticated session on the device.
- User must be on Usage Analysis Dashboard view.
- User must be a System Administrator.

Succ

11.5a Test Plan:

- Execution Steps:
  - User is in User Analysis Dashboard View.
- Test 1 Integration Test
  - Dashboard correctly outputs the calculation of metrics to be measured within our system.

Success Case:

- The calculation on the data retrieved is correctly done. I.e. The percentage of usage for a particular component.

Failure Case:

- The data calculations are not understandable

        - The data calculated has no output, just garbage
        - The data calculated gives negative results.

    Execution Step: 1 minute.

- Test 2 End to End Test
    - An Administrator selects a graph to view

        Success Case:
        - The admin is shown the proper graph that they selected to view.

        Failure Case:
        - The graph the administrator has selected is not showing.
        - The administrator is not shown the correct graph.
        - The correct visualization is not shown the given the calculated values (graph is inaccurate, showing wrong values or displaying a readable graph)
        - Account's sensitive information is logged or represented somehow on the graph or in the analysis
    - An Administrator updates or refreshes a visualization

        Success Case:
        - The graph will be refreshed and updated with the current information (i.e. If a user creates a new registration then that means loggin that information will alter the current number holders with BitOHealth)

        Failure Case:
        - The graph is not updated. New information is not retrieved upon refresh of the user analysis dashboard by an administrator.

        Execution Time: 2 minutes.
    - The data required for the User dashboard is stored/present in the database for retrieval.

        Success Case:
        - The necessary data for the dashboard can be queried

        Failure Case:
        - The data needed is not being logged by the logging feature
        - The data that is meant to be queried for the dashboard is not present in the datastore or the information is insufficient.

        Execution Time: 2 minutes.

11.6 Datastore

Assumptions:

- Persistent data store must be active
- Persistent data store must accessible by the system
- User must have an active authenticated session on the device
- User must be on Datastore view
- User must be a system administrator

11.6a Test Plan:

- Execution Steps:
    - System administrator is in an active authenticated session
    - System administrator waits for an action or data to appear on Datastore View

- Test 1 End to End Testing:

    Success Case:

    - The data store will store all data
    - The datastore will store all data required for feature functionality
    - The datastore will store all necessary data that is produced from certain features.

    Failure Case:

    - The datastore does not store data
    - The datastore jumbles data and inaccurately store data.
    - The datastores unnecessary data that is not needed for future querying.
    - Features that access data in the data store will not alter the data, causing loss of data integrity

    Execution Time: 5 minutes.

11.7 Account Deletion

Assumptions:

- User must have an active authenticated session
- User must be on account deletion view
- User has permission to delete account

11.7a Test Plan:

- Execution Steps:
    - User is in Account Deletion View

- Test 1 Integration Testing
    - A user (account holder or System admin) attempts to delete account
      Success Case:
        - The user (account holder) is deleted from the data store and is simultaneously logged out

      Failure Case:
        - The user (account holder) record remains in the data store.
        - The user remains logged in after successful account deletion.

  Execution Time: 2 minutes.

11.8 Account Recovery

Assumptions:

- User must not have an active authenticated session on the device, otherwise the user is unable to perform the operation
- User must be in the account recovery view.

11.8a Test Plan:

- Execution Steps:
    - User is on Log In View
    - User selects Account Recovery
    - User provides valid username and associated email
- Test 1 Integration Testing
    - A user (account holder) attempts to recover to an active or disabled account

      Success Case:
        - The user (account holder) is logged in and a system message displays “Account Recovery completed successfully” within 5 seconds of completion.

      Failure Case:
        - User inputs invalid username and/or email.
        - Account recovery email is sent too late.
        - User inputs all the required fields, but no account recovery email is sent to them.
        - User receives account recovery email, but does not click the link.
        - User receives account recovery email, clicks the link, and account is not recovered.
        - User receives account recovery email, clicks the link, and incorrect account is recovered.

Execution Time: 4 minutes.

12. Authorization/Security

This section will consist of mainly automated functions which will run when running our Testing Plan. Authentication and security is hard to test through only navigating the website.

Requirements: Our internal testing files, Python 3.8 and above , Selenium
Execution Steps: Download testing folder. Execute main file.

Test Case	Steps to create?	Type?	Execution Time	Success	Fail
Authorized User Check	Log into a user account, and get the path/link for a Record. Sign out, and sign into another account. In address bar of browser enter the Record path.	Security Test	1 Minute	403 Error: User does not have access.	User is allowed to view a Record
XSS Testing	In user input locations such as where you enter your Email to log in, you enter a string such as <script>alert('XSS')</script>. You then do this multiple times with different	Security Test	5 minutes	No XSS Vulnerabilities found.	XSS Vulnerability found.

	common xss attacks.				
Brute Force Testing	Get proxies from team members. Use python to create a multithreaded requests application that tries logging into one account from many proxies.	Stress Test	5 minutes	Brute force attack occurring.	Brute force attack not detected.

Cost, Effort, and Resources

Summary of Test Features and Resources

No.	Features	# of Testing Plan	Time (min)	Resource
1	Medication/Health Recorder	14	20	Dante
2	Reminders	8	10	Angel
3	Health Locator	8	10	Rami
4	Medication Lookup	13	14	Emily
5	Weight Management/Calorie Counter	22	30	Sebastian
6	Track Records	5	5	Jacob
7	Diet Recommendations	5	6	Dante
8	Hot Topics	3	3	Angel
9	Registration	3	3	Rami
10	Log in	8	13	Emily
11	Registration	3	11	Jacob

12	Authentication	5	7	Dante
13	Logging	1	3	Emily
14	User Management	2	15	Angel
15	User Analysis Dashboard	3	5	Jacob
16	Datastore	1	5	Rami
17	Account Deletion	1	2	Sebastian
18	Account Recovery	1	4	Dante

Summary of Efforts and Cost

No.	Resource Name	Effort (min)	Cost ($30/hr) (0.5/1min)
1	Dante	37	18.5
2	Angel	28	14
3	Jacob	21	10.5
4	Sebastian	32	16
5	Emily	30	15
6	Rami	18	9
Totals		166 (2.8 hrs)	$83

Total Estimated Time to complete the testing is approximately 3 hrs (166 minutes) with a total cost of $83 with the assumption of 30$/hr per resource.