Software Implementation and Testing Document

For

Group 11

Version 1.0

Authors:
Isaiah Alex
Charlie Penner
Gordan Leadbetter
Eli Bendavid

1. Programming Languages (5 points)

We utilized Python for the main application and backend. We used HTML, CSS, and JavaScript for the front end. We used Python because it was easy to pick up and use as a programming language and some of us had experience with it. HTML was our choice for the front end as it was simple to use and with the capability of using CSS and JavaScript to style it and add functions to the frontend.

2. Platforms, APIs, Databases, and other technologies used (5 points)

We used Flask for setting up the front end and allowing us to link html pages which it was capable of rendering. We used SQLite for the database to allow us to have user registration and having users track their recipes by allowing users to add and create recipes which is stored for them. We also use a Recipe Search API from Edemam which entails its own recipe database that allows users to search for recipes based off a given ingredient. The link for this API is: https://developer.edamam.com/edamam-recipe-api/. We also adapted code from another API: https://github.com/JarbasAl/py_edamam/blob/master/py_edamam/__init__.py, to help us send requests for recipe searches from Python. We also sourced and adapted CSS from: https://codepen.io/colorlib/pen/rxddKy to allow us to have a nicer front end and styling to go from.

3. Execution-based Functional Testing (10 points)

We were able to test for user registration by creating numerous accounts and verified that we were able to log in to each account. We tested Calorie Tracking by creating a field in the user registration which allows them to set their goal. Also, they can add the calories from a recipe in their database to their tracker to track how much they have eaten. We were able to test the Create Recipe functionality by creating multiple recipes and verifying that they were stored in the database as it was presented to the user on their homepage. We tested the Edit Recipe by verifying that we were able to change the recipe stored in the database and it showed changes on the user's homepage. We tested the Admin functionalities by testing if we were able to delete recipes from other user's accounts, show all the users accounts that were created on the admin page and have it display all the users recipes. We also tested that the user was able to delete their stored recipes by creating a button which deletes it from the database and verified that worked by showing the database and ensuring the deleted recipe was not there. We didn't test for the Create shopping list or Edit Shopping list as we ended up not completing that functionality.

4. Execution-based Non-Functional Testing (10 points)

We were able to test the security of it by creating a function to hide the entered password as well as storing the saved password as a hash in our database. This is confidential to the user and why it was a high priority for us. We tested the performance and reliability by utilizing all of the functions and seeing that the web application was able to handle all the requests and we also saw that the database was storing all the user information and recipes between shutting down the application. Data backup was tested by restarting the app and seeing that all the information is stored. We followed good coding practices by including comments in our code to increase the software quality. Our usability was tested and improved by creating CSS for our HTML pages and JavaScript to switch between forms for the login and user creation allowing for a more streamlined look. It is straightforward and easy to navigate. We tested the scalability by having numerous amounts of users and being able to log into each one. We tested accessibility by making everything easy to use as well as using large text so everyone would be able to see their information.

5. Non-Execution-based Testing (10 points)

We utilized pair programming to work out bugs and had at least two people work on each function to ensure that our code was bug free and to fix out bugs. We had group walkthroughs, so everyone was aware of the state of our project and how everything ran together.