**Grocery App**

**Requirements**

* Load existing grocery data
* Save new grocery data
* Generate a grocery list
* Provide user-friendly UI

Before we can load data, we need to be able to store it. Data will be stored in an initially unencrypted file that stores simple formatted strings. The strings will always be clean before they are written to the file, once the data has been read it will then need to be loaded into meal objects.

**UML**

**A diagram of a meal

Description automatically generated**

**Load Existing Grocery Data**

* Create a load class that extracts raw data meals from a set data file and loads it into POJOs

*\*\*Important that the file and location is never changed*

**Save New Grocery Data**

* Whenever a user is done interacting and updating/adding meals, data must be saved to the same data file that load uses

*\*\*Important that the file and location is never changed*

**Generate a User List**

* When a user requests, the program should generate X number of meals in a list and all necessary ingredients for the meals in a separate list.
  + If the ingredient list contains more than one instance of an ingredient, the list should not contain duplicates but display a number next to it to indicate how many. For example, two cheese would be “cheese (2).”
  + The program should keep track of the last data a meal was used, if the meal was used in the past 2 weeks, it should search for an option that has not yet been used.
  + The program should allow a user to select a meal they do not like, and regenerate only that meal.

**Provide a user-friendly UI**

* This application is not meant for technical users, but rather users that have little to no knowledge. It should provide an interactive and intuitive element for each requirement.

**Development Strategy**

* Agile

**Development Plan**

**Backend (Model)**

* Write each entity class and test
* Write loader
  + Create a sample test file and test loader
* Write Save
  + Test Save
* Create a run class that handles the program logic and combines functionalities of entity, loader, save, and generate list.
  + Test all methods except generate list
* Write Generate List class and methods
  + Test Generate
* Test the entire backend v.0.0.1

**Frontend (View)**

* Create a visual drawing of what the website should look like
* Implement all design and ensure that navigation, descriptions, and errors are in place

**Connector (View-Model)**

* Connect View and Model and ensure all error checking is completed
* Test program