Test Coverage Plan

The Test Coverage Plan is designed to prescribe the scope, approach, and high-level overview of all testing activities of the FoodPants project. The plan identifies the items to be tested, the features to be tested, and the types of testing to be performed. Unit testing will be performed using JUnit. Integration testing will be performed by analyzing the UI to ensure that the interfaces among the subsystems operate correctly. System testing will be performed by analyzing the UI to make sure all requirements are met for the user.

(1) PANTRY USE CASES (Scope: Pantry system):

- 1.1 Manage pantry (add/edit/remove items manually)
    - Test successful adding of food item (Expected Result: food item should appear in pantry with correct quantity and name)
    - Test successful editing of food item details (Expected Result: food item name or quantity should appear updated to match new entered details)
    - Test successful removal of food items (Expected Result: food item should no longer appear in pantry)
    - Test adding of food item with missing details (Expected Result: program should wait until user enters all missing details or exits without adding)
    - Test adding of food item that already exists (Expected Result: program should add new quantity to already existing quantity of food item that is already in the pantry)
    - Test adding of food item that has new food type (Expected Result: program should prompt use to add a new food type)
    - Test editing of food item to match name of another food item that exists in the pantry (Expected Result: food item is removed from pantry and quantity of removed food item is added to other food item)
- 1.2 Search pantry (by name of food type)
    - Test searching pantry for specific food item (Expected Result: food item with name containing user’s query should appear in results)
    - Test searching pantry for query that does not match any food item (Expected Result: program should tell user no match exists)
    - Test searching pantry with query that matches multiple food items (Expected Result: all food items with name containing user’s query should appear in results)
- 1.3 Consume specific item (add to nutrition log)
    - Test successful consumption of 1 of food item in pantry (Expected Result: food item should be updated with decremented quantity)
    - Test successful consumption of more than 1 of a food item in pantry (Expected Result: food item should be updated with previous quantity – amount consumed)
    - Test consumption of quantity amount of food item in pantry (Expected Result: quantity left is 0 meaning food item should be removed from pantry)
    - Test consumption of more than quantity amount of food item in pantry (Expected Result: negative quantity left meaning food item should be removed from pantry)

(2) NUTRITION LOG USE CASES (Scope: Nutrition Log system):

- 2.1 Manage nutrition log directly
    - Test successful adding of item to nutrition log (Expected Result: new item appears in nutrition log)
    - Test successful editing of nutrition item to nutrition log (Expected Result: details of item in nutrition log are updated matching the edits that were made)

    - Test successful removal of nutrition item to nutrition log (Expected Result: item no longer appears in nutrition log)
    - Test adding of item to nutrition log with duplicate details (Expected Result: new item should not be added to nutrition log and existing nutrition item should be updated with nutritional info of original item + nutritional info of new item)
- 2.2 Set nutritional goals
    - Test successful adding of a nutrition goal (Expected Result: goal should appear in nutrition goals as well as percentage of progress towards the goal)
    - Test successful editing of a nutrition goal (Expected Result: goal should appear with updated details)
    - Test successful removal of a nutrition goal (Expected Result: goal should no longer appear in nutrition goals)
    - Test adding a duplicate nutrition goal (Expected Result: goal should not be added and user should be warned that the goal already exists and they can update the existing goal if they wish)
    - Test editing nutrition goal to match another nutrition goal (Expected Result: goal should not be updated and user should be warned that another goal already exists with the same details)
- 2.3 View nutrition report
    - Test successful viewing of nutrition report (Expected Result: all nutrition log items appear in report)
    - Test viewing of nutrition report with no nutrition logs (Expected Result: user is told there is no nutrition log items to report)

(3) RECIPE USE CASES (Scope: Recipe system):
- 3.1 Create
    - Test successful creation of recipe (Expected Result: recipe appears in list of recipes with correct details)
    - Test creation of recipe with duplicate name (Expected Result: recipe should not be added and user should be warned that recipe with same name already exists but they can rename one)
    - Test creation of recipe with new food types (Expected Result: user should be prompted to add new food types and then recipe will appear in list of recipes with correct details)
- 3.2 View (all/one) +Search
    - Test successful viewing of recipes (Expected Result: all existing recipes are visible to user)
    - Test viewing of recipes when none exist (Expected Result: user should be told there are no existing recipes)
    - Test successful search of an existing recipe by its name (Expected Result: recipe with name containing query should be displayed to user)
    - Test search for a recipe that does not exist (Expected Result: user should be told there are no matching recipes)
    - Test search with a keyword contained in multiple recipe names (Expected Result: all recipes with name containing query should be displayed to user)
- 3.3 Add items from recipe to shopping list (optional add even if already in pantry)
    - Test successful adding of all items from recipe to shopping list (Expected Result: all ingredients for recipe should appear on shopping list)
    - Test successful adding of items in recipe not in pantry to shopping list (Expected Result: all ingredients not already in the pantry with sufficient quantity for the recipe should appear in the shopping list)
    - Test adding of all items to shopping list when there are existing shopping list items (Expected Result: recipe items are appended to shopping list)

- Test adding of all items to shopping list when there are existing shopping list items for ingredients in the recipe (Expected Result: recipe items are appended to shopping list and duplicate items are merged with two quantities added)
- Test adding of items not in pantry to shopping list when there are existing shopping list items (Expected Result: recipe items are appended to shopping list)
- Test adding of items not in pantry to shopping list when there are existing shopping list items for ingredients in the recipe (Expected Result: recipe items are appended to shopping list and duplicate items are merged with two quantities added)

- 3.4 Edit (“Delete” is a button within the “Edit” menu)
    - Test successful editing of recipe (Expected Result: recipe is updated with changed details)
    - Test successful removal of recipe (Expected Result: recipe no longer appears in list of recipes)
    - Test editing of recipe to match another recipe’s name (Expected Result: recipe is not updated and user is warned that they cannot change one recipe to have the same name as another)
- 3.5 Get recommended (recommend food that you can make using stuff in your pantry)
    - Test successful retrieval of recommended recipes (Expected Result: random and unique recommended recipes are visible to the user)
    - Test retrieval of recommended recipes when no recipes exist (Expected Result: no recommended recipes are returned or visible)
    - Test retrieval of recommended recipes when only one recipe exists (Expected Result: only one recommended recipe is returned and visible as opposed to seeing multiple of the same recipe as recommended)
- 3.6 Cook/consume (ask user how much they ate, how much leftover → leftovers to pantry)
    - Test successful cooking of recipe (Expected Result: quantity of items in pantry is updated to reflect how much of each item a user consumed and leftover items are added to pantry)
    - Test cooking of recipe with leftovers of new food type (Expect Result: user is prompted to add new food type and leftovers are added to pantry)
    - Testing cooking of recipe that requires a greater quantity of ingredients than is available in pantry (Expected Results: ingredients all used up are removed from the pantry)
    - Testing cooking of recipe that requires the exact amount of ingredients that are available in the pantry (Expected Results: ingredients with 0 quantity remaining in pantry are removed)

(4) SHOPPING LIST USE CASES (Scope: Shopping List system):

- 4.1 Manage list (add/edit/remove items manually)
    - Test successful adding of item to shopping list (Expected Result: new item appears in shopping list)
    - Test successful editing of item in shopping list (Expected Result: item in shopping list is updated with changed values)
    - Test successful removal of item in shopping list (Expected Result: item no longer appears in shopping list)
    - Test adding of item that already exists in shopping list to list (Expected Result: duplicated item is not added to shopping list and existing item’s quantity to buy is updated by adding to it duplicate item’s quantity to buy)
    - Test adding of new food to shopping list (Expected Result: user is prompted to add new food type, then item appears in shopping list)
    - Test editing item in shopping list to match another item (Expected Result: current item is removed from shopping list and other item’s quantity to buy is updated by adding to it current item’s quantity to buy)

- 4.2 Export (Print/PDF)

  - Test successful exporting of shopping list to pdf (Expected Result: pdf is created with all of shopping list details)
  - Test export of empty shopping list to pdf (Expected Result: no pdf is created and user is told shopping list is empty)
  - Test successful exporting of shopping list to txt (Expected Result: txt is created with all of shopping list details)
  - Test export of empty shopping list to txt (Expected Result: no txt is created and user is told shopping list is empty)
  - Test successful exporting of shopping list to csv (Expected Result: csv is created with all of shopping list details)
  - Test export of empty shopping list to csv (Expected Result: no csv is created and user is told shopping list is empty)
  - Test export of shopping list to an invalid format (Expected Result: no file is created is created and user is told export format is invalid)
- 4.3 Mark item(s) as purchased (individual items one by one or Extension: all at once)
  - Test successful marking of one item as purchased (Expected Result: selected item is visibly checked off as purchased)
  - Test successful marking of all items as purchased (Expected Result: all items are visibly checked off as purchased)
  - Test marking of all items as purchased when an item is already marked as purchased (Expected Result: all items are visibly checked off as purchased)
  - Test marking all items as purchased when only one item exists in a list (Expected Result: single item in shopping list is visibly checked off as purchased)
  - Test marking all items as purchased when all items in shopping list are already marked as purchased (Expected Result: no visible change)
  - Test marking all items as purchased when no items exist in shopping list (Expected Result: no visible change)

(5) STARTUP USE CASES (Scope: Startup):
- 5.1 Startup (Level: Summary, extends several)
  - Test successful registration (Expected Result: user instance is created, user details are logged, startup tutorial is started)
  - Test registration with already existing name (Expected Result: user is told they must use a unique name)

(6) FOOD TYPE DB USE CASES (Scope: Food Type system):
- 6.1 Manage list of food types from within Food Type DB Page
  - Test successful editing of food type (Expected Result: food type details are updated in database)
  - Test successful removal of food type (Expected Result: food type is removed from database)
  - Test editing of food type to match another food type's name (Expected Result: food type is not updated and user is told that there is another food type with the same name already)
- 6.2 Create new food type (extends "Manage list of food types") (Level: Subfunction)
  - Test successful creation of new food type (Expected Result: food type and correct details are added to database)
  - Test creation of already existing food type with same name (Expected Result: food type is not added and user is told they have a food type with the same name already)
  - Test creation of food type with same name but different case (Expected Result: food type is not added and user is told they have a food type with the same name already)