Decoders

Test Plan - Project 1

Keval Varia(Product Owner)

Nick Sidaris(Scrum Master)

Dustin Vuong

Austin Madrano

Adam Weesner

* Purpose
  + To ensure that all parts of the program are working as intended both independently and with conjunction to other parts of the project.
* Strategy
  + Test with expected cases, test with unexpected cases to test contingency handling, test different parts of the program that might affect other parts of the program to ensure all parts of the program are working in unison correctly.
* Assumptions
  + The part of the program that is being tested has gui implemented correctly
  + All initial data(restaurants their menus) is in the database
  + The database in the correct folder with the correct name
  + The program is connecting successfully with the database
* Scope
  + Test the all the admin functionality
    - Admins can add new restaurants via user input or the recommended ones(this can only done if the initial 10 are in the database)
    - Admins can add or remove menu items for a certain restaurant
    - Admins can change the price of an item for a certain restaurant
  + Test all the saddleback student functionality
    - Test all 3 trips
      * Visit all restaurants in most efficient order, start with the restaurant closest to saddleback
      * Pick starting restaurant and total number of restaurants to visit(including the starting one) and visit in most efficient order
      * Pick all restaurants and visit in most efficient order, starting at the restaurant closest to Saddleback
* Testing types
  + The majority of the testing will be done since we can use the Gui to verify most of the testing. The only exceptions are things like the adding new restaurants, which will require some white box testing. The reason for this is we have to manually look in the database for certain data that is not displayed to the Gui.

1. Administrative functions on their own
   1. Administrators can add new restaurants by user input
      1. Description : Administrators can add new Restaurants by user input. They will need to enter restaurant's name, its distance to saddleback, then input its distance to all previous restaurants in the database. This test will involve both white and black box testing, as the new restaurant will immediately appear in the table in the admin tab(as long as the addition is successful). But the data base must be opened in a separate program to verify that all of the distances were updated correctly.
      2. Cases:
         1. Tests with a new restaurant name
         2. Test with blank restaurant name
         3. Test with a duplicate restaurant name
      3. Expected results/process:
         1. The administrator will login, and navigate to the restaurants tab. They use the gui elements to input the name, distance to saddleback and the new restaurants distance to all previous restaurants in the database. The button to add the restaurant will appear.
            1. Tests with a new restaurant name: A notification will appear that the addition was successful and refresh the table. The database will be updated with the latest info
            2. Test with blank restaurant name: A notification will appear asking the user to fill all fields
            3. Test with duplicate restaurant name: A notification will appear that the admin entered a restaurant is already in the database. The database has a unique constraint on the restaurant name
   2. Administrators can add new restaurants from the given input file of additional restaurants
      1. Description: Administrators have the ability to add the recommended restaurants(Sonic, Pizza Hut) to the database along with their corresponding menu items with the push of a button. The distances to other restaurants will be updated as well. This is only to be done if these restaurants do not exist in the database and the data base only consists of the initial 10 restaurants. This test will also include both black and white box testing since we must examine the data base to ensure the
      2. Cases/Tests:
         1. Attempt to add when we know those restaurants aren’t there and only the initial restaurants are in the database
         2. Add when we know they aren’t there or more than 10 restaurants are in the database
      3. Expected Results
         1. The administrator logs in and navigate to the restaurants tab and clicks the “add recommended restaurants” button
            1. A notification will appear telling the administrator that the addition was successful
            2. A notification will appear telling the user that the operation could not be done/ has already been completed(These changes are persistent so closing the program doesn’t reset the database)
   3. Administrators can add and remove menu items from a specific restaurant
      1. Description: Administrators can add a new menu item with the gui elements. They must enter the item’s name and price. They can also select an item and click remove to delete it from the restaurant and the database. Once a restaurant has 8 items, no more items can be added to that restaurant until there is a deletion. Having two items with the same name is not allowed. This test is all black box testing since we can use the gui to verify the success of the operation(s)
      2. Cases/Tests:
         1. Add a menu item on a restaurant with less than 8 items it it’s menu and not the same name as a previous item
         2. Add a menu item with the same name as a previous item
         3. Add a menu item with a blank name
         4. Add menu items and get a restaurant up to 8 items and try to add more
         5. Remove a menu item
      3. Expected results/Process
         1. Administrator logs in and navigates to the menu tab, selects the restaurant they want to work with and goes to the add item section, there they must fill out the item name and price, the item will be added to whatever restaurant is selected in the combo box in the top left of the menu tab.
            1. A notification will appear saying that the item addition was successful and the menu table will refresh to add the new item
            2. A notification will appear saying that the item being added already exists
            3. A notification will appear and ask the admin to fill all fields
            4. As long as the new items don’t match a previous item name for a particular restaurant. Once the item count for a restaurant is 8, the add button will be hidden and replaced with text saying that the limit has been reached, removing items will re show the button.
         2. Administrator logs in, goes to the menu tab and selects a restaurant to work with. They then select the item they want to remove with the item combo box. Then click the remove button
            1. A notification will appear saying the item was removed, and the table will refresh and the item that was deleted will not be there.
   4. Administrators can update a item’s price
      1. Description: Admins can select an item and update it’s price. This will be black box testing since we can use the Gui to verify the success of the operation
      2. Cases/Tests
         1. Remove an item from a restaurant with less than 8 items
         2. Remove an item from a restaurant with 8 items
      3. Expected Results/ Process
         1. Administrator logs in, goes to the menu tab and selects a restaurant to work with. They then select the item they want to update with the items combo box. They type the new price into the price spinbox then click update
            1. A notification will appear showing the operation was a successful. The table will refresh and show the new price for whatever item was updated
            2. A notification will appear saying that the remove was successful, the button to add items will reappear
2. Saddleback student functions on their own
   1. Test going on a trip(regardless of what kind of trip).
      1. Description: A saddleback student can go on a trip to different restaurants and see their menu items. Ie travel from restaurant to restaurant then be take to a review tab after the trip is complete. This test will be black box testing since we can use the gui to verify that the program is working correctly. This test also does not account for the different trips, just general going on a trip
      2. Cases/Tests (n/a for this test)
      3. Expected Results/ process
         1. A saddleback student starts the program and goes on a trip(doesn’t matter which trip for this test)
            1. The user will be taken to the trips tab once they start a trip. They can see the menu for the restaurant they are at(buy items comes in a later test). Once they visit all restaurants they will be taken to the review tab, there they will see how much they spent in total on the trip, and how much they spent at each restaurant for the trip. The current receipt is updated after each successful item purchase and the grand total will update each time the saddleback student moves to the next restaurant
   2. Test buying item while on trip
      1. Saddleback student can buy an item by selecting an item in the combobox, or by clicking on an item in the menu table. They then select the quantity they want then hit buy.
      2. Tests/Cases:
         1. Quantity is 1 or greater
         2. Quantity is 0
      3. Expected Results
         1. The receipt table will update and add the purchase on, the current receipt will update as well. (Grand total will update once they leave the restaurant)
         2. Message will display asking the saddleback student to buy at least one item.
   3. Test default trip
      1. Description: Saddleback student will visit all restaurants in the most efficient order. The student will start the program and click the default trip button. This will show the start button, clicking it will begin the trip. As with previous case the user will visit each restaurant and be taken to the review tab once complete. This time the restaurants will be visited in the most efficient order
      2. Tests/Cases(n/a this is dependent on the data, the result should always be in efficient order regardless)
      3. Expected Results
         1. The trip will begin with the restaurant closest to saddleback then visit the rest in the most efficient order.
   4. Test Custom Trip 1
      1. Description: This trip involves picking a starting restaurant and then the total number of restaurants you want to visit on the trip(this includes the starting one picked). Trip will begin the trip at the selected starting restaurant then visit the closest restaurants until the number of restaurants visited is equal to the specified value selected before the trip started.
      2. Tests/Cases
         1. Visit the max number of restaurants
         2. Visit the minimum number of restaurants
         3. Visits a number of restaurants between 1 and max
      3. Expected Results
         1. All restaurants will be visited starting with the selected first one, then the rest in the most efficient order
         2. Visit just the first restaurant selected
         3. Visit the first restaurant selected, then the next closest restaurants and stop once the number specified have been visited
   5. Test custom trip 2
      1. Description: This trip will let a student select all the restaurants they want to visit. They start the program and select custom trip 2. They then double click a restaurant to add it to the visit list they can start the trip once they have at least one restaurant in the visit list.
      2. Tests/Cases
         1. Test with only one restaurant
         2. Test with all restaurants
         3. Test with number between 1 and max number
      3. Expected Results
         1. Only the first restaurant will be visited
         2. All restaurants will be visited in most efficient order, just like the default trip
         3. The selected restaurants will be visited in the most efficient order