Task 0: Explain what you are doing/ going to accomplish

Create a receipt page that will display the total amount of food sold during the lunch time, and the price. Possibly a payment page, and finally an end of transaction success page.

Task 1: Sketch interface design

*Draft a rough design for the interface that allows the user to trigger functionality in task 1, while also annotating where the information in task 2 will be displayed. Create another sketch listing the interface widgets used to create the interface.*

It will just be a card with all the food sold and the price, and a credit card page, and a success page.

Task 2: Identify any classes required

*Explain what the class will represent, plus listing what information will be stored in the class and any functions the class will have.*

Canteen\_content.food\_sold

Canteen\_content.food\_cost

Canteen\_content.food\_cart

Task 3: Identify information to be displayed

*What information will the interface need to display to the user?*

How much of a food item had been sold, and the cost of all the prices added up. A few button to get to other pages.

Task 4: Identify user inputs

*What program functions can the user trigger through the interface?*

A submit button on the order page  
A continue button

Maybe a form?

And a back button

Task 5: Identify any constants or existing data if required

N/A

Task 6: Identify indexed data structures

Contents\_list

Task 7: Determine what calculations are necessary

*Write out the calculations the program will have to compute.*

Plus and minus

Task 8: Develop a modular structure for your program

*Describe any functions that the computer program will have, identifying any sub-functions where required.*

Receipt – this will be the receipt page that shows everything that has been sold.

Pay – This could be a credit card page

Task 9: Define the functions identified

*Describe the functions for both the main program and any classes in terms of input and/or output where required. You may choose to do this with flow charts or pseudo-code (not Python code!). Add in additional steps or explanations using sequential, conditional, iterative statements where required. Identify global and/or local variables.*

AT ROUTE (‘/receipt’)

AT VIEW (‘receipt’)

PROGRAM plus\_stock

SET data TO dict (contents\_list EQUALS contents)

RETURN data

Task 10: Address any relevant implications such as usability, functionality, legal/ethical requirements.

The user should be able to click a button to get to the receipt page, from there, there will be a continue button or a back button to continue shopping. If the continue button is pressed it will take the user to a payment page, from there they click another button and it will take them to a success page and that will be the end of the transaction.

Task 11: Document test cases for testing the program

*Document any testing that can be used to test your program. If any input is inputted using the keyboard, describe the expected input, plus any exceptional, boundary or invalid cases.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Test case** | **Expected Outcome** | **Actual Outcome** | **Fix** |
| Click the submit button | Goes to receipt page | Goes to receipt page | NONE |
| Total cost | Total cost | Broken | Created new variable total\_cart and set to 0. Used a for loops and plused the canteen\_content.food\_cart |
| Total sold | Total sold | broken | Created new variable total\_sold and set to 0. Used a for loops and plused the canteen\_content.food\_sold |

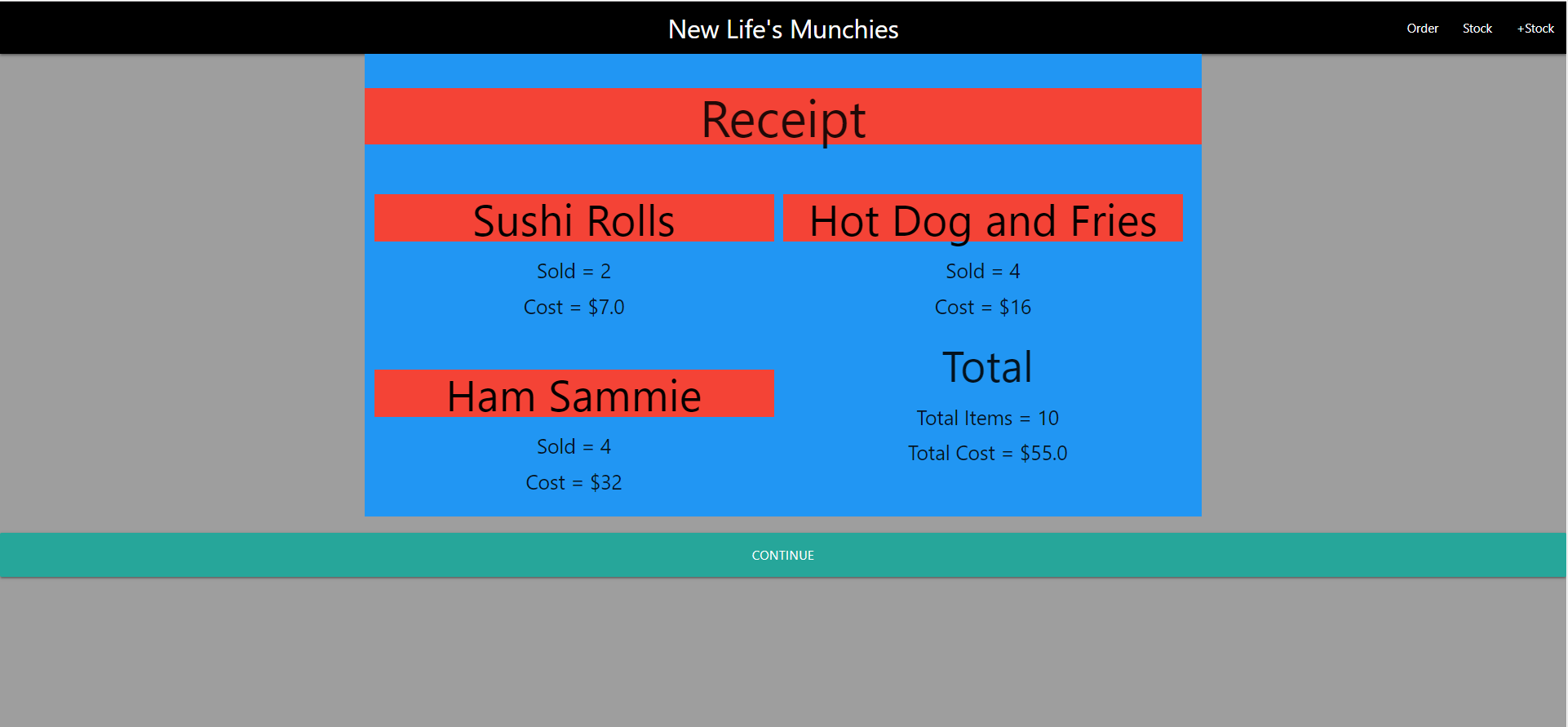
Task 12: Refine the plan

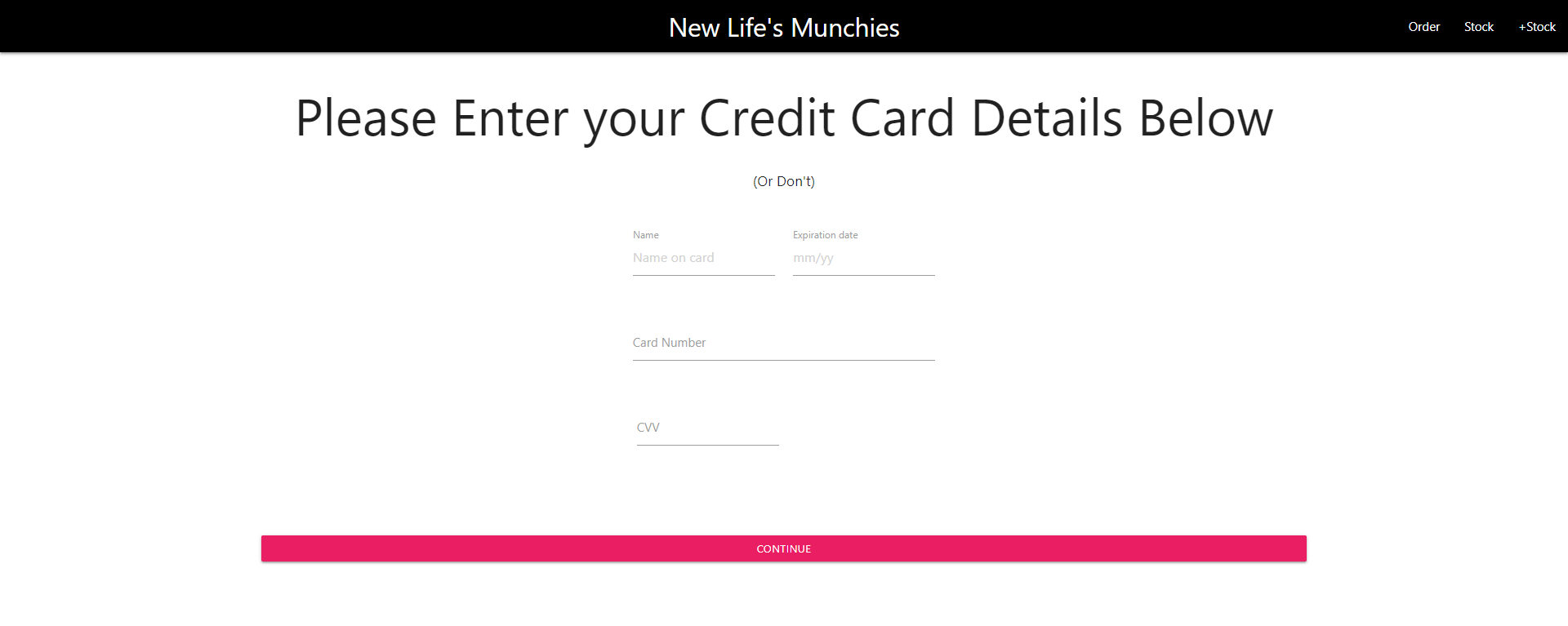
*Note any modifications here when iterating through the development cycles.*

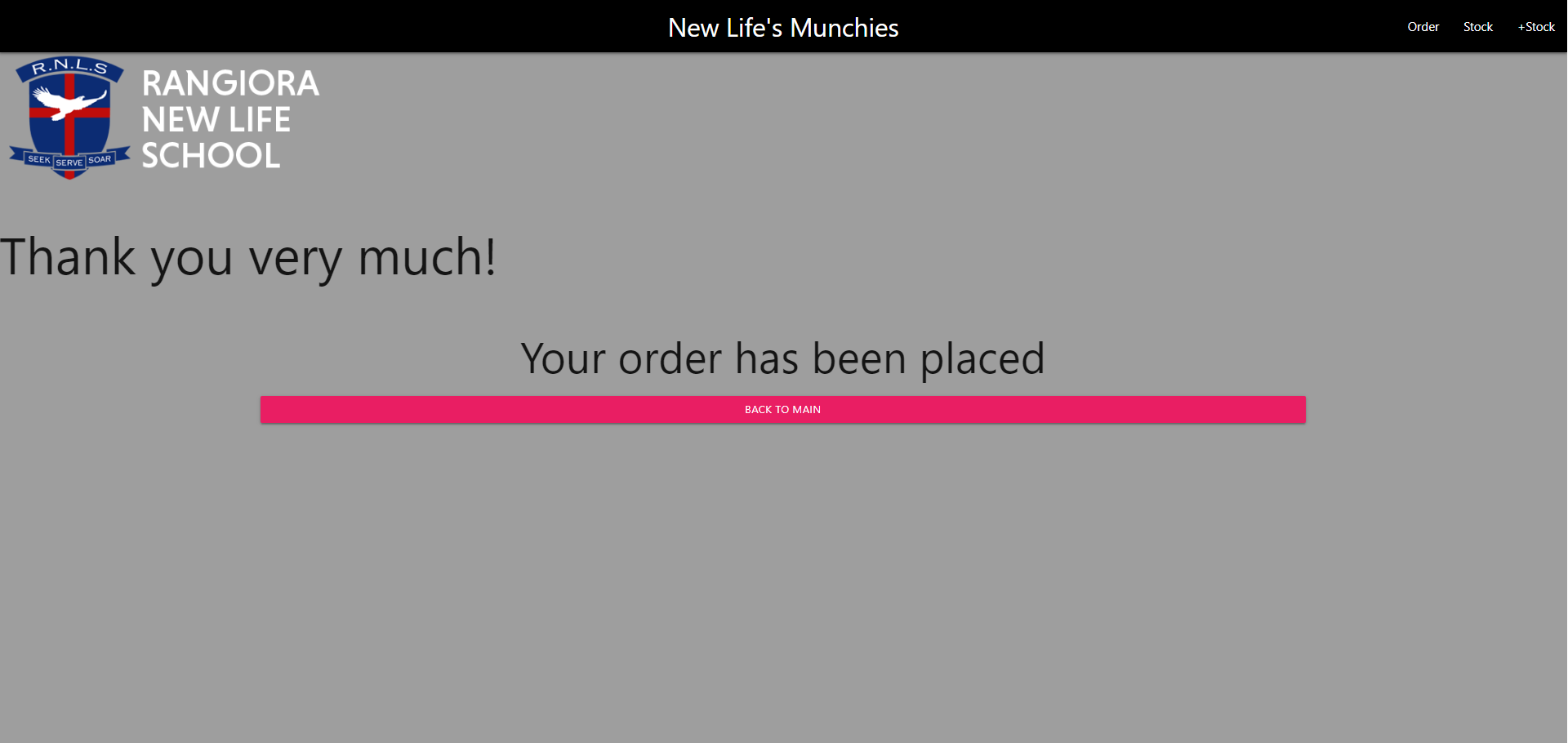
I have made it so the user cannot stock up more than 100

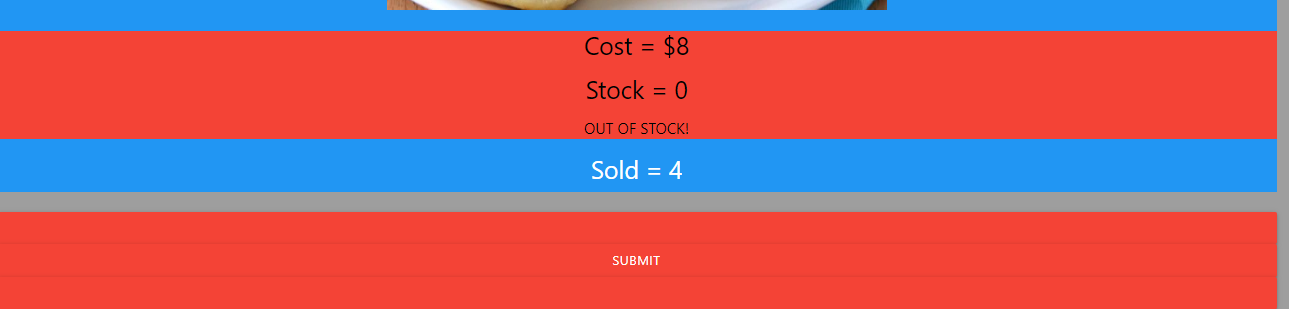
Task 13: Document testing

*Show screenshots of your program working with descriptions of each image. These images should test the tests cases listed above.*









Task 14 : Evaluation

*How did your version turn out*

This version has turned out very well, and I like it. I had problems trying to get the total cost and total sold, however this was a very simple fix. I just needed to take a step back and think. I created a new variable and did python in html.

