Ver 1.0

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

Get my python server running, including constructing my class and menu.

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.*

N/A

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.*

Will have a class that contains my menu and the variables. Will help with displaying my info to the user when linking python to HTML.

Task 3: Identify information to be displayed

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

N/A

Task 4: Identify user inputs

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

N/A

Task 5: Identify any constants or existing data if required

Test data:

* Sushi Roll pack - Starting with 5
* Hot dog and Chips - starting with 12
* Ham and Cheese Sandwiches - 4 in stock

Task 6: Identify indexed data structures

N/A

Task 7: Determine what calculations are necessary

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

N/A

Task 8: Develop a modular structure for your program

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

N.A

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.*

From bottle import run, route, view, get, post, request, static\_file

Form itertools import count

Class canteen\_food

Set \_ids to count(0)

Define \_\_initialise\_\_(self, name, image, stock, description)

Set self.id to next(self.\_ids)

Set self.food\_name to name

Set self.food\_image to image

Set self.food\_stock to stock

Set self.food\_price to price

Set self.food\_description to description

Set Canteen\_test to [

Canteen\_food(“Sushi Roll Pack”, “image”, “5”, “price” “description”)

Canteen\_food(“Hot Dog and chips”, “image”, “12”, “price” “description”)

Canteen\_food(“Ham and Cheese Sandwich”, “image”, “4”, “price” “description”)

Run(set host to “0.0.0.0”, set port to 8080, set reloader to true, set debug to true)

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

N/A

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.*

If there is no error messages then I can expect the code to be working, although I will have no way of testing this till version 2 when I create my index page.

Task 12: Refine the plan

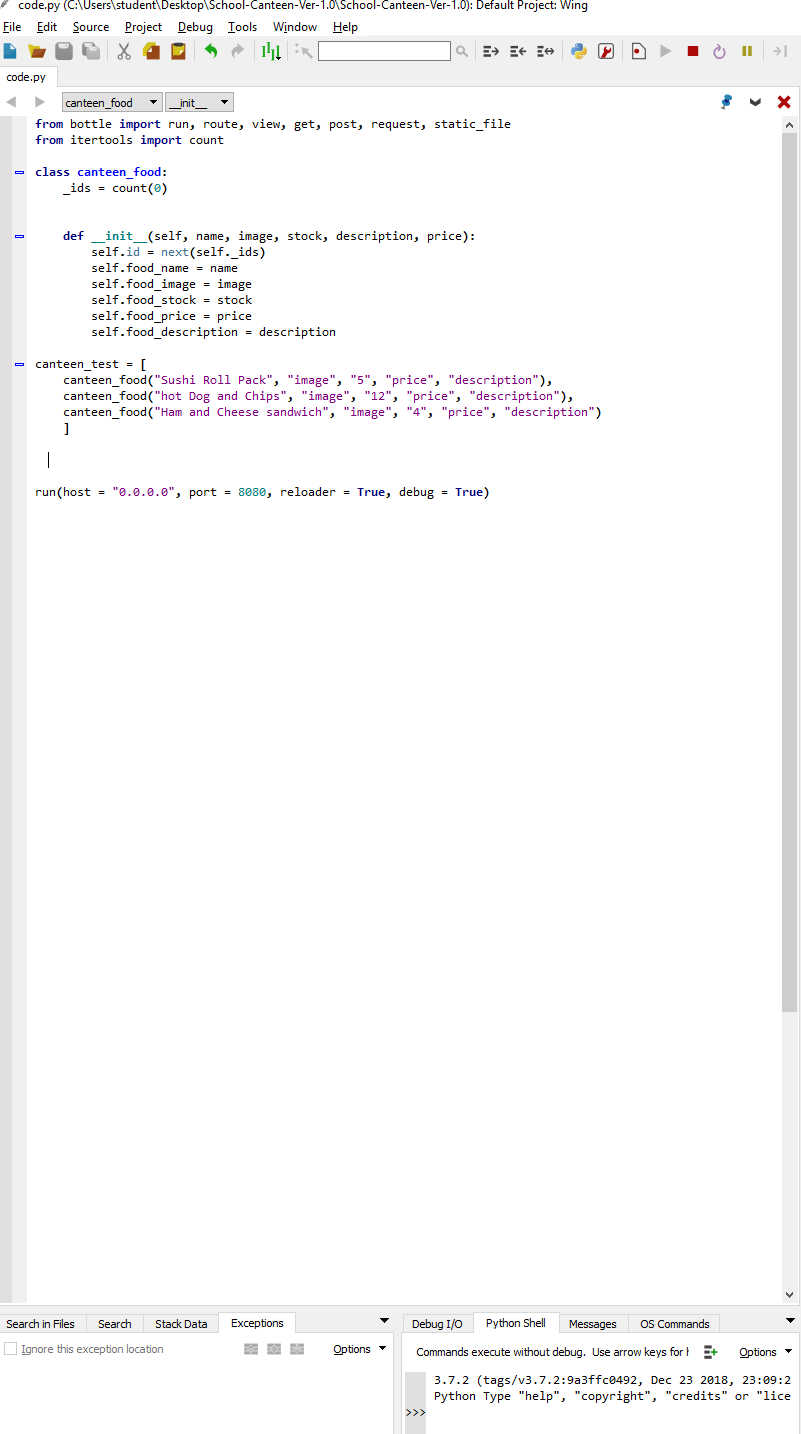
*Note any modifications here when iterating through the development cycles.*

*Needed capital T for true on the last line* Run(set host to “0.0.0.0”, set port to 8080, set reloader to true, set debug to true). Also when giving arguments to my class I forgot to include price so I was using but not giving an argument.

Task 13: Document testing

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

The below picture shows my program running without any errors. I will know just how well this works in my next version when I add my index page.



Task 14 : Evaluation

*How did your version turn out*

Good, will know the full extent of how well it works when I add my index page in the next version.

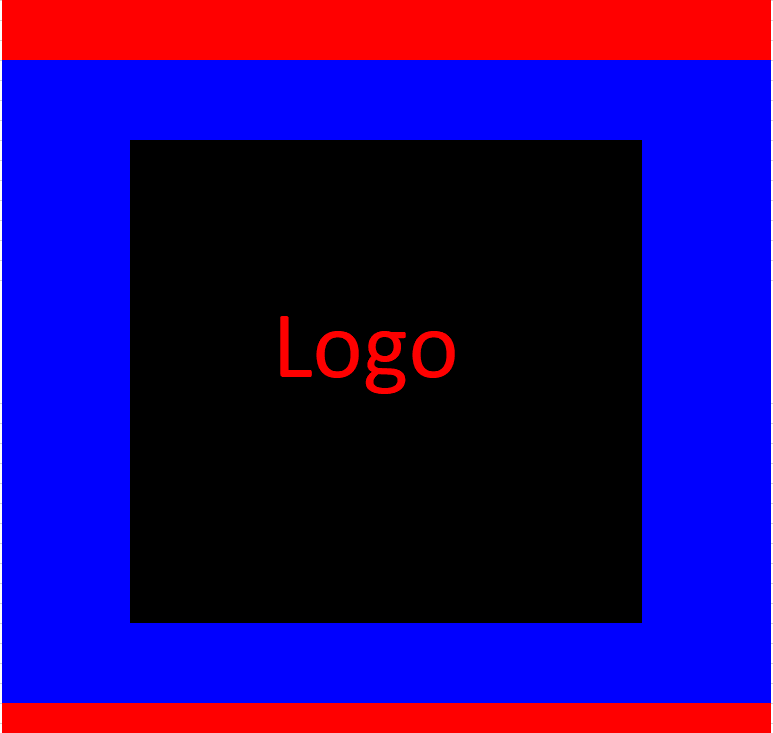
**VER 2.0**

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

To display an index page to the user, with the appropriate colours and nav bar, also footer.

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.*



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\_food, contains my python server and my test data. Will use to display webpage to the user.

Task 3: Identify information to be displayed

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

My index page, with a nav bar, footer, logo, title and the right colours.

Task 4: Identify user inputs

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

They will be able to click the links on the nav bar but they will not lead anywhere at this stage.

Task 5: Identify any constants or existing data if required

* Test data:
* Sushi Roll pack - Starting with 5
* Hot dog and Chips - starting with 12

Ham and Cheese Sandwiches - 4

Task 6: Identify indexed data structures

N/A

Task 7: Determine what calculations are necessary

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

N/A

Task 8: Develop a modular structure for your program

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

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 (“/”)

At view(“index”)

Define function index():

Pass

For index page:

% include shared / header.html

% include shares / navbar.html

% include shared / footer.html

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

The website should be easy to navigate around and have clear option paths.

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.*

Run the server and go to my web page, if it loads it works, if it doesn’t it doesn’t.

Task 12: Refine the plan

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

Forgot to add the function that allows pictures

To display images:

At route(‘/picture/<filename>’)

Define saved\_pics(filename):

Return static\_file(filename, root = ‘./images’)

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*

Very good, once I figured out why the picture didn’t load and fiddled with the red colour a little bit it came together quite nicely.

**Ver 3.0**

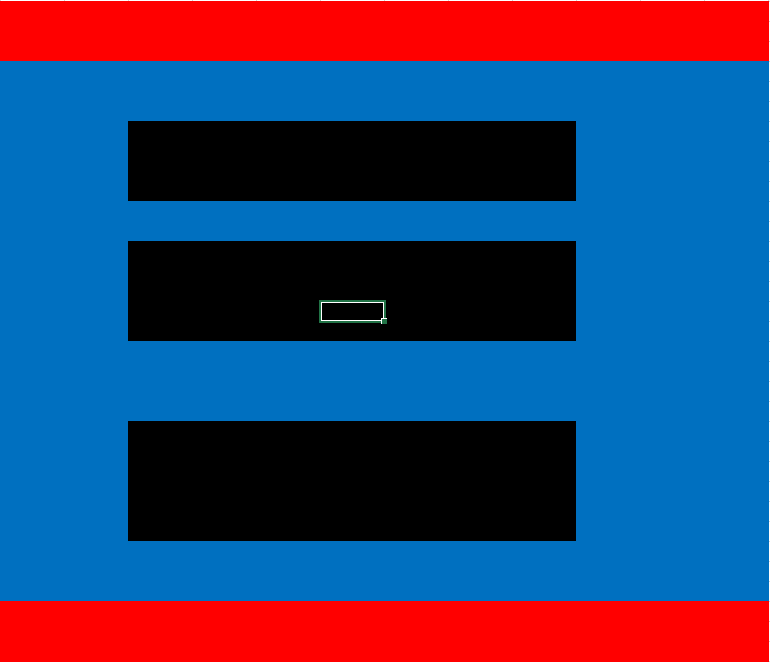
Task 0 : Explain what you are doing/ going to accomplish

Link my navbar to my other pages which will be terms and conditions as well as an order page for the food.

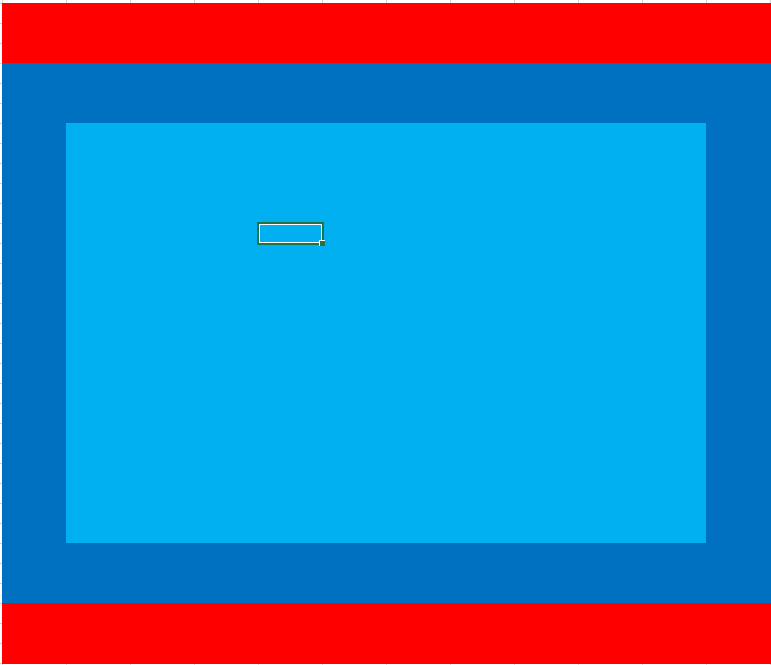
Task 1: Sketch interface design

*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.*

Menu page



Terms and conditions page, light blue represents where the writing will go.



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\_food, contains my python server and my test data. Will use to display webpage to the user.

Task 3: Identify information to be displayed

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

The index page which was created in the last version. The links on the navbar will have to be easily accessible and easily seen. The links will go to pages which are displayed nicely and the terms and conditions will be easily deciphered.

Task 4: Identify user inputs

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

They will be able to navigate around my website using the nav bar.

Task 5: Identify any constants or existing data if required

* Test data:
* Sushi Roll pack - Starting with 5
* Hot dog and Chips - starting with 12

Ham and Cheese Sandwiches - 4

Task 6: Identify indexed data structures

N/A

Task 7: Determine what calculations are necessary

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

N/A

Task 8: Develop a modular structure for your program

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

Has the initialize function

The function for each of my web pages which attatches the decorators and means I can do other stuff inside my webpages.

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 (‘/food’)

At view (“food”)

Define menu\_page():

Set data to dict (food\_list = canteen\_test)

Return data

At route (“/tanc”)

At view (“tanc”)

Define terms\_page():

pass

python in my HTML

%include('shared/header.html')

%include('shared/navbar.html')

%include('shared/footer.html')

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

Legal – terms and conditions need to be good, easily read and legal.

Functionality – all links need to go to the right place.

Usability – website needs to be easily navigated.

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.*

The user should only be able to be able to go to different pages in my website so all links needs to link to the right place.

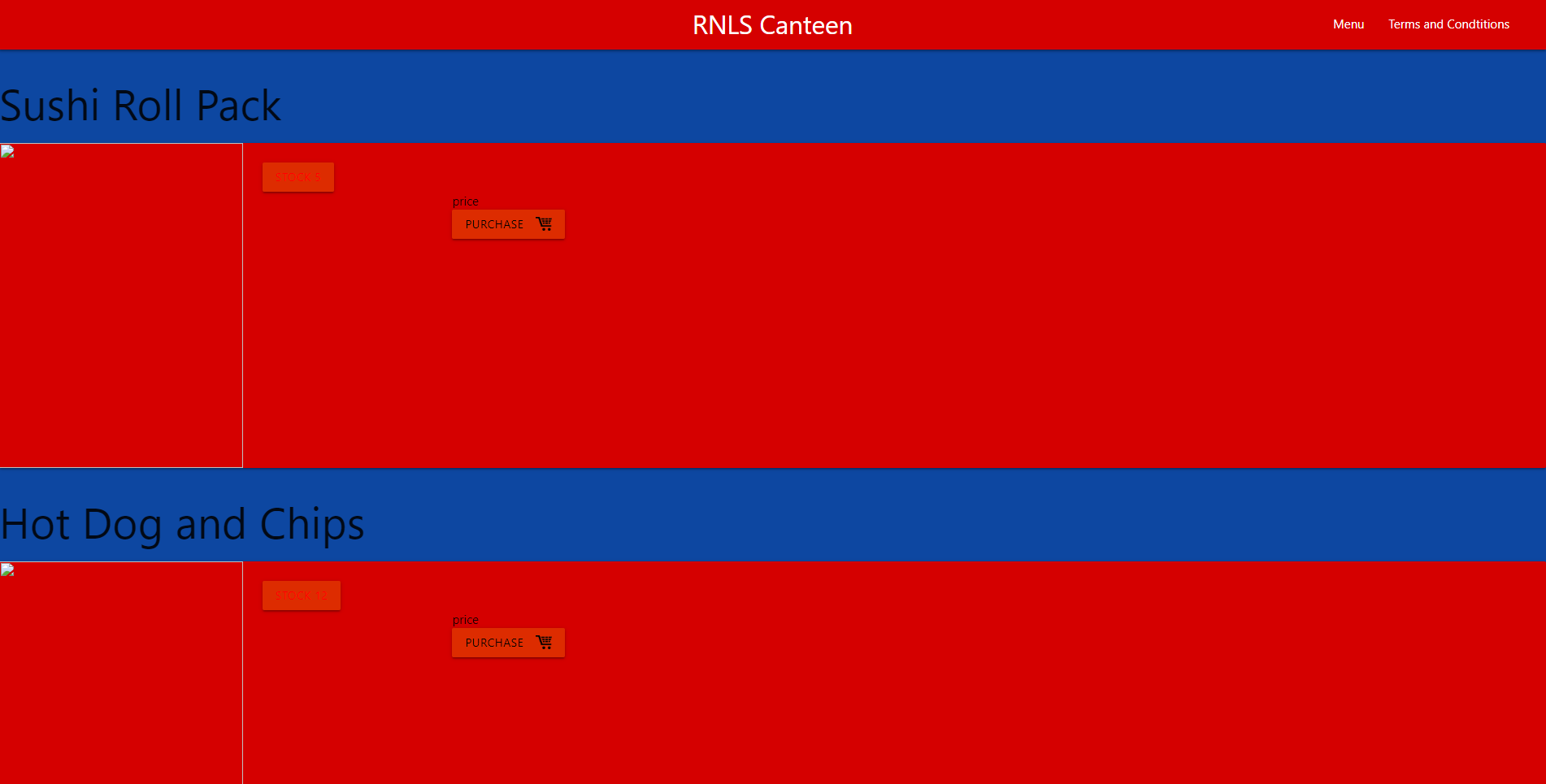
Task 12: Refine the plan

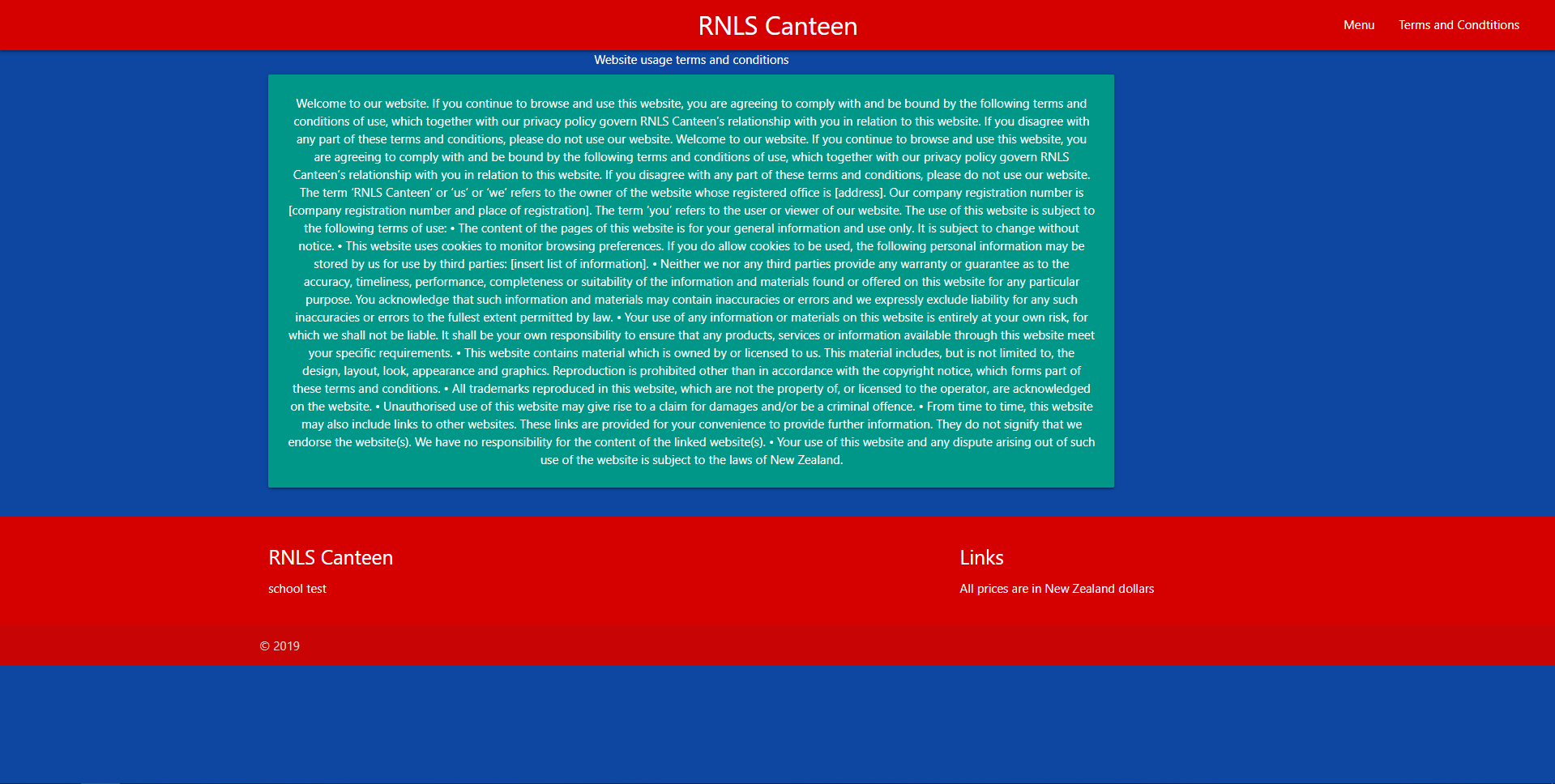
*Note any modifications here when iterating through the development cycles.*

Had to add a container around my card with the terms and conditions to center it. But didn’t work very well. Will come back and look at in a different version.

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*

Good, everything worked excepted centering the card. Will have to address in a later version.

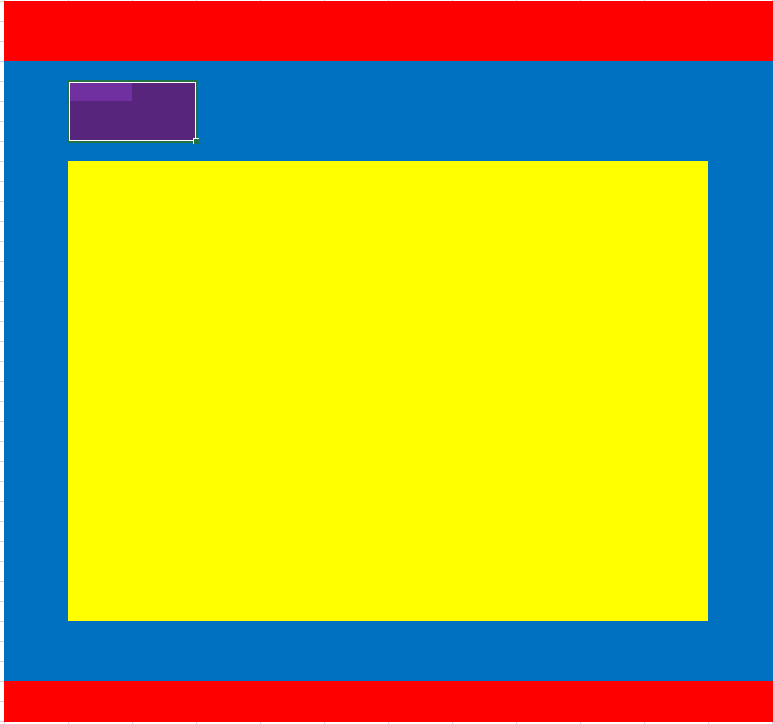
Ver 4.0

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

In this version I will be adding the pictures, description and stock to my menu. Along with a buy button that links to a success page if food is bought successfully.

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.*



Purple represents the back button and yellow represents the picture.

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\_food class contains test data.

Task 3: Identify information to be displayed

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

The success page will be crucial as they need to know what is happening.

Task 4: Identify user inputs

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

They will be able to order food.

Task 5: Identify any constants or existing data if required

Test data

* Sushi Roll pack - Starting with 5
* Hot dog and Chips - starting with 12

Ham and Cheese Sandwiches - 4

Task 6: Identify indexed data structures

Task 7: Determine what calculations are necessary

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

When I add a cart I will have to add prices to each other.

Task 8: Develop a modular structure for your program

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

Adding the cart will require its own function and the success page will need its own function also.

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.*

Set food\_id to integer (food\_id)

Set found\_food to none

For food in canteen\_test DO

If food.id is equal to food\_id DO

Set found\_food to food

Set data to dictionary(set food to found\_food)

Found\_food.food\_stock -= 1

Return data

In success page

%include('shared/header.html')

%include('shared/navbar.html')

%include('shared/footer.html')

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

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.*

Stock decreases by one each time, success page works and the purchase button disables when stock is equal to 0.

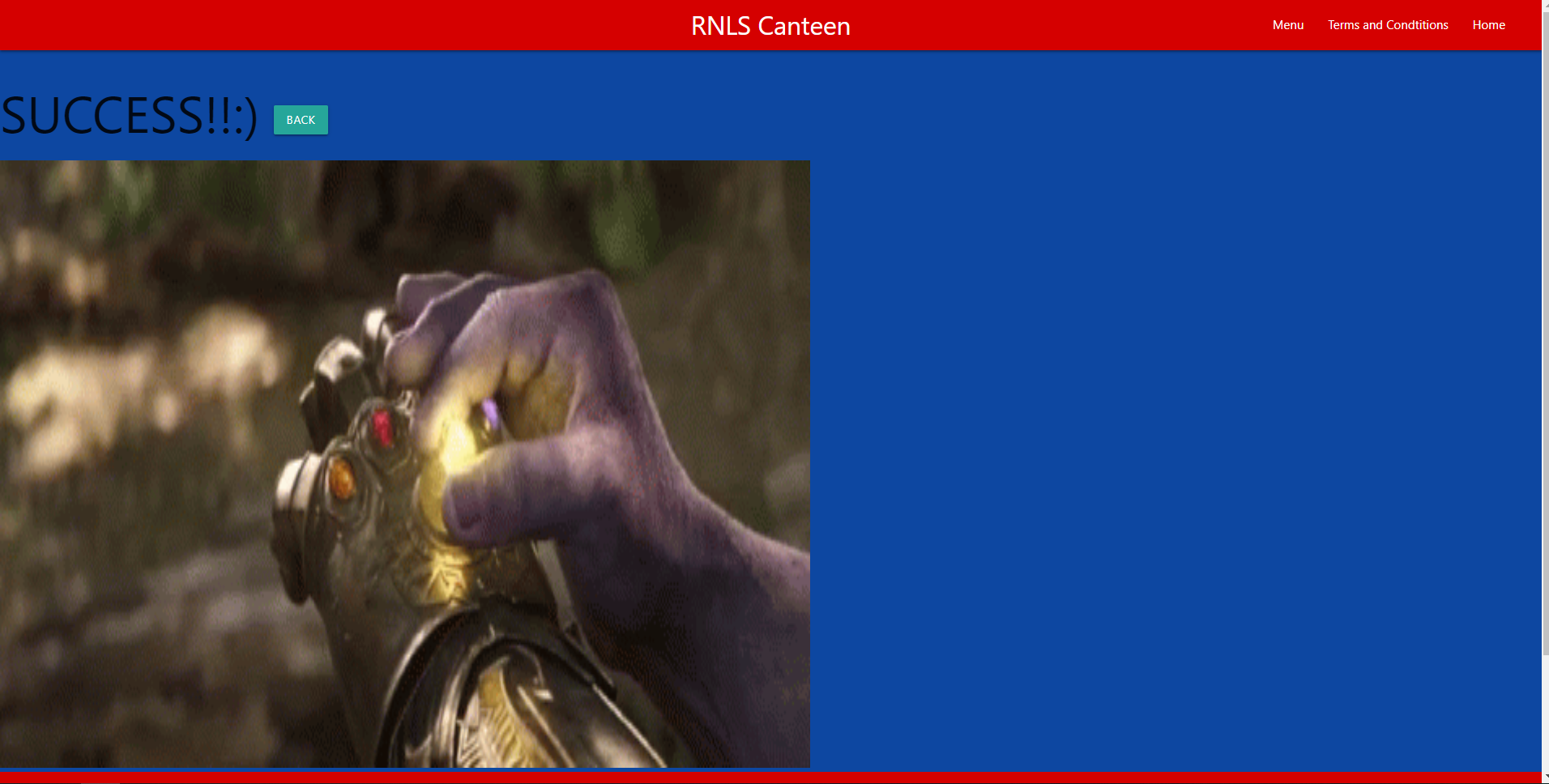
Task 12: Refine the plan

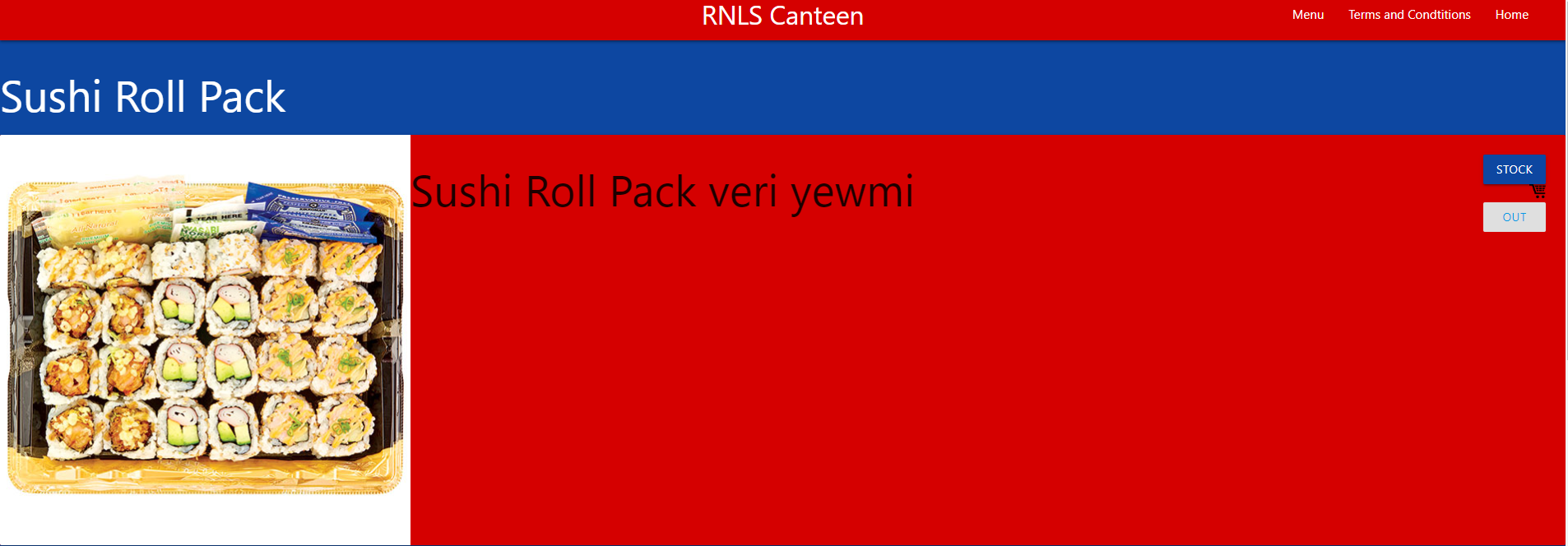
*Note any modifications here when iterating through the development cycles.*

Had an issue where my titles for the food was being linked to the success page but was just a non-closed <a> tag.

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*

Good, everything worked.

**Ver 5.0**

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

Add the restock function

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.*

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.*

Task 3: Identify information to be displayed

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

Task 4: Identify user inputs

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

Task 5: Identify any constants or existing data if required

Task 6: Identify indexed data structures

Task 7: Determine what calculations are necessary

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

Task 8: Develop a modular structure for your program

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

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.*

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

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.*

Task 12: Refine the plan

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

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*