Ver1.0 Bottle Framework

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

I am going to create the sign-up.py file that will use bottle to create the framework for my comic book store webpage.

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.*

No interface design yet, not applicable.

Created index.html page, however this will be for testing purposes with no functionality or design added yet.

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.*

ComicBook

* id
* Title
* Stock
* Price
* description

Task 3: Identify information to be displayed

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

Not applicable, this is server side

Task 4: Identify user inputs

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

Not applicable, no inputs required

Task 5: Identify any constants or existing data if required

Not applicable to this version

Task 6: Identify indexed data structures

comicBooks which will be a dictionary of test data comicbook objects to be used for the website

comicBooks = [

ComicBook(title,stock,price,description)

Repeat…

]

Task 7: Determine what calculations are necessary

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

Not applicable, no calculations necessary yet

Task 8: Develop a modular structure for your program

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

Only \_\_init\_\_ will be the initial function for the class. This creates the objects of the class

As well as def index():

Pass

This sets up the index page without passing anytihng

Def server\_static(filename)

Return static\_file(filename, root=”./assets/images”)

This allows me to link to images within the 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.*

PROGRAM START

IMPORT BOTTLE FUNCTIONS RUN,ROUTE,GET,REQUEST,VIEW,STATIC\_FILE

IMPORT ITERTOOLS FUNCTION COUNT

MAKE class called ComicBook:

Set ids\_ = count function of 0

DEFINE \_\_init\_\_ with parameters (self,title,stock,price,description):

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

Set self.title to title

Set self.stock to be stock

Set self.price to be price

Set self.description to be description

END

END

SET comicBooks to be a dictionary of:

ComicBook with parameters ( "Water Woman", 0, 19.99, "This book sucks, don't buy it." )

ComicBook with parameters ("Not Water Woman", 13, 15.99, "This book is very good thus you should buy it.")

ComicBook with parameters ("Wagon Wars", 26, 8.49, "In a world full of wagons, one wagon will destroy them all!")

END

MAKE route to ‘/image/<filename>’

DEFINE server\_static to take filename:

RETURN static\_file( filename, with root “./assets/images’)

END

MAKE route to ‘/.’

MAKE view to ‘index’

DEFINE index():

PASS

END

RUN with host as “localhost”, port equals 8080, debug equals True

END PROGRAM

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

Usability:

Add comments to make my code easier to read and understand for myself as I code as well as anyone who is assessing my code.

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 python then access localhost:8080 in browser to make sure it connects.

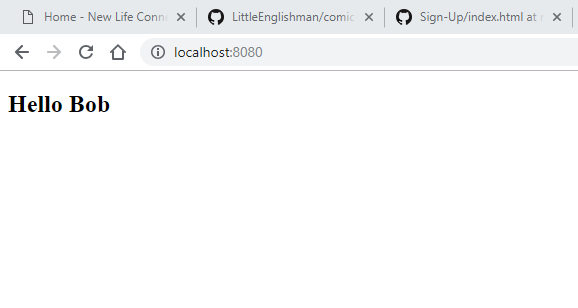
*This worked first time so no further action needed.*

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.*

**

Accessing localhost:8080 works as shown above