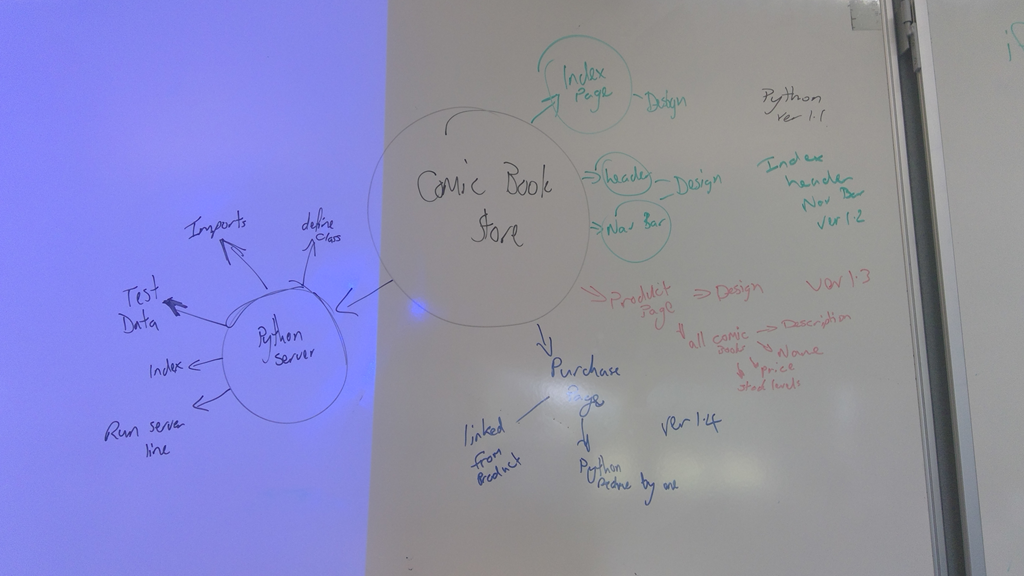
|  |
| --- |
| Learning Objective:  **To understand what we have learnt**    Success Criteria:  **Completed the Must Do & Should Do  & Could Do** |
| **Do Now - Get set up.**    Create a Github called Comic book store.      Read the following Task for this week    **Task**  Develop a comic book store program to meet the specifications below.  See Appendix 1 for planning guide.  **Program Details**  The comic book stores the following comics:  Super Dude - Starting with 8 in stock  Lizard Man - Starting with 12 in stock  Water Woman - Starting with 3 in stock  The user should be able to sell a comic one at a time, reducing the stock by one.  The interface should notify the user if the comic has been sold successfully.  The interface should notify the user with an error message if the comic has not been sold if there is not enough stock.  The interface should display:  The number of comics sold.  The current stock levels of all comics (at once). If the stock levels change at any point, the interface should update.  The user should be able to restock a chosen comic.  The user should be able to input how many copies the comic is being restocked with. For example, restock 10 copies of Super Dude at once.  There is no limit to the amount of comics the store can stock.  The program should display relevant error messages for appropriate situations.    Note: There does not need to be any functionality for the user to add a new comic book to the program. However available comic books should easily be editable by editing the program code. |



**Appendix 1: Planning Guide**

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

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.*