**CS 225 P0 Project Proposal Name: \_Aiden McCollum\_\_\_**

|  |  |
| --- | --- |
| **Provide an overall description of the project.** | |
| This project will be an inventory management system that allows a user to track items as they go through the purchasing, shipping, and organizing process. When the code starts running, the program will output a message saying “Welcome to the Inventory Management System! Select an option to get started:”. The following options will then be outputted with their corresponding numerical assignments: (1) Add an item, (2) Find an Item, (3) List all items, (4) list all products, (99) quit program. The user will then be prompted to input a whole number value to select an option. Error handling will check to ensure that the inputted value is a whole number and one of the options. If not, the program will prompt for a new response.  If the user selects option 1, the program will prompt the user to add an item. The user will prompt the user to input a name (string type), quantity (int type), cost (double type), item type (product or expense) and status of the item. The status of the item will be selected from the following options: (1) Purchased, (2) In Transit, (3) Arrived, (4) In Stock, (5) Out of stock. When the user adds a part into the system, a unique part number will be generated, stored in the database, and output a response saying “Item ####### added to tracking system”.  If the user selects the find item option, the program will output a message saying “type the part number of the item you would like to find: “. Once the user types a valid part number, the program will output all the details about the found item. It will output the data in the following format:  “Item: (name here) | Item Type: (type here) | Quantity: (num here) | Unit Cost: (num here) | Total Cost: (num here) | Status: (string here) “  If the user selects the list all items option, the same output as demonstrated above will be repeated on a new line for each item. If the user selects the list all products option, the same output will be displayed, but only for the items that are labelled as products. This will be preceded by an output saying the number of products currently in stock.  If the user selects to quit the program, it will promptly terminate execution. | |
| **Describe how the project will implement the project requirements.** | |
| **File I/O** | The .csv file will be used as a database for storing all of the information about the items tracked in the warehouse. There will be name, part number, quantity, unit cost, total cost, type, and status columns in the file. There will also be columns for the attributes of the subclasses, which is described more in depth in the inheritance section. Each row will be a unique item. As items are added to the program, the provided information will be appended to a row in the file. When the program is loaded, all the rows of information will be extracted from the file to load in all the relevant items. |
| **Exception**  **Handling** | When the user enters values for the quantity and unit cost, exception handling will be used to ensure that the value entered is a positive value of integer type (for quantity) or of double type (for cost). If the value entered is not a positive number, an NumberFormatException error will be thrown and the user will be prompted to enter a valid number again. |
| **Inheritance** | Inheritance will be used in this program to help differentiate different types of items. An abstract “Item” class will be created with the basic setters, getters, and essential methods needed for all items. A “product” subclass will be extended from the item class, which will have extra methods for indicating the sale price (bool) and supplier (String). An “expense” subclass will be extended from the item class to store items that are not sellable products, but still expenses for the business (office supplies, software costs, etc). This subclass will contain methods for determining the type of expense. |