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. Users can also assign them to custom-created projects so that they can track all the parts in that project. Additionally, users can add a Person of Contact (POC) for each project, so that employees can easily check to see which projects they are working on and the parts within that project.
. When the code starts running, the program will output a message saying “Welcome to the Inventory Management System! Would you like to: (1) Log In (2) Sign Up”. If the user selects to log in, the system will prompt the user to enter an employee ID and password. The program will use error handling to make sure that the employee ID and password are valid strings and that the employee ID and password match with the associated account. If the employee ID or password are wrong, it will throw an error saying “Username or password is incorrect” and then it’ll prompt for the ID and password again. Once it is correct, it will enter the menu described below. If the user selects to sign up, they will be prompted to input the following values: employee ID(String), Password(String), and name (String). Once valid strings have been inputted, the program will step into the menu below.

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, (5) Create a project, (6) Find project by ID, (7) Find my projects, (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), project ID(optional)(String) 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. If the user types in an ID for the project, the program will check to see if there exists a project with that ID. If so, it will set the projectID value to the user input. Otherwise, or if the user does not enter a project ID, there will be no value assigned to the projectID attribute. The item ID will also be added to the project’s data in the database. 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) “. Users will also be prompted if they would like to edit the part. If they select to edit the part, it will prompt for new values (same process as item creation). If the user doesn’t want to edit a specific field in the item edit prompt, they will be able to press enter to skip the editing.
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 create a project, the program will prompt for a projectID (int), a project name (String), projectLead (String), and a project description (String). Once completed, the project will be added to the file storage and initialized with the relevant information. The system will output a string with all the details about the newly created project.
If the user selects the find project option, the program will output a message saying "type ID of the project you want: ". Once the user types a valid project ID, the program will output all the details about the found project. It will output the data in the following format:
"Project: (name here) | ID: (ID here) | Project Lead: (employee name here) | Description (description here) Items: (list of items printed in following rows) "
If the user selects to list all their projects, the program will search the database for projects with their projectLead value equal to the user's ID. The program will then output in a similar format to that highlighted above but will output all the projects associated with the current user and wont print the items for each of those projects.

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. Additionally, there will be a .csv file used as a database for storing the employee ID, name, and password for each user. This information will be written to when users are created and read when users are trying to log in. Finally, there will be a .csv file used as a database for storing the project information. Every time a project is created, a new row will be added to the database. When items are assigned to a project, the item ID will be appended to the list of the project and write to the file. These values will be read when the user searches for projects.
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. There will also be exception handling to make sure that strings that are being parsed to numbers (for example for ID inputs) are valid and can actually be converted properly.

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.