OOP Project {All sections}

Anyone found in copying the project from any other group from any section, both the groups will get F grade in Course as well as in the Lab.

For your information projects of all sections will be evaluated by only one panel committee including Course + Lab instructors to check the quality of code and plagiarism.

Note: Carefully read the following instructions.

- 2 students are allowed per group. No cross-section is allowed.
- Submit your roll number in each submission.
- Using all concepts of OOP is mandatory. And concepts should be used correctly.
- Read the documentation and discuss it with your groupmate on zoom auto record and submit link of the video in the first submission
- Your code should be clear and add exceptions where required [to avoid system to crash or accept incorrect data/options]
- Naming Conventions should clarify the purpose of variables and functions being used.
  - Names of Classes should start with a Capital Letter.
  - Names of Variables should start from a Small Letter.
  - Identifiers should be named using camelCase.
- Project is to be submitted as a zip file.
- Use 3 file structure for implementation
- It is mandatory to keep data in files as mentioned in each section of project
- Submit the whole project by one member of the group
  - Project Deliverable 1[05/12/2021]
  - Final Product Deliverable completed and integrated with all

files[26/12/2021], files should have data to check and evaluate each module

- **Submission information**
  - **Submission will be through google classroom. Lab teachers will make portals for submission.**
  - **In the first part, you must submit code having all classes.**
  - **In the second part, you need to submit the final project code [write roll numbers of group mates in a comment at top] txt files and readme file contain the information about modules completed and the sequence to run the program.**
  - **Use windows Form to get bonus marks**

GOOD LUCK ☺

Final Project statement

Your task is to implement a complete Inventory Management System (IMS). IMS are centralized systems used by shop keepers, customers, and administrators to collect information needed to manage all core features of an inventory's operations such as add products, view products, administrative tasks. maintain customers data records, shopkeeper's data records, product records etc. Mostly, IMS contains functionalities of three major entities i.e. (Administrators, Shop keepers and customers).

Major entities along with their functionalities are described below:

Divide the modules in start with your group member and everyone is responsible to his own completed modules. Make sure you combine the project from time to time to avoid any integration issues later.

You need to design the hierarchy of classes to be defined in the system first. All the components should be implemented in OOP concepts. [constructers, Inheritance, virtual function, friend-class/functions, static data members, polymorphism, operator overloading, three files' structures etc.]

1) Administrator Module:

The main job responsibility of an administrator is to ensure the efficient performance of all people (shop keeper and customer). Following are key features for admin
Here are features (functionalities) of administrator that your project should contains:

- **Sign into system**

  Show signing page to each type of user and ask for username and password. The username and password should be unique for each type of member.
  Username and password admin/admin.

- **Add new Shop Keeper**

  To add a new shop keeper, your program should ask some credentials from the user to register a shop keeper. Credentials you must ask includes:

  S-id (Shop keeper id) should be in proper format as string to use in future modules [All the info should be taken in same order]

  - First Name
  - Last Name
  - S-id (should be auto generated by the system in increasing order) [format should be Shopkeeper e.g., 0001 for first shopkeeper and 1001 for one thousand and one shopkeeper number]
  - Username
  - Email
  - Password
  - Registration Date

- Gender
- Blood Group
- Contact No
- Address

After this, you must record all the above-mentioned data save data within a file named as **"shopkeeper.txt"** present in the same directory.

- **Add new Customer**

Similarly, to add a new customer, your program should ask some credentials from the user to register a customer. Credentials you must ask includes:

- First Name
- Last Name
- C-id (should be auto generated by the system in increasing order) [format should be Customer e.g., 0001 for first customer and 1001 for one thousand and one customer number]
- Registration Date
- Gender
- Contact No
- Blood Group
- Address
- Products selected
    - Category of products
    - Number of products
- Bill paid status (true/ false)

After this, you must record all the above-mentioned data save data within a file named as **"customer.txt"** present in the same directory.

- **Add new Product**

To add new product following steps must be full filled

- Select category
- Product Name
- Description of product
- Product Category Name
    1. Sports
    2. Garments
    3. Eatables
    4. Medicines
    5. Fashion
- P-id (should be auto generated by the system in increasing order) [format should be product from which category+std_number e.g., 010001 for sports category and 1001 for product number and 020003 for category of Garments and 0003 product number]
- Size of product(optional) e.g., Small, Medium, Large

- Color of product(optional) e.g., Blue, Brown, Black
- Product quantity (if quantity is zero show out of stock)

After this, you must record all the above-mentioned data save data within a file named as **"products.txt"** present in the same directory.

- **Edit Shopkeeper details**

Your program should be capable of editing the shopkeeper's basic details. The details that administrator can edit includes:

- First Name
- Last Name
- Address
- Contact No
- Username
- Email
- Password

The system should display a shopkeepers list and then admin enter/select shopkeeper's id number to edit and then update the above fields for that specific shopkeeper. Once the user updates the shopkeeper's record, your program should update this information against the id you have selected in file "**shopkeeper.txt**".

- **Edit Customer details**

Your program should be capable of editing the customer's basic details. The details that administrator can edit includes:

- First Name
- Last Name
- Address
- Contact No
- Username
- Email
- Password

The system should display a customer list and then admin enter/select customer's id number to edit and then update the above fields for that specific customer. Once the user updates the customer's record, your program should update this information against the id you have selected in file "**customer.txt**".

Same as shopkeeper updating records.

- **Edit Product details**

Your program should be capable of editing the product basic details. The details that administrator can edit includes:

- Name of product
- Description of product
- Number of products
- Size of product(optional) e.g Small, Medium, Large
- Color of product(optional) e.g Blue, Brown, Black

- **Delete Shopkeeper**

  The system should display a shopkeepers list and then admin enter/select shopkeeper's id number to delete that specific shopkeeper. Once the user updates the shopkeeper's record, your program should update this information against the id you have selected in file "**shopkeeper.txt**".

- **Delete Customer**

  The system should display a shopkeepers list and then admin enter/select customer's id number to delete that specific customer. Once the user updates the customer's record, your program should update this information against the id you have selected in file "**customer.txt**".

- **Delete Product**

  The system should display a products list and then admin enter/select product's id number to delete that specific product. Once the user updates the product's record, your program should update this information against the id you have selected in file "**product.txt**".

2) Shopkeeper Module:

Customer module is to implement operations regarding customer which are discussed below.

- **Register to system**

  To add a new shopkeeper, your program should ask some credentials from the user to register a shopkeeper. Credentials you must ask includes:

  S-id (Shop keeper id) should be in proper format as string to use in future modules [All the info should be taken in same order]

  - First Name
  - Last Name
  - S-id (should be auto generated by the system in increasing order) [format should be Shopkeeper e.g., 0001 for first shopkeeper and 1001 for one thousand and one shopkeeper number]
  - Username
  - Email
  - Password
  - Registration Date
  - Gender
  - Blood Group
  - Contact No
  - Address

  After this, you must record all the above-mentioned data save data within a file named as

"shopkeeper.txt" present in the same directory.

- **Sign in to system**

Show signin menu to each type of user and ask for username and password. The username and password should be unique for each type of member.
Username and passwords of all shopkeepers in readme file which should be attached to project submissions.

- **Add new Product**

To add new product following steps must be full filled

- Select category
- Product Name
- Description of product
- Product Category Name
    6. Sports
    7. Garments
    8. Eatables
    9. Medicines
    10. Fashion
- P-id (should be auto generated by the system in increasing order) [format should be product from which category+std_number e.g., 010001 for sports category and 1001 for product number and 020003 for category of Garments and 0003 product number]
- Size of product(optional) e.g., Small, Medium, Large
- Color of product(optional) e.g., Blue, Brown, Black
- Product quantity (if quantity is zero show out of stock)

After this, you must record all the above-mentioned data save data within a file named as **"products.txt"** present in the same directory.

- **Edit Product details**

Your program should be capable of editing the product basic details. The details that administrator can edit includes:

- Name of product
- Description of product
- Number of products
- Size of product(optional) e.g Small, Medium, Large
- Color of product(optional) e.g Blue, Brown, Black

- **View Products**

This feature views all the specific products added by the specific shopkeeper.

3) Customer Module:-

Customer module is to implement operations regarding customer which are discussed below.

- **Register to system**

To add a new customer, your program should ask some credentials from the user to register a customer. Credentials you must ask includes:

C-id (Customer id) should be in proper format as string to use in future modules [All the info should be taken in same order]

- First Name
- Last Name
- C-id (should be auto generated by the system in increasing order) [format should be customer e.g., 0001 for first customer and 1001 for one thousand and one customer number]
- Username
- Email
- Password
- Registration Date
- Gender
- Blood Group
- Contact No
- Address

After this, you must record all the above-mentioned data save data within a file named as **"customer.txt"** present in the same directory.

- **Sign in to system**

Show signin menu to each type of user and ask for username and password. The username and password should be unique for each type of member.

Customer will input the total money he/she right after login.
Username and passwords of all customers in readme file which should be attached to project submissions.

- **Purchase a Product**

First list of all products will be shown to customer. Then he/ she can select the product. After selecting the product now, the quantity is also shown to the customer. Stock is automatically update after selecting the product. Multiple products will be selected and add in cart of customer. Customer can also select the product size or color if available.

- **Add to cart**

  After moving to cart all products with quantity will be visible to customer. He/ She can update the quantity of product and delete the product from cart. Each product carry 2% sales tax with 200 delivery charges will be automatically added. All the products available in add to cart must be visible to customer anytime. If customer login again to system, then all add to cart products must be visible to customer.

  All add to cart products of customer will be saved in single **"addToCart.txt"** file with customer id.

- **Checkout**

  When checkout is selected first the total amount of all products and customer available amount will be check if the product amount exceeds then show message "Not enough Cash" otherwise. All products will be updated in stock and customer add following details for shipment

  - Shipment Address

  After successfully checkout show message "Thank you for shopping" and show list all products again to customer.

GOOD LUCK ☺