Assignment - Creating a Shopping App Using Python

Overview

In this project, participants will put the full spectrum of Python programming fundamentals to practice, from data preparation, conditional statements, and loops.The focus is on building the backend functionality of a simple shopping applicationfor an e-commerce website. Participants will use Python to manage login, add orupdate product categories, and manage items on user carts.

Instructions:

- Review the learning materials in Fundamentals of Python Programming
- Carefully read the situation, tasks, actions, and result sections to grasp theassignment fully
- Complete and submit your assignment via the Learning Management System(LMS)
- Follow the provided guidelines closely, ensuring your report includes allrequired analyses and interpretations

Situation:

You are a Python developer, building a shopping application or e-commerce application with login and public login features on the Python platform. The application should include categories such as footwear, clothing, and electronics. Youshould be able to add and update categories in the application. Additionally, the application must allow users to add or remove items from their cart. Finally, the program needs to support various payment options including UPI and debit cards. This is a backend implementation, and UX/UI and database connectivity are not required.

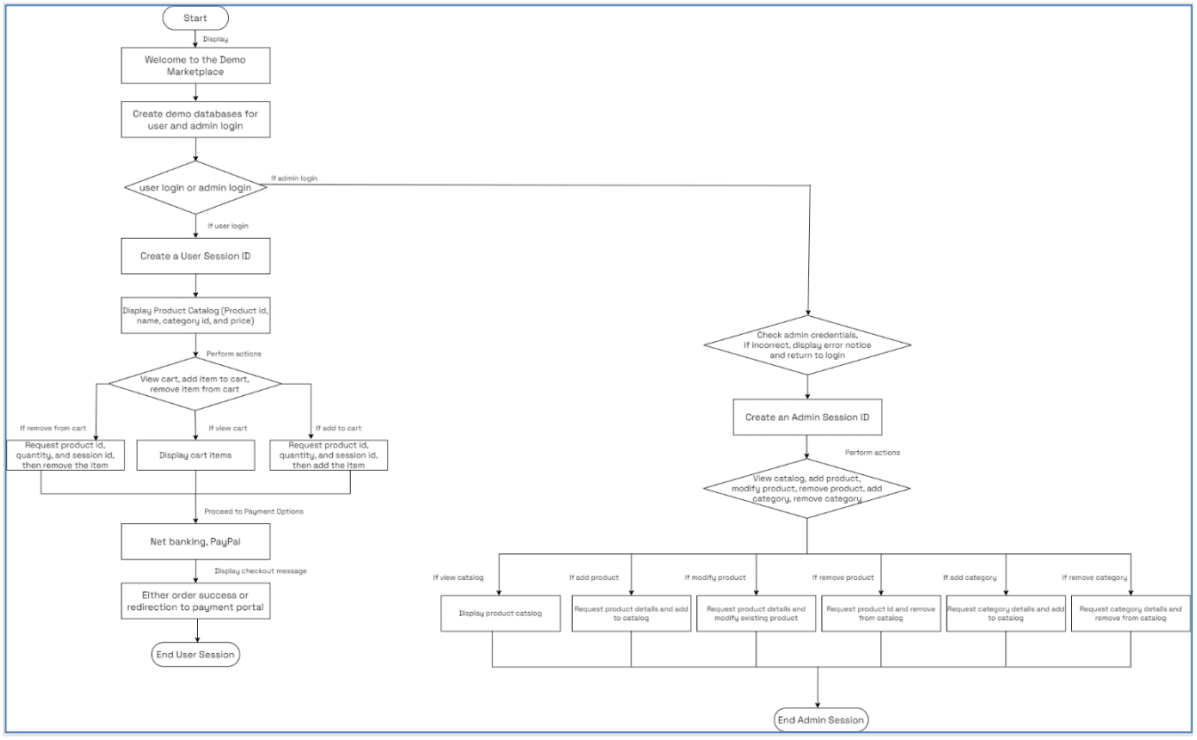
Task

- Display a welcome message in the e-commerce application, such as "Welcometo the Demo Marketplace"
- Create user login and admin login after writing the code for the welcome message. Create demo databases for “user” and “admin” verification and “session ID” creation. Once the databases are created, create admin and userlogins
- Create a sample product catalog with three to four product categories likeBoots, Coats, Jackets, and Caps. The dummy database of the catalog should contain **product ID**, **name**, **category ID**, and **price** for each item. Both usersand administrators can view the catalog
- User login rights should include the ability to view cart contents, add items tothe cart, and remove items from the cart. Users should be able to add or deleteitems in the cart using session ID, product ID, and quantity
- Provide demo payment checkout options such as **Net banking** and **PayPal**. After selecting the payment option, display a checkout message like "Your order is successfully placed" or "You will be shortly redirected to the portal to make a payment of $20".
- Only the admin can log in using their credentials. If the admin tries to log in using another set of credentials, an error notice must appear
- The admin should not be able to perform the same functions as the user. An error should appear if the admin tries to carry out those tasks
- The admin should be able to add new products to the catalog using the selected attributes. The program should also allow the admin to modify existing products using an admin session ID
- The admin should have the ability to remove any existing products from thecatalog
- Finally, the admin should be able to add a new category of product and deleteexisting categories from the catalog, considering the dynamic demands of themarket
- Users should not have access to the admin's rights as described above

Note: The focus of this implementation is on the backend functionality, and there is no requirement for UX/UI or database connectivity.

Design Flowchart

Use the following design flow chart as a reference while building your softwareapplication using Python.



Action

1. **Open the Python environment:**
   - Open the Python File and start your work in a blank code file.

2. **Print the welcome message:**

   - Write the code print("Welcome to the Demo Marketplace") in your file todisplay the welcome message

3. **Define the user and admin databases:**

   - Create two dictionaries named user_db and admin_db to store theusernames and passwords

4. **Define the user login function:**

   - Write a function called **user_login**
   - Inside the function, use the input function to get the user's username andpassword
   - Check if the entered username exists in the **user_db** and if the passwordmatches
   - Print "**Login successful as user**" or "Login failed" accordingly

5. **Define the admin login function:**

   - Write a function called admin_login similar to the user login function.
   - Get the admin's username and password
   - Verify the credentials against the **admin_db**

- Print **login status**

6. **Define the session ID generator function:**

   - Create a function named generate_session_id
   - Inside the function, simply return a fixed session ID like "12345"

7. **Add the functionality to login as a user or an admin:**

   - Use the input function to ask the user whether they want to log in as auser or admin
   - Based on the input, call either the user_login or admin_login function

8. **Create the product catalog:**

   - Define a list called catalog that contains dictionaries with details for eachproduct
   - Each dictionary should have keys like "id", "name", "category", and "price"

9. **Define the catalog display function:**

   - Write a function named display_catalog
   - Loop through the catalog list and print product details in a tabular format

10. **Create the cart management functionalities:**

    - Create an empty list named cart to represent the user's cart
    - Define functions display_cart, add_to_cart, and delete_from_cart
    - display_cart: Display the items in the cart
    - add_to_cart: Add items to the cart with session ID, product ID, andquantity
    - delete_from_cart: Remove items from the cart using session ID andproduct ID

11. **Create a user interaction loop:**

    - Start a loop to repeatedly interact with the user
    - Show options to view cart, add to cart, delete from cart, or log out
    - Use the input to get the user's choice
    - Depending on the choice, call the corresponding functions defined earlier

12. **Implement the checkout functionality:**

    - Define a function called checkout
    - Ask the user to select a payment method using the input function
    - Based on the payment method, print a message indicating successfulpayment or redirection for payment

13. **Implement the admin login verification functionality:**

    - Create a function named admin_login to verify admin credentials.
    - Check if the entered admin username exists in admin_db and if thepassword matches.
    - Implement the admin verification functionality when the user enters thelogin credentials.

14. **Define a function named admin_func:**

    - Display options for admin tasks like adding products or updatingproducts.
    - Use the input function to obtain the admin's selection.
    - Depending on the choice, call functions to perform admin actions.

- Add restrictions to ensure admin actions are not accessible to normalusers.
- In the functions related to cart management and catalog management,add conditions to check if the user type is "admin" before proceeding.

15. **Differentiate user and admin experience:**

    - Inside the user interaction loop, ask if the user is an admin or a regularuser.
    - Use conditional statements to direct the user to the appropriate set ofactions.

16. **Run the program and verify the results:**

    - Run the program by executing the file in your Python environment.
    - Follow the prompts and inputs to simulate user interactions and adminfunctions.

Result

Save and submit your solution file (.ipynb file format) in LMS.