Problem Scenario

You have to develop a shopping application or e-commerce application which has a Login platform. The applications that have been developed should also include categories, such as 3–4 for footwear, clothing, electronics, etc. It should be possible to add and update categories in the application. Additionally, it must contain a feature that allows you to add or remove items from your cart. Finally, the program needs to support a variety of payment options, including UPI and debit cards. This should be only backend implementation, and UX/UI and database connectivity are not required.

In [5]:
```
# Mocked Databases
users = [{'username': 'Admin', 'password': 'Admin@1234'}]
logged_in_users = []
categories = {'footwear': [{'product_id': 'footwear-001', 'brand': 'Crocs', 'name': 'LiteRide 360', 'type': 'Clogs', 'unit_price': 2000}
                           , {'product_id': 'footwear-002','brand': 'Nike', 'name': 'Air Jordan', 'type': 'Sneakers', 'unit_price': 4000}],
              'clothing': [{'product_id': 'clothing-001','brand': 'Lacoste', 'name': 'Sport 3D', 'type': 'T-Shirt', 'unit_price': 3500},
                           {'product_id': 'clothing-002', 'brand': 'Peter England', 'name': 'Slim Fit', 'type': 'Formal Shirt', 'unit_price': 1500}],
              'electronics': [{'product_id': 'electronics-001', 'brand': 'Apple', 'name': 'iPhone 16 Pro Max', 'type': 'Mobile', 'unit_price': 75000},
                              {'product_id': 'electronics-002', 'brand': 'Samsung', 'name': '55 OLED 4K TV', 'type': 'TV', 'unit_price': 40000}]}
user_carts = []
user_orders = []
payments_methods = ['Credit_Card', 'Debit_Card', "Cash", "UPI"]

# Helper Variables
Password_MinLen = 6
Password_MaxLen = 20
Password_Special_Chars = ['!', '@', '#', '$', '%', '&', '_', '.']
Invalid_Pwd_Msg = "Invalid password.\nPassoword format required:\n1) The password must contain atleast one upper case, one lower case, one number, and one special character from the followings: {}\n2) The minimum length of the must password mus
```

Function Definitions

User Management

Helper functions for password strength check during sign up

In [6]:
```
import string
import re

# Function for checking passoword lenth
def pwd_len_check(password):
    password_length = len(password)
    return password_length >= Password_MinLen and password_length <= Password_MaxLen

# Function for checking whether the password contains atleast one lower case, one upper case and one numric character
def pwd_has_upper_lower_number(password):
    return len(set(string.ascii_lowercase).intersection(password)) != 0 and len(set(string.ascii_uppercase).intersection(password)) != 0 and len(set(string.digits).intersection(password)) != 0

# Function to check whether the password contains any of the required special characters or not
def pwd_has_allowed_spl_chars(password):
    pwd_regex = re.compile('[!@#$%&*_.]')
    return pwd_regex.search(password) != None
```

Sign Up function definition

In [7]:
```
def create_user(username, password, password_confirmation):

    # First check whether the username provided already exists in the db or not
    for user in users:
        if(user['username'] == username):
            print('User \"{}\" already exists. Please login if already an user or try sign up with a different username.'.format(username))
            return False
    #Check whether the password and re-confirm password matches or not
    if(password != password_confirmation):
        print('Password and password reconfirmation do not match. Please try again')
        return False

    # We can do an additional check for the password strength.
    # The password must contain atleast one upper case, one lower case, one special character and one number.
    # The minimum length of the must password must be 6 characters and the maximum length can be 20 characters

    # List of validator functions previously defined for password check
    pwd_validators = [pwd_len_check, pwd_has_upper_lower_number, pwd_has_allowed_spl_chars]

    # Loop through the validator functions to check the passord strength
    for validator in pwd_validators:
        if not validator(password):
            print(Invalid_Pwd_Msg.format(' '.join(Password_Special_Chars), Password_MinLen, Password_MaxLen))
            return False

    # Finally, after all the above constraints are satisfied, add the user the users list
    users.append({'username': username, 'password': password})
    print("Congratulation, user successfully created. Please login to use the shopping application.")
    return True
```

Do Signup with user input

In [8]:
```
# Function to take user input for signup
def user_sign_up():
    username = input('Enter Username: ')
    password = input('Enter Password: ')
    password_confirmation = input('Re-confirm Password: ')
    create_user(username = username, password = password, password_confirmation = password_confirmation)
```

Login function definition

In [9]:
```
# Function for the user authentication logic
def authenticate_user(username, password):
    for user in users:
        #Check whether user is already logged in or not
        current_user = {'username': username, 'password': password}
        if(current_user in logged_in_users):
            print('{} is already logged in!'.format(current_user['username']))
            return False

        #Validate the inputs to authenticate user login
        if(user['username'] == username and user['password'] == password):
            print('Hello {}, welcome to Shopping Cart Application!'.format(username))
            logged_in_users.append(user)
            return True
    print('Access Denied! Invalid username or password')
    return False
```

Do Login with user input

In [10]:
```
# Function to take user input for user login
def user_login():
    username = input('Enter Username: ')
    password = input('Enter Password: ')
    authenticate_user(username = username, password = password)
```

Logout function definition

In [11]:
```
# Logout logic for already logged in user
def logout(username):
    logged_in_user = None
    for user in logged_in_users:
        if(user['username'] == username):
            logged_in_user = user
            break

    if(logged_in_user != None):
        logged_in_users.remove(logged_in_user)
        print('{} has been successfully logged out of the system'.format(logged_in_user['username']))
        return True
    else:
        print('{} is not yet logged in'.format(username))
        return False
```

Do Logout with user input

In [12]:
```
# Function to take user input for logout
# This function takes username of the user to be logged out
def user_logout():
    username = input('Enter username to logout: ')
    logout(username)
```

Check User Login Status

In [13]:
```
def check_user_login(username):
    is_logged_in = False
    for user in logged_in_users:
        if user['username'] == username:
            is_logged_in = True
            break
    return is_logged_in
```

Formatted Print Functions

Print All Category-wise Products

In [14]:
```
def print_all_category_products():
    for category_name in categories.keys():
        print('\nCategory - {}'.format(category_name))
        category_products = categories[category_name]
        print('{:<20} {:<20} {:<20} {:<20} {:<20}'.format('Product ID', 'Brand','Product Name','Product Type','Unit Price'))
        for prod in category_products:
            print('{:<20} {:<20} {:<20} {:<20} {:<20}'.format(prod['product_id'], prod['brand'], prod['name'], prod['type'], prod['unit_price']))
```

Print All Products of a Category

In [15]:
```
def print_category_products(category_name):
    if category_name in categories:
        print('\nCategory - {}'.format(category_name))
        category_products = categories[category_name]
        print('{:<20} {:<20} {:<20} {:<20} {:<20}'.format('Product ID', 'Brand','Product Name','Product Type','Unit Price'))
        for prod in category_products:
            print('{:<20} {:<20} {:<20} {:<20} {:<20}'.format(prod['product_id'], prod['brand'], prod['name'], prod['type'], prod['unit_price']))
    else:
        print('No Category Found with name: {}'.format(category_name))
```

Print a Product

In [16]:
```
def print_product(product):
    if product != None:
        print('\n{:<20} {:<20} {:<20} {:<20} {:<20}'.format('Product ID', 'Brand','Product Name','Product Type','Unit Price'))
        print('{:<20} {:<20} {:<20} {:<20} {:<20}'.format(product['product_id'], product['brand'], product['name'], product['type'], product['unit_price']))
    else:
        print('Invalid product')
```

Inventory Modifications/Viewing (can be done only by 'Admin' user)

Check Admin User

In [17]:
```
def check_admin():
    admin = input('Enter admin username: ')

    # Check whether the user is Admin or not
    if len(admin) == 0 or admin != 'Admin':
        print('Only Admin can perform this action')
        return False

    is_admin_logged_in = check_user_login(admin)

    if not is_admin_logged_in:
        print('\"{}\" is not yet logged in'.format(admin))

    return is_admin_logged_in
```

Add new category (Admin login required)

In [18]:
```
def add_category():
    # Check whether the user who wants to add category is an Admin or not
    # Only an Admin can add category
    # To add a category the Admin has to login first
    if check_admin():
        new_category = input('Enter New Category: ')
        if new_category not in categories:
            categories[new_category] = []
            print('Category \"{}\" successfully added to the inventory'.format(new_category))
            return True
        else:
            print('Category \"{}\" already exists'.format(new_category))
    else:
        return False
```

Update existing category name (Admin login required)

In [29]:
```
def update_category():
    # Check whether the user who wants to update category name is an Admin or not
    # Only an Admin can update category name
    # To update a category name the Admin has to login first
    if check_admin():
        existing_name = input('Enter Existing Category Name: ')
        new_name = input('Enter New Category Name: ')

        if existing_name in categories:
            categories[new_name] = categories.pop(existing_name)
            print('Category name successfully updated from \"{}\" to \"{}\"'.format(existing_name, new_name))
            return True
        else:
            print('Category \"{}\" not found'.format(existing_name))
            return False
    else:
        return False
```

Delete a category (Admin login required)

In [20]:
```
def delete_category():
    # Check whether the user who wants to delete category is an Admin or not
    # Only an Admin can delete a category
    # To delete a category the Admin has to login first
    if check_admin():
        all_categories = categories.keys()
        print('Delete any category following categories\n{}'.format(', '.join(all_categories)))
        delete_category_name = input('Enter Category Name: ')

        if delete_category_name in all_categories:
            # Check if the category contains any products or not
            # If a category contains products then please reconfirm the deletion of the category from the Admin
            if len(categories[delete_category_name]) > 0:
                delete_confirmation = input('\nThe Category - {} contains {} products, do you want to still delete it? Confirm with Y/N: '.format(delete_category_name, len(categories[delete_category_name])))

                if delete_confirmation.upper() == 'Y':
                    del categories[delete_category_name]
                    print('Category \"{}\" deleted successfully'.format(delete_category_name))
                    return True

                else:
                    print('You did not confirm deletion, skipping category deletion')
                    return False

            else:
                # If it is an empty category delete without confirmation
                del categories[delete_category_name]
                print('Category \"{}\" deleted successfully'.format(delete_category_name))
                return True

        else:
            print('Category \"{}\" not found, please try again'.format(delete_category_name))
            return False

    else:
        return False
```

Add a new product to a category (Admin login required)

In [24]:
```python
def add_category_product():
    if check_admin():
        all_categories = categories.keys()
        print('Add a product to any one of the following categories\n{}'.format(', '.join(all_categories)))
        category_name = input('Enter Category Name: ')

        #Check whether the category name prodived exists or not
        if category_name in categories:
            print('Provide the new product details for Category \"{}\"'.format(category_name))
            # Take inputs for brand name, product name and product type
            brand_name = input('Enter Brand Name: ')
            if(len(brand_name) == 0):
                print('Brand Name can not be empty, Please try again')
                return False

            product_name = input('Enter Product Name: ')
            if(len(product_name) == 0):
                print('Product Name can not be empty, Please try again')
                return False

            product_type = input('Enter Product Type: ')
            if(len(product_type) == 0):
                print('Product Type can not be empty, Please try again')
                return False

            category_products = categories.get(category_name)
            for product in category_products:
                if(product['brand'] == brand_name and product['name'] == product_name and product['type'] == product_type):
                    print('Product already exists. Please add a new product')
                    return False
            # Take input for unit price of the product
            unit_price = input('Enter Unit Price (Non-negetive integer): ')

            #Sanity check for unit price
            if not unit_price.isnumeric() or int(unit_price) < 0:
                print('Invalid unit price. Unit price must be non-negetive integer')
                return False

            # Auto generate product id based on the category name of the product
            product_id = '{}-00{}'.format(category_name, len(category_products)+1)

            # Finally add the new product for the category
            categories[category_name].append({'product_id': product_id, 'brand': brand_name, 'name': product_name, 'type': product_type, 'unit_price': int(unit_price)})
            print('Product successfully added to category \"{}\"'.format(category_name))

            # Print all the category products after adding the product
            print_category_products(category_name)

            return True
        else:
            print('Category \"{}\" not found. Please try again'.format(category_name))
            return False
    else:
        return False
```

Udate a product from a category (Admin login required)

In [25]:
```
def update_category_product():
    if check_admin():
        all_categories = categories.keys()
        print('\nUpdate a product from any one of the following categories\n{}'.format(', '.join(all_categories)))
        category_name = input('Enter category: ')
        # Check whether the category name provided exists or not
        if category_name in categories:
            # All product ids from the given category
            product_id_list = list(map(lambda x: x['product_id'], categories[category_name]))
            # If a category has no product, then return
            if len(product_id_list) == 0:
                print('No product found in the category')
                return False

            # Print all the category products for the reference of admin to update one
            print_category_products(category_name)

            product_id = input('\nEnter Product ID to update (Refer the above table to get the Product ID): ')
            if product_id in product_id_list:
                existing_product = next((x for x in categories[category_name] if x['product_id'] == product_id), None)
                if existing_product != None:
                    print('\n Follow product is found, please proceed to update')
                    print_product(existing_product)

                    # Take inputs for brand name, product name and product type
                    print('\nPlease provide the updated info for the product attributes')
                    print('N.B. If you do not want to update any specific attribute give blank input for the corresponding attribute')
                    brand_name = input('Enter Brand Name: ')
                    product_name = input('Enter Product Name: ')
                    product_type = input('Enter Product Type: ')
                    unit_price = input('Enter Unit Price (Non-negetive integer): ')

                    # Set existing product attributes as default when blank input is given
                    if(len(brand_name) == 0):
                        brand_name = existing_product['brand']

                    if(len(product_name) == 0):
                        product_name = existing_product['name']

                    if(len(product_type) == 0):
                        product_type = existing_product['type']

                    if(len(unit_price) == 0):
                        unit_price = existing_product['unit_price']

                    # Sanity check for unit price
                    if isinstance(unit_price, str):
                        if not unit_price.isnumeric() or int(unit_price) < 0:
                            print('Invalid unit price. Unit price must be non-negetive integer')
                            return False

                    # Finally, Update the product
                    udpate_product = next((prod for prod in categories[category_name] if prod['product_id'] == product_id), None)
                    if udpate_product != None:
                        udpate_product['brand'] = brand_name
                        udpate_product['name'] = product_name
                        udpate_product['type'] = product_type
                        udpate_product['unit_price'] = int(unit_price)

                        print('\nProduct List After Update')
                        print_category_products(category_name)
                        return True

                    else:
                        print('\nFailed to update product, please try again')
                        return False

                else:
                    print('Product ID \"{}\" not found in Category \"{}\"'.format(product_id, category_name))
                    return False

            else:
                print('Product ID \"{}\" not found in Category \"{}\"'.format(product_id, category_name))
                return False
        else:
            print('Category \"{}\" not found. Please try again'.format(category_name))
            return False
    else:
        return False
```

Delete a product from a category (Admin login required)

In [93]:
```
def delete_category_product():
    if check_admin():
        all_categories = categories.keys()
        print('\nDelete a product from any one of the following categories\n{}'.format(', '.join(all_categories)))
        category_name = input('Enter category: ')
        # Check whether the category name provided exists or not
        if category_name in categories:
            # All product ids from the given category
            product_id_list = list(map(lambda x: x['product_id'], categories[category_name]))
            # If a category has no product, then return
            if len(product_id_list) == 0:
                print('No product found in Category \"{}\"'.format(category_name))
                return False

            # Print all the category products for the reference of admin to update one
            print_category_products(category_name)

            delete_product_id = input('Enter the Product ID of the Product to be deleted: ')

            if delete_product_id in product_id_list:
                for index in range(len(categories[category_name])):
                    if categories[category_name][index]['product_id'] == delete_product_id:
                        del categories[category_name][index]
                        print('Product deleted successfully from Category \"{}\"'.format(category_name))
                        print('\n Category after deletion of the product is as below:')
                        print_category_products(category_name)
                        return True

            else:
                print('No product found with Product ID \"{}\"'.format(delete_product_id))
                return False
        else:
            print('Category \"{}\" not found. Please try again'.format(category_name))
            return False
    else:
        return False
```

Show Orders of All Users (Admin login required)

In [94]:
```
def show_all_users_orders():
    if check_admin():
        username_list =  list(map(lambda item: item['username'], list(filter(lambda user: user['username'] != 'Admin', users))))
        if len(username_list) > 0:
            for username in username_list:
                print_user_orders(username)
        else:
            print('No user found')
        return True

    else:
        return False
```

Delete User (Admin login required)

In [95]:
```
def delete_user():
    if check_admin():
        username = input('Enter the username to be deleted: ')
        if(username != 'Admin'):
            user_to_be_deleted = None
            for user in users:
                if user['username'] == username:
                    user_to_be_deleted = user
                    break

            if user_to_be_deleted != None:
                # User found perform deleted
                # Remove cart entry for the user to be deleted
                user_cart_items = list(filter(lambda item: item['username'] == username, user_carts))
                if(len(user_cart_items) > 0):
                    for item in user_cart_items:
                        user_carts.remove(item)

                # Remove user from logged_in_users list
                for logged_in_user in logged_in_users:
                    if logged_in_user['username'] == user_to_be_deleted['username']:
                        logged_in_users.remove(logged_in_user)
                        break

                # Finally, remove the user from the users list
                for user in users:
                    if user['username'] == user_to_be_deleted['username']:
                        users.remove(user)
                        break

                print('\"{}\" deleted successfully'.format(username))
            else:
                print('\"{}\" is not a user'.format(username))
        else:
            print('Can not delete the Admin user')
            return False
    else:
        return False
```

Functions for Shopping (General Users)

Validate User Login

In [96]:
```
def validate_user_session():
    username = input('Enter the Username: ')

    if len(username) == 0:
        print('Username can not be empty')
        return None

    is_logged_in = check_user_login(username)

    if not is_logged_in:
        print('\"{}\" is not yet logged in'.format(username))
        return None

    return username
```

User Cart Section

Print Cart Info for User (User login required)

In [97]:
```
def print_user_cart(username):
    is_logged_in = check_user_login(username)

    if is_logged_in:
        print('\nCart Info of \"{}\":\n'.format(username))
        user_cart_items = list(filter(lambda item: item['username'] == username, user_carts))
        if(len(user_cart_items) > 0):
            print('{:<20} {:<20} {:<20} {:<20} {:<20}'.format('Product ID', 'Product Name', 'Category', 'Unit Price', 'Quantity'))
            for prod in user_cart_items:
                print('{:<20} {:<20} {:<20} {:<20} {:<20}'.format(prod['product_id'], prod['product_name'], prod['category'], prod['unit_price'], prod['quantity']))
        else:
            print('No items in cart')
    else:
        print('\"{}\" is not yet logged in'.format(username))
        return False
```

Cart Info for User (User login required)

In [98]:
```
def user_cart_info():
    username = validate_user_session()
    if username != None:
        print_user_cart(username)
        return True
    else:
        return False
```

Add Products to Cart (User login required)

In [99]:
```
def add_to_cart():
    # First take username and validate if the user is logged in or not
    username = validate_user_session()
    if username != None:
        add_more_product = 'Y'
        print('\nChoose from the below avalable products')
        print_all_category_products()
        print('\n')

        while add_more_product.upper() == 'Y':
            cart_product_id = input('Enter the ID of Product you want to add to your cart: ')
            cart_category = None
            cart_product = None
            cart_item = None
            # Check if the product id provided is valid or not
            for category in categories:
                cart_category = category
                for prod in categories[category]:
                    if prod['product_id'] == cart_product_id:
                        cart_product = prod
                        cart_item = {'username': username, 'product_id': prod['product_id'], 'product_name': prod['name'], 'category': cart_category, 'unit_price': prod['unit_price'], 'quantity': 0}
                        break

            if  cart_product != None and cart_item != None:
                print('\nProduct found')
                print_product(cart_product)

                quantity = input('Enter quantity (Non-negetive integer): ')
                if quantity.isnumeric() and int(quantity) > 0:
                    # Add the product to the cart for specific username
                    cart_item['quantity'] = quantity

                    # Find the user cart items
                    existing_cart_item = next((item for item in user_carts if item['username'] == username and item['product_id'] == cart_item['product_id']), None)
                    # If the item is already present in the cart then just add the the current quantity with the existing quantity else add the item to the cart
                    if existing_cart_item == None:
                        user_carts.append(cart_item)
                    else:
                        existing_cart_item['quantity'] = int(existing_cart_item['quantity']) + int(cart_item['quantity'])


                    print('Product \"{}\" from category \"{}\" has been successfully added to the cart!'.format(cart_product['name'], cart_category))
                else:
                    print('Invalid quantity. Quantity must be non-negetive integer')
            else:
                print('Product not found with ID \"{}\"\n'.format(cart_product_id))

            add_more_product = input('\nDo you want to add more products to your cart? Enter \"Y\" to add more: ')

        print_user_cart(username)
        return True
    else:
        return False
```

Delete Specific Items from User's Cart (User login required)

In [100…
```
def remove_cart_items():
    # First take username and validate if the user is logged in or not
    username = validate_user_session()
    if username != None:
        user_cart_items = list(filter(lambda item: item['username'] == username, user_carts))
        delete_more_product = 'Y'
        print_user_cart(username)
        print('\n')
        while(delete_more_product.upper() == 'Y'):
            if(len(user_cart_items) == 0):
                print('No items in cart found for User \"{}\"'.format(username))
                break

            found_product = None
            cart_product_id = input('Enter the ID of Product you want to delete from your cart: ')
            for item in user_cart_items:
                if item['product_id'] == cart_product_id:
                    found_product = item
                    break

            if found_product != None:
                user_carts.remove(found_product)
                print('Product successfully removed from cart"\n')
            else:
                print('Product not found with ID \"{}\"\n'.format(cart_product_id))


            user_cart_items = list(filter(lambda item: item['username'] == username, user_carts))
            print('\nUpdated Cart info after item deletion')
            print_user_cart(username)

            delete_more_product = input('\nDo you want to delete more products to your cart? Enter \"Y\" to delete more: ')

        return True
    else:
        return False
```

Clear User Cart (User Login required)

In [101…
```
# Clears the cart items for the given logged in user
def clear_user_cart():
    # First take username and validate if the user is logged in or not
    username = validate_user_session()
    if username != None:
        user_cart_items = list(filter(lambda item: item['username'] == username, user_carts))
        if(len(user_cart_items) > 0):
            for item in user_cart_items:
                user_carts.remove(item)
            print('Successfully deleted all the items from the cart of \"{}\"'.format(username))
        else:
            print('Cart of \"{}\" is already empty'.format(username))
    else:
        return False
```

User Order Section

Print User Order(s)

In [102…
```
def print_user_orders(username):
    user_order_list = list(filter(lambda item: item['username'] == username, user_orders))
    if len(user_order_list) > 0:
        print('\nOrder List of \"{}\"'.format(username))
        for order in user_order_list:
            print('\nOrder ID: {}, Order Date: {}, Order Total: {}, Payment Method: {}'.format(order['order_id'], order['order_time'], order['order_total'], order['payment_method']))
            print('Order Items:')
            order_items = order['order_items']
            if len(order_items) > 0:
                print('{:<20} {:<20} {:<20} {:<20} {:<20}'.format('Product ID', 'Product Name', 'Category', 'Unit Price', 'Quantity'))
                for item in order_items:
                    print('{:<20} {:<20} {:<20} {:<20} {:<20}'.format(item['product_id'], item['product_name'], item['category'], item['unit_price'], item['quantity']))
            else:
                print('No order items found')
    else:
        print('No order found for \"{}\"'.format(username))
```

Show User Order(s) (User login required)

In [103…
```
def show_user_orders():
    # First take username and validate if the user is logged in or not
    username = validate_user_session()
    if username != None:
        print_user_orders(username)
        return True
    else:
        return False
```

Place Order from Cart (User login required)

In [105…
```
import datetime

def place_order():
    # First take username and validate if the user is logged in or not
    username = validate_user_session()
    if username != None:
        # Check whether user has any item(s) added to his/her cart or not
        user_cart_items = list(filter(lambda item: item['username'] == username, user_carts))
        if len(user_cart_items) > 0:
            # Check all the products in the cart still exists or not
            # Use case: If a product from a category is added to a user cart,
            # and then the product is deleted or the category itself is deleted along with all the products in it by the Admin
            deleted_products_from_cart = []
            for item in user_cart_items:
                # First check whether cart item category exists or not, if category does not exist then add cart item to deleted_products_from_cart
                if not item['category'] in categories:
                    deleted_products_from_cart.append({'product_id': item['product_id'], 'product_name': item['product_name'], 'category': item['category']})
                else:
                    # Then check whether the product exist in the category or not
                    product_id_list = list(map(lambda x: x['product_id'], categories[item['category']]))
                    if not item['product_id'] in product_id_list:
                        deleted_products_from_cart.append({'product_id': item['product_id'], 'product_name': item['product_name'], 'category': item['category']})

            if(len(deleted_products_from_cart) > 0):
                # There are deleted products
                print('The following product(s) in the cart not longer exists, please remove them from the cart before placing your order')
                print('{:<20} {:<20} {:<20}'.format('Category', 'Product ID', 'Product Name'))
                for deleted_product in deleted_products_from_cart:
                    print('{:<20} {:<20} {:<20}'.format(deleted_product['category'], deleted_product['product_id'], deleted_product['product_name']))

                return False

            order_items = []
            total_amount_payable = 0
            for item in user_cart_items:
                cart_item_price = int(item['quantity']) * int(item['unit_price'])
                total_amount_payable += cart_item_price
                order_items.append({'product_id': item['product_id'], 'product_name': item['product_name'], 'category': item['category'], 'unit_price': item['unit_price'], 'quantity': item['quantity']})

            # Choose Payment Method and Pay
            print_user_cart(username)
            print('\nTotal Payable amount is: {}'.format(total_amount_payable))
            print('\nPlease choose from the below payment methods to place your order')
            print('{}'.format(', '.join(payments_methods)))
            payment_mode = input('Enter Payment Mode: ')
            if payment_mode in payments_methods:
                # Create an order entry for the user
                order_id = 'order-00{}'.format(len(user_orders) + 1)
                user_orders.append({'order_id': order_id, 'username': username, 'order_time': datetime.datetime.now(), 'order_total': total_amount_payable, 'payment_method': payment_mode, 'order_items': order_items})
                 # Clear user cart
                for item in user_cart_items:
                    user_carts.remove(item)

                print('Order placed successfully for \"{}\". Thank You for Shopping with us'.format(username))

            else:
                print('\nOops..! Invalid Payment method, please try again')
                return False

        else:
            print('No cart item found for \"{}\"'.format(username))

        return True
    else:
        return False
```

==========

==========

Function Calls

User Management funtion calls

Call user_sign_up (Multiple users can sign up into the system)

In [108…
```
# Call user_sign_up() for user 1
# This create a new user Saurav in the system
user_sign_up()
```

```
Congratulation, user successfully created. Please login to use the shopping application.
```

In [109…
```
# Call user_sign_up() for user 2
# Use case when the user enters an invalid password
user_sign_up()
```

```
Invalid password.
Passoword format required:
1) The password must contain atleast one upper case, one lower case, one number, and one special character from the followings: ! @ # $ % & _ .
2) The minimum length of the must password must be 6 characters and the maximum length can be 20 characters
```

In [110…
```
# Call user_sign_up() for user 2
# With correct password format
# This creates another new user SG in the system
user_sign_up()
```

```
Congratulation, user successfully created. Please login to use the shopping application.
```

Print users list

In [111…
```
# Here, we can see that there are total 3 users in the system, the Admin user being the default one which is pre-seeded in the system
print(users)
```

```
[{'username': 'Admin', 'password': 'Admin@1234'}, {'username': 'Saurav', 'password': 'Password@1234'}, {'username': 'SG', 'password': 'Password@1234'}]
```

Call user_login

In [113…
```
# Any user including the Admin needs to log into the system first to do their respective operations
# Log in as Admin
user_login()
```

```
Hello Admin, welcome to Shopping Cart Application!
```

Print Logged in user list

In [114…
```
# Here we can see there is only one user logged in the system which is the Admin
print(logged_in_users)
```

```
[{'username': 'Admin', 'password': 'Admin@1234'}]
```

Call user_logout

In [121…
```
# This function logs out the user from the system
user_logout()
```

```
Admin has been successfully logged out of the system
```

In [122…
```
# Here we see that no more user is logged in the system
# Logged in userlist is empty
print(logged_in_users)
```

```
[]
```

In [123…
```
# Again we need to log into the system as Admin as logout is called previously for Admin
user_login()
```

```
Hello Admin, welcome to Shopping Cart Application!
```

Print Logged in user list

In [125…
```
# Here we see again there is one user in the logged_in_users list which is the Admin, post the above login call
print(logged_in_users)
```

```
[{'username': 'Admin', 'password': 'Admin@1234'}]
```

Inventory Management function calls (only Admin user can modify inventory)

(for every inventory operation the user needs to verify himself/herself as Admin first)

Add New Category (Admin access only)

In [126…
```
# Add a category the user need to verify as Admin and the system checks whether Admin is logged into the system or not
# Adding a new category "furniture"
add_category()
```

```
Category "furniture" successfully added to the inventory
```

Out[126]: True

Show Categories (Admin access only)

In [129…
```
# Now we see a new category "furniture" is being added to the sytem along with the other pre-seeded categories
# However, the category does no contain any product yet, which will be added in the forth coming function calls
print_all_category_products()
```

```
Category - footwear
Product ID          Brand               Product Name        Product Type        Unit Price
footwear-001        Crocs               LiteRide 360        Clogs               2000
footwear-002        Nike                Air Jordan          Sneakers            4000

Category - clothing
Product ID          Brand               Product Name        Product Type        Unit Price
clothing-001        Lacoste             Sport 3D            T-Shirt             3500
clothing-002        Peter England       Slim Fit            Formal Shirt        1500

Category - electronics
Product ID          Brand               Product Name        Product Type        Unit Price
electronics-001     Apple               iPhone 16 Pro Max   Mobile              75000
electronics-002     Samsung             55 OLED 4K TV       TV                  40000

Category - furniture
Product ID          Brand               Product Name        Product Type        Unit Price
```

Update Category Name (Admin access only)

In [130…
```
# This function updates the name of any existing category
# Here we see that the the category "furniture" is renamed to "furnitures"
update_category()
```

```
Category name successfully updated from "furniture" to "furnitures"
```

Out[130]: True

Print Inventory Stock (Admin access only)

In [132…
```
# Printing the inventory to verify the change
print_all_category_products()
```

```
Category - footwear
Product ID          Brand               Product Name        Product Type        Unit Price
footwear-001        Crocs               LiteRide 360        Clogs               2000
footwear-002        Nike                Air Jordan          Sneakers            4000

Category - clothing
Product ID          Brand               Product Name        Product Type        Unit Price
clothing-001        Lacoste             Sport 3D            T-Shirt             3500
clothing-002        Peter England       Slim Fit            Formal Shirt        1500

Category - electronics
Product ID          Brand               Product Name        Product Type        Unit Price
electronics-001     Apple               iPhone 16 Pro Max   Mobile              75000
electronics-002     Samsung             55 OLED 4K TV       TV                  40000

Category - furnitures
Product ID          Brand               Product Name        Product Type        Unit Price
```

In [139…
```
# Adding another category "common" to the system
add_category()
```

```
Category "common" successfully added to the inventory
```

Out[139]: True

In [140…
```
# Printing the inventory to check whether category "common" is added to the inventory or not
print_all_category_products()
```

```
Category - footwear
Product ID           Brand               Product Name         Product Type         Unit Price
footwear-001         Crocs               LiteRide 360         Clogs                2000
footwear-002         Nike                Air Jordan           Sneakers             4000

Category - clothing
Product ID           Brand               Product Name         Product Type         Unit Price
clothing-001         Lacoste             Sport 3D             T-Shirt              3500
clothing-002         Peter England       Slim Fit             Formal Shirt         1500

Category - electronics
Product ID           Brand               Product Name         Product Type         Unit Price
electronics-001      Apple               iPhone 16 Pro Max    Mobile               75000
electronics-002      Samsung             55 OLED 4K TV        TV                   40000

Category - furnitures
Product ID           Brand               Product Name         Product Type         Unit Price

Category - common
Product ID           Brand               Product Name         Product Type         Unit Price
```

Delete Category (Admin access only)

In [141…
```
# Deleting a category from the system
# Deleting a category will delete all the products in the category as well
delete_category()
```

```
Delete any category following categories
footwear, clothing, electronics, furnitures, common

Category "common" deleted successfully
```

Out[141]: True

Print Inventory Stock (Admin access only)

In [142…
```
# No we can see that the category "common" has been removed from the inventory
print_all_category_products()
```

```
Category - footwear
Product ID           Brand               Product Name         Product Type         Unit Price
footwear-001         Crocs               LiteRide 360         Clogs                2000
footwear-002         Nike                Air Jordan           Sneakers             4000

Category - clothing
Product ID           Brand               Product Name         Product Type         Unit Price
clothing-001         Lacoste             Sport 3D             T-Shirt              3500
clothing-002         Peter England       Slim Fit             Formal Shirt         1500

Category - electronics
Product ID           Brand               Product Name         Product Type         Unit Price
electronics-001      Apple               iPhone 16 Pro Max    Mobile               75000
electronics-002      Samsung             55 OLED 4K TV        TV                   40000

Category - furnitures
Product ID           Brand               Product Name         Product Type         Unit Price
```

Add a Product to a Category (Admin access only)

In [144…
```
# Add a product a category
# Here we are adding a product to the category "furnitures"
add_category_product()
```

```
Add a product to any one of the following categories
footwear, clothing, electronics, furnitures

Provide the new product details for Category "furnitures"

Product successfully added to category "furnitures"

Category - furnitures
Product ID           Brand               Product Name         Product Type         Unit Price
furnitures-001       Godrej Interio      King Bed             Bed                  55000
```

Out[144]: True

Print Inventory Stock (Admin access only)

In [145…
```
# Printing inventory list the check the product entry in category "furnitures"
print_all_category_products()
```

```
Category - footwear
Product ID           Brand               Product Name         Product Type         Unit Price
footwear-001         Crocs               LiteRide 360         Clogs                2000
footwear-002         Nike                Air Jordan           Sneakers             4000

Category - clothing
Product ID           Brand               Product Name         Product Type         Unit Price
clothing-001         Lacoste             Sport 3D             T-Shirt              3500
clothing-002         Peter England       Slim Fit             Formal Shirt         1500

Category - electronics
Product ID           Brand               Product Name         Product Type         Unit Price
electronics-001      Apple               iPhone 16 Pro Max    Mobile               75000
electronics-002      Samsung             55 OLED 4K TV        TV                   40000

Category - furnitures
Product ID           Brand               Product Name         Product Type         Unit Price
furnitures-001       Godrej Interio      King Bed             Bed                  55000
```

In [146...]
```
# Add a product a category
# Here we are adding another product to the category "furnitures"
add_category_product()
```

```
Add a product to any one of the following categories
footwear, clothing, electronics, furnitures

Provide the new product details for Category "furnitures"

Product successfully added to category "furnitures"

Category - furnitures
Product ID           Brand               Product Name         Product Type         Unit Price
furnitures-001       Godrej Interio      King Bed             Bed                  55000
furnitures-002       Godrej Interio      Queen Bed            Bed                  45000
```

Out[146]: True

In [147...]
```
# Printing inventory list the check the product entry in category "furnitures", now there are two products in the category "furnitures"
print_all_category_products()
```

```
Category - footwear
Product ID           Brand               Product Name         Product Type         Unit Price
footwear-001         Crocs               LiteRide 360         Clogs                2000
footwear-002         Nike                Air Jordan           Sneakers             4000

Category - clothing
Product ID           Brand               Product Name         Product Type         Unit Price
clothing-001         Lacoste             Sport 3D             T-Shirt              3500
clothing-002         Peter England       Slim Fit             Formal Shirt         1500

Category - electronics
Product ID           Brand               Product Name         Product Type         Unit Price
electronics-001      Apple               iPhone 16 Pro Max    Mobile               75000
electronics-002      Samsung             55 OLED 4K TV        TV                   40000

Category - furnitures
Product ID           Brand               Product Name         Product Type         Unit Price
furnitures-001       Godrej Interio      King Bed             Bed                  55000
furnitures-002       Godrej Interio      Queen Bed            Bed                  45000
```

Update Product Info of a Category (Admin access only)

In [148...]
```
# Update product info
# To update any particular attibute of a product enter value for that particular attribute and keep other attributes blank
update_category_product()
```

```
Update a product from any one of the following categories
footwear, clothing, electronics, furnitures

Category - furnitures
Product ID           Brand               Product Name         Product Type         Unit Price
furnitures-001       Godrej Interio      King Bed             Bed                  55000
furnitures-002       Godrej Interio      Queen Bed            Bed                  45000

 Follow product is found, please proceed to update

Product ID           Brand               Product Name         Product Type         Unit Price
furnitures-002       Godrej Interio      Queen Bed            Bed                  45000

Please provide the updated info for the product attributes
N.B. If you do not want to update any specific attribute give blank input for the corresponding attribute
```

```
Product List After Update

Category - furnitures
Product ID          Brand               Product Name        Product Type        Unit Price
furnitures-001      Godrej Interio      King Bed            Bed                 55000
furnitures-002      Godrej Interio      Queen Bed           Bed                 42000
```

Out[148]: True

Print Inventory Stock (Admin access only)

In [149...
```
# Print all the products to check the change
print_all_category_products()
```

```
Category - footwear
Product ID          Brand               Product Name        Product Type        Unit Price
footwear-001        Crocs               LiteRide 360        Clogs               2000
footwear-002        Nike                Air Jordan          Sneakers            4000

Category - clothing
Product ID          Brand               Product Name        Product Type        Unit Price
clothing-001        Lacoste             Sport 3D            T-Shirt             3500
clothing-002        Peter England       Slim Fit            Formal Shirt        1500

Category - electronics
Product ID          Brand               Product Name        Product Type        Unit Price
electronics-001     Apple               iPhone 16 Pro Max   Mobile              75000
electronics-002     Samsung             55 OLED 4K TV       TV                  40000

Category - furnitures
Product ID          Brand               Product Name        Product Type        Unit Price
furnitures-001      Godrej Interio      King Bed            Bed                 55000
furnitures-002      Godrej Interio      Queen Bed           Bed                 42000
```

Delete a Product From a Category (Admin access only)

In [150...
```
# Delete a product from a category
# Here we delete a product from the category "furnitures"
delete_category_product()
```

```
Delete a product from any one of the following categories
footwear, clothing, electronics, furnitures

Category - furnitures
Product ID          Brand               Product Name        Product Type        Unit Price
furnitures-001      Godrej Interio      King Bed            Bed                 55000
furnitures-002      Godrej Interio      Queen Bed           Bed                 42000

Product deleted successfully from Category "furnitures"

 Category after deletion of the product is as below:

Category - furnitures
Product ID          Brand               Product Name        Product Type        Unit Price
furnitures-001      Godrej Interio      King Bed            Bed                 55000
```

Out[150]: True

Print Inventory Stock (Admin access only)

In [151...
```
# Printing the inventory to check the product has been deleted from the "category" furnitures
print_all_category_products()
```

```
Category - footwear
Product ID          Brand               Product Name        Product Type        Unit Price
footwear-001        Crocs               LiteRide 360        Clogs               2000
footwear-002        Nike                Air Jordan          Sneakers            4000

Category - clothing
Product ID          Brand               Product Name        Product Type        Unit Price
clothing-001        Lacoste             Sport 3D            T-Shirt             3500
clothing-002        Peter England       Slim Fit            Formal Shirt        1500

Category - electronics
Product ID          Brand               Product Name        Product Type        Unit Price
electronics-001     Apple               iPhone 16 Pro Max   Mobile              75000
electronics-002     Samsung             55 OLED 4K TV       TV                  40000

Category - furnitures
Product ID          Brand               Product Name        Product Type        Unit Price
furnitures-001      Godrej Interio      King Bed            Bed                 55000
```

Show all users orders (Admin access only)

In [152...]
```
# This shows all the orders of a users
# There are no orders yet by any user so it shows empty for both the users "Saurav" and "SG"
show_all_users_orders()
```

```
No order found for "Saurav"
No order found for "SG"
```

Out[152]: True

In [153...]
```
# Lets sign up another user to demonstrates the user delete functionality
user_sign_up()
```

```
Congratulation, user successfully created. Please login to use the shopping application.
```

In [156...]
```
# Printing the list of users to check the entry of "DemoUser" in the system
print(users)
```

```
[{'username': 'Admin', 'password': 'Admin@1234'}, {'username': 'Saurav', 'password': 'Password@1234'}, {'username': 'SG', 'password': 'Password@1234'}, {'username': 'DemoUser', 'password': 'Demo@1234'}]
```

Delete User (Admin access only)

In [157...]
```
# Delete a system user, Admin can not be deleted
delete_user()
```

```
"DemoUser" deleted successfully
```

In [159...]
```
# Printing users list to check whether "DemoUser" is removed from the system or not
print(users)
```

```
[{'username': 'Admin', 'password': 'Admin@1234'}, {'username': 'Saurav', 'password': 'Password@1234'}, {'username': 'SG', 'password': 'Password@1234'}]
```

Shopping Function Calls for General Users

Cart Function Calls

Add to Cart (Logged in user only)

In [162...]
```
# For adding an item the user needs to log in first
# Let's login with the user "Saurav"
user_login()
```

```
Hello Saurav, welcome to Shopping Cart Application!
```

In [163...]
```
# It allows a user to add a product to his/her carts
# Let's try adding some products to the cart of user "Saurav"
add_to_cart()
```

```
Choose from the below avalable products

Category - footwear
Product ID          Brand               Product Name        Product Type        Unit Price
footwear-001        Crocs               LiteRide 360        Clogs               2000
footwear-002        Nike                Air Jordan          Sneakers            4000

Category - clothing
Product ID          Brand               Product Name        Product Type        Unit Price
clothing-001        Lacoste             Sport 3D            T-Shirt             3500
clothing-002        Peter England       Slim Fit            Formal Shirt        1500

Category - electronics
Product ID          Brand               Product Name        Product Type        Unit Price
electronics-001     Apple               iPhone 16 Pro Max   Mobile              75000
electronics-002     Samsung             55 OLED 4K TV       TV                  40000

Category - furnitures
Product ID          Brand               Product Name        Product Type        Unit Price
furnitures-001      Godrej Interio      King Bed            Bed                 55000



Product found

Product ID          Brand               Product Name        Product Type        Unit Price
footwear-001        Crocs               LiteRide 360        Clogs               2000
Product "LiteRide 360" from category "furnitures" has been successfully added to the cart!
Product found

Product ID          Brand               Product Name        Product Type        Unit Price
clothing-001        Lacoste             Sport 3D            T-Shirt             3500
```

```
Product "Sport 3D" from category "furnitures" has been successfully added to the cart!

Cart Info of "Saurav":

Product ID          Product Name        Category            Unit Price          Quantity
footwear-001        LiteRide 360        footwear            2000                1
clothing-001        Sport 3D            clothing            3500                1
```

Out[163]: True

In [165…
```
# For adding an item the user needs to log in first
# Let's login with another user "SG"
user_login()
```

```
Hello SG, welcome to Shopping Cart Application!
```

In [168…
```
# It allows a user to add a product to his/her carts
# Let's try adding some products to the cart of user "SG"
# Here the product with ID "footwear-001" has been added twice by the user "SG" and the entry in the cart for "footwear-001" remains 1 just the cart quantity is incremented
add_to_cart()
```

```
Choose from the below avalable products

Category - footwear
Product ID          Brand               Product Name        Product Type        Unit Price
footwear-001        Crocs               LiteRide 360        Clogs               2000
footwear-002        Nike                Air Jordan          Sneakers            4000

Category - clothing
Product ID          Brand               Product Name        Product Type        Unit Price
clothing-001        Lacoste             Sport 3D            T-Shirt             3500
clothing-002        Peter England       Slim Fit            Formal Shirt        1500

Category - electronics
Product ID          Brand               Product Name        Product Type        Unit Price
electronics-001     Apple               iPhone 16 Pro Max   Mobile              75000
electronics-002     Samsung             55 OLED 4K TV       TV                  40000

Category - furnitures
Product ID          Brand               Product Name        Product Type        Unit Price
furnitures-001      Godrej Interio      King Bed            Bed                 55000



Product found

Product ID          Brand               Product Name        Product Type        Unit Price
footwear-001        Crocs               LiteRide 360        Clogs               2000

Product "LiteRide 360" from category "furnitures" has been successfully added to the cart!

Product found

Product ID          Brand               Product Name        Product Type        Unit Price
footwear-002        Nike                Air Jordan          Sneakers            4000

Product "Air Jordan" from category "furnitures" has been successfully added to the cart!

Product found

Product ID          Brand               Product Name        Product Type        Unit Price
electronics-001     Apple               iPhone 16 Pro Max   Mobile              75000

Product "iPhone 16 Pro Max" from category "furnitures" has been successfully added to the cart!

Product found

Product ID          Brand               Product Name        Product Type        Unit Price
furnitures-001      Godrej Interio      King Bed            Bed                 55000

Product "King Bed" from category "furnitures" has been successfully added to the cart!

Product found

Product ID          Brand               Product Name        Product Type        Unit Price
footwear-001        Crocs               LiteRide 360        Clogs               2000

Product "LiteRide 360" from category "furnitures" has been successfully added to the cart!

Cart Info of "SG":

Product ID          Product Name        Category            Unit Price          Quantity
footwear-001        LiteRide 360        footwear            2000                3
footwear-002        Air Jordan          footwear            4000                1
electronics-001     iPhone 16 Pro Max   electronics         75000               1
furnitures-001      King Bed            furnitures          55000               1
```

Out[168]: True

Show Cart Info (Logged in user only)

In [164…
```
# Show cart info for user "Saurav"
user_cart_info()
```

```
Cart Info of "Saurav":

Product ID          Product Name        Category            Unit Price          Quantity
footwear-001        LiteRide 360        footwear            2000                1
clothing-001        Sport 3D            clothing            3500                1
```

Out[164]: True

In [168…
```
# Show cart info for user "SG"
user_cart_info()
```

```
Cart Info of "SG":

Product ID          Product Name        Category            Unit Price          Quantity
footwear-001        LiteRide 360        footwear            2000                3
footwear-002        Air Jordan          footwear            4000                1
electronics-001     iPhone 16 Pro Max   electronics         75000               1
furnitures-001      King Bed            furnitures          55000               1
```

Out[169]: True

Delete Specific Items from User's Cart (Logged in user only)

In [170…
```
# Remove any product from a user's cart
# Here we are removing a product from the cart of the user "SG"
remove_cart_items()
```

```
Cart Info of "SG":

Product ID          Product Name        Category            Unit Price          Quantity
footwear-001        LiteRide 360        footwear            2000                3
footwear-002        Air Jordan          footwear            4000                1
electronics-001     iPhone 16 Pro Max   electronics         75000               1
furnitures-001      King Bed            furnitures          55000               1


Product successfully removed from cart"


Updated Cart info after item deletion

Cart Info of "SG":

Product ID          Product Name        Category            Unit Price          Quantity
footwear-001        LiteRide 360        footwear            2000                3
footwear-002        Air Jordan          footwear            4000                1
furnitures-001      King Bed            furnitures          55000               1
```

Out[170]: True

In [171…
```
# Updated cart info for user "SG" removal of the product from the cart
user_cart_info()
```

```
Cart Info of "SG":

Product ID          Product Name        Category            Unit Price          Quantity
footwear-001        LiteRide 360        footwear            2000                3
footwear-002        Air Jordan          footwear            4000                1
furnitures-001      King Bed            furnitures          55000               1
```

Out[171]: True

Clear the cart of specific user (Logged in user only)

In [172…
```
# It clears all the entries from the cart of a user
# In this case we are removing all the entries from the cart of the user "Saurav"
clear_user_cart()
```

```
Successfully deleted all the items from the cart of "Saurav"
```

In [173…
```
# Let's check the cart info of "Saurav" post clear cart call
user_cart_info()
```

```
Cart Info of "Saurav":

No items in cart
```

Out[173]: True

User Order Function Calls

Place User Order from Cart (Logged in user only)

In [174…
```
# Place a order from user cart
# Let's now try placing order for "Saurav" whose cart is empty
place_order()
```

```
No cart item found for "Saurav"
```

Out[174]: True

In [175…
```
# Place a order from user cart
# Let's now try placing order for "SG" whose cart has products
# The order is placed successfully after entering a valid payment method
place_order()
```

```
Cart Info of "SG":

Product ID           Product Name         Category             Unit Price           Quantity
footwear-001         LiteRide 360         footwear             2000                 3
footwear-002         Air Jordan           footwear             4000                 1
furnitures-001       King Bed             furnitures           55000                1

Total Payable amount is: 65000

Please choose from the below payment methods to place your order
Credit_Card, Debit_Card, Cash, UPI

Order placed successfully for "SG". Thank You for Shopping with us
```

Out[175]: True

Show User Order(s) (Logged in user only)

In [176…
```
# This shows the orders of the current user
# In this case we are showing the orders for the user "SG"
show_user_orders()
```

```
Order List of "SG"

Order ID: order-001, Order Date: 2024-10-10 09:24:48.346462, Order Total: 65000, Payment Method: Credit_Card
Order Items:
Product ID           Product Name         Category             Unit Price           Quantity
footwear-001         LiteRide 360         footwear             2000                 3
footwear-002         Air Jordan           footwear             4000                 1
furnitures-001       King Bed             furnitures           55000                1
```

Out[176]: True

Finally we call the fuction to show the orders of all the users (Admin access only)

In [177…
```
# This shows all the orders of a users
# There are no orders yet by any user so it shows empty for both the users "Saurav" and "SG"
show_all_users_orders()
```

```
No order found for "Saurav"

Order List of "SG"

Order ID: order-001, Order Date: 2024-10-10 09:24:48.346462, Order Total: 65000, Payment Method: Credit_Card
Order Items:
Product ID           Product Name         Category             Unit Price           Quantity
footwear-001         LiteRide 360         footwear             2000                 3
footwear-002         Air Jordan           footwear             4000                 1
furnitures-001       King Bed             furnitures           55000                1
```

Out[177]: True