Developer Name: Suraj Kumar Samanta

Camera Rental Application Documentation

Overview

The Camera Rental Application is an application that allows users to rent cameras for a specified duration and manage their wallet balance. Users can view available cameras, rent a camera, add money to their wallet, check their wallet balance, and perform additional operations related to camera management. The application helps users browse through available cameras, rent one, and ensure they have a sufficient wallet balance for the rental.

Features

The Camera Rental Application provides the following features:

1.View Available Cameras:
Users can view a list of cameras along with their details, including Brand, Model, Per-day-Rental Amount, and Availability status.

2.Rent a Cameras:
Users can select a camera from the available list and rent it for a specified duration. The application ensures that the user has a sufficient wallet balance to complete the rental.

3.Add Money to Wallet:
Users can deposit money into their wallet to increase their balance for future camera rentals.

4.Check Wallet Balance:
Users can check their current wallet balance to ensure they have enough funds for camera rentals.

5.Add New Camera:
Users with appropriate permissions can add new cameras to the list of available cameras.

6. Remove Camera:
Users with appropriate permissions can remove cameras from the list of available cameras.

7.Return to Main Menu:
Users can navigate back to the main menu to access various features of the application.

8.Close the Application:
Users can exit the application.

Main Classes and Methods

The application consists of several main classes and methods:

1.CameraRentalApp:

- **add Camera (Camera camera):** Adds a new camera to the list of available cameras.
- **remove Camera (int camera Index):** Removes a camera from the list.
- **displayAvailableCameras ():** Displays the list of available cameras.
- **rent Camera (int camera Index):** Allows the user to rent a camera if the balance is sufficient.
- **deposit Money (double amount):** Deposits money into the user's wallet.
- **checkWalletBalance ():** Checks the current wallet balance.
- **displayMainMenu ():** Displays the main menu and allows users to interact with the application's features.

2.Camera:

- **is Rented ():** Checks if the camera is rented (true) or available (false).

3.Wallet:

- **get Balance ():** Retrieves the current balance of the user's wallet.
- **set Balance (double balance):** Sets the wallet balance.
- **deposit (double amount):** Deposits a specified amount into the wallet.

4.CameraMain:

- The main class for running the application. It presents the user with a menu to choose various features and operations.

Algorithm: Camera Rental Application

1.Start

2.Initialize the list of available cameras and user wallet balance.

3.Display Welcome Screen

Print the application name and developer details.

4.Display Main Menu

Print options: View Available Cameras, rent a Camera, Add Money to Wallet, Check Wallet Balance, Add New Camera, Remove Camera, Return to Main Menu, Close the Application.

5.User Input

Accept user choice.

6.Process User Choice

Case 1: View Available Cameras

If the list of cameras is not empty:

Print details of each camera: Brand, Model, Rental Amount, and Availability Status.

Else, print "No Data Present at This Moment."

Case 2: Rent a Camera

If the list of cameras is not empty:

Prompt user to enter the number of the camera to rent.

Check if the selected camera is available (not rented) and the user has sufficient balance.

If available, deduct the rental amount from the user's wallet and mark the camera as rented.

If not, print "Insufficient Wallet Amount. Please deposit money."

Else, print "No Data Present at This Moment."

Case 3: Add Money to Wallet

Prompt user to enter the amount to deposit.

Add the entered amount to the user's wallet balance.

Print "Deposit successful. Current wallet balance: X".

Case 4: Check Wallet Balance

Print the user's current wallet balance.

Case 5: Add New Camera

Prompt user to enter camera details: Brand, Model, Per-day Rental Amount.

Create a new camera object with the entered details and add it to the list of available cameras.

Print "New camera added successfully."

Case 6: Remove Camera

If the list of cameras is not empty:

Print details of each camera with an index.

Prompt user to enter the number of the camera to remove.

Remove the selected camera from the list.

Print "Camera removed successfully."

Else, print "No Data Present at This Moment."

Case 7: Return to Main Menu
Continue to the next iteration of the loop.
Case 8: Close the Application
Print "Thank you for using. Goodbye!" and exit the program.
Default Case:
Print "Invalid choice. Please try again."
7.Loop
Continue displaying the main menu and processing user input until the user chooses to close the application.
8.End

Conclusion

The Camera Rental Application is designed to provide users with a convenient way to rent cameras and manage their wallet balance. It includes features to view available cameras, rent cameras, deposit money, check wallet balances, add new cameras, and remove cameras. The application's structured flow and well-defined classes make it user-friendly and efficient.

The program can be expanded and improved with additional features and enhancements as needed.