CAMERA RENTAL APPLICATION WRITE UP

Product Overview:

The peer-to-peer camera rental application, named rentmycam.io, aims to provide a platform for users to rent cameras from other users. It facilitates the renting process by maintaining a list of available cameras, allowing users to select and rent cameras, and managing wallet balances for rental payments. The application will be developed using Java, incorporating various Java concepts such as data structures, exception handling, collections, and sorting techniques for optimal performance.

Application Flow:

1. Welcome Screen:
    - Display the application name and developer details.
    - Provide options for users to access various features of the application.
2. Main Context:
    - Display a menu with the following options: a. List Cameras b. Rent a Camera c. Wallet Management d. Close Application
3. List Cameras:
    - Retrieve the list of available cameras.
    - Display the camera details, including brand, model, and per-day rental amount.
    - If the list is empty, display a message: "No Data Present at This Moment."
4. Rent a Camera:
    - Prompt the user to select a camera from the available list.
    - Verify if the user's wallet balance is sufficient to cover the per-day rental amount.
    - If the balance is sufficient, deduct the amount from the wallet and proceed with the rental.
    - If the balance is insufficient, display a message: "Insufficient wallet amount."
5. Wallet Management:
    - Provide options to add or view the balance in the user's wallet.
    - Display the current wallet balance.
    - Allow the user to deposit an amount to increase the wallet balance.
    - Display a message indicating the status of the deposit operation.
6. Close Application:
    - Terminate the application execution.

Sprints Plan:

To complete the development of the peer-to-peer camera rental application, it is advisable to plan the implementation over multiple sprints. The number and duration of sprints will depend on the complexity and scope of the project. Considering a moderate level of complexity, we propose the following sprint plan:

Sprint 1:

- Set up Git and create a GitHub account for version control and issue tracking.
- Create the welcome screen with the application name and developer details.
- Implement the main context menu with options for camera listing, renting, wallet management, and application closure.

Sprint 2:

- Implement the functionality to list available cameras with their details.
- Handle the scenario when the camera list is empty.
- Implement exception handling for any potential errors.

Sprint 3:

- Implement the camera rental functionality, including verification of wallet balance and deduction of rental amount.
- Implement the insufficient wallet amount scenario handling.
- Refine exception handling and error messaging.

Sprint 4:

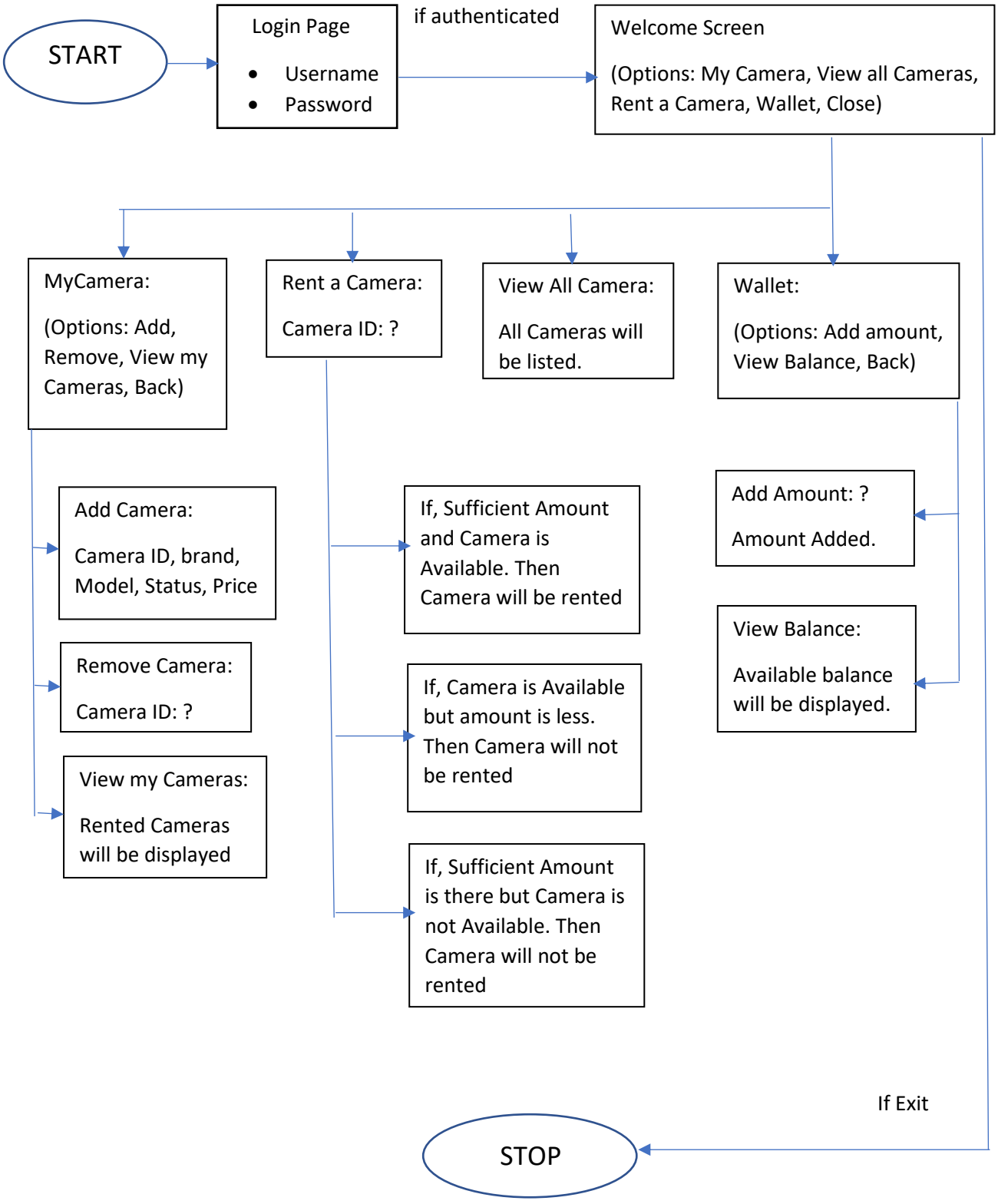
- Implement wallet management features, including viewing and depositing funds.
- Optimize source code using appropriate collections and sorting techniques.
- Conduct thorough testing and bug fixing.

Technologies and Concepts:

The rentmycam.io application will be built using Java and will incorporate the following concepts:

- Java Object-Oriented Programming principles
- Data structures for sorting and searching
- Exception handling
- Collections for efficient data management

FLOWCHART:


START
Login Page
Username
Password
if authenticated
Welcome Screen
(Options: My Camera, View all Cameras, Rent a Camera, Wallet, Close)
MyCamera:
(Options: Add, Remove, View my Cameras, Back)
Rent a Camera:
Camera ID: ?
View All Camera:
All Cameras will be listed.
Wallet:
(Options: Add amount, View Balance, Back)
Add Camera:
Camera ID, brand, Model, Status, Price
Remove Camera:
Camera ID: ?
View my Cameras:
Rented Cameras will be displayed
If, Sufficient Amount and Camera is Available. Then Camera will be rented
If, Camera is Available but amount is less. Then Camera will not be rented
If, Sufficient Amount is there but Camera is not Available. Then Camera will not be rented
Add Amount: ?
Amount Added.
View Balance:
Available balance will be displayed.
If Exit
STOP