Ride Sharing Platform Documentation

Kshitij Jaiswal
B20CS028

Overview:

The Ride Sharing Platform is a simple program that simulates a ride-sharing service, allowing users to register as drivers or riders, offer or request rides, and interact with the system. The program uses object-oriented principles with interfaces and classes to model users, rides, and the ride-sharing platform.

Classes and Interfaces:

1. `UserInterface` (Interface):

- Methods:
  - `getUsername()`: Abstract method to get the username of a user.
  - `getPassword()`: Abstract method to get the password of a user.

2. `DriverInterface` (Interface, extends `UserInterface`):

- Methods:
  - `offerRide(String destination)`: Method for a driver to offer a ride to a specific destination.
  - `acceptRideRequest(Rider rider)`: Method for a driver to accept a ride request from a rider.
  - `rejectRideRequest(Rider rider)`: Method for a driver to reject a ride request from a rider.

3. `RiderInterface` (Interface, extends `UserInterface`):

- Methods:
  - `requestRide(String destination)`: Method for a rider to request a ride to a specific destination.
  - `receiveRideRequest(Driver driver)`: Method for a rider to receive a ride request from a driver.
  - `acceptRideRequest(Driver driver)`: Method for a rider to accept a ride request from a driver.

- `rejectRideRequest(Driver driver)`: Method for a rider to reject a ride request from a driver.

4. `User` (Class, implements `UserInterface`):

- Fields:
  - `username`: The username of the user.
  - `password`: The password of the user.
- Methods:
  - `getUsername()`: Implementation of `getUsername` from `UserInterface`.
  - `getPassword()`: Implementation of `getPassword` from `UserInterface`.
- Constructor:
  - `User(String username, String password)`: Constructor to initialize the username and password.

5. `Driver` (Class, extends `User`, implements `DriverInterface`):

- Fields:
  - `offeredRides`: List of rides offered by the driver.
  - `rideRequests`: List of ride requests received by the driver.
- Methods:
  - `offerRide(String destination)`: Implementation of `offerRide` from `DriverInterface`.
  - `acceptRideRequest(Rider rider)`: Implementation of `acceptRideRequest` from `DriverInterface`.
  - `rejectRideRequest(Rider rider)`: Implementation of `rejectRideRequest` from `DriverInterface`.
  - `getOfferedRides()`: Get the list of rides offered by the driver.
  - `getRideRequests()`: Get the list of ride requests received by the driver.
- Constructor:
  - `Driver(String username, String password)`: Constructor to initialize the username, password, and lists.

6. `Rider` (Class, extends `User`, implements `RiderInterface`):

- Fields:
  - `confirmedRide`: The ride confirmed by the rider.
  - `requestedDestination`: The destination requested by the rider.
- Methods:

        - `requestRide(String destination)`: Implementation of `requestRide` from `RiderInterface`.
        - `receiveRideRequest(Driver driver)`: Implementation of `receiveRideRequest` from `RiderInterface`.
        - `acceptRideRequest(Driver driver)`: Implementation of `acceptRideRequest` from `RiderInterface`.
        - `rejectRideRequest(Driver driver)`: Implementation of `rejectRideRequest` from `RiderInterface`.
        - `getConfirmedRide()`: Get the confirmed ride.
    - Constructor:
        - `Rider(String username, String password)`: Constructor to initialize the username, password, and fields.

7. `Ride` (Class):

- Fields:
    - `driver`: The username of the driver offering the ride.
    - `destination`: The destination of the ride.
- Methods:
    - `getDriver()`: Get the username of the driver.
    - `getDestination()`: Get the destination of the ride.
- Constructor:
    - `Ride(String driver, String destination)`: Constructor to initialize the driver and destination.

8. `RideSharingPlatform` (Class):

- Fields:
    - `drivers`: Map of driver usernames to `Driver` objects.
    - `riders`: Map of rider usernames to `Rider` objects.
- Methods:
    - `registerDriver(String username, String password)`: Register a new driver on the platform.
    - `registerRider(String username, String password)`: Register a new rider on the platform.
    - `isDriverRegistered(String username)`: Check if a driver is registered.
    - `isRiderRegistered(String username)`: Check if a rider is registered.
    - `login(String username, String password, Class<? extends UserInterface> userType)`: Login a user based on type (driver or rider).

- `getAvailableDriversForDestination(String destination)`: Get a list of drivers offering rides to a specific destination.
- `getDrivers()`: Get the map of drivers.
- `getRiders()`: Get the map of riders.

9. `Main` (Class):

- Method:
  - `main(String[] args)`: The main method to run the ride-sharing platform simulation.

How to Run:

Compile the Code:

- Open a terminal or command prompt.
- Navigate to the directory containing the Java file (`Main.java`).
- Compile the code using the command: `javac Main.java`

Run the Program:

- After compiling, run the program with the command: `java Main`
- Follow the prompts to interact with the ride-sharing platform.
- Choose whether you are a Rider or a Driver.
- For registration, enter the requested information.
- For login, enter the username and password.
- Follow the prompts to offer or request rides, accept or reject ride requests, and interact with the system.
- To exit, enter 'yes' when prompted.

Sample Interaction:

- Below is a sample interaction to give you an idea:

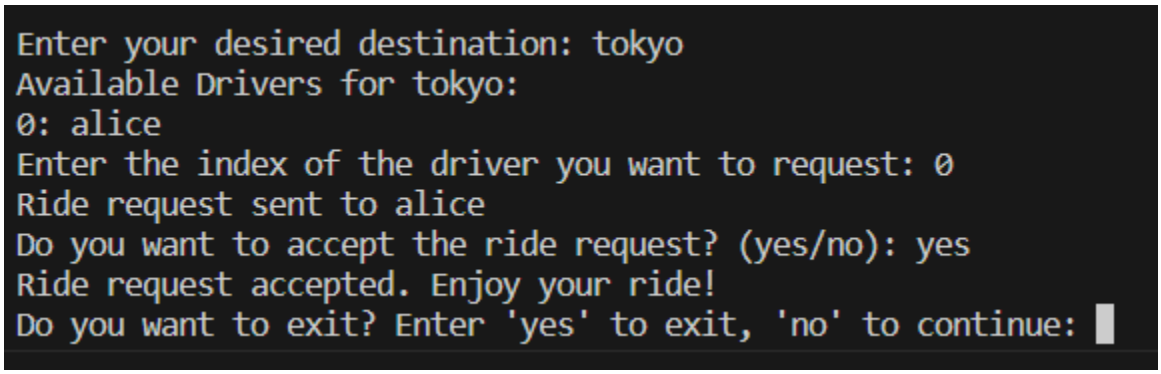
Driver registration and login –

```
PS C:\Users\ACER\Desktop\oops> javac Main.java
PS C:\Users\ACER\Desktop\oops> java Main
Are you a Rider or a Driver? Enter 'r' for Rider, 'd' for Driver: d
--------DRIVER LOGIN/REGISTRATION-------
Are you a new user? Enter 'yes' for new registration, 'no' for login: yes
Enter your username: alice
Enter password: 123
Driver registration successful!

Enter username for login: alice
Enter password for login: 123
Driver login successful!

Enter your offered destination: tokyo
No ride requests at the moment.
Do you want to exit? Enter 'yes' to exit, 'no' to continue: no
Are you a Rider or a Driver? Enter 'r' for Rider, 'd' for Driver: r
```

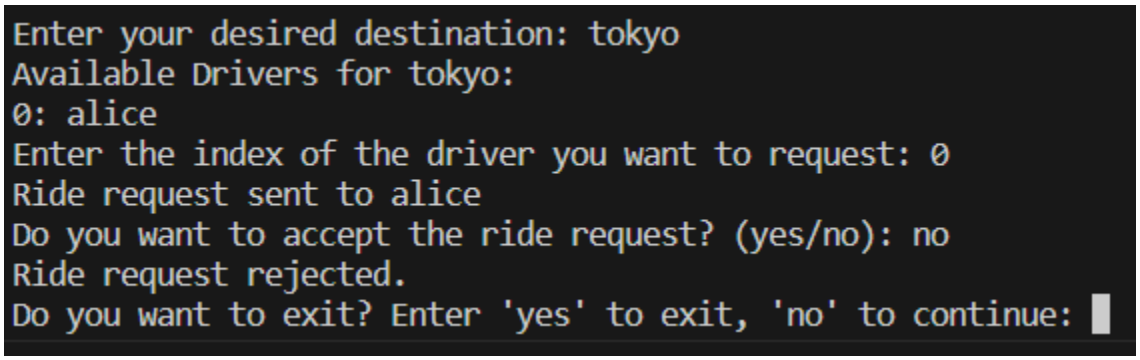
Rider registration and login –

```
Are you a Rider or a Driver? Enter 'r' for Rider, 'd' for Driver: r
-------RIDER LOGIN/REGISTRATION-------
Are you a new user? Enter 'yes' for new registration, 'no' for login: yes
Enter your username: bob
Enter password: 456
Rider registration successful!

Enter username for login: bob
Enter password for login: 456
Rider login successful!
```

Sending of Ride Request by the Rider followed by acceptance of the same by the Driver



Rejection of the Ride Request by the Driver



Note:

- The program may require user input, so follow the prompts and provide necessary information.
- Ensure that you have Java installed on your system.