SHRR

Q1. SwiftRide is a ride-sharing application that connects users for on-demand transportation. Every user is represented by a User class, which contains attributes such as user ID, name, email, phone, and password, and includes basic operations like logging in, logging out, and updating their profile. Users can either be Riders or Drivers. Riders have an additional attribute called default payment and can request rides, view their ride history, and rate drivers. Drivers, on the other hand, have attributes like license number, status, and a reference to their current vehicle. They can accept ride requests, update the status of trips, and view earnings. Each vehicle has a unique ID, model, license plate number, type, and current status, and can be assigned to a driver. When a Rider initiates a ride request, a RideRequest is created containing the request ID, pickup and drop-off locations, time of request, and status. Once accepted by a driver, a Trip is created with attributes like trip ID, start and end times, fare, status, and a Route. The Route, which is part of the Trip, includes the route ID, starting and ending locations, estimated duration, and distance. Each location is described by latitude, longitude, and a readable address. The system ensures low-latency response time for matching Riders and Drivers even during peak hours, as part of its performance requirement. After a trip ends, both Riders and Drivers can leave feedback, which stores the rating ID, score, comment, and timestamp. Drivers also receive Earning Reports that summarize their total number of trips, total earnings, and the time period covered, and can be generated or exported as needed. The application must ensure secure storage of user credentials and sensitive data using encryption to comply with its security requirements. Additionally, SwiftRide is required to maintain 99.9% uptime to ensure high availability for users relying on the service at any time of the day.

a. **Design** a class diagram. Add multiplicity according to your preference. **[12]**

b. Find three functional and three non-functional requirements **from the above scenario**. **[3]**

Non-functional requirements

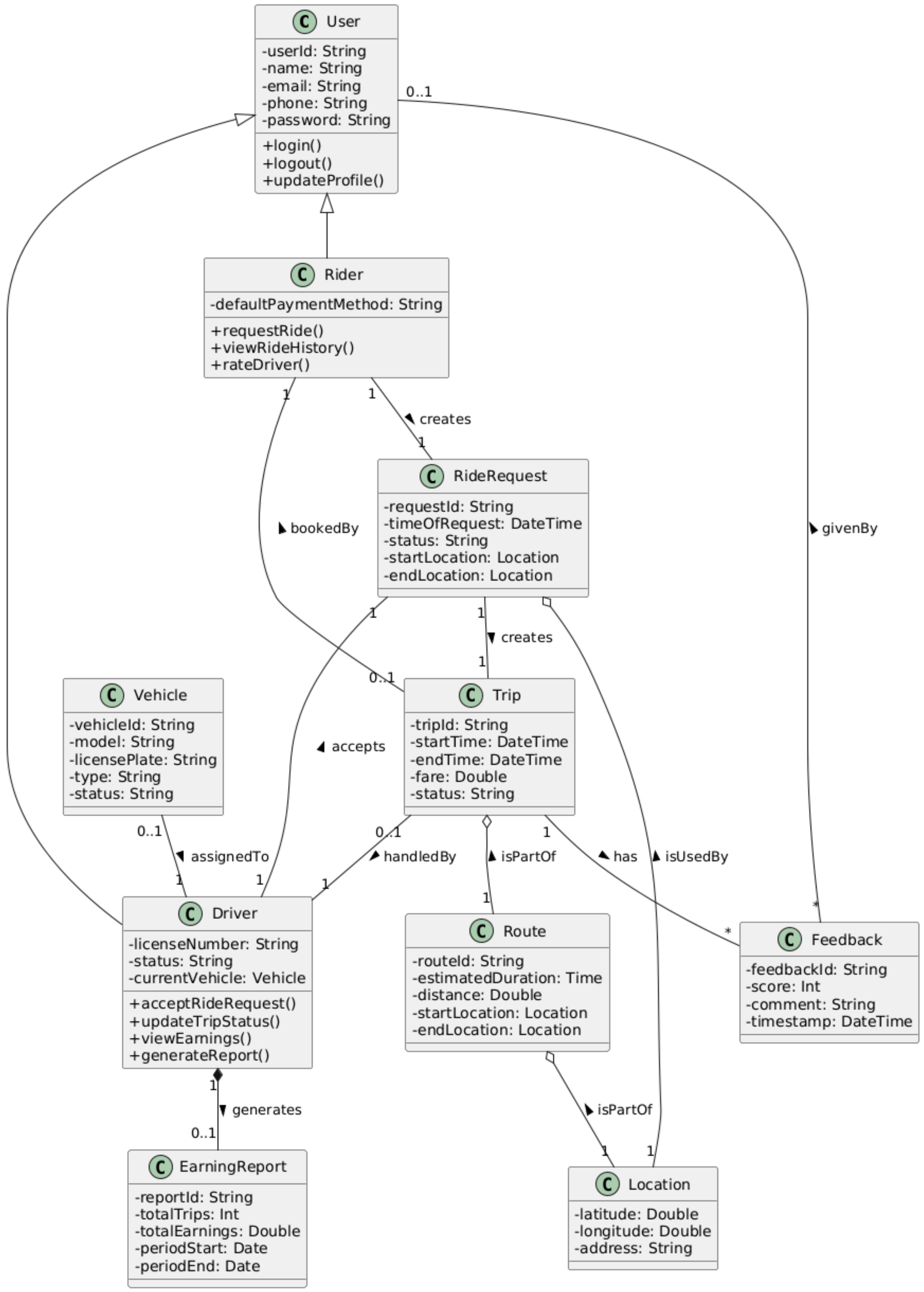
Class Name

Operations/methods, or functional requirements

Attributes

Check the next page for the solution.

Tentative Solution:


User
-userId: String
-name: String
-email: String
-phone: String
-password: String
+login()
+logout()
+updateProfile()
Rider
-defaultPaymentMethod: String
+requestRide()
+viewRideHistory()
+rateDriver()
RideRequest
-requestId: String
-timeOfRequest: DateTime
-status: String
-startLocation: Location
-endLocation: Location
Trip
-tripId: String
-startTime: DateTime
-endTime: DateTime
-fare: Double
-status: String
Vehicle
-vehicleId: String
-model: String
-licensePlate: String
-type: String
-status: String
Driver
-licenseNumber: String
-status: String
-currentVehicle: Vehicle
+acceptRideRequest()
+updateTripStatus()
+viewEarnings()
+generateReport()
Route
-routeId: String
-estimatedDuration: Time
-distance: Double
-startLocation: Location
-endLocation: Location
Feedback
-feedbackId: String
-score: Int
-comment: String
-timestamp: DateTime
EarningReport
-reportId: String
-totalTrips: Int
-totalEarnings: Double
-periodStart: Date
-periodEnd: Date
Location
-latitude: Double
-longitude: Double
-address: String
creates
bookedBy
givenBy
creates
accepts
assignedTo
handledBy
isPartOf
has
isUsedBy
generates
isPartOf
0..1
1
*
0..1