<u>RideShare</u>

1. **Description:** Implement a ride-sharing application with the below-expected features.
2. **Features:**
    a. The application allows users to share rides on a route.
    b. Users can either offer a shared ride (Driver) or consume a shared ride (Passenger).
    c. Users can search and select one from multiple available rides on a route with the same source and destination.
3. **Requirements:**
    a. Application should allow user onboarding.
        i. add_user(user_detail)
            a. Add basic user details
        ii. add_vehicle(vehicle_detail)
            a. Add the user's vehicle(s) details
    b. User should be able to offer a shared ride on a route with details.
        i. offer_ride(ride_detail)
            a. Ride will have details like vehicle, origin, destination, available seats. (A ride will have no intermediate stops.)
    c. Users can select a ride from multiple offered rides using a selection strategy. (A user can only request a ride (only for 1 or 2 people))
        i. select_ride(source, destination, seats, selection_strategy)
            a. Prefered Vehicle (Activa/Polo/XUV)
            b. Most Vacant.
    d. System should be able to end the ride. User can only offer a ride for a given vehicle, once there are no active offered rides for that vehicle.
        i. end_ride(ride_details)
    e. Find total rides offered/taken by all users.
        i. print_ride_stats()

4. **Bonus Question:**

    If the user’s origin/destinations are not available directly but it’s possible via multiple rides, then the application should output multiple rides. (Example: for input: Bangalore to Mumbai, the output can be Bangalore to Goa and Goa to Mumbai)

5. **Other Notes:**
    a. Write a driver class for demo purposes. Which will execute all the commands in one place in the code and test cases.
    b. Do not use any database or NoSQL store, use in-memory data-structure for now.
    c. Do not create any UI for the application.
    d. Please prioritize code compilation, execution, and completion.
    e. Work on the expected output first and then add good-to-have features of your own.
6. **Expectations:**
    a. Make sure that you have a working and demonstrable code.
    b. Make sure that the code is functionally correct.
    c. Use of proper abstraction, modeling, separation of concerns is required.
    d. Code should be modular, readable and unit-testable.
    e. Code should easily accommodate new requirements with minimal changes.
    f. Proper exception handling is required.
7. **Sample Test Cases:**
    a. Onboard 5 users
        i. add_user(“Rohan, M, 36”); add_vehicle(“Rohan, Swift, KA-01-12345)

   ii. add_user(“Shashank, M, 29”); add_vehicle(“Shashank, Baleno, TS-05-62395)
   iii. add_user(“Nandini, F, 29)
   iv. add_user(“Shipra, F, 27”) ; add_vehicle(“Shipra”, Polo, KA-05-41491); add_vehicle(“Shipra, Activa KA-12-12332”)
   v. add_user(“Gaurav, M, 29)
   vi. add_user(“Rahul, M, 35); add_vehicle(“Rahul”, “XUV”, KA-05-1234);

b. Offer 4 rides by 3 users
   i. offer_ride(“Rohan, Origin=Hyderabad, Available Seats=1, Vehicle=Swift, KA-01-12345, Destination= Bangalore”)
   ii. offer_ride(“Shipra, Origin=Bangalore, Available Seats=1, Vehicle=Activa KA-12-12332, Destination=Mysore”)
   iii. offer_ride(“Shipra, Origin=Bangalore, Available Seats=2, Vehicle=Polo, KA-05-41491, Destination=Mysore”)
   iv. offer_ride(“Shashank, Origin=Hyderabad, Available Seats=2, Vehicle=Baleno, TS-05-62395, Destination=Bangalore”)
   v. offer_ride(“Rahul, Origin=Hyderabad, Available Seats=5, Vehicle=XUV, KA-05-1234, Destination=Bangalore”)
   vi. offer_ride(“Rohan, Origin=Bangalore, Available Seats=1, Vehicle=Swift, KA-01-12345, Destination=Pune”)
      a. This call should fail, since a ride has already been offered by this user for this vehicle.

c. Find rides for 4 users
   i. select_ride(“Nandini, Origin=Bangalore, Destination=Mysore, Seats=1, Most Vacant”)
      a. 2(c) is the desired output.
   ii. select_ride(“Gaurav, Origin=Bangalore, Destination=Mysore, Seats=1, Preferred Vehicle=Activa”)
      a. 2(b) is the desired output
   iii. select_ride(“Shashank, Origin=Mumbai, Destination=Bangalore, Seats=1, Most Vacant”)
      a. No rides found
   iv. select_ride(“Rohan, Origin=Hyderabad, Destination=Bangalore, Seats=1, Preferred Vehicle=Baleno”)
      a. 2(d) is the desired output

        v. select_ride(“Shashank, Origin=Hyderabad, Destination=Bangalore, Seats=1,Preferred Vehicle=Polo”)
            a. No rides found

d. End Rides
    i. end_ride(2-a)
    ii. end_ride(2-b)
    iii. end_ride(2-c)
    iv. end_ride(2-d)

e. Find total rides by user: Rides Taken: Rides that have were taken and have been marked as “ended” Rides Offered: Rides that were offered and have been marked as “ended”. Note: Even if the offered ride was not taken by any user, it counts as an offered ride.
    i. print_ride_stats()
        a. Nandini: 1 Taken, 0 Offered
        b. Rohan: 1 Taken, 1 Offered
        c. Shashank: 0 Taken, 1 Offered
        d. Gaurav: 1 Taken, 0 Offered
        e. Rahul: 0 Taken, 0 Offered
        f. Shipra: 0 Taken, 2 Offered