**The story**

Uber has decided to leave the Chisinau market, so there is an opportunity to develop a software solution in order to continue to provide these services without interruption to the citizens of Chisinau.

You are required to develop the backend system which supports the driver's Mobile App.

**Domain description:**

Each driver will be registered in the system and each driver will register his car. There can be one single active car per driver, but there is also a requirement to keep the history of vehicles which were previously registered with each driver. Any driver can acquire a trip. Many drivers can compete for the same trip, but only one will acquire a specific trip(the first to acquire). After each trip, each driver can receive ratings (i.e.: stars from 0 to 5). For each driver there will be available the current ratings score, but also the rating obtained on each trip. Any driver can see he's list of payments and their status. Any driver can see the list of trips. Any driver can see: the total trips price per day, total trips time per day, average cost per trip.

The payment process will occur in two steps:

1. ﻿﻿﻿Some amount will be authorized at the start of the trip(this will be triggered when the trip starts: i.e.: when the drivers acquires the trip)

2. ﻿﻿﻿A confirmation will the total amount will occur at the end of the trip(the confirmation will be triggered when the trip finishes)

The payment service MUST also be developed by you and MUST expose an API for the authorization and confirmation operations. Both operations are asynchronous, so payment service will have to notify back the triggering system, upon completion of payment authorization and confirmation, with the outcome.

Notification will occur using a HTTP API.

Note: For the Payment Service, a simplified version of the implementation is required (i.e.: accept a payment authorization/confirmation request, and notify back automatically after some delay with the payment status)

For driver the following info are required to be stored:

* ID - unique identifier of the driver
* Name
* Phone number

For the vehicle the following info are required to be stored:

* Name
* Color
* Registration number
* Register timestamp

For the payment the following info are required to be stored:

* ﻿﻿ID - unique identifier of the payment
* ﻿﻿Paid price
* ﻿﻿Start - payment initiation timestamp
* ﻿﻿End - payment confirmation timestamp

For the trip the following info are required to be stored:

• ﻿﻿ID - unique identifier of the trip

• ﻿﻿Start location

• ﻿﻿End location

For the payment the following info are required to be stored:

* ID - unique identifier for payment
* Price
* Reward(i.e.: amount received as reward)
* Status(I.e.: PENDING\_AUTHORIZATION, PENDING\_CONFIRMATION, SUCCEEDED, FAILED and reason)

The system MUST expose following APIs to the driver's mobile app, to supply the following info:

• ﻿﻿retrieve the driver profile(driver, vehicles, current rating score, average trip price)

• ﻿﻿register a vehicle

• ﻿﻿acquire trip

• ﻿﻿confirm trip termination

• ﻿﻿list trips(trip info, payment info)

• ﻿﻿list analytics: the total trips price per day, total trips time per day, average cost per trip

The payment service MUST expose the following APis for :

* ﻿﻿payment authorization
* ﻿﻿payment confirmation

**Characteristics for the developed systems:**

* ﻿﻿Resilience(e.g.: API clients and servers MUST be resilient)
* ﻿﻿The application SHOULD be able to scale easily(i.e.: adding more instances).

**Notes:**

* There is no mention of the Client(ie.: the person ordering a trip), as it is not in the scope of this problem. But you can model it in any way you can image in order to have a consistent model. It will not be part of the evaluation, just do whatever you consider necessary that it would help you in designing the driver's model.
* When a requirement is not fully clear, you are allowed to make any assumption and take the simplest design decision. However the decisions SHOULD not eliminate any of the characteristics mentioned above.

**Other constraints:**

* ﻿﻿The application MUST be developed in any of Java or Kotlin languages
* ﻿﻿The application can use any framework, but preferably SpringBoot
* ﻿﻿The source code MUST be available through GitHub in a single private repository and shared with Orange
* ﻿﻿The application MUST use a persistent storage and any database type can be used for this purpose
* ﻿﻿The application MUST be self-contained (i.e.: someone MUST be able to run the application without installing additional software, except Java/Kotlin. externally internet available databases or docker containers for the database are allowed)