

Base de Dados

Ride-sharing App

17 de Março de 2019

2MIEIC01 – Grupo 104

Francisco Batista up201604320@fe.up.pt

João Rocha up201708566@fe.up.pt

Tiago Alves up201603820@fe.up.pt

***Context***

In this project we pretend to create a database for an application to arrange rides between different users. It’s intended that the database is capable to store and manage all the necessary information for the complete operation of the application.

Our idea for the usage of the *app* is that it will be a platform where any user with a vehicle can register, define the details of his trip like starting and ending times and locations, and the other users without a vehicle can join him and get a ride with the intentions of splitting the travel costs. We’ll assume that Users can not share cars and we’ll not include conditions like half-trips, small deviations or extra luggage in the app, as it’s more of a logical problem and not a database related matter.

***Specifications***

Any person using the *app* will be a **User**. The users will have a unique UID, username, password, rating, photo and email address. Users will be able to exchange many messages between themselves. They will be stored as **Message** with a date and text.

There will be also the **Vehicle** class with the attributes plate and colour, which is generalized by **Model** that is composed by name, maxCapacity and avgConsumption. Each model has a **Maker** where it’s described its name and countryOfOrigin. The Vehicle class is a *composition* of the User class. Where a user can own and drive as many vehicles as they wish, but they will be deleted from the database at the same time as the user is associated. This way a Vehicle will always have a user associated as a *driver*.

There is also the **Map** class that has a Path that is created by the two **Address** associated. One is the *starting point* and the other is the *ending point*. The addresses are composed by streetName and a doorNumber.

The vehicle, users and the map are associated to a central class named **Trip**. It will have a trip ID (TID), startTime, endTime, and cost per passenger (costPassenger) that will be calculated based on distance and number of passengers. Each trip can only have a Map, a Vehicle, where the driver associated will be its driver, and it can have passengers as long as they don’t surpass the Vehicle’s maxCapacity. There’s will be also a **PaymentMethod** class associated with each User – Trip that will be a generalization of **Cash** and **BankTransfer**. The accountNumber will be stored by the BankTransfer class.

The class Trip itself it’s a generalization of two classes: The **Punctual** and the **Regular** trips. Both are associated with **Weekday**, a class that has a name and it’s can only be named one of the seven days of the week (Sunday, Monday, …, Saturday). The difference between both trips is that the Punctual must be associated with just one Weekday and the regular can be associated with at least one day of the week. The regular class also has an attribute called onHolidays, which informs if the trip is also done in holidays or not.