Information Management and Systems Engineering MS1

Nikita Antonov 01348746

November 2019

1 Business Model Outline

I have a database for a new startup. It is an aggregator car rental points, located around the world. Every rental point represents the company (for example Hertz or Europcar) and has an address. So we can choose not only between the car rental points of specific company or in a specific city, but among all possible variants. Every rental point has different cars. Every car belongs naturally only to one rental point. But this points can be partners of each other. If two rental points are partners, user can take a car by one of them and leave it at another one. Car still belongs to the first rental point and a special worker will drive the car back to its station. Every car has a maker, model and body (for example Audi A6 Sedan). Car has its own car information card with rental information. Here it is possible to save and update information about number of rents of a car, mileage, its current place. Of course, every car has only one information card and every card belongs only to one car.

Every car has a specific type of an engine. Engine has a type and PS. It is a weak entity, because engine can not working without a car. And one type of engine can be in different cars (for example 1.4 Diesel 120 PS can be in Skoda Octavia, Skoda Superb and VW Golf).

There are users in this app. They have name, login and password. Every user can naturally rent different cars, and cars can be rented by different users.

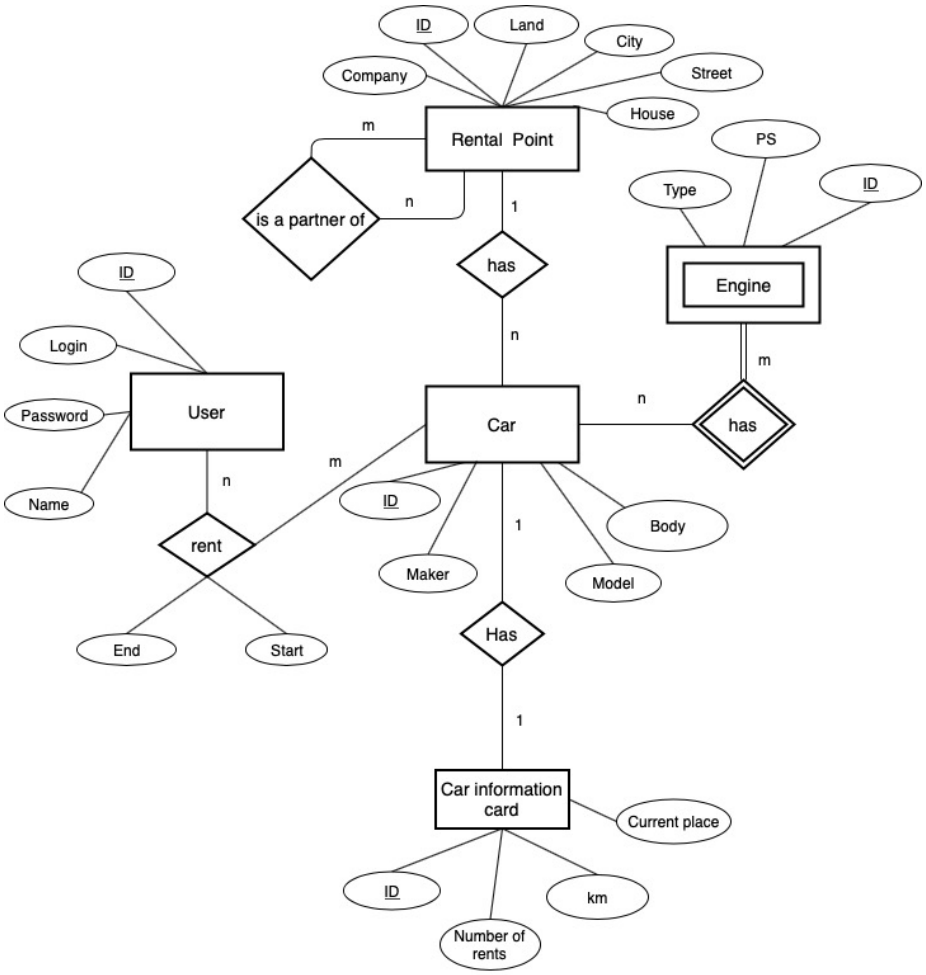

Figure 1: ER Diagram

2 Simple Use-Cases

2.1 Registration of a user

- Objective: the use-case of a user registration through web in a database of an app.
- Short description: in this use-case user can validate his credentials (login and password), if he has it already, or create the new one, if he is a new user of this system.
- Preconditions: database and web-app should run correctly, web-page for a registration should be available.
- Expected execution: if the user is already registered, he should simply give his credentials in a form on the web-page. It sends it to a database. If login and password are correct, then user can enter his account. If not, he should try it one more time. If he isn't already registered, when he can fill the form on the web-page and it will save the information in a database. Next he can log in as described above.
- Postconditions: New user is saved in a database and can log his account with this credentials. If credentials are not correct, he cannot go further as to a login-page.

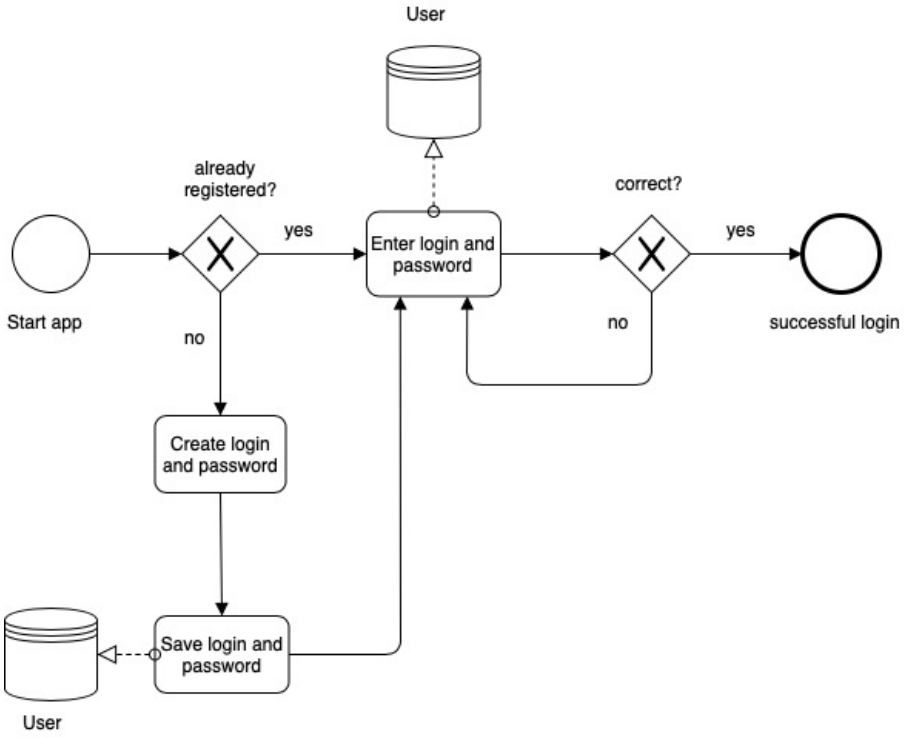

Figure 2: BPMN-Diagram for a "Registration of a user" use-case

2.2 Rent a car (mane use-case)

- Objective: the use-case to rent a car in this app.
- Short description: in this use-case user can rent a car in a specific city, and choose between possible variants.
- Preconditions: database and web-app should run correctly, web-pages should be available, user should be already registered.
- Expected execution: user logs in. He chooses land and city. If there are no rental points in this city, he should try the another one. After that he see all available cars in this city. He chooses the suitable one. If there are this car model with different engines available, then he chooses the specific engine. Finally he types start and end date of this rent. Then he has rented a car.
- Postconditions: user rented the car for specified dates, all this information is saved in a database.

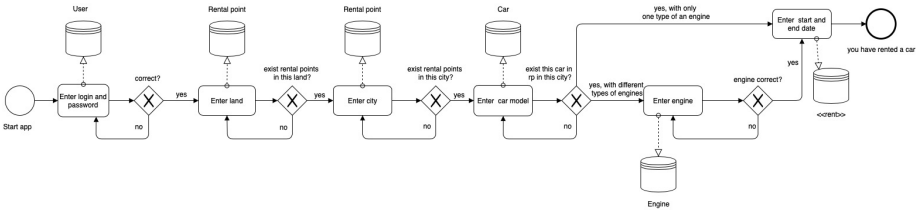

Figure 3: BPMN-Diagram for a "Rent a car" mane use-case

2.3 Check the availability of rental points in the specific city

- Objective: the use-case to check the availability of rental points in the specific city.
- Short description: in this use-case user can type the city name and check rental points there. If there are more than one, he can sort them.
- Preconditions: database and web-app should run correctly, web-pages should be available, user should be already registered, information about rental points should be saved and be up-to-date.
- Expected execution: user logs in. Then he entered the city. Database checks rental points there. If it didn't find any of them there, user see, that nothing found. If it found the information, user can see it.
- Postconditions: user has an actual information about rental points in this city.

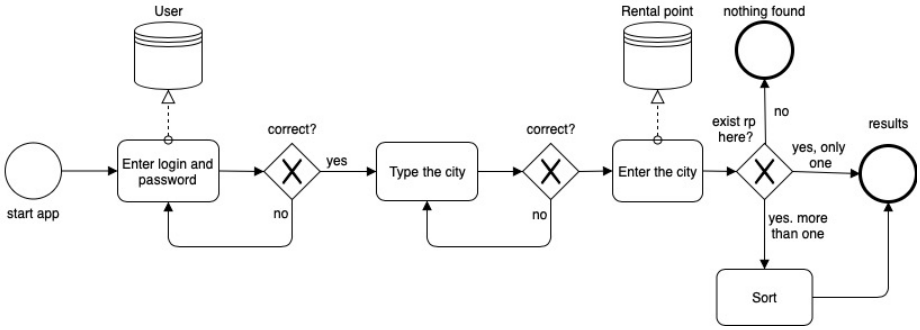

Figure 4: BPMN-Diagram for a "Check the availability of rental points in the specific city" use-case

2.4 Check the partners of the specific rental point

- Objective: the use-case to check the partners of the specific rental point.

- Short description: in this use-case user can select the rental point and check its possible partners.
- Preconditions: database and web-app should run correctly, web-pages should be available, user should be already registered, information about rental points and their partners should be saved and be up-to-date.
- Expected execution: user logs in. Then he select the rental point. Database checks partners of this rental point. If it didn't find any of them there, user see, that nothing found. If it found the information, user can see the list of the partners.
- Postconditions: user has an actual information about rental point partners.

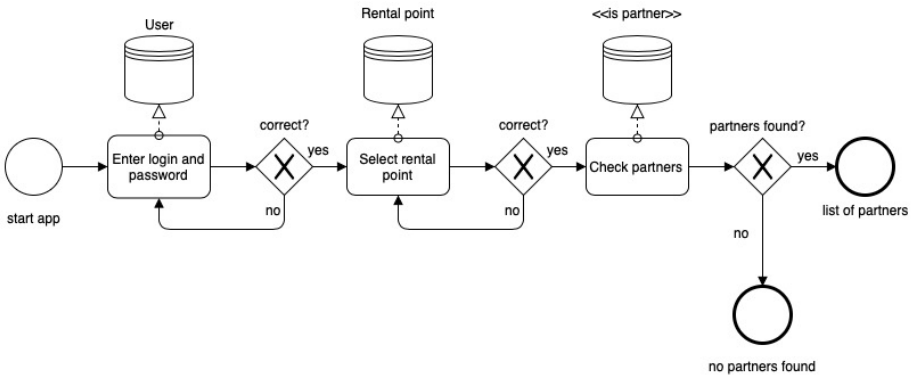

Figure 5: BPMN-Diagram for a "Check the partners of the specific rental point" use-case

3 Elaborate Use-Case

3.1 Report "How often the specific user takes car from a specific land?"

- Objective: the use-case to make the report "How often the specific user takes car from a specific land?"
- Short description: in this use-case it is possible to count, how much the specific user rented cars from a specific land (from all rental points in this land).
- Preconditions: database and web-app should run correctly, user should exists, he had to rent a car at least one time.

- Expected execution: first, user should be successfully registered (see Registration use-case).Then we are searching this user. Next, we are looking for all the cars, that he had rented. Then we proof the rental points of this cars. In this rental points we search the land, and count the suitable one. Then we return the results. If he never booked a car there, we return naturally 0.
- Postconditions: we have the number of car rental of this user in the specific country or an information, that he never rented a car there.

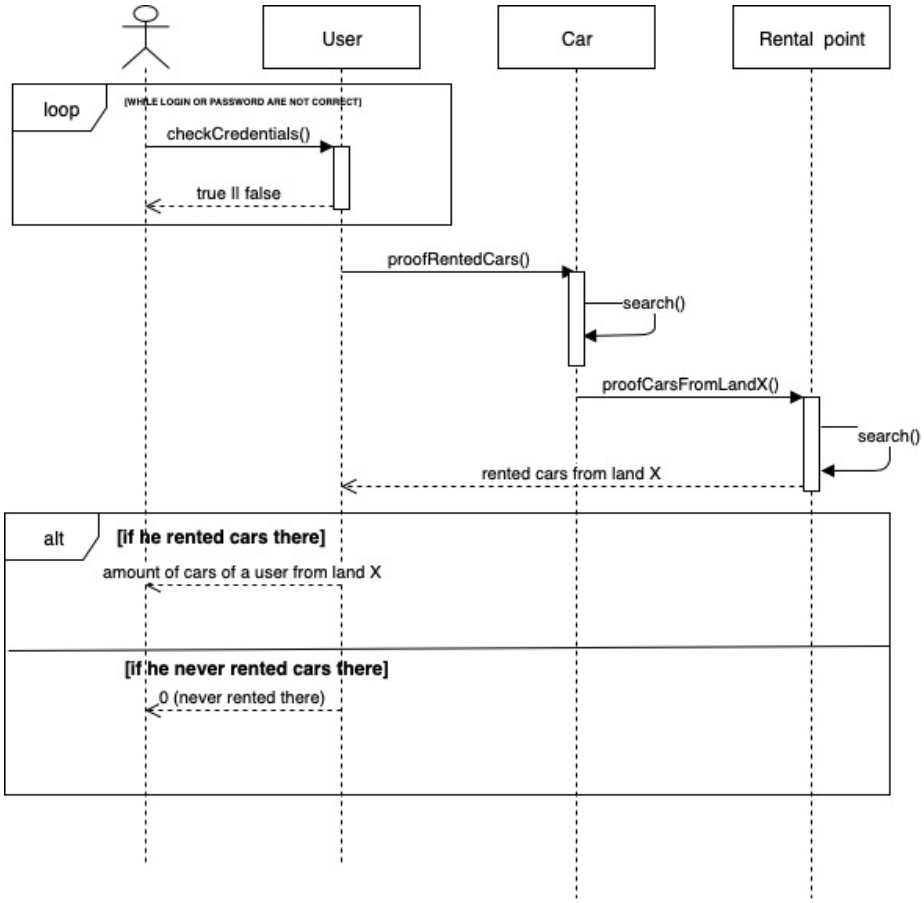

Figure 6: UML Sequence-Diagram for a elaborate Use-Case(report)