Project report

Client/Server System

Rental Car Company

Dementie Bors, 279948

Justinas Jancys, 280151

Nicoleta Sova, 267069

Supervisors: Steffen Vissing Andersen

Ole Ildsgaard Hougaard

VIA University College Horsens

28347 characters

Software engineering

2nd semester

2019

Table of content

Abstract

In daily life people are appealing to rent a car companies to do their things quickly but it is often used to travel. Thus, that system was built for clients who want to book a car. Depending on the area, client has to fill in the credentials and system will show which cars are available at that day. This system was written in Java programming language & SQL query language, which means that all data will be stored in a database, that means security for data, accessibilities. The whole system is maintained by manager, which could add, delete, edit cars from the database. The result of the program did not fulfill all requirements, because there were links with banks, however none of these requirements were vital for program’s functionality.

1 Introduction

Europcar (Europcar, 2019) is a car rental company that offers a large spectrum of cars and extra services. It has been active for about 2 years in this industry. The company provides cars for people who want to rent it for some time. They are used to doing this in an office. If a customer desires to acquire a car for some time, then the customer has to go to Europcar office, fill out a form: identity, driver’s license, give a time period for which they want to rent a car, choose a car and only then they get the car. This takes time and is an old way of doing it. furthermore, this is a problem not just for the company but also for the clients. They have to come right to the office and bring all the documents with them, which mostly causes problems because clients forget something and at the end they don’t come back and choose other rental company.

Europcar wants to grow and get to the next level in order to get more clients and make their way of working easier. That is why Europcar asked us to create a program that would make their lives easier for them and for their clients.

Europcar is used to doing everything on paper. It is not as effective as doing everything online. Europcar wants a product that would be more efficient and quicker - thus making their customers lives easier. Europcar want an online version, so that every client could have the possibility to rent a car from the comfort of their home.

Working on this project will help the group to grow at a new level and asses the knowledge into practice. Also it will help us to understand how a team works in real-life situation. The company chooses us in order to find and solve the problem that they have now.

2 Analysis

In the following requirements to the functional requirements it is described what had to be done until deadline. Each requirement is categorized by priority (Critical, High, Medium, Mid-low, Low).

Requirements

2.1 Functional Requirements

ID	Priority	Estimate	Item
1	Critical	20 h	As a user, I want to be able to rent a car in order to drive.
2	High	15 h	As a user, I want to be able to select a certain type of vehicle in order to find a vehicle that suits my needs.
3	Medium	10 h	As a user, I want to be able to select additional services in order to ease my journey.
5	High	20 h	As a user, I want to be able to select the number of days I want to rent the vehicle in order to reserve the vehicle.
6	Mid low	10 h	As a user, I want to be able to see the pricing of different vehicles in order to select a vehicle inside my budget.
7	Medium	5 h	As a user, I want to be able to cancel my reservation in order to not rent a vehicle.
8	Low	15 h	As a user, I want to be notified if changes occur to my reservation, in order to be able to change my reservation
9	High	25 h	As a user, I want to receive a booking confirmation by email, in order to have proof of reservation.
10	Low	20 h	As an administrator, I want to be able to select the prices of different cars, in order to control prices.
11	High	15 h	As an administrator, I want to be able to access my customers credentials in order to be able to contact them.
12	High	10 h	As an administrator, I want to be able to see what vehicles are currently rented out, in order to see what is available.
13	Mid low	10 h	As an administrator, I want to be able to select extra services provided, in order to ease the journey of the customer.

14	Medium	10 h	As a user, I want to be able to add money to my wallet

2.2 Non-Functional Requirements

4	Low	10	As a user, I want the system to save my credentials in order to make it easier next time renting a vehicle.

2.3 Use case diagram

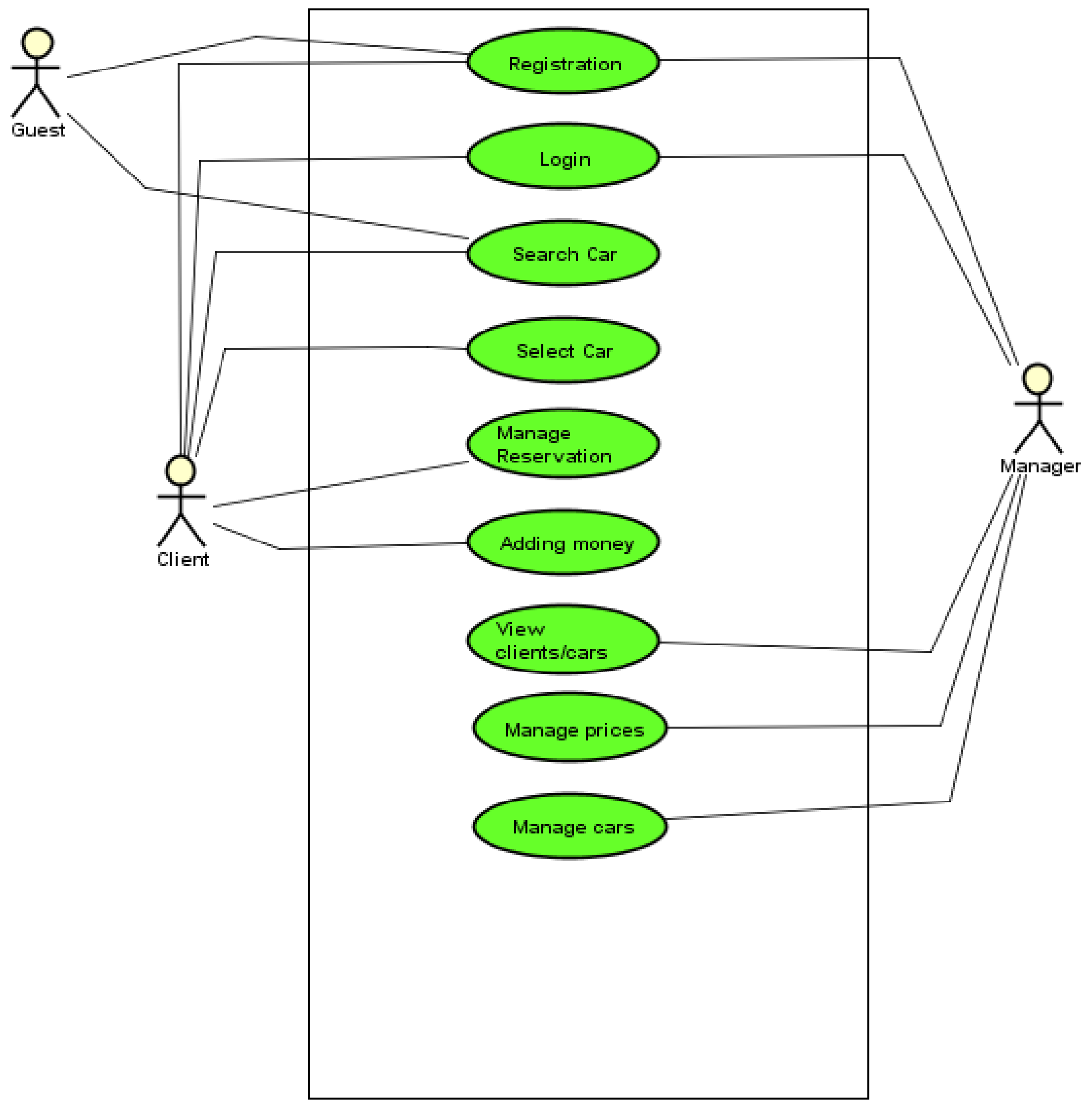


Figure 1 Use case diagram

2.4 Use case descriptions

Item	Value
Use case	Registration
Summary	New users can register in the system and then the users will be able to book a car.
Actor	Client Guest Manager
Precondition	
Postcondition	The user is added to the database
Base sequence	1. The user needs to fill in the credentials (First name, last name, date of birth, Email, confirm Email, password, confirm password, Registry address, ZIP code and phone number). 2.Click Create Account.
Branch sequence	* At any time during step, 1 and 2 clients can cancel the registration and goes back to login window. 1.a. If the user didn't fill out all the credential will appear an error with text (All fields must be filled out). 1.b. If the user inserted the wrong form of email system will show an error with text (Invalid email address). 1.b.a. If the user inserted the different email the system will show an error with text (Emails do not match). 1.c. If the user inserted the different password the system will show an error with text (Passwords do not match).
Exception sequence	None
Sub use case	
Note	None

Item	Value
Use case	Login
Summary	The User can login into the system if he has an account already.
Actor	Client Manager
Precondition	The client must be on the system.
Postcondition	The user was logged in the system
Base sequence	1.Insert the email. 2.Insert the password. 3.Click login
Branch sequence	3.a. If the user will click login without email and password, the system will show an error with text (Username or password is empty). 3.b. If the user will use the wrong email or password the system will show an error with text (Username or password is incorrect).
Exception sequence	If server was not started the system will not work in any cases.
Sub use case	
Note	None

Item	**Value**
Use case	Search Car
Summary	The Client searches for a certain type of vehicle in order to find a vehicle that suits his/her needs
Actor	Client Guest
Precondition	The client must be on the system.
Postcondition	A list of available vehicles is shown
Base sequence	1.The client clicks "Rent a car" tab. 2.Client chooses the city from a dropdown button. 3.Client chooses the wanted type of fuel. 4.Client chooses the pick-up date from calendar. 5.Client inserts the return date. 6.Client selects the check box ‘older than 23 y.o’. 7.Client press the "Search" button. 8. The system shows the available cars based on search and "Back" button. 9.The client choose between the available cars.
Branch sequence	* At any time during step, 2-6 client can cancel the search and goes to login view. 5. If the client doesn't choose a type of fuel the system shows all available cars and ignores fuel types 13. a. When "Back" button is clicked the system brings you to step (2).
Exception sequence	If server was not started the system will not work in any cases.
Sub use case	
Note	As a user, I want to be able to select a certain type of vehicle in order to find a vehicle that suits my needs.

Item	**Value**
Use case	Select Car
Summary	The user can select a car or car/s
Actor	Client
Precondition	The client must be logged in
Postcondition	The car is added to the database
Base sequence	1.The client select which car to book then clicks Rent. 2.The system will show a pop-up that was sent an email. 3.The system will asks if the client wants extra services. 4. The user can select which extra services wants then clicks Add. 5. The system will go in search car window.
Branch sequence	3.a. The user can click Yes then step (4). 3.b. The user can click No then step (5).
Exception sequence	None
Sub use case	
Note	None

Item	Value
Use case	Manage Reservation
Summary	The client can delete reservation
Actor	Client
Precondition	The client must be logged into the system.
Postcondition	Rent will be deleted from the database
Base sequence	1.The client needs to go in My Cars tab. 2.The client can delete a reservation. 3. The system will show a pop up that the system sent an email regarding the changes of reservation.
Branch sequence	2.a To delete a car user should click on the button 2 times. 3.a. The user has an option just to press Ok and the system will keep the client with updated table on My cars window.
Exception sequence	If server was not started the system will not work in any cases.
Sub use case	
Note	

Item	Value
Use case	Adding Money
Summary	The user can add money in their wallet.
Actor	Client
Precondition	The client must be logged in the system.
Postcondition	Money is added in the database.
Base sequence	1. The client can see the current balance from first page after login. 2.Press '+' to add money. 3. A text field and a button appear. 4. Enter a sum to the text field. 5.Press 'Add' button
Branch sequence	If server was not started the system will not work in any cases.
Exception sequence	None
Sub use case	
Note	None

Item	Value
Use case	View clients/cars
Summary	View which cars are available, which customers has ever booked a car.
Actor	Manager
Precondition	The client must be logged in the system.
Postcondition	
Base sequence	1.In "Available" cars tab, "Booked cars" tab, "Booked tab" the manager can sort all columns from the table and change order of columns.
Branch sequence	None
Exception sequence	If server was not started the system will not work in any cases.
Sub use case	
Note	None

Item	Value
Use case	Manage cars
Summary	The administrator can add, edit daily price and city for cars
Actor	Manager
Precondition	The manager must be logged in the system.
Postcondition	The changes are saved in the database.
Base sequence	Edit car: 1. In "Edit Cars" tab the manager can select which car to edit by selecting the radio button next to the car. 2. Press 'edit' button 3. Replace the old values with new. 4. Press save. Add a car: 1. The manager can add a car pressing the "Add car" button, the system will show a new window. 2. The manager needs to fill in the text fields. 3. Click "Save" and the system will store the data in database and returns to the administrator view.
Branch sequence	Edit car: 4.a. If the text fields will be empty the system will show an error with text (All fields must be filled out.) Add a car: 4.a. If one of the text fields will be empty the system will show an error with text (All fields must be filled out.) 4.b. If all fields are filled out after pressing the "Add car" button the system will save the data in database.
Exception sequence	If server was not started the system will not work in any cases.
Sub use case	
Note	None

2.5 Link between requirements and use cases

Registration –

Login –

Search car – 1, 5

Select car – 1, 2, 3, 5, 6, 9, 12, 13

Manage reservations – 7, 8

Adding money – 14

View clients/car – 11

Manage cars – 10

3 Design

Domain model:

In the following diagram the domain model is shown. This diagram shows every table that will be implemented with every detail.

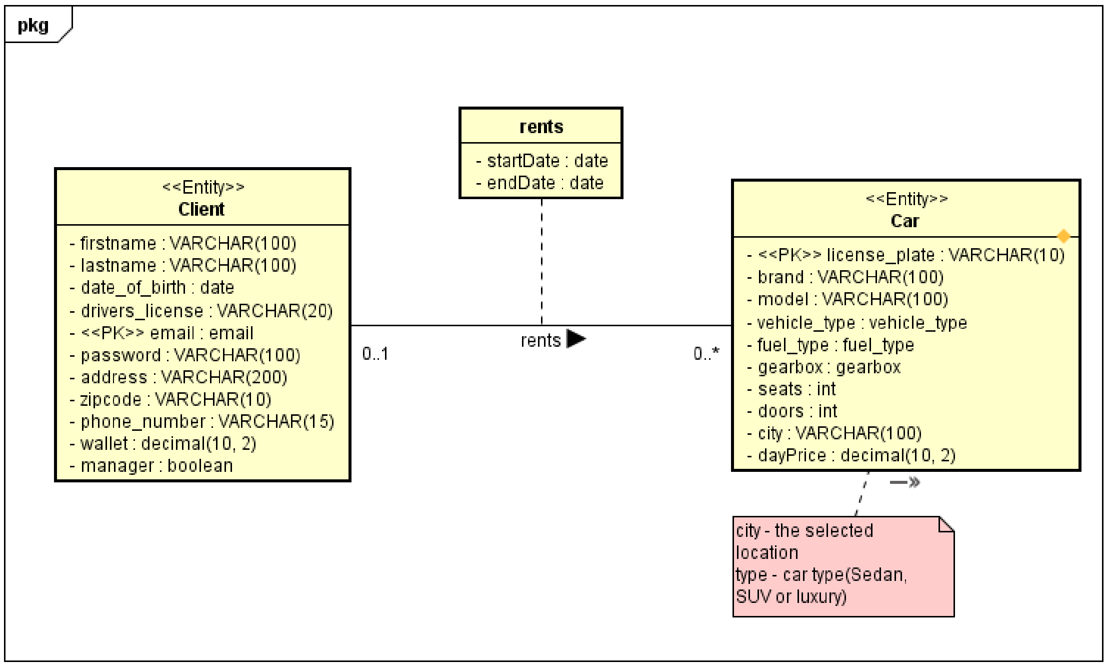


Figure 2 Domain model

UML diagram:

We used the MVVM(Model-View-ViewModel) pattern in making the UML diagram.

To see the full UML diagram look at appendix B.

Sequence diagram:

The sequence diagram below shows how the renting of a car works.

So first the client selects the city, the dates the user wants to rent a car, the type of fuel (if needed) and that the client is 23 and older. Then the system retrieves the available cars with the met specifications. The client then selects the vehicles that meet his/her demands. The cars are booked in the database and a notification is shown. Then the system asks if the client wants any extra services. If the client presses no, then the system goes back to the search car view. If the client presses yes, then the extra services are shown for the client to choose from. After choosing the extra services it goes back to the search car view.

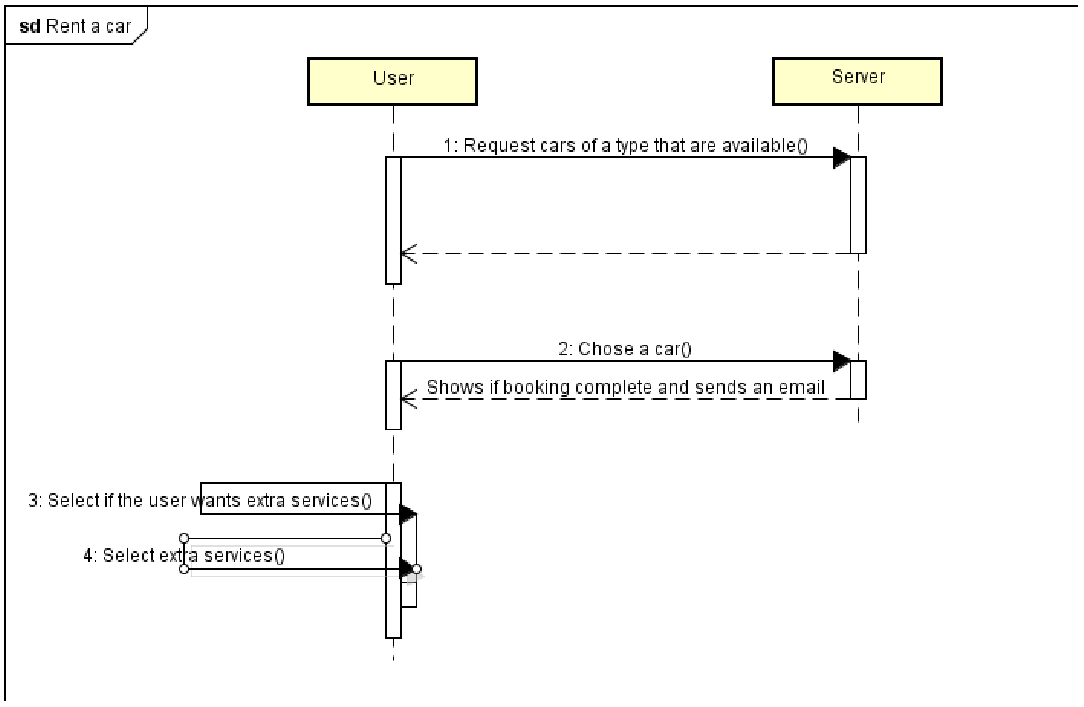

sd Rent a car
User
Server
1: Request cars of a type that are available()
2: Chose a car()
Shows if booking complete and sends an email
3: Select if the user wants extra services()
4: Select extra services()

Figure 3 Sequence diagram

Activity diagram:

Rent a car:

The activity diagram shown below is for the renting of a car. When a client logs in and wants to rent some cars. These are the steps that he takes:

First things first, the client selects the city that he wants to pick up the car. After that the client sets the dates for when the car should be booked, selects the fuel type if needed and the client must select the 23 years old or older check box. After that the system retrieves the available cars from the database. If none of the cars suits the needs of the client, then the user can press the back button and go back to the search view. To book some cars the client selects the check boxes of the cars that are correct, and presses rent. When the booking is complete a notification is shown that an email is sent to the client.

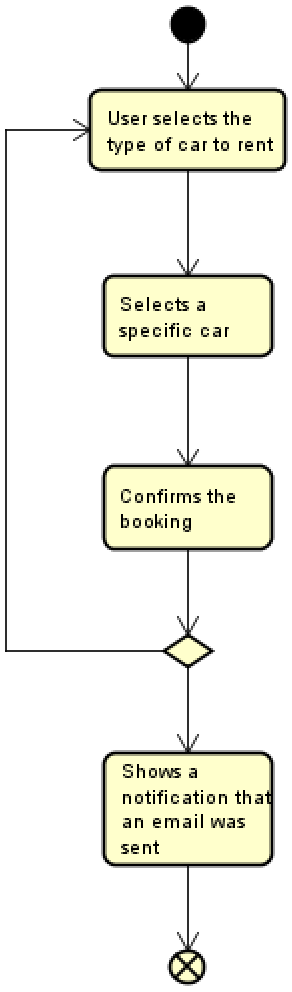

User selects the type of car to rent
Selects a specific car
Confirms the booking
Shows a notification that an email was sent

Figure 4 Activity diagram

Design patterns

RMI pattern:

Java Remote Method Invocation (Java RMI) is a Java API that performs remote method invocation. The server create a stub and sends it to the client to work on. That way the client can call method on the server object remotely.

MVVM pattern:

The project is based on the MVVM pattern. Every .fxml file has its own controller (the view) where everything happens related to the view, for example: button pressed, switch views, change the size of an element, etc. Everything for the view that is related with the data and calculation is calculated in the view model. The model is the main part of the program. It stores everything in it, converts from one type to others, etc.

4 Implementation

Login

The login class is the first step to access our program.

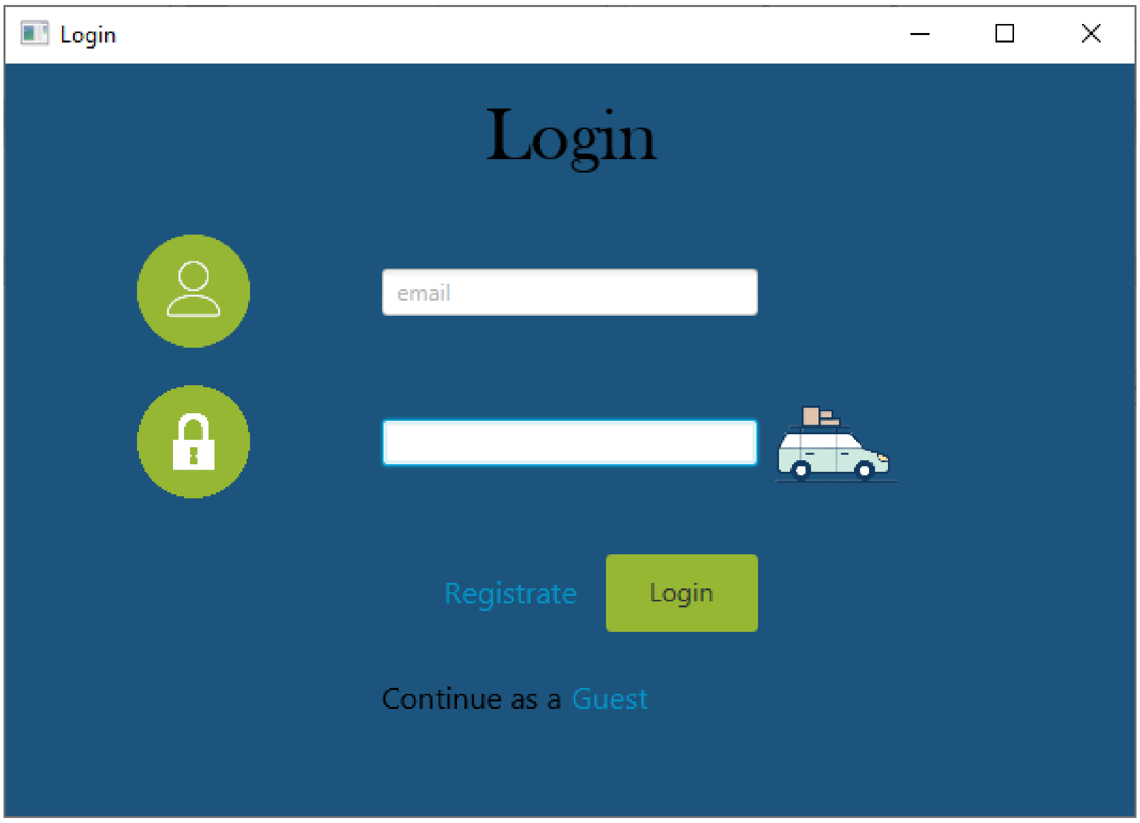
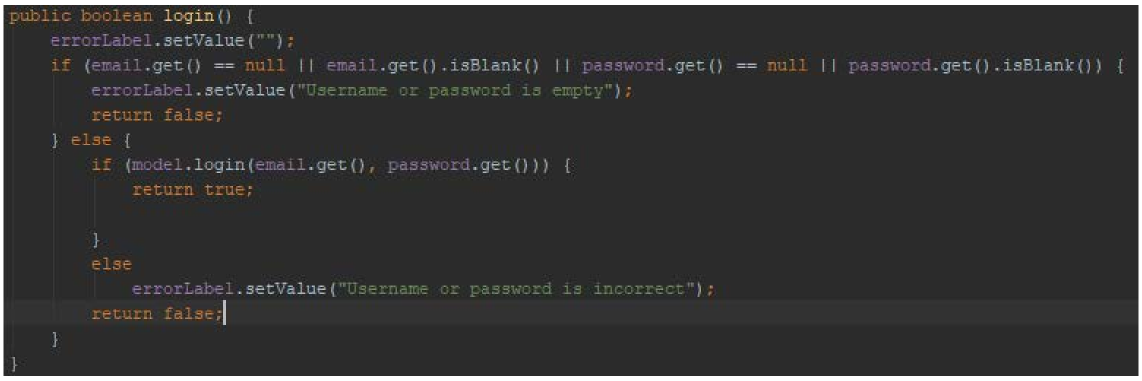


Figure 5 Login view

The system check if the text fields are filled out.



```
public boolean login() {
    errorLabel.setValue("");
    if (email.get() == null || email.get().isBlank() || password.get() == null || password.get().isBlank()) {
        errorLabel.setValue("Username or password is empty");
        return false;
    } else {
        if (model.login(email.get(), password.get())) {
            return true;

        }
        else
            errorLabel.setValue("Username or password is incorrect");
        return false;
    }
}
```

Figure 6 Login code

If they are filled out the email and the password are sent to the database, where all of the information about the user is retrieved.

The user is logged in and the search car view is opened.

Registration

To be able to login to the system, a user must register first.

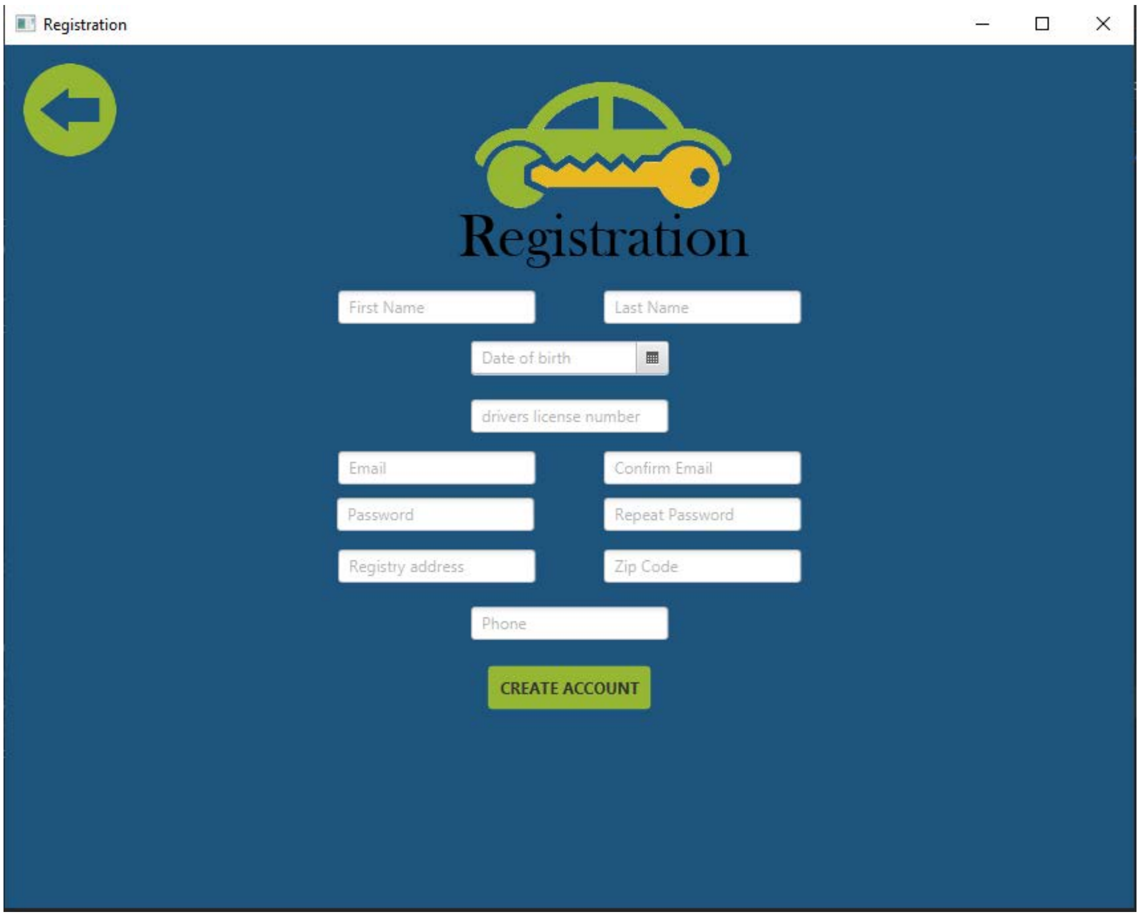
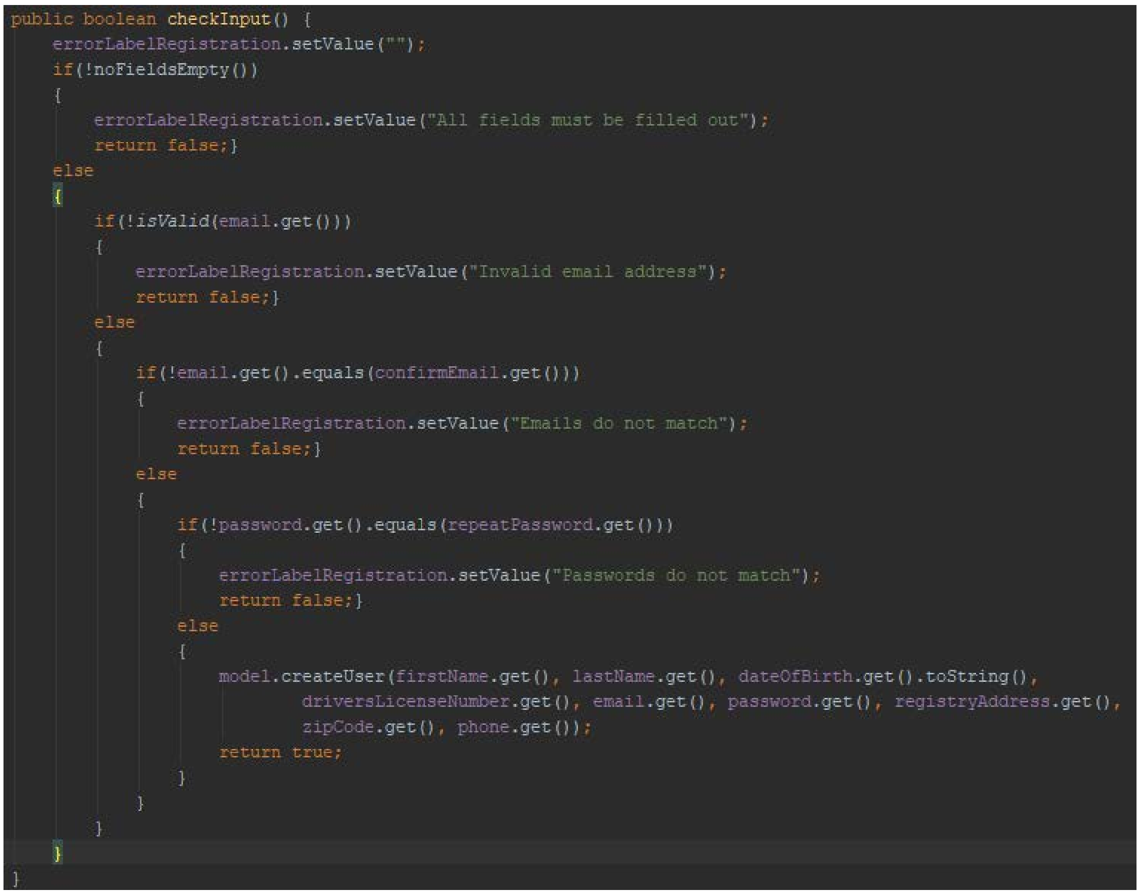


Figure 7 Registration view

The user has to fill every single text field in order to register to the system.



```java
public boolean checkInput() {
    errorLabelRegistration.setValue("");
    if(!noFieldsEmpty())
    {
        errorLabelRegistration.setValue("All fields must be filled out");
        return false;}
    else
    {
        if(!isValid(email.get()))
        {
            errorLabelRegistration.setValue("Invalid email address");
            return false;}
        else
        {
            if(!email.get().equals(confirmEmail.get()))
            {
                errorLabelRegistration.setValue("Emails do not match");
                return false;}
            else
            {
                if(!password.get().equals(repeatPassword.get()))
                {
                    errorLabelRegistration.setValue("Passwords do not match");
                    return false;}
                else
                {
                    model.createUser(firstName.get(), lastName.get(), dateOfBirth.get().toString(),
                            driversLicenseNumber.get(), email.get(), password.get(), registryAddress.get(),
                            zipCode.get(), phone.get());
                    return true;
                }
            }
        }
    }
}
```

Figure 8 Registration validation code

Once the user registers, the login window is opened.

Search car as a guest

In order to just get the list of available cars, the user does not have to log in.

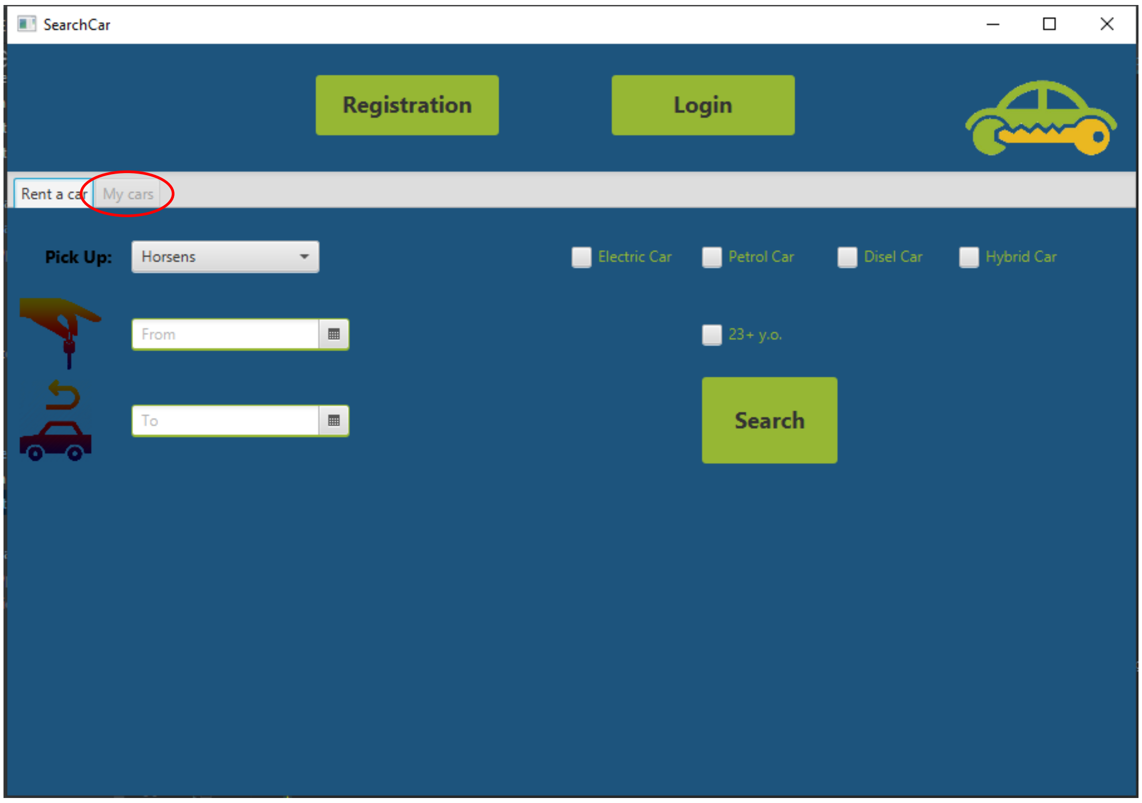
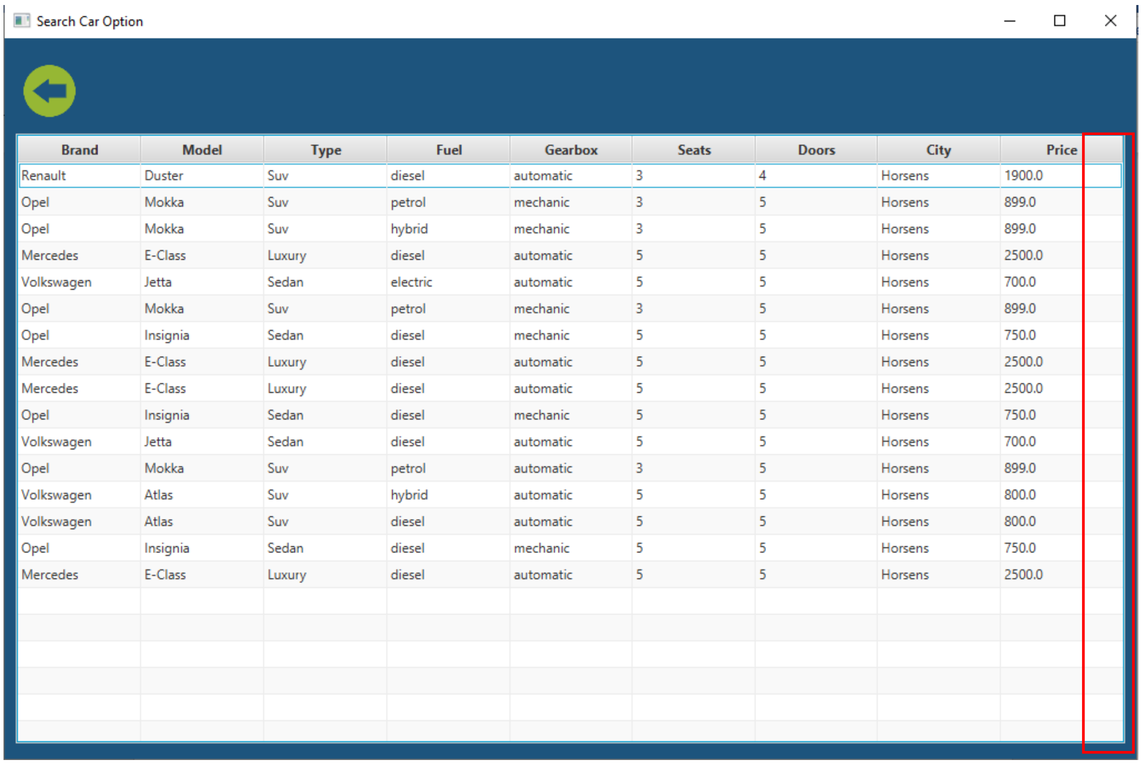


Figure 9 Guest search car view

As a guest, the user can use a minimized version of the system. They can search for the available cars, but they can not rent any of the vehicles. Also, the tab with all rented cars is disables.



Brand	Model	Type	Fuel	Gearbox	Seats	Doors	City	Price
Renault	Duster	Suv	diesel	automatic	3	4	Horsens	1900.0
Opel	Mokka	Suv	petrol	mechanic	3	5	Horsens	899.0
Opel	Mokka	Suv	hybrid	mechanic	3	5	Horsens	899.0
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Horsens	2500.0
Volkswagen	Jetta	Sedan	electric	automatic	5	5	Horsens	700.0
Opel	Mokka	Suv	petrol	mechanic	3	5	Horsens	899.0
Opel	Insignia	Sedan	diesel	mechanic	5	5	Horsens	750.0
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Horsens	2500.0
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Horsens	2500.0
Opel	Insignia	Sedan	diesel	mechanic	5	5	Horsens	750.0
Volkswagen	Jetta	Sedan	diesel	automatic	5	5	Horsens	700.0
Opel	Mokka	Suv	petrol	automatic	3	5	Horsens	899.0
Volkswagen	Atlas	Suv	hybrid	automatic	5	5	Horsens	800.0
Volkswagen	Atlas	Suv	diesel	automatic	5	5	Horsens	800.0
Opel	Insignia	Sedan	diesel	mechanic	5	5	Horsens	750.0
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Horsens	2500.0

Figure 10 Guest searching for a car

Search car view

Once the user is logged in, they can search for car and look at their future reservations.

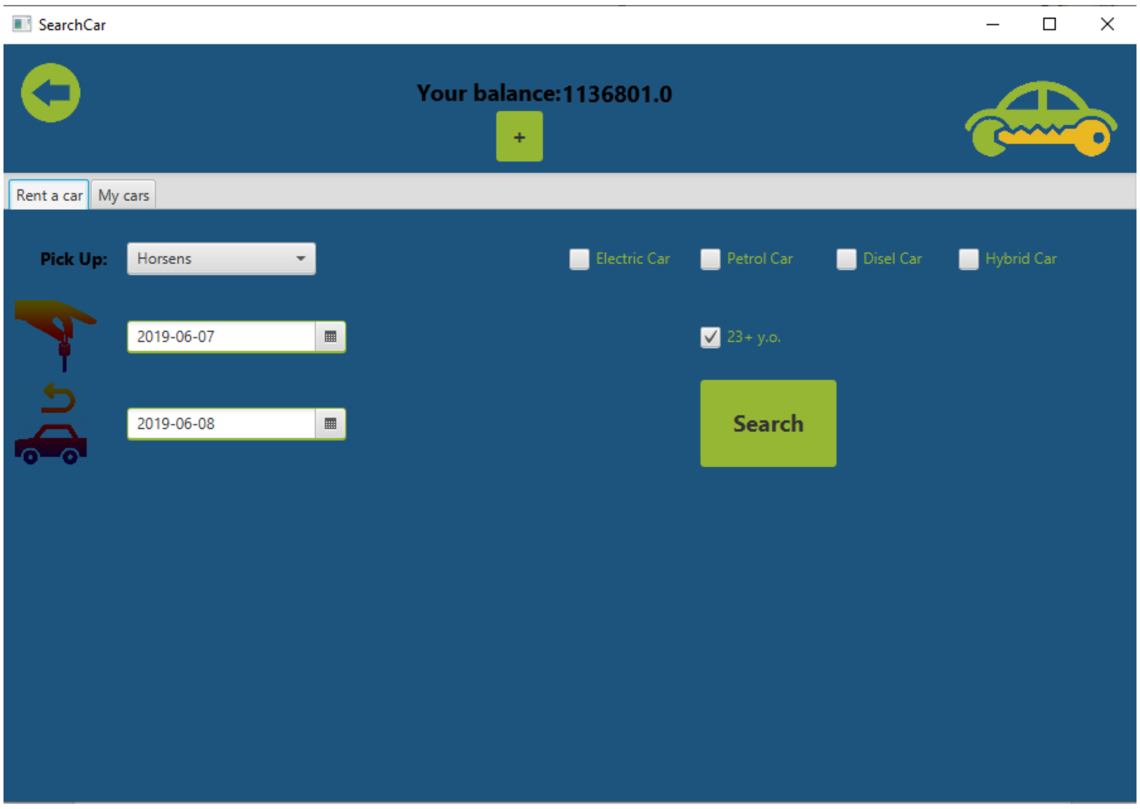
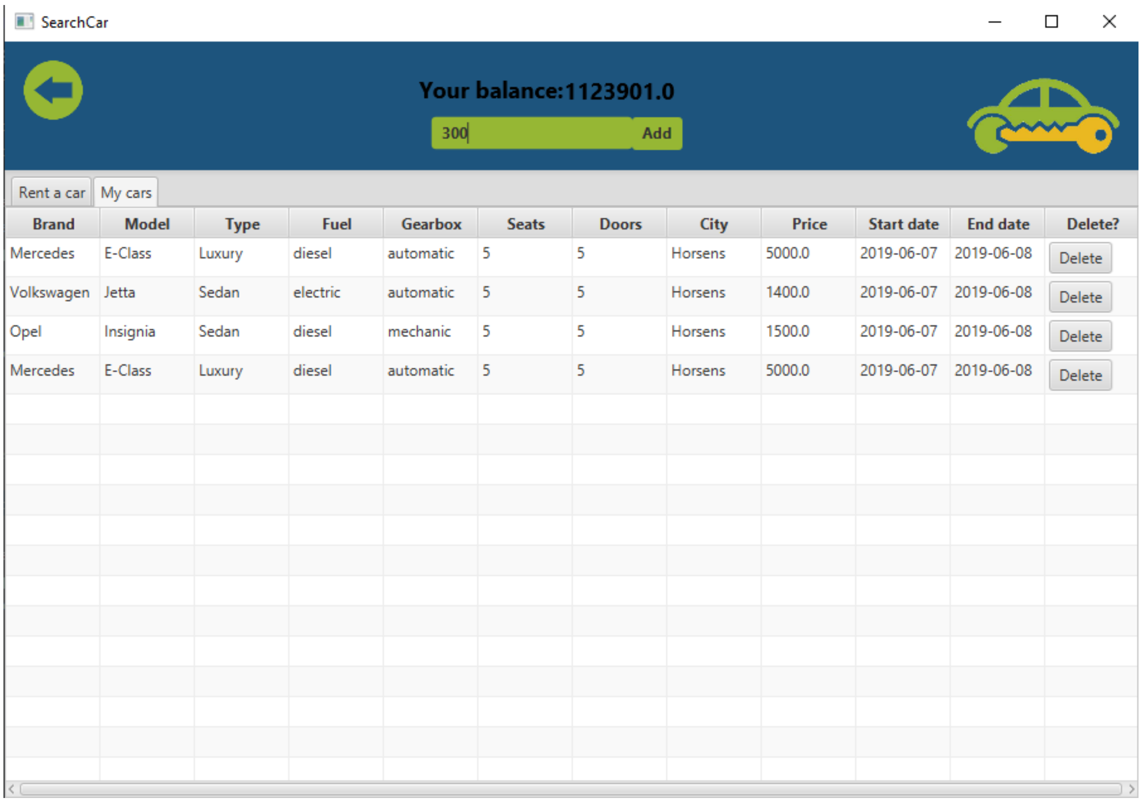


Figure 11 User search car view

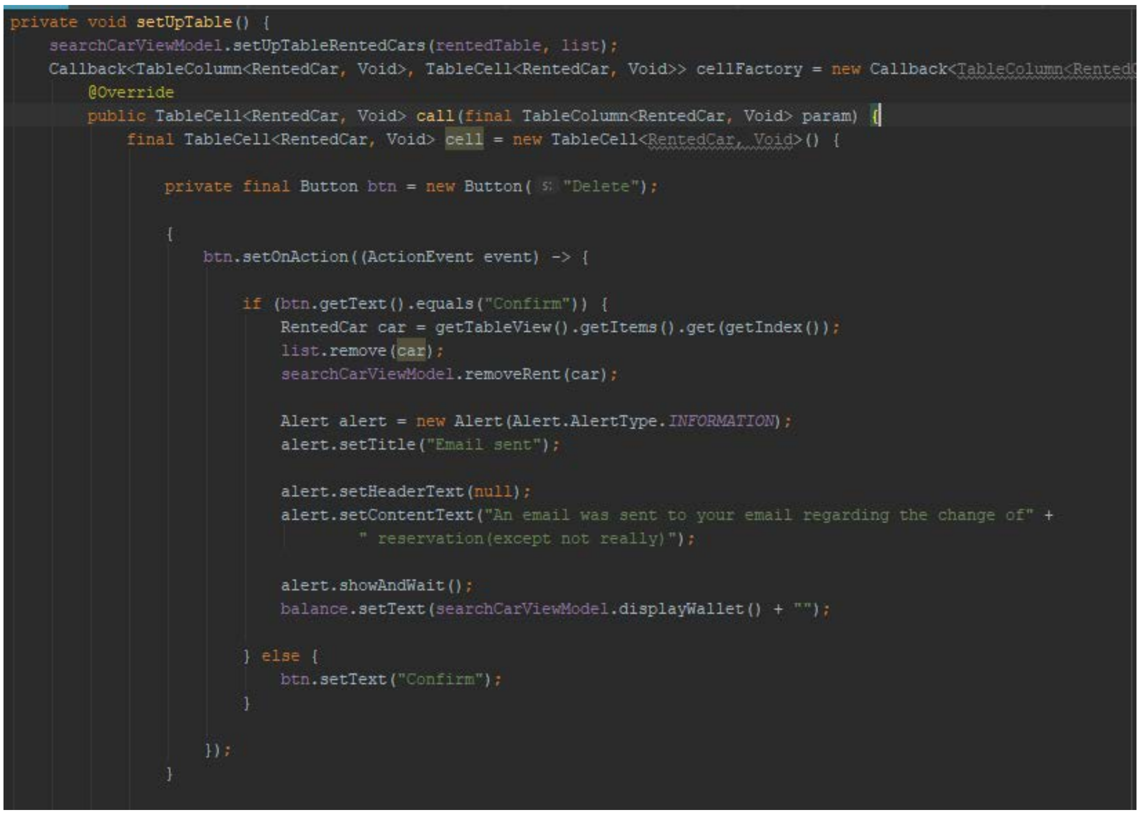
From this window the user can search for available cars, view their reservations and add money to their wallet.



Brand	Model	Type	Fuel	Gearbox	Seats	Doors	City	Price	Start date	End date	Delete?
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Horsens	5000.0	2019-06-07	2019-06-08	Delete
Volkswagen	Jetta	Sedan	electric	automatic	5	5	Horsens	1400.0	2019-06-07	2019-06-08	Delete
Opel	Insignia	Sedan	diesel	mechanic	5	5	Horsens	1500.0	2019-06-07	2019-06-08	Delete
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Horsens	5000.0	2019-06-07	2019-06-08	Delete

Figure 12 Users rented cars

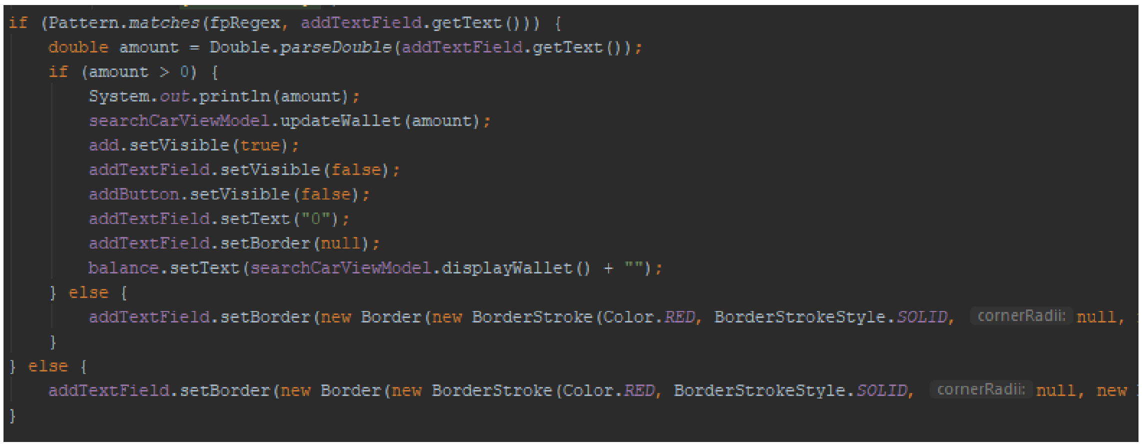
To cancel a reservation, the client needs to press the delete button twice.



```java
private void setUpTable() {
    searchCarViewModel.setUpTableRentedCars(rentedTable, list);
    Callback<TableColumn<RentedCar, Void>, TableCell<RentedCar, Void>> cellFactory = new Callback<TableColumn<RentedC
        @Override
        public TableCell<RentedCar, Void> call(final TableColumn<RentedCar, Void> param) {
            final TableCell<RentedCar, Void> cell = new TableCell<RentedCar, Void>() {

                private final Button btn = new Button( s: "Delete");

                {
                    btn.setOnAction((ActionEvent event) -> {

                        if (btn.getText().equals("Confirm")) {
                            RentedCar car = getTableView().getItems().get(getIndex());
                            list.remove(car);
                            searchCarViewModel.removeRent(car);

                            Alert alert = new Alert(Alert.AlertType.INFORMATION);
                            alert.setTitle("Email sent");

                            alert.setHeaderText(null);
                            alert.setContentText("An email was sent to your email regarding the change of" +
                                    " reservation(except not really)");

                            alert.showAndWait();
                            balance.setText(searchCarViewModel.displayWallet() + "");

                        } else {
                            btn.setText("Confirm");
                        }

                    });
                }
```

Figure 13 Users rented cars code

To add money to the wallet the client needs to write a number to add. If anything besides numbers is written, then the text field is highlighted red.

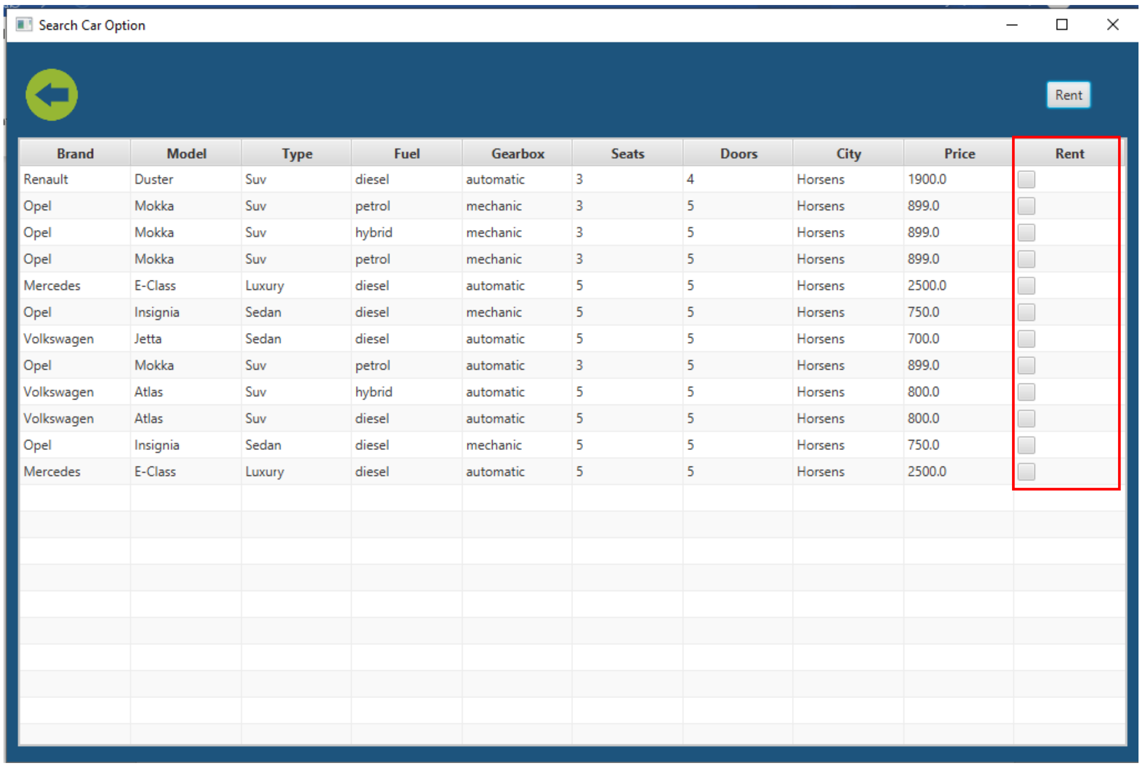


```
if (Pattern.matches(fpRegex, addTextField.getText())) {
    double amount = Double.parseDouble(addTextField.getText());
    if (amount > 0) {
        System.out.println(amount);
        searchCarViewModel.updateWallet(amount);
        add.setVisible(true);
        addTextField.setVisible(false);
        addButton.setVisible(false);
        addTextField.setText("0");
        addTextField.setBorder(null);
        balance.setText(searchCarViewModel.displayWallet() + "");
    } else {
        addTextField.setBorder(new Border(new BorderStroke(Color.RED, BorderStrokeStyle.SOLID,  cornerRadii: null,
    }
} else {
    addTextField.setBorder(new Border(new BorderStroke(Color.RED, BorderStrokeStyle.SOLID,  cornerRadii: null, new
}
```

Figure 14 User add money code

Select car/cars

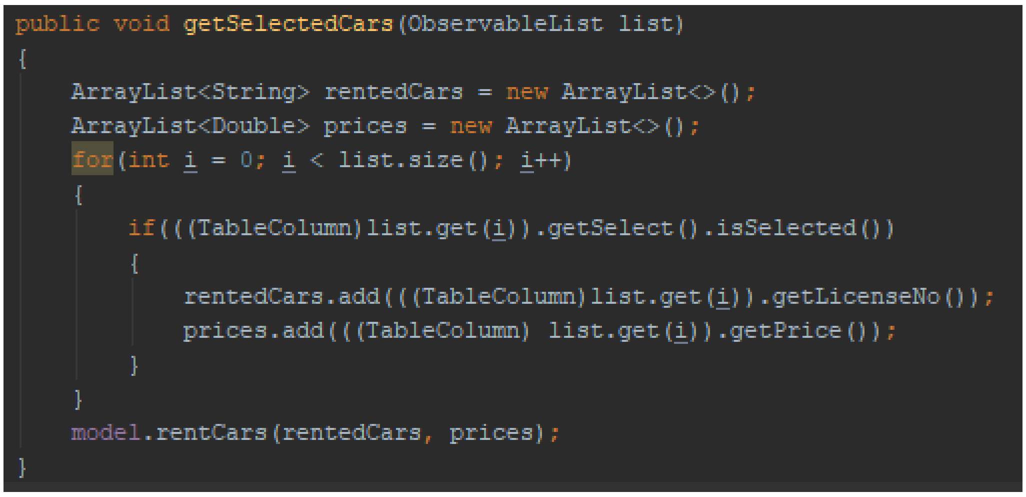
After searching for a car, the view with all of the available cars is shown.



Brand	Model	Type	Fuel	Gearbox	Seats	Doors	City	Price	Rent
Renault	Duster	Suv	diesel	automatic	3	4	Horsens	1900.0	☐
Opel	Mokka	Suv	petrol	mechanic	3	5	Horsens	899.0	☐
Opel	Mokka	Suv	hybrid	mechanic	3	5	Horsens	899.0	☐
Opel	Mokka	Suv	petrol	mechanic	3	5	Horsens	899.0	☐
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Horsens	2500.0	☐
Opel	Insignia	Sedan	diesel	mechanic	5	5	Horsens	750.0	☐
Volkswagen	Jetta	Sedan	diesel	automatic	5	5	Horsens	700.0	☐
Opel	Mokka	Suv	petrol	automatic	3	5	Horsens	899.0	☐
Volkswagen	Atlas	Suv	hybrid	automatic	5	5	Horsens	800.0	☐
Volkswagen	Atlas	Suv	diesel	automatic	5	5	Horsens	800.0	☐
Opel	Insignia	Sedan	diesel	mechanic	5	5	Horsens	750.0	☐
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Horsens	2500.0	☐

Figure 15 User car renting

The difference between the guest and the logged in user is that the logged in user is able to rent from one to many cars.



```java
public void getSelectedCars(ObservableList list)
{
    ArrayList<String> rentedCars = new ArrayList<>();
    ArrayList<Double> prices = new ArrayList<>();
    for(int i = 0; i < list.size(); i++)
    {
        if(((TableColumn)list.get(i)).getSelect().isSelected())
        {
            rentedCars.add(((TableColumn)list.get(i)).getLicenseNo());
            prices.add(((TableColumn) list.get(i)).getPrice());
        }
    }
    model.rentCars(rentedCars, prices);
}
```

Figure 16 Renting car select code

When the user selects the cars that he wants to rent and presses the rent button extra service view is opened.

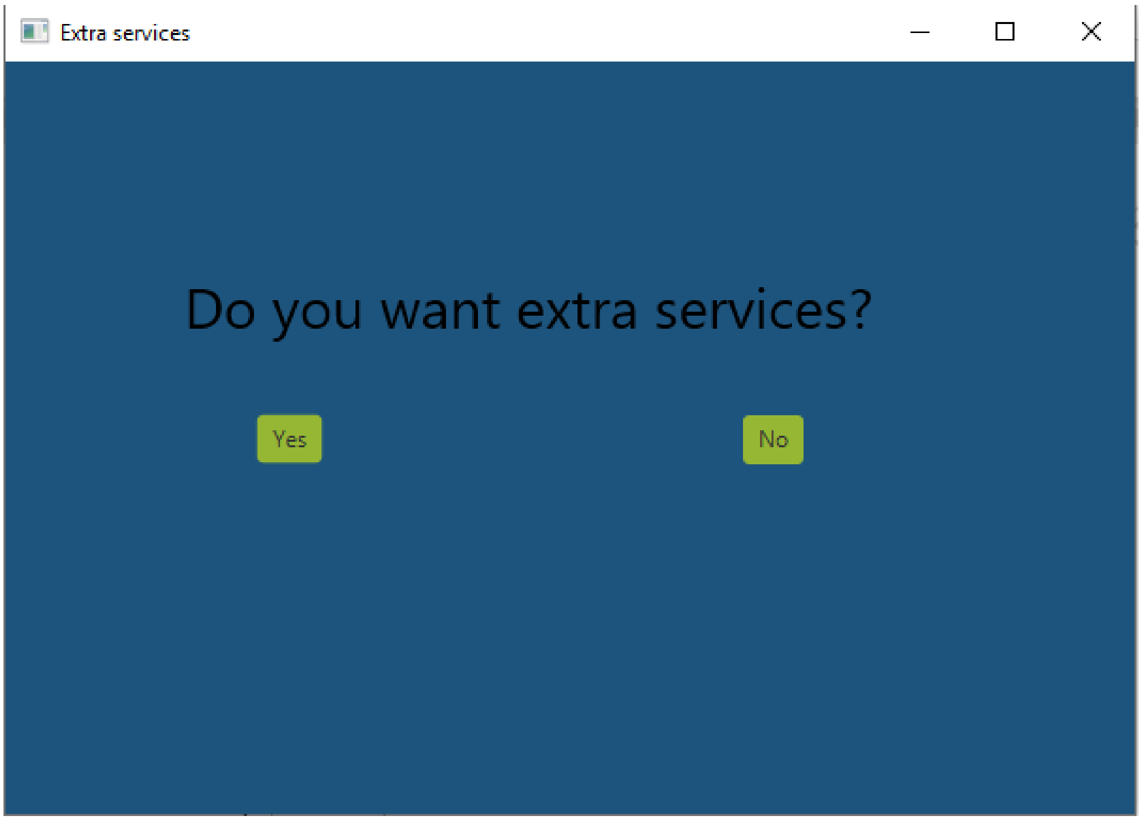


Figure 17 Extra services view

If the user presses no then it goes back to the search car view.

If user presses yes, then it proceeds to select extra services.

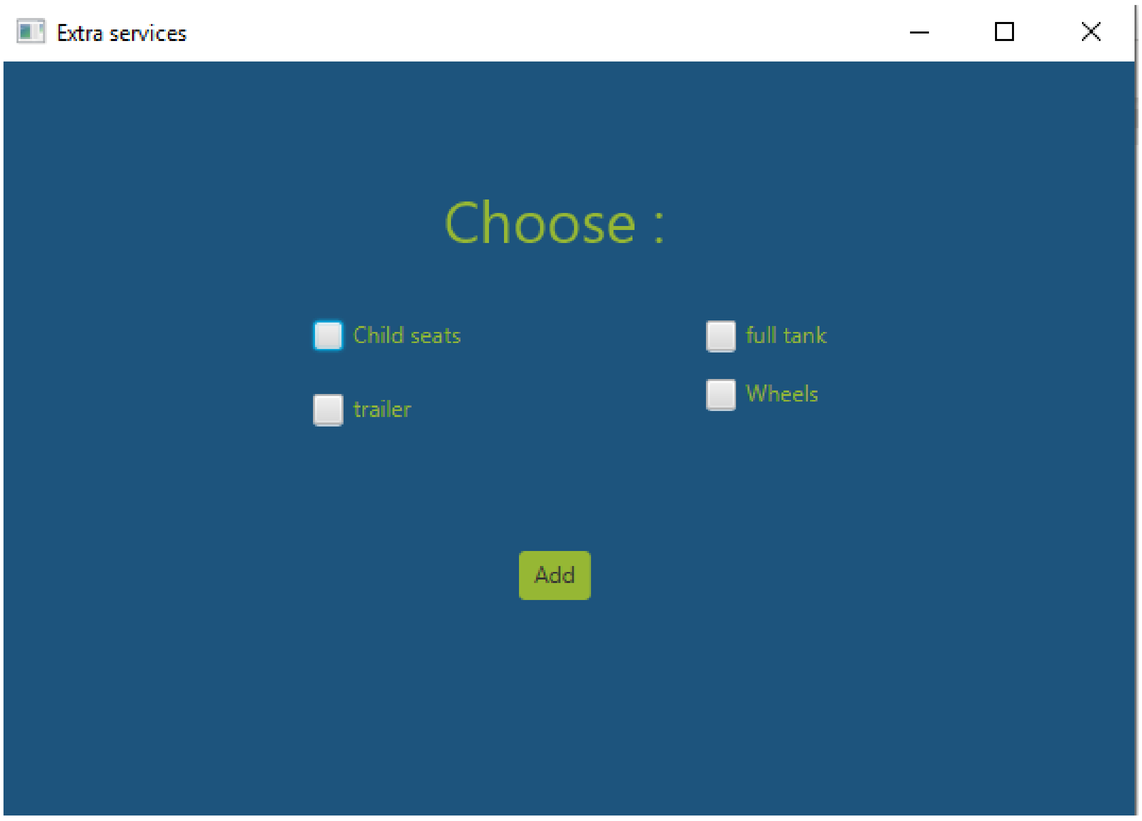
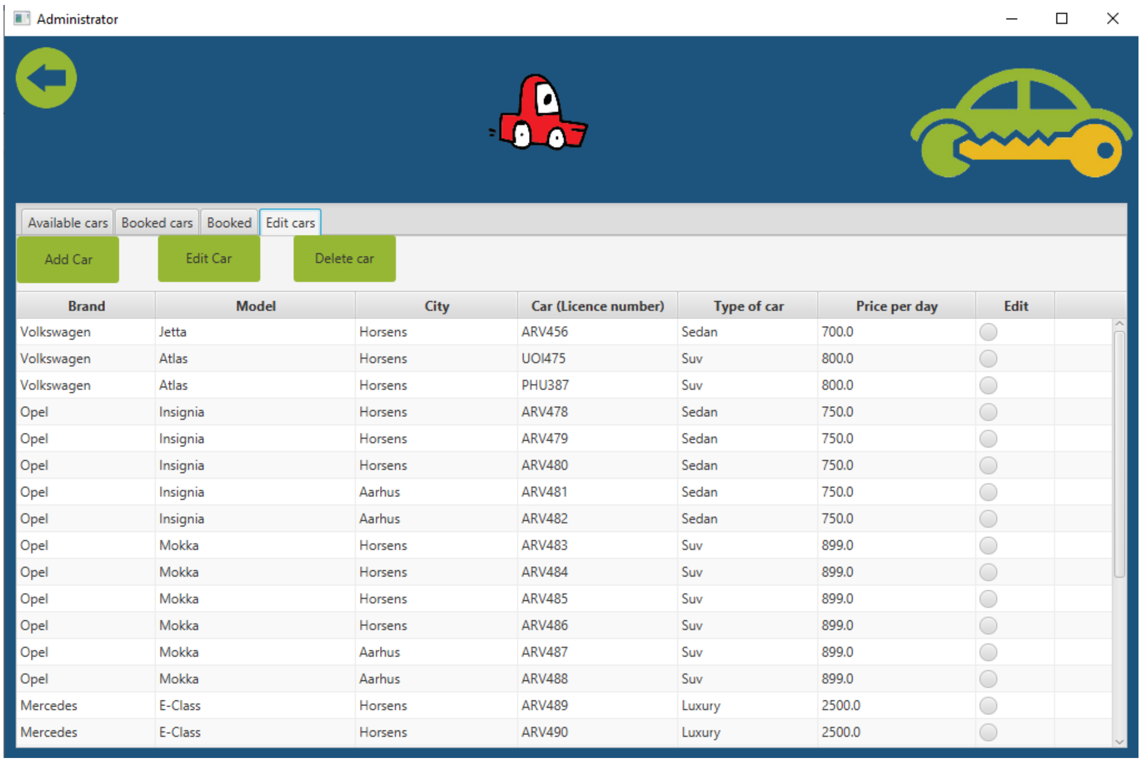


Figure 18 Extra services select services view

Administrator view

When the administrator logs in, the administrator view is opened.



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Volkswagen	Jetta	Horsens	ARV456	Sedan	700.0	○
Volkswagen	Atlas	Horsens	UOI475	Suv	800.0	○
Volkswagen	Atlas	Horsens	PHU387	Suv	800.0	○
Opel	Insignia	Horsens	ARV478	Sedan	750.0	○
Opel	Insignia	Horsens	ARV479	Sedan	750.0	○
Opel	Insignia	Horsens	ARV480	Sedan	750.0	○
Opel	Insignia	Aarhus	ARV481	Sedan	750.0	○
Opel	Insignia	Aarhus	ARV482	Sedan	750.0	○
Opel	Mokka	Horsens	ARV483	Suv	899.0	○
Opel	Mokka	Horsens	ARV484	Suv	899.0	○
Opel	Mokka	Horsens	ARV485	Suv	899.0	○
Opel	Mokka	Horsens	ARV486	Suv	899.0	○
Opel	Mokka	Aarhus	ARV487	Suv	899.0	○
Opel	Mokka	Aarhus	ARV488	Suv	899.0	○
Mercedes	E-Class	Horsens	ARV489	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV490	Luxury	2500.0	○

Figure 19 Administrator view

From that view the administrator can check which cars are available today(available today), which cars are booked today(booked cars), all of the clients that have ever booked a car(booked), and a tab with all of the cars(edit cars). In the edit cars tab the administrator can add a new car.

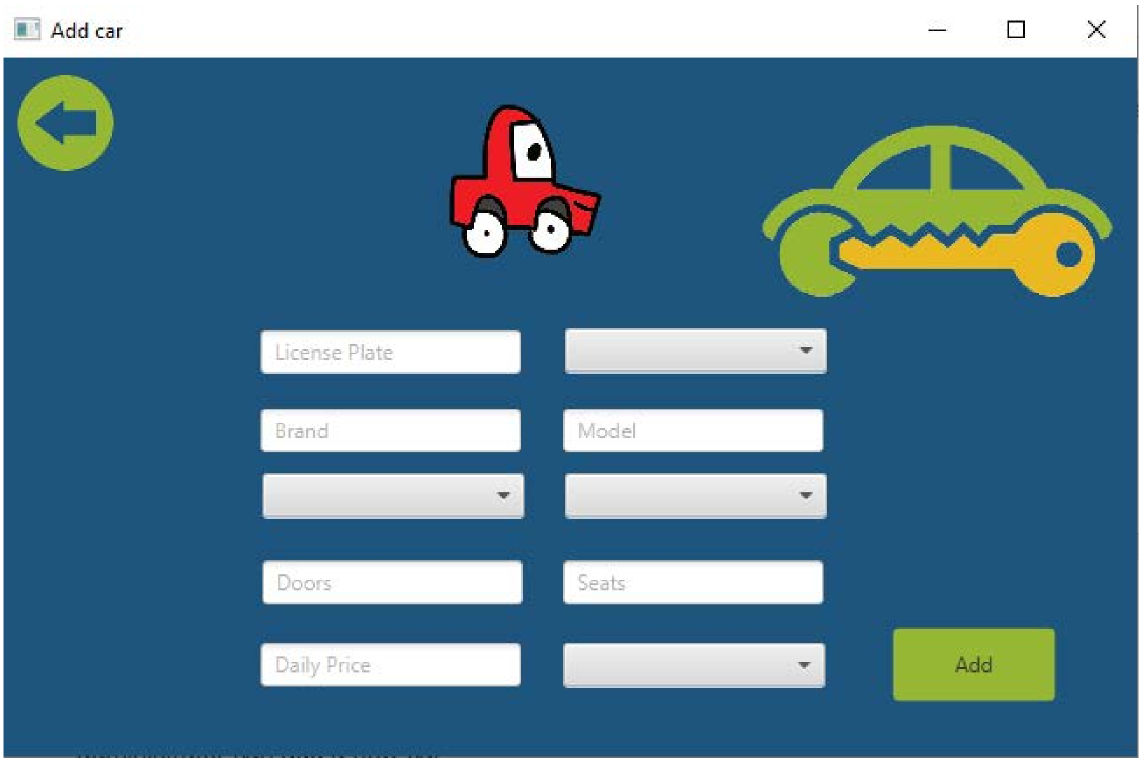


Figure 20 Administrator add new car view

When all of the fields are filled out and the add button is pressed the system creates a new car and adds it to the database.

Besides creating new cars, the administrators car delete cars by selecting them in the edit car tab.

Furthermore, the user can edit an existing cars price or the city it is located in.

To be able to edit a car the user has to select a car at first.

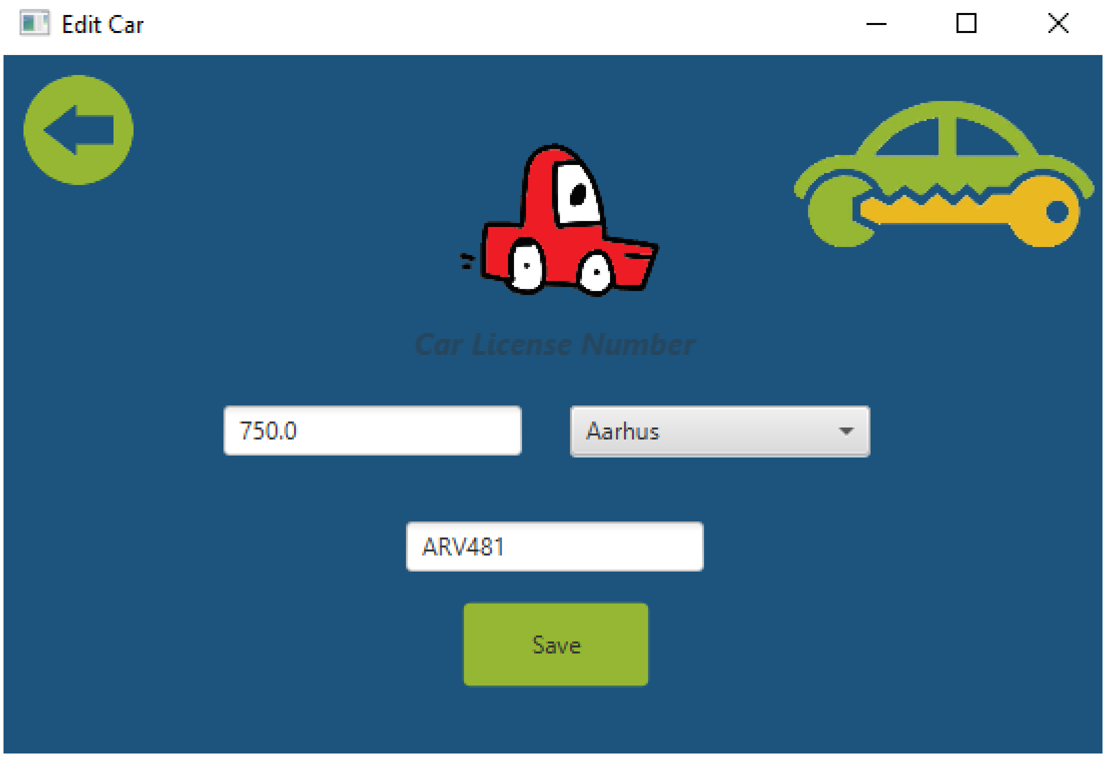


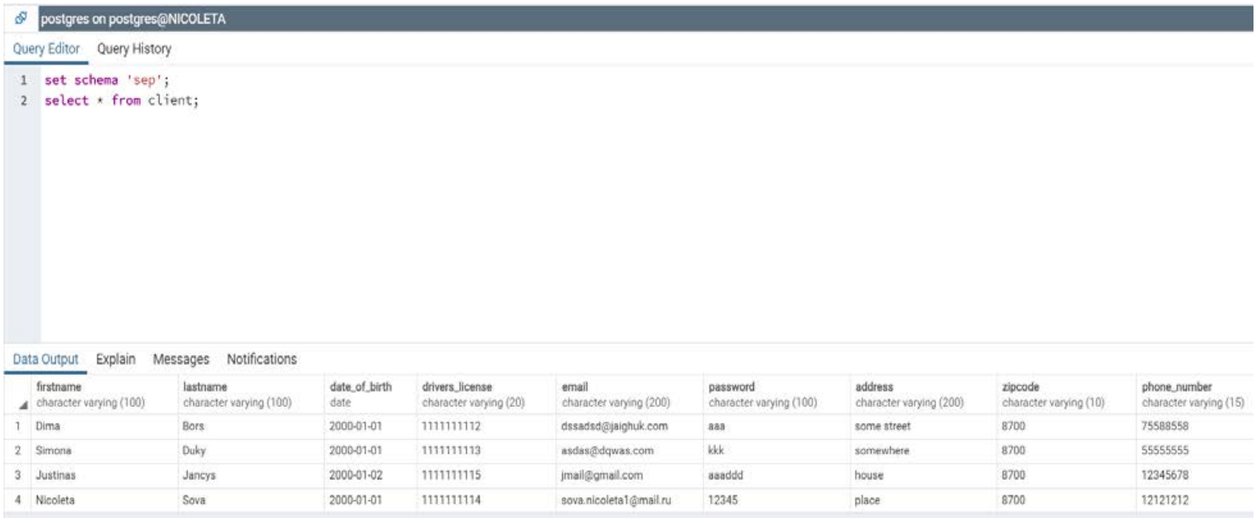
Figure 21 Administrator edit car

The window opens with old information is show. The price and the city can be changed. Once the save button is pressed, then the changes are saved to the database.

Testing

Login

There is database with all registered clients



```sql
set schema 'sep';
select * from client;
```

	firstname character varying (100)	lastname character varying (100)	date_of_birth date	drivers_license character varying (20)	email character varying (200)	password character varying (100)	address character varying (200)	zipcode character varying (10)	phone_number character varying (15)
1	Dima	Bors	2000-01-01	1111111112	dssadsd@jaighuk.com	aaa	some street	8700	75588558
2	Simona	Duky	2000-01-01	1111111113	asdas@dqwas.com	kkk	somewhere	8700	55555555
3	Justinas	Jancys	2000-01-02	1111111115	jmail@gmail.com	aaaddd	house	8700	12345678
4	Nicoleta	Sova	2000-01-01	1111111114	sova.nicoleta1@mail.ru	12345	place	8700	12121212

Figure 22 Database with all clients

In order to rent a car, we must be login. Here is an example of Login with correct email and password.

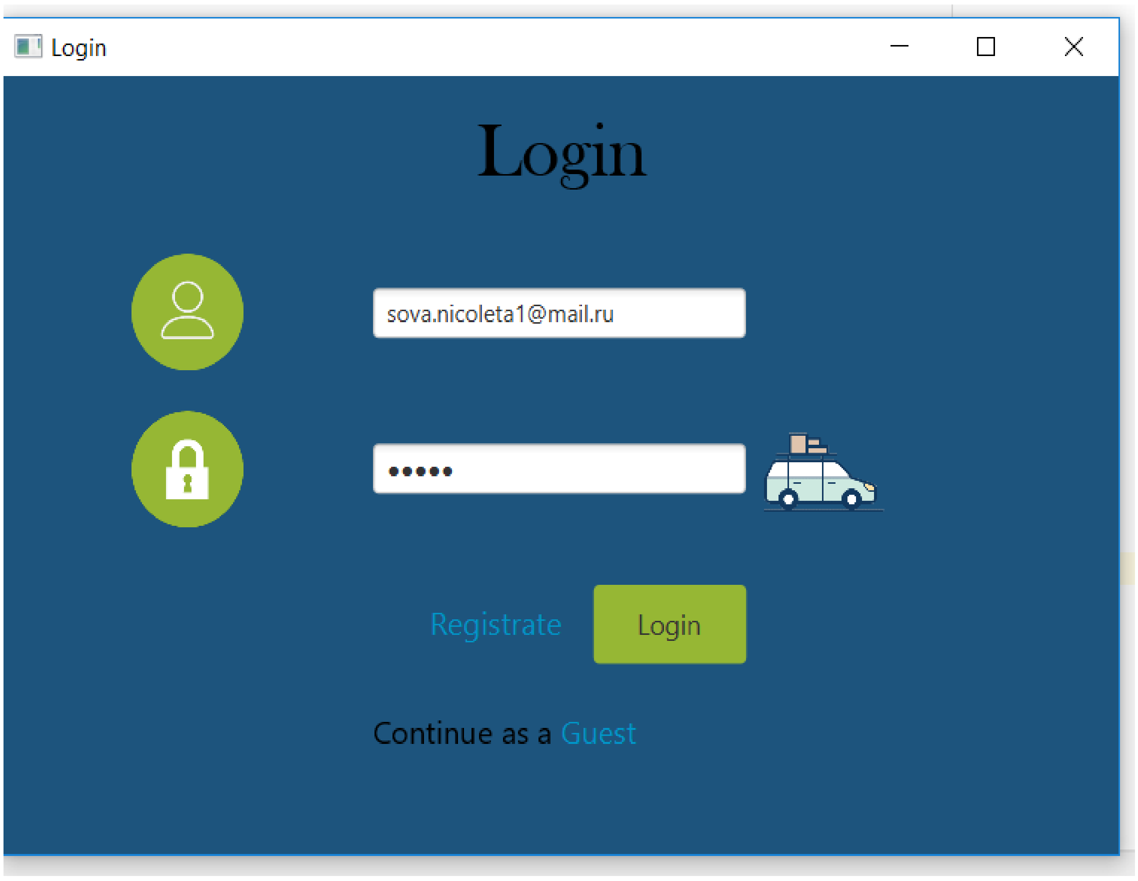


Figure 23 Login view

After entering the correct email and password this window appears.

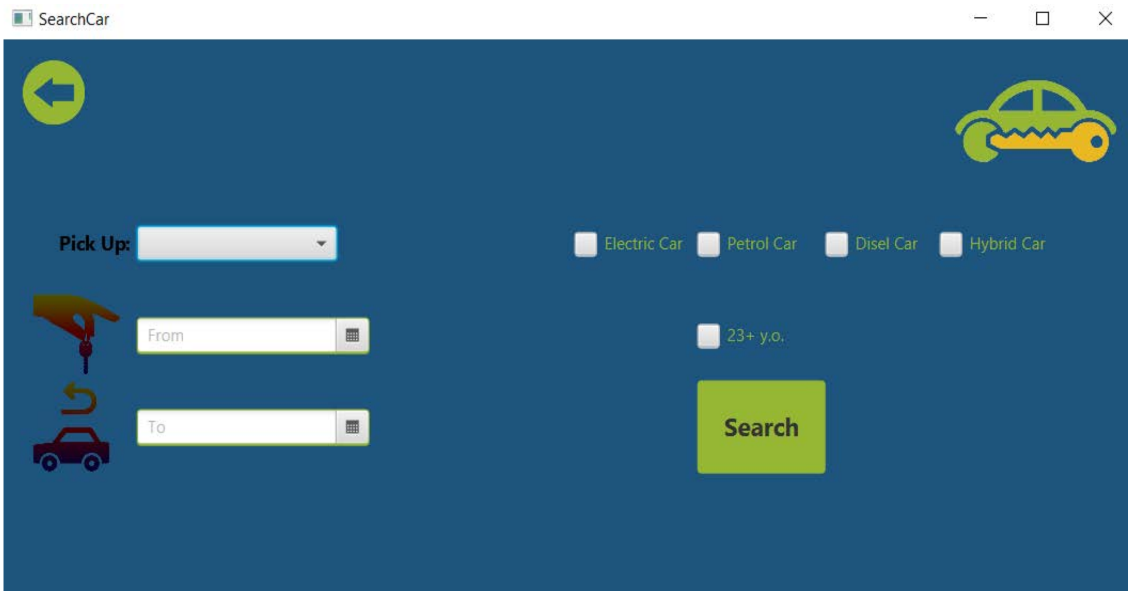


Figure 24 Logged in search car view

Login with incorrect password and email (an invalid email is insert).

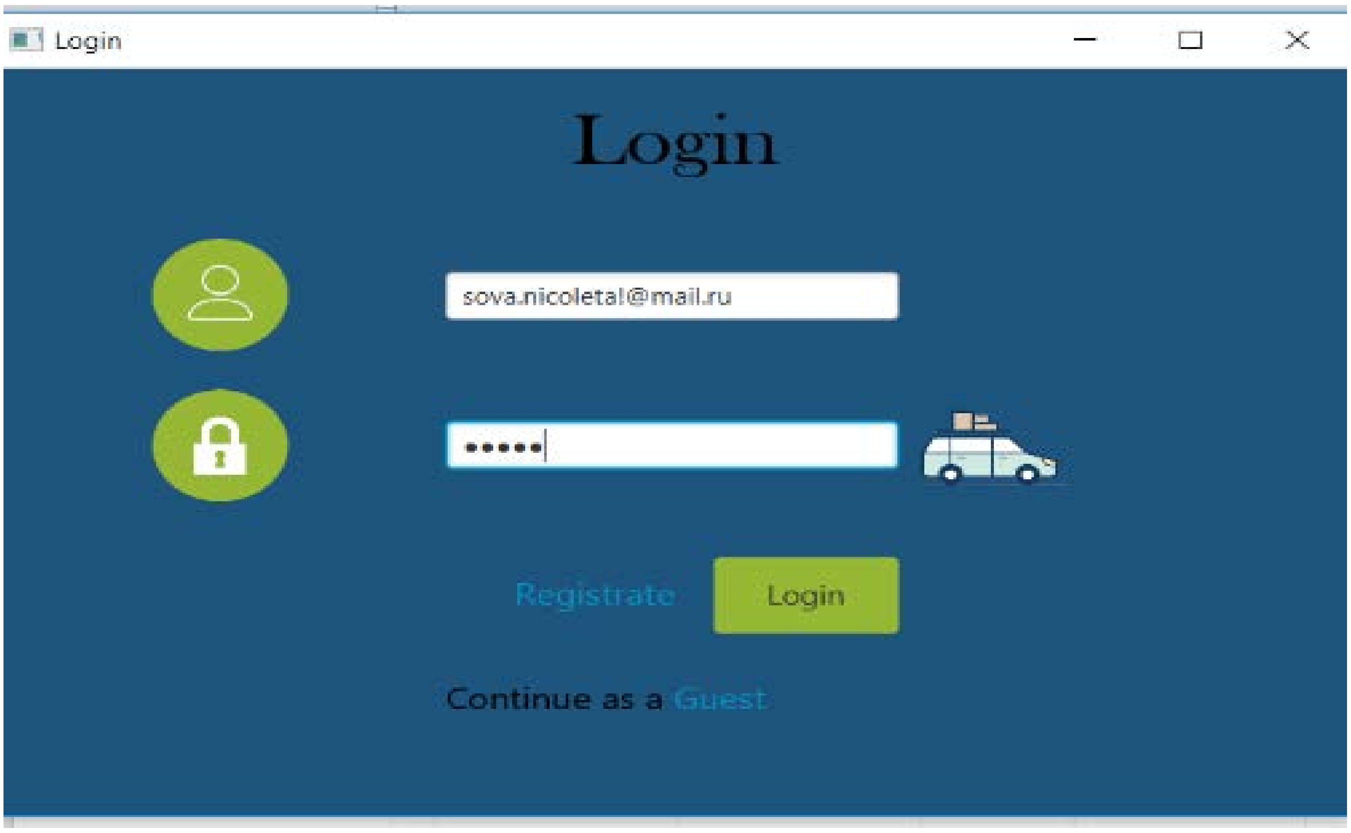
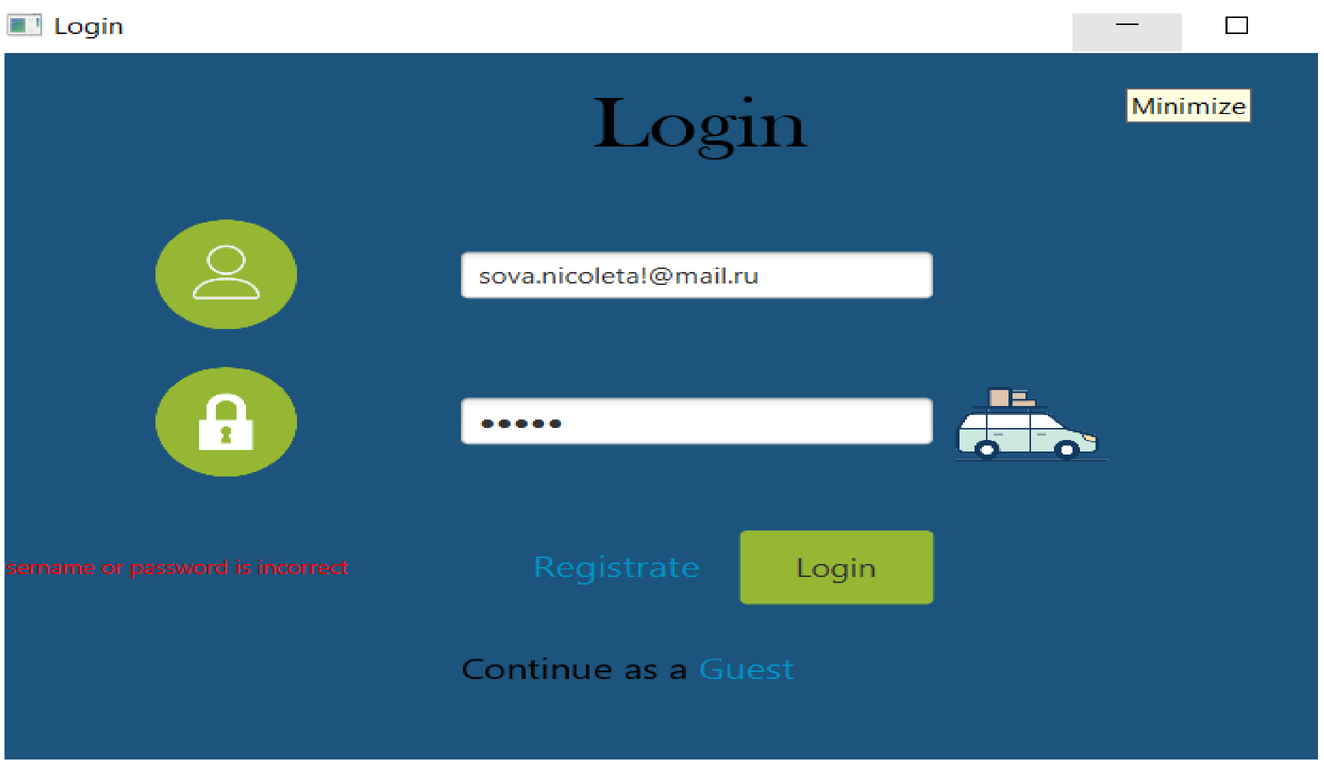


Figure 25 Login view test1

A warning that said (username or password incorrect)



sova.nicoleta!@mail.ru

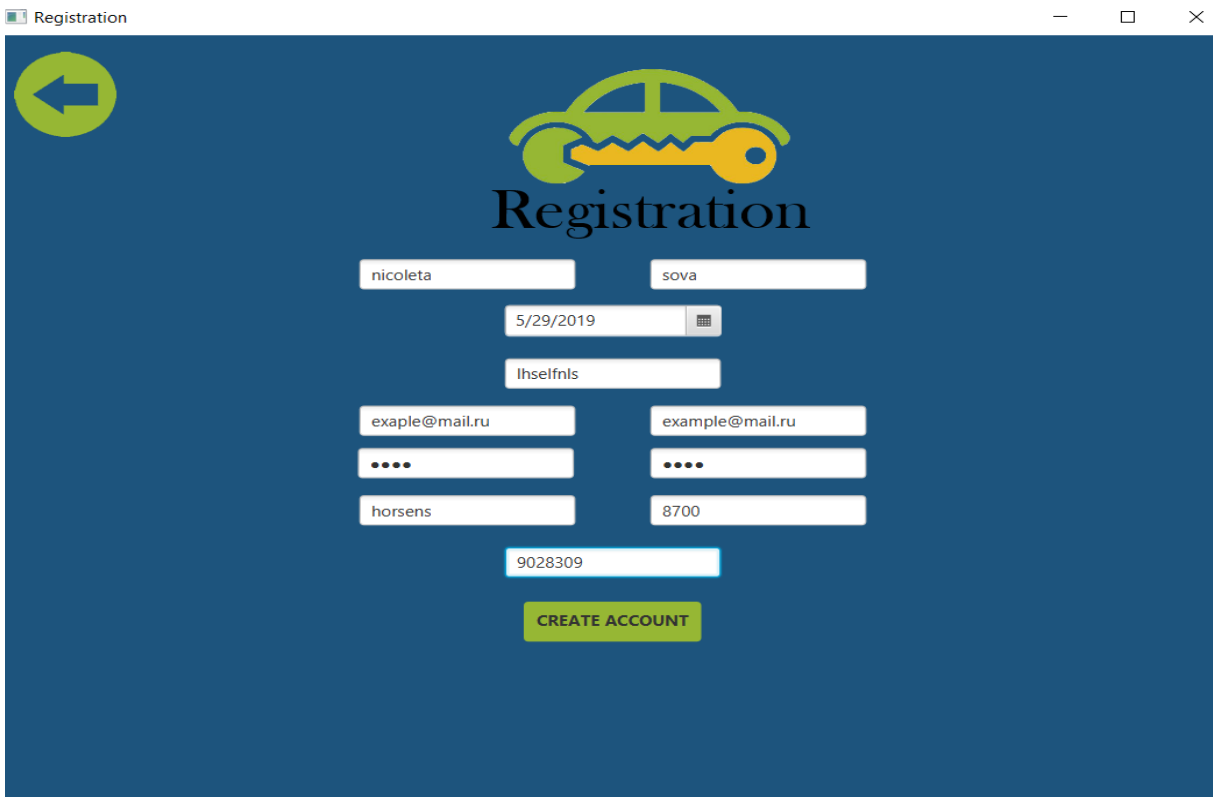
●●●●●

sername or password is incorrect

Figure 26 Login view test2

If you want to register:

Registration with all fields completed:



nicoleta

sova

5/29/2019

lhselfnls

exaple@mail.ru

example@mail.ru

●●●●

●●●●

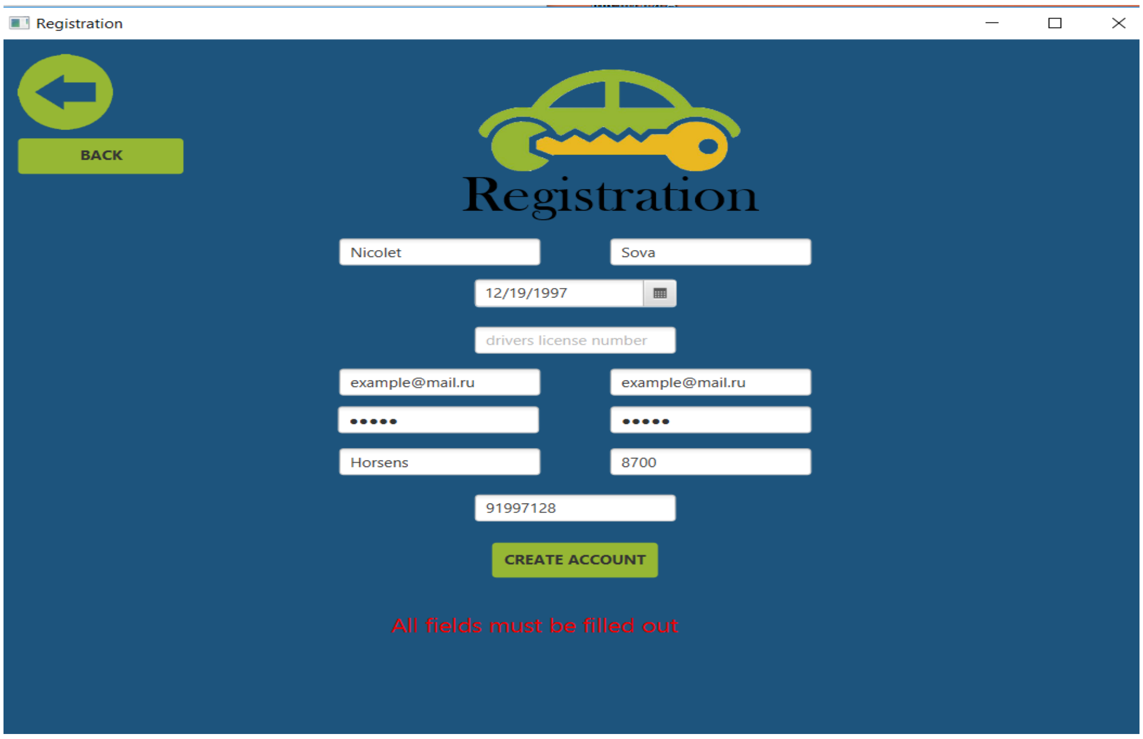
horsens

8700

9028309

Figure 27 Registration test1

Registration without completing all the fields(a warring appears)



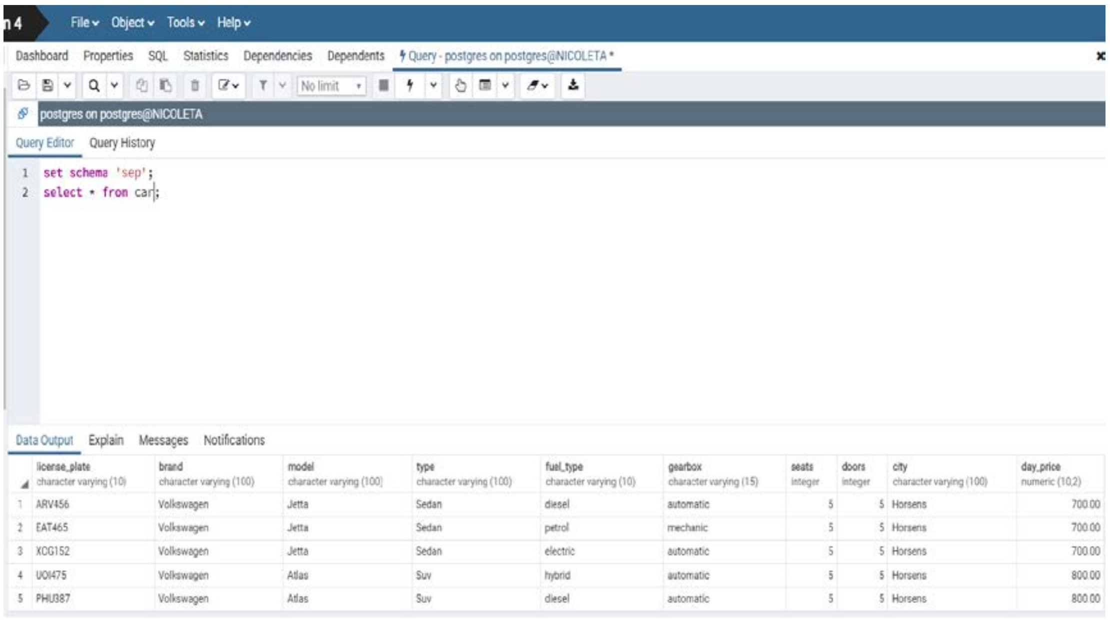
Nicolet | Sova
12/19/1997
drivers license number
example@mail.ru | example@mail.ru
••••• | •••••
Horsens | 8700
91997128

All fields must be filled out

Figure 28 Registration test2

Search a car:

All available cars in the database are shown in the image below



```
set schema 'sep';
select * fron car;
```

	license_plate character varying (10)	brand character varying (100)	model character varying (100)	type character varying (100)	fuel_type character varying (10)	gearbox character varying (15)	seats integer	doors integer	city character varying (100)	day_price numeric (10,2)
1	ARV456	Volkswagen	Jetta	Sedan	diesel	automatic	5	5	Horsens	700.00
2	EAT465	Volkswagen	Jetta	Sedan	petrol	mechanic	5	5	Horsens	700.00
3	XCG152	Volkswagen	Jetta	Sedan	electric	automatic	5	5	Horsens	700.00
4	UOI475	Volkswagen	Atlas	Suv	hybrid	automatic	5	5	Horsens	800.00
5	PHU387	Volkswagen	Atlas	Suv	diesel	automatic	5	5	Horsens	800.00

Figure 29 Database with all cars

After choosing the desired city, date, type of cars and are you more than 23 the search is made. If the client choses a type of car in a period of time that it is already rented by someone else, that car will not be shown.

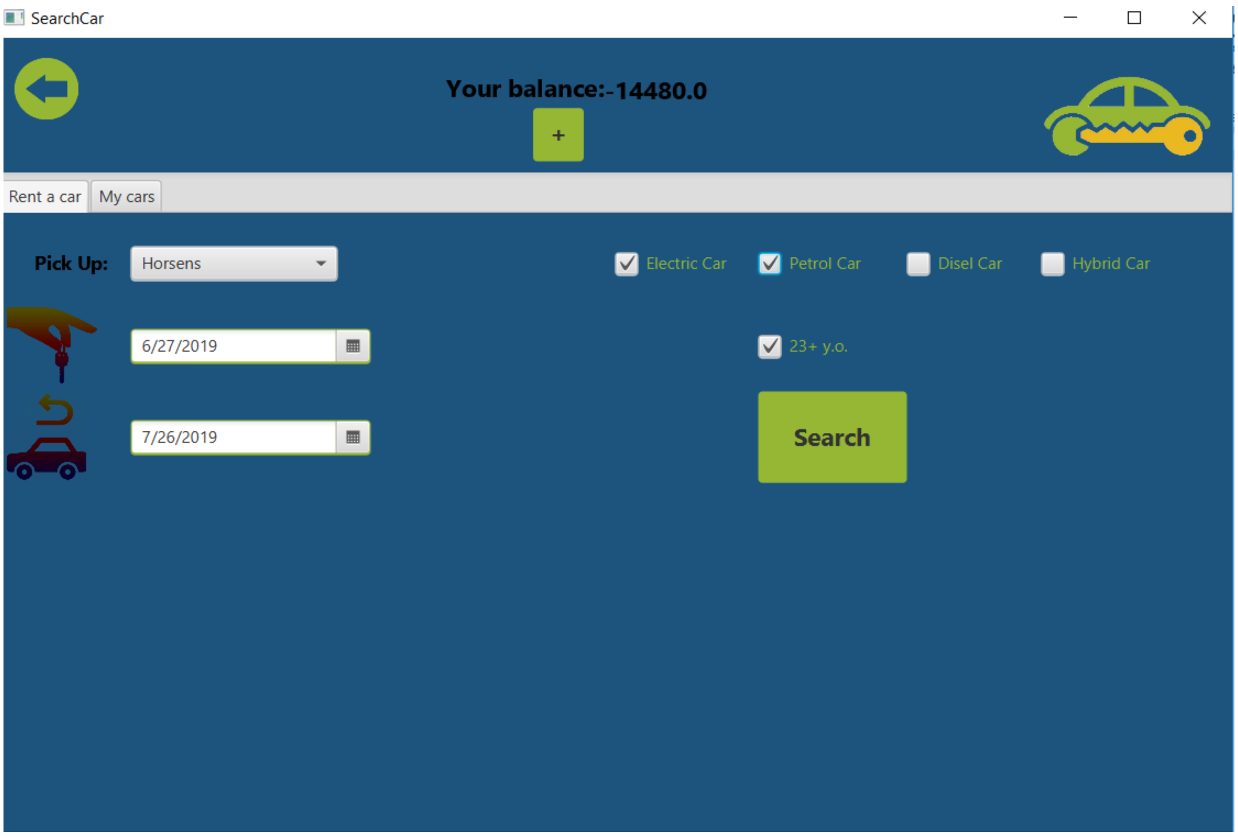
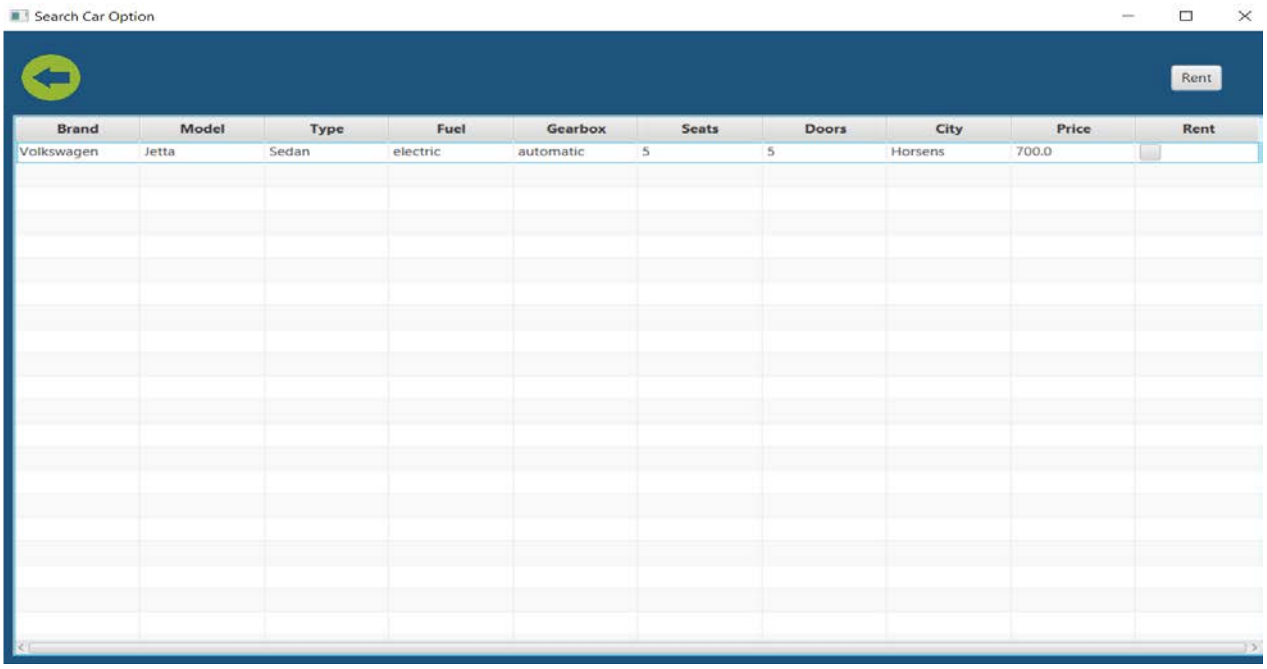


Figure 30 Search car as a user view

Based on search the available cars are shown:

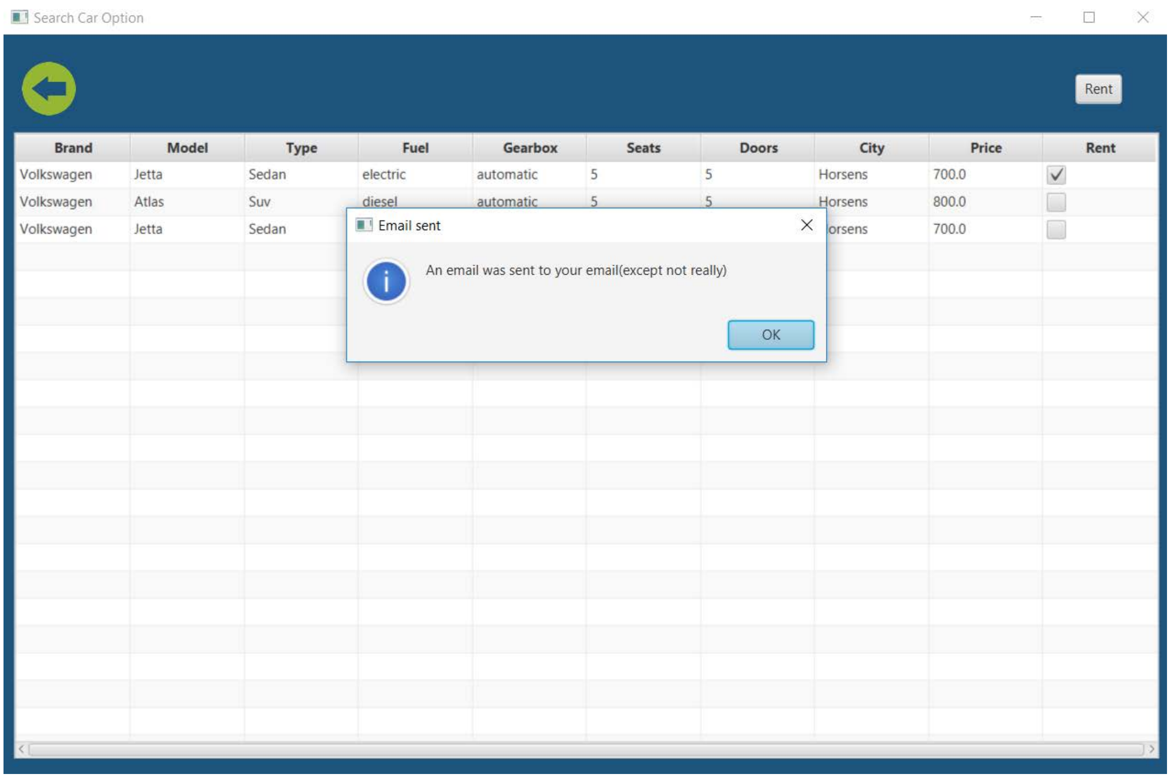


Brand	Model	Type	Fuel	Gearbox	Seats	Doors	City	Price	Rent
Volkswagen	Jetta	Sedan	electric	automatic	5	5	Horsens	700.0	☐

Figure 31 Search window with data from the database

Rent a car >

The client chooses between offered cars and receive an email confirmation



Brand	Model	Type	Fuel	Gearbox	Seats	Doors	City	Price	Rent
Volkswagen	Jetta	Sedan	electric	automatic	5	5	Horsens	700.0	✓
Volkswagen	Atlas	Suv	diesel	automatic	5	5	Horsens	800.0	☐
Volkswagen	Jetta	Sedan					orsens	700.0	☐

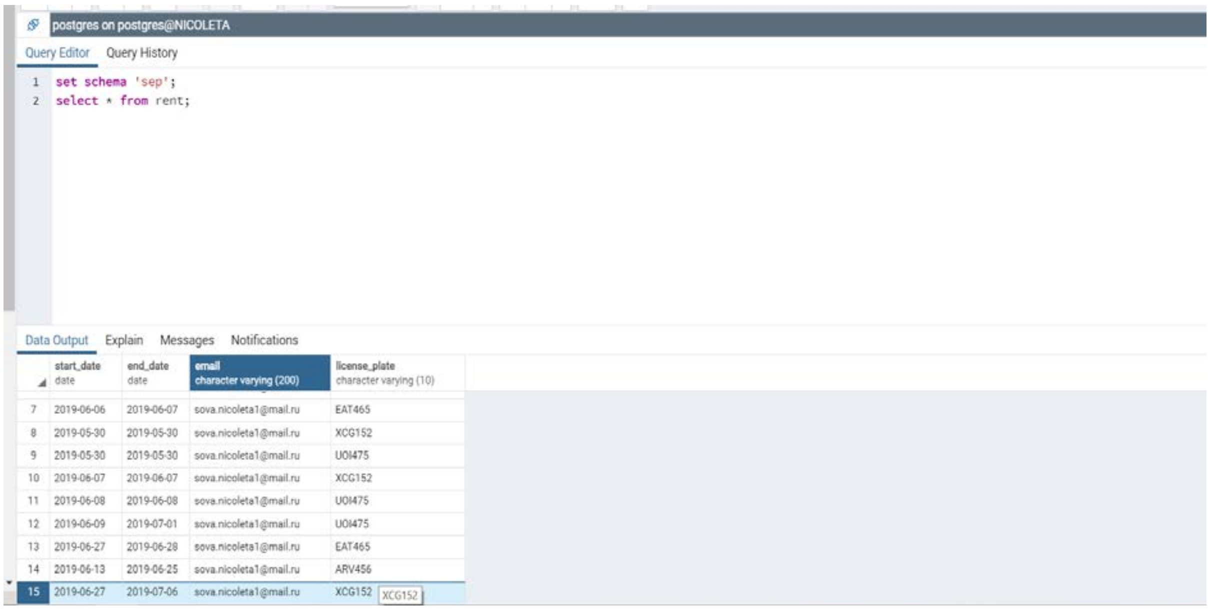
Email sent

An email was sent to your email(except not really)

OK

Figure 32 Search car renting a car

In database we can see that the car has been rented.



```
set schema 'sep';
select * from rent;
```

	start_date date	end_date date	email character varying (200)	license_plate character varying (10)
7	2019-06-06	2019-06-07	sova.nicoleta1@mail.ru	EAT465
8	2019-05-30	2019-05-30	sova.nicoleta1@mail.ru	XCG152
9	2019-05-30	2019-05-30	sova.nicoleta1@mail.ru	UOI475
10	2019-06-07	2019-06-07	sova.nicoleta1@mail.ru	XCG152
11	2019-06-08	2019-06-08	sova.nicoleta1@mail.ru	UOI475
12	2019-06-09	2019-07-01	sova.nicoleta1@mail.ru	UOI475
13	2019-06-27	2019-06-28	sova.nicoleta1@mail.ru	EAT465
14	2019-06-13	2019-06-25	sova.nicoleta1@mail.ru	ARV456
15	2019-06-27	2019-07-06	sova.nicoleta1@mail.ru	XCG152

Figure 33 Database with the rented car

Adding money in the wallet:

We see that the balance on the account is not enough to rent a car, so we have to add money.

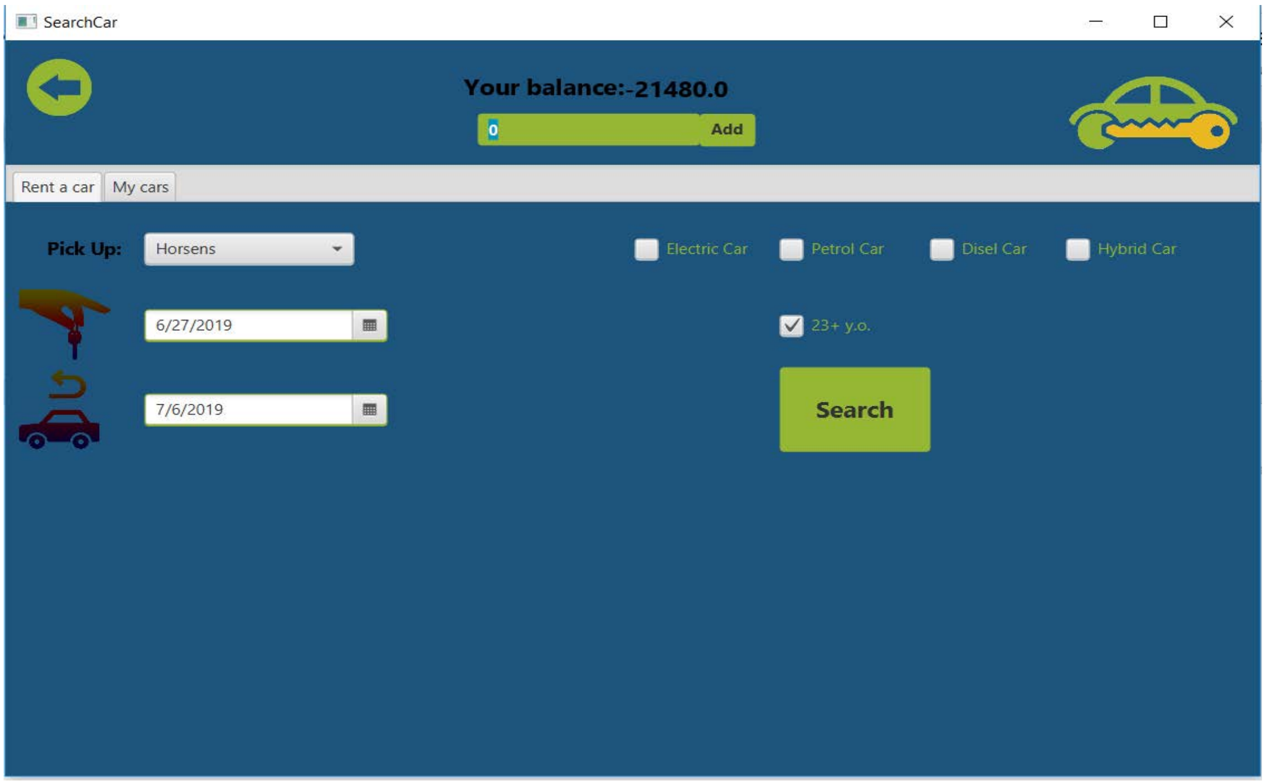


Figure 34 User view adding money to the users wallet

After adding an amount of money, the balance is changed.

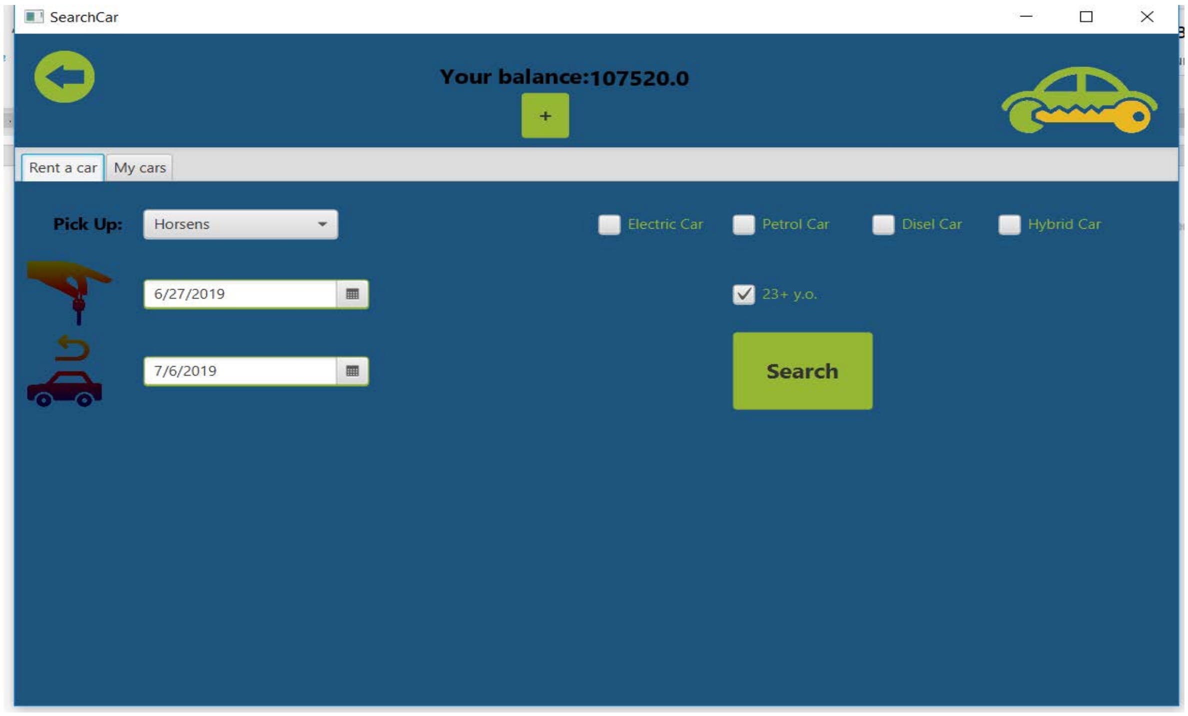
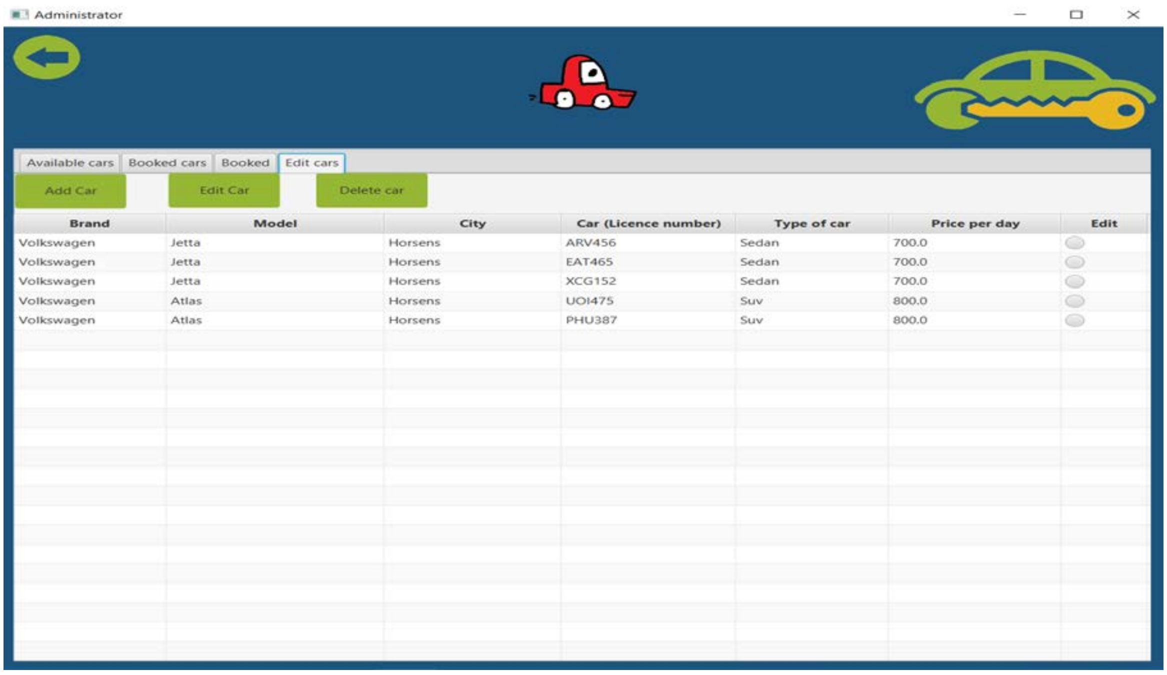


Figure 35 User view wallet changed

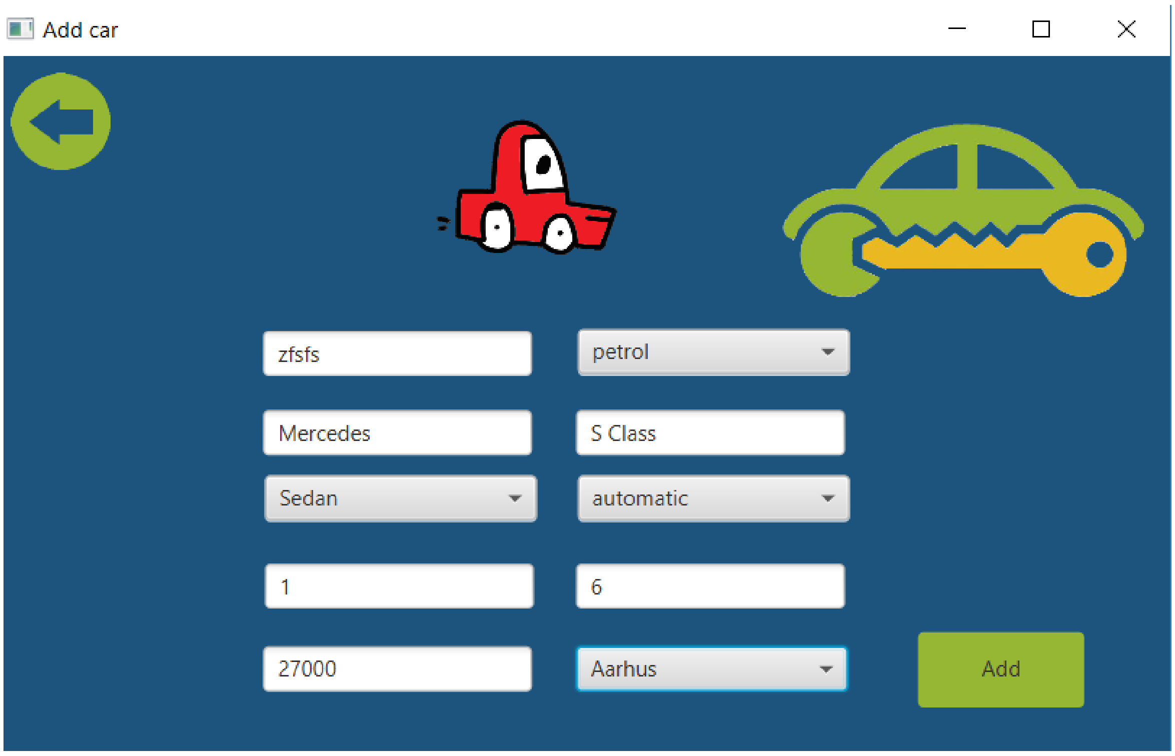
When you login as administrator and want to add a car, press in Add car



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Volkswagen	Jetta	Horsens	ARV456	Sedan	700.0	○
Volkswagen	Jetta	Horsens	EAT465	Sedan	700.0	○
Volkswagen	Jetta	Horsens	XCG152	Sedan	700.0	○
Volkswagen	Atlas	Horsens	UOI475	Suv	800.0	○
Volkswagen	Atlas	Horsens	PHU387	Suv	800.0	○

Figure 36 Administrator view edit car tab

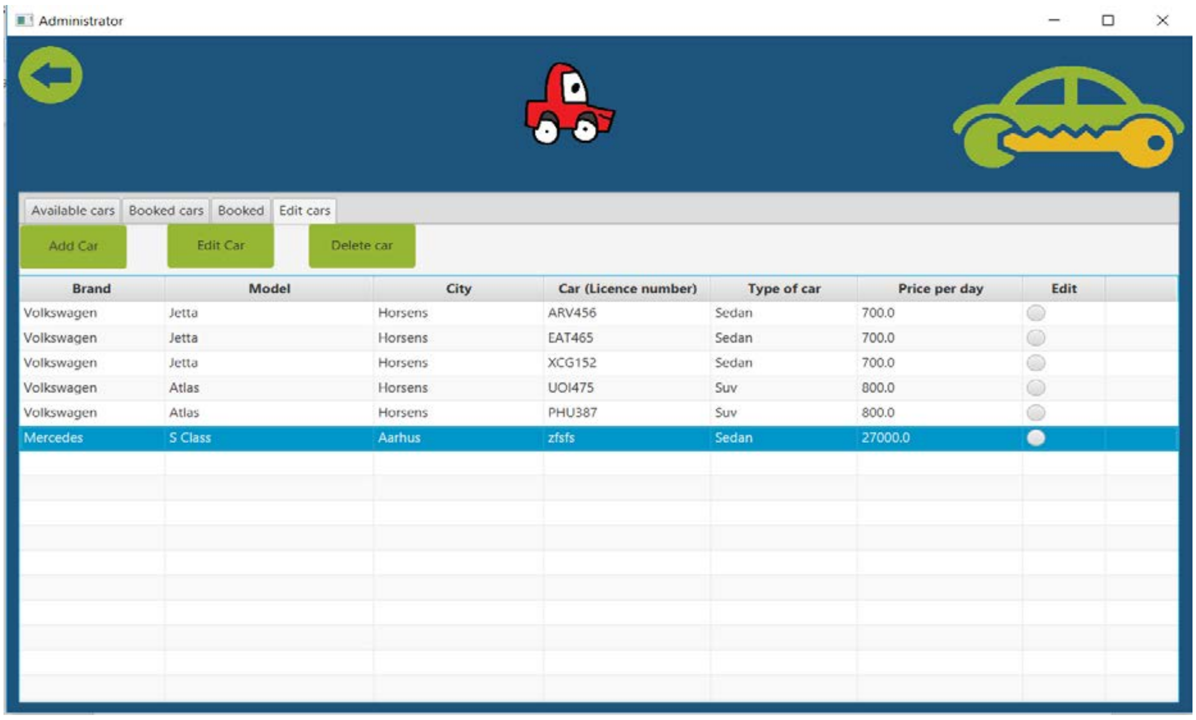
Adding a car with all the field completed:



zfsfs	petrol
Mercedes	S Class
Sedan	automatic
1	6
27000	Aarhus

Figure 37 Administrator add new car view

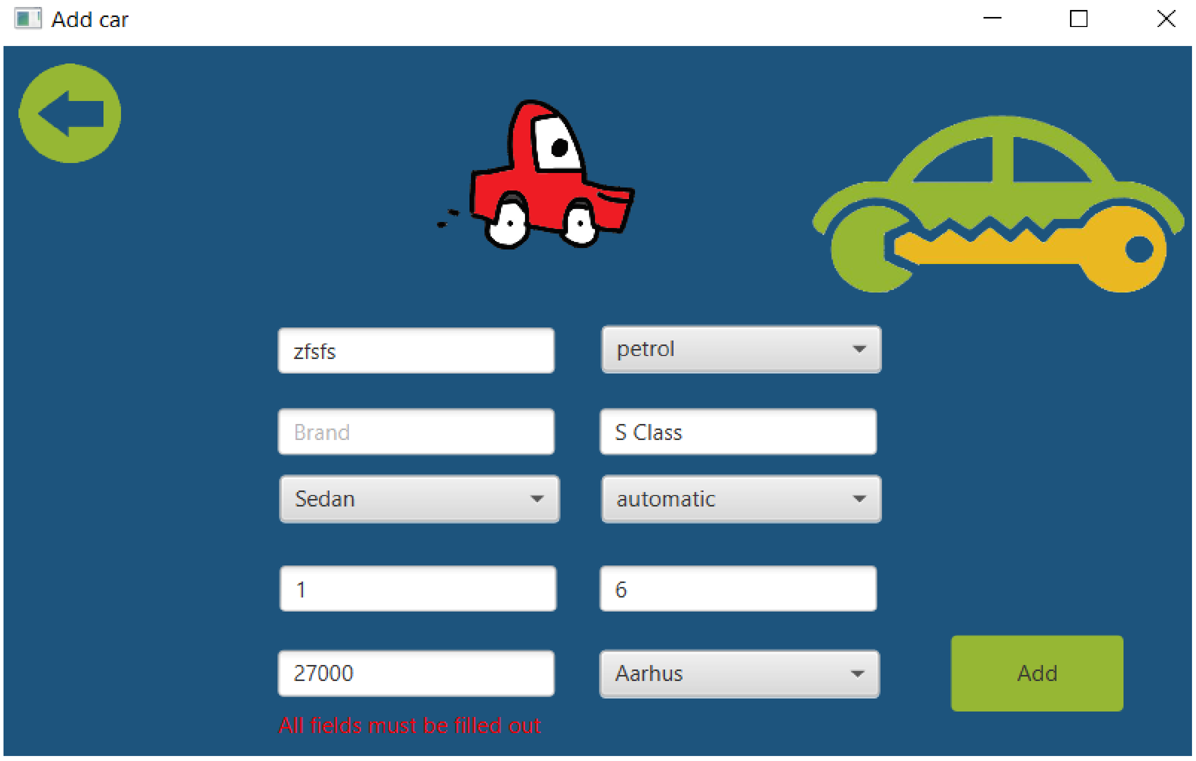
The car is added



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Volkswagen	Jetta	Horsens	ARV456	Sedan	700.0	○
Volkswagen	Jetta	Horsens	EAT465	Sedan	700.0	○
Volkswagen	Jetta	Horsens	XCG152	Sedan	700.0	○
Volkswagen	Atlas	Horsens	UOI475	Suv	800.0	○
Volkswagen	Atlas	Horsens	PHU387	Suv	800.0	○
Mercedes	S Class	Aarhus	zfsfs	Sedan	27000.0	○

Figure 38 Administrator main view edit car tab - new car added

Adding a car without all fields completed (error appears)



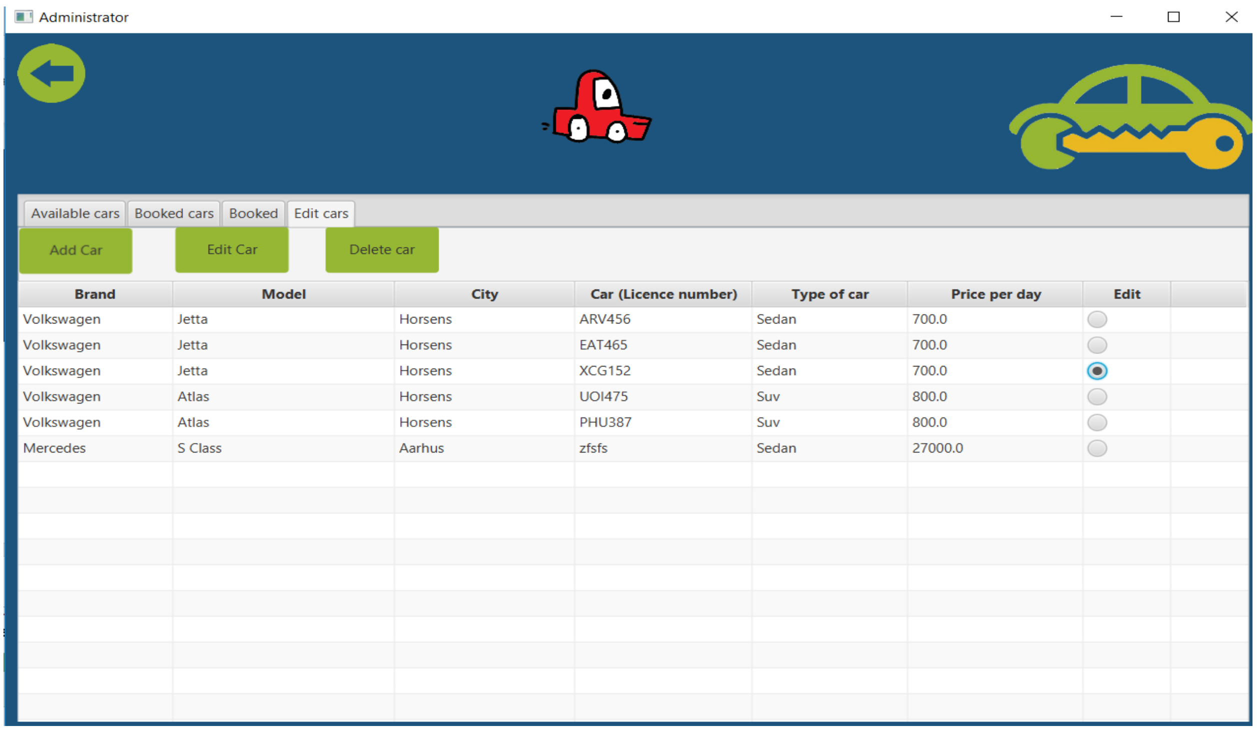
zfsfs	petrol
Brand	S Class
Sedan	automatic
1	6
27000	Aarhus

All fields must be filled out

Figure 39 Adding new car error

Edit a car:

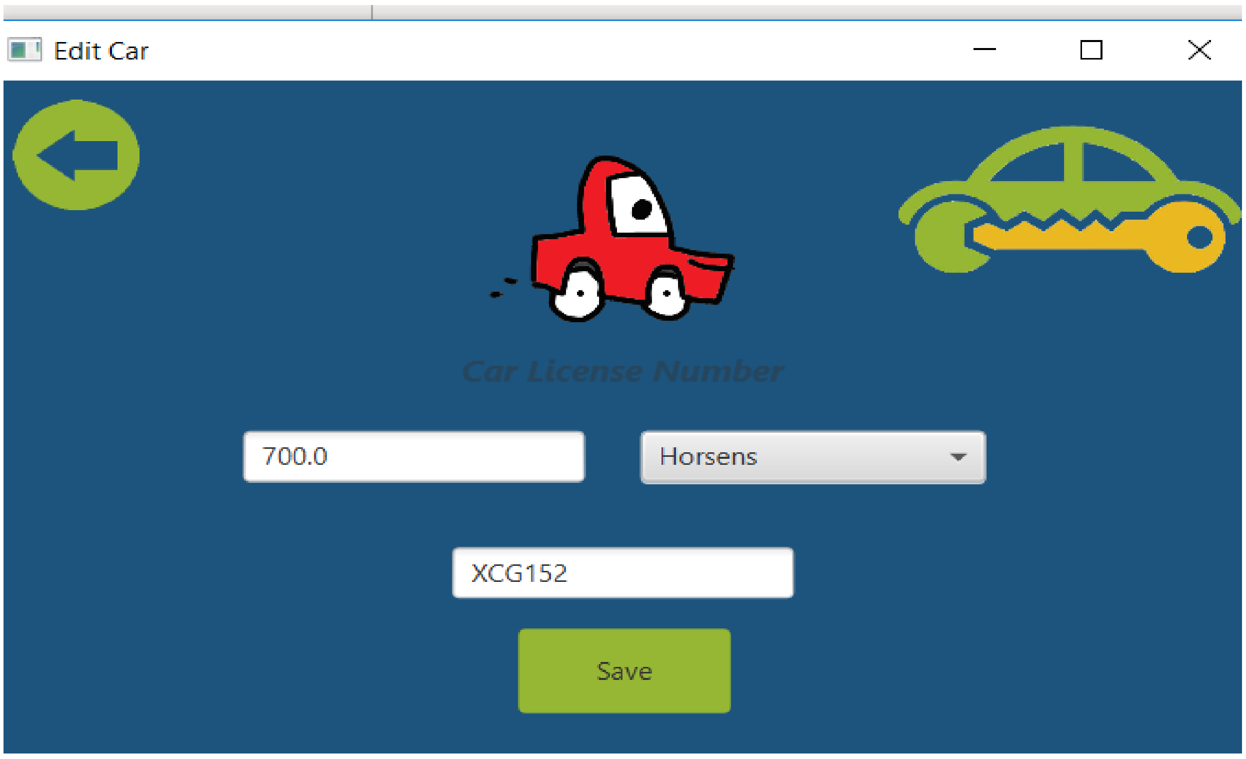
For doing this a car must be selected and then the button "add" pressed



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Volkswagen	Jetta	Horsens	ARV456	Sedan	700.0	○
Volkswagen	Jetta	Horsens	EAT465	Sedan	700.0	○
Volkswagen	Jetta	Horsens	XCG152	Sedan	700.0	●
Volkswagen	Atlas	Horsens	UOI475	Suv	800.0	○
Volkswagen	Atlas	Horsens	PHU387	Suv	800.0	○
Mercedes	S Class	Aarhus	zfsfs	Sedan	27000.0	○

Figure 40 Administrator main view edit tab

Editing the old data



Price: 700.0
City: Horsens
Licence number: XCG152

Figure 41 Administrator edit a car view

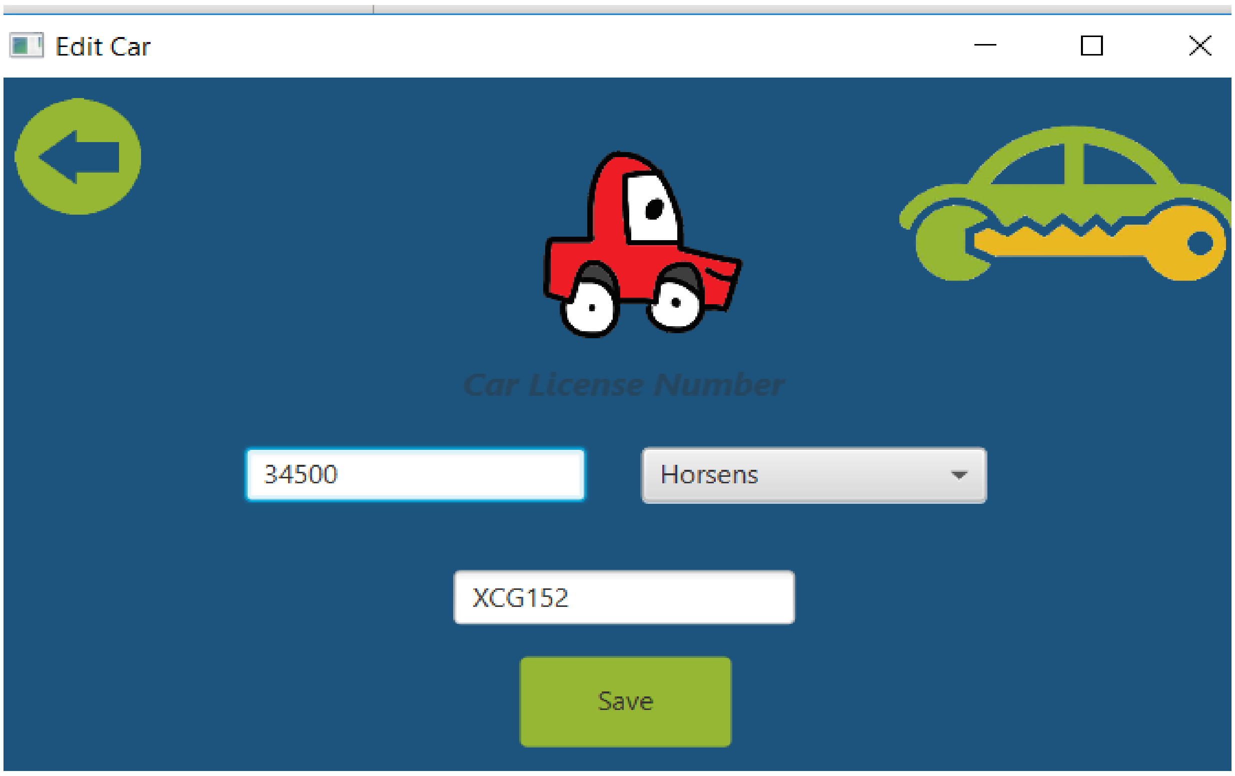
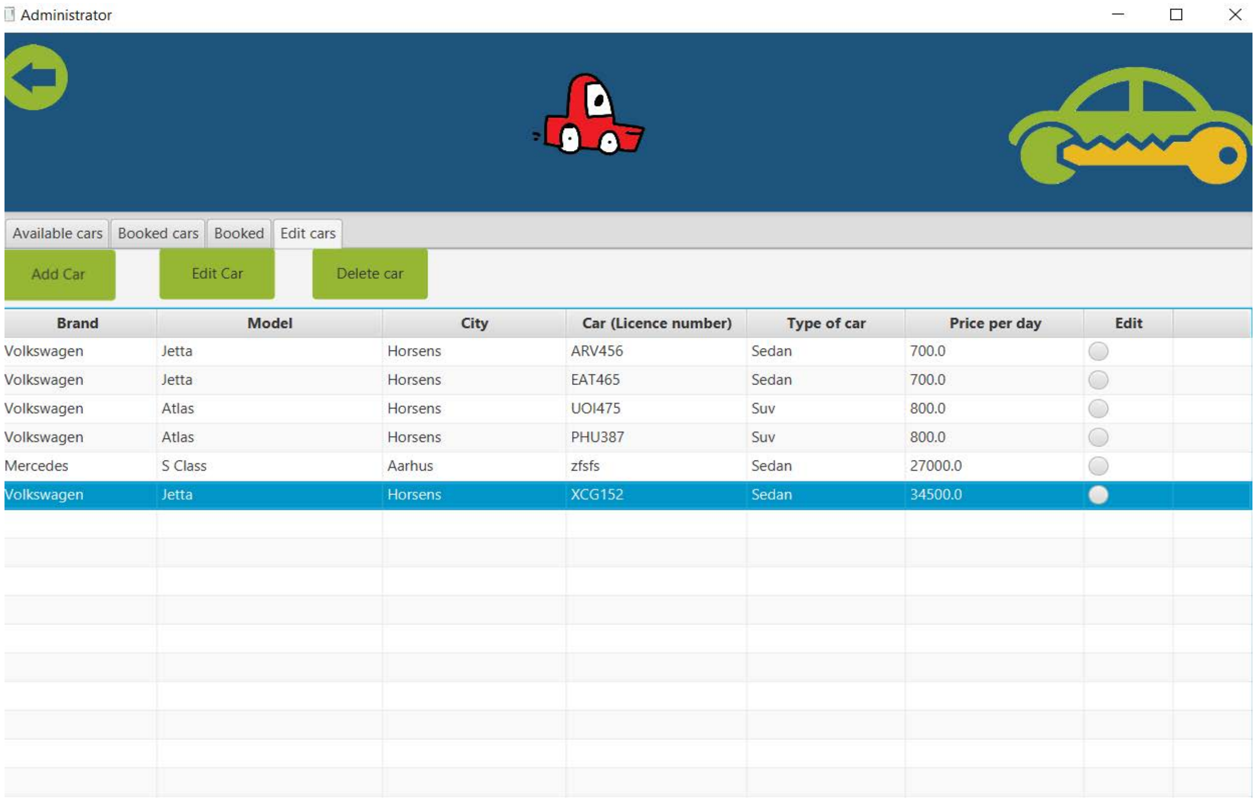


Figure 42 Administrator changed car data

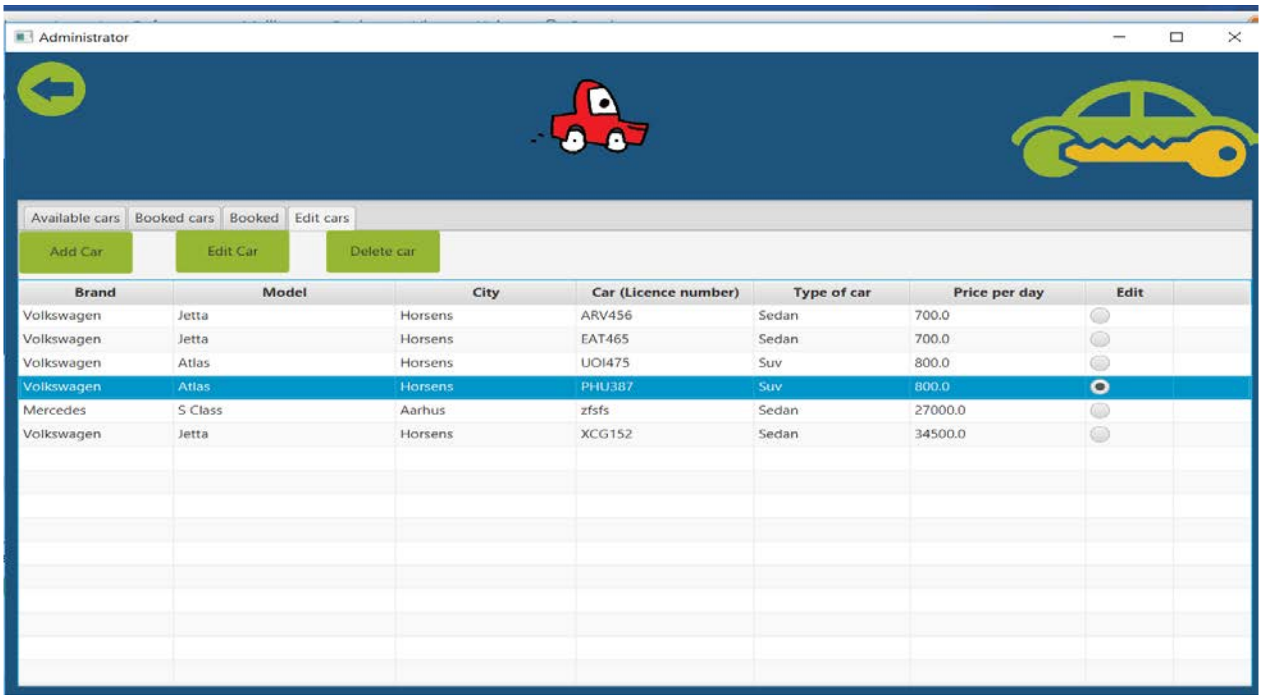
The changed data for the car.



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Volkswagen	Jetta	Horsens	ARV456	Sedan	700.0	○
Volkswagen	Jetta	Horsens	EAT465	Sedan	700.0	○
Volkswagen	Atlas	Horsens	UOI475	Suv	800.0	○
Volkswagen	Atlas	Horsens	PHU387	Suv	800.0	○
Mercedes	S Class	Aarhus	zfsfs	Sedan	27000.0	○
Volkswagen	Jetta	Horsens	XCG152	Sedan	34500.0	○

Figure 43 Administrator main view changed car

Delete a car:



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Volkswagen	Jetta	Horsens	ARV456	Sedan	700.0	○
Volkswagen	Jetta	Horsens	EAT465	Sedan	700.0	○
Volkswagen	Atlas	Horsens	UOI475	Suv	800.0	○
Volkswagen	Atlas	Horsens	PHU387	Suv	800.0	●
Mercedes	S Class	Aarhus	zfsfs	Sedan	27000.0	○
Volkswagen	Jetta	Horsens	XCG152	Sedan	34500.0	○

Figure 44 Administrator main view delete car

This part of deleting a car doesn't work. Wen the button delete is pressed it takes you to the “edit” window.

4.1 Test Specifications

ID	Priority	Estimate	Item	
1	Critical	20 h	As a user, I want to be able to rent a car in order to drive.	🟩
2	High	15 h	As a user, I want to be able to select a certain type of vehicle in order to find a vehicle that suits my needs.	🟩
3	Medium	10 h	As a user, I want to be able to select additional services in order to ease my journey.	🟩
5	High	20 h	As a user, I want to be able to select the number of days I want to rent the vehicle in order to reserve the vehicle.	🟩
6	Mid low	10 h	As a user, I want to be able to see the pricing of different vehicles in order to select a vehicle inside my budget.	🟩
7	Medium	5 h	As a user, I want to be able to cancel my reservation in order to not rent a vehicle.	🟨
8	Low	15 h	As a user, I want to be notified if changes occur to my reservation, in order to be able to change my reservation	🟨
9	High	25 h	As a user, I want to receive a booking confirmation by email, in order to have proof of reservation.	🟨
10	Low	20 h	As an administrator, I want to be able to select the prices of different cars, in order to control prices.	🟩
11	High	15 h	As an administrator, I want to be able to access my customers credentials in order to be able to contact them.	🟩
12	High	10 h	As an administrator, I want to be able to see what vehicles are currently rented out, in order to see what is available.	🟩
13	Mid low	10 h	As an administrator, I want to be able to select extra services provided, in order to ease the journey of the customer.	🟩
14	Medium	10 h	As a user, I want to be able to add money to my wallet	🟨

5 Conclusions

The purpose of this project was to create a system that will make the rental car company's life easier. The system is able to satisfy customer needs when he wants to ret a car or administrator needs when he wants to make it for his client. During the project some changes have been made and others gained priority. All the changes that have been done were made in order to make the system better.

The class diagrams were implemented as shown and later tested.

In conclusion, the program has been completed except some errors with the money related requirements.

6 Project future

Since our project has some errors with money, we need to fix those problems. Delete part is not working and also, we want to make it work.
Also, we would redesign the GUI to look better and be more interactive. We could try to make a full system and sell it to real rental companies.

7 Sources of information

Andersen, S. V., n.d. [Online].
Europcar, 2019. *Europcar.* [Online]
Available at: https://www.europcar.com/business/business-services
[Accessed 28 February 2019].
Larman, C., 2004. *Applying UML and Patterns.* 3rd ed. Westford, Massachusetts : Pearson Education.
NISO, 2010. *Scientific and Technical Reports -,* Baltimore: National Information Standards Oganization.
VIA Engineering, in preparation. *Confidential Student Reports,* s.l.: s.n.
VIA, 2018. *VIA Engineering Guidelines - Project Description Guideline.* [Online]
Available at:
https://studienet.via.dk/projects/Engineering__project_methodology/General/Guidelines/2018%20Project%20Description%20(Appendix%201)%20VIA%20Engineering%20Guidelines.pdf
[Accessed 28 February 2019].

8 Appendix A - A Project Description

Background description:

Europcar (Europcar, 2019) is a car rental company that offers a large spectrum of cars and extra services. It has been active for about 2 years in this industry. The company provides cars for people who want to rent it for some time. They are used to doing this in an office. If a customer desires to acquire a car for some time, then the customer has to go to Europcar office, fill out a form: identity, driver's license, give a time period for which they want to rent a car, choose a car and only then they get the car. This takes time and is an old way of doing it. furthermore, this is a problem not just for the company but also for the clients. They have to come right to the office and bring all the documents with them, which mostly causes problems because clients forget something and at the end they don't come back and choose other rental company.

Europcar wants to grow and get to the next level in order to get more clients and make their way of working easier. That is why Europcar asked us to create a program that would make their lives easier for them and for their clients.

Europcar is used to doing everything on paper. It is not as effective as doing everything online. Europcar wants a product that would be more efficient and quicker - thus making their customers lives easier. Europcar want an online version, so that every client could have the possibility to rent a car from the comfort of their home.

Working on this project will help the group to grow at a new level and asses the knowledge into practice. Also it will help us to understand how a team works in real-life situation. The company chooses us in order to find and solve the problem that they have now.

Definition of purpose:

The purpose of this project is to create a program with a client server communication, in which customers can rent cars that the rental company provide. Customers will be able to select and reserve vehicles through the client server by registering, resulting in the rental company having this information of their customers.

Problem Statement:

Main problem:

What are the needs for a car rental company, running on a client server?

Sub problems:

- What needs to be known about the client in order to rent a car?
- What information should be provided about the cars?
- How will the rental period be managed?
- What extra services are available? (insurance, fuel, etc.)
- What happens when a client cancels a reservation?

Delimitations:

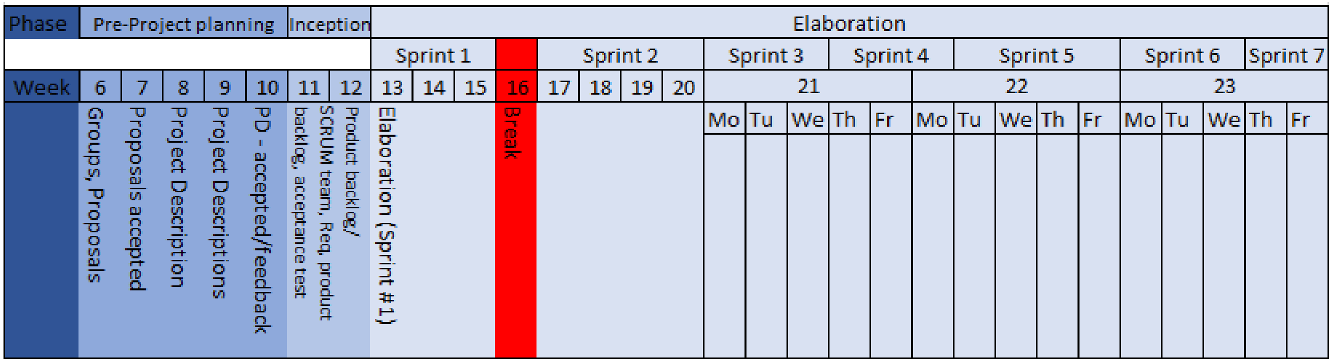
- There will be no money transactions available.
- The program only counts the amount the user needs to pay, but the user never pays.
- No damage or extra issues with the car.
- Not focus on the returning of the car or pick-up location.

Choices of models and methods:

What **- partial problem**	**Why** **-Study the problem**	**Which** **-Methods/models/ theories**	**Who** **-has the main responsibility for this point**
What needs to be known about the client in order to rent a car?	Because there must exist a form to complete for company's database to register the clients and show further information.	Provided information from the company	Lucas
What information should be provided about the cars?	Clients must have a brief information about the selected data.	Provided information about the company	Justinas
How will the rental period be managed?	For the company's interests on	The rental period will be managed by	Dima

	managing expenses for the provided services.	using data bases and the knowledge from SDJ2	
What happens when a client cancels a reservation?	It affects company's profit.	Newly acquired knowledge from SDJ2 about listeners	Simona
What extra services are available? (insurance, fuel, etc.)	Some clients want to take some precautions regarding accidents, fuel and other	The database will have all the necessary information	Nicoleta

Time schedule:



Phase	Pre-Project planning					Inception		Elaboration																						
								Sprint 1				Sprint 2				Sprint 3			Sprint 4			Sprint 5				Sprint 6			Sprint 7	
Week	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21					22					23				
																Mo	Tu	We	Th	Fr	Mo	Tu	We	Th	Fr	Mo	Tu	We	Th	Fr
	Groups, Proposals	Proposals accepted	Project Description	Project Descriptions	PD - accepted/feedback	Product backlog/ SCRUM team, Req, product backlog, acceptance test		Elaboration (Sprint #1)			Break																			

Risk assessment:

Risks	**Description**	**Likelihood scale:1-5** Scale:1-5 5=high risk	**Severity** Scale: 1-5 5= high risk	**Product of likelihood and severity**	**Risk mitigation e.g. Preventive & Responsive actions**	**Identifiers**	**Responsible**
Risk 1	Lack of time before hand-in	5	5	25	Time-scheduling	Blaming others	Simona
Risk 2	Lack of knowledge	3	4	12	Reading, searching the information	Making excuses	Dementie
Risk 3	Delays	4	5	20	Time controlling	Laziness	Nicoleta
Risk 4	Less meetings	2	4	10	Communication in the group	Absence	Lucas
Risk 5	Personal problems	4	5	20	Communication, meetings	Lack of experience	Justinas
Risk 6	miscommunication	5	5	25	Understanding, communication	Afraid to talk and do not recap	Dementie

Sources of information:

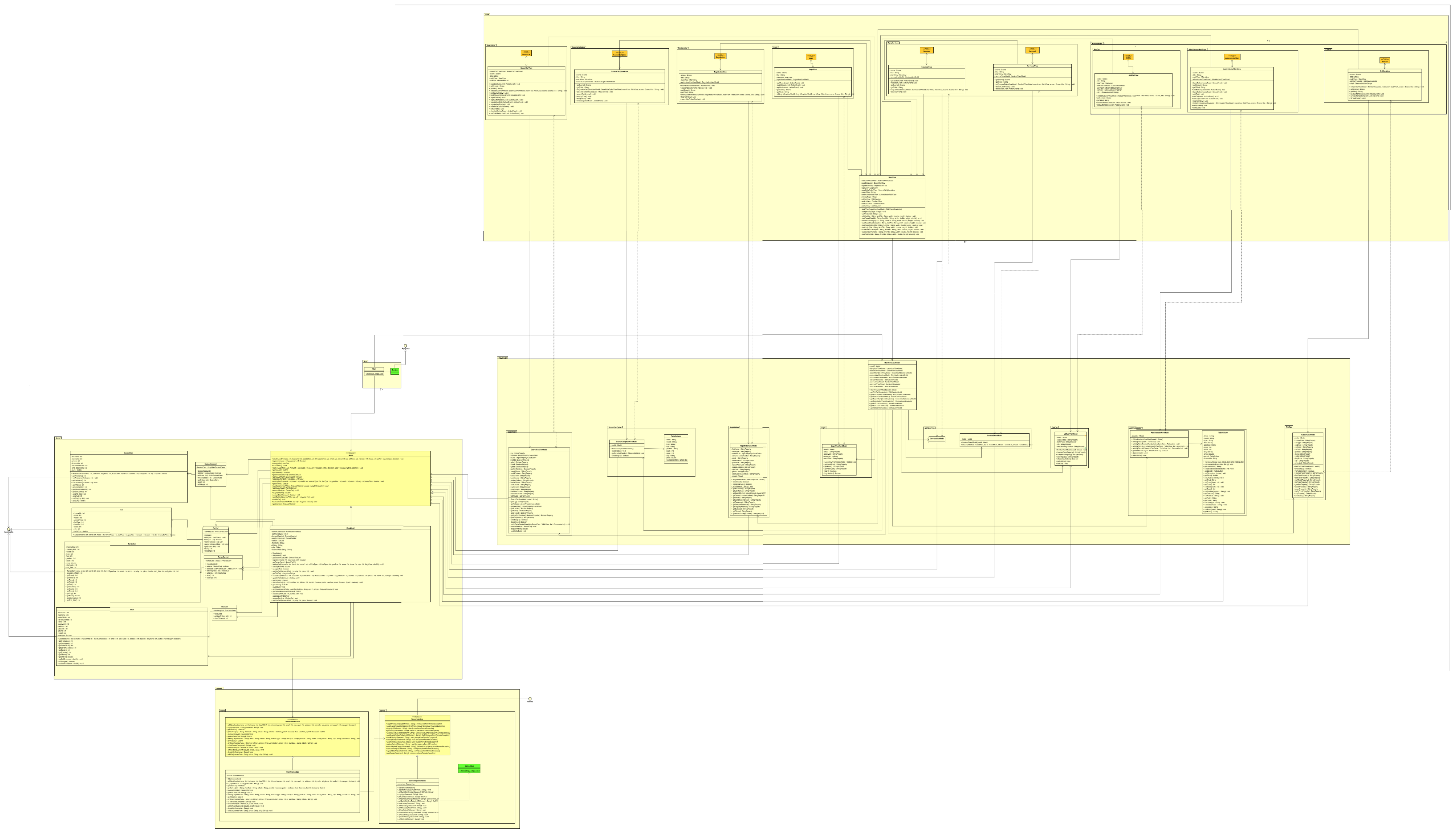
Europcar, 2019. *Europcar.* [Online]
Available at: https://www.europcar.com/business/business-services
[Accessed 28 February 2019].

NISO, 2010. *Scientific and Technical Reports -,* Baltimore: National Information Standards Oganization.

VIA Engineering, in preparation. *Confidential Student Reports,* s.l.: s.n.

VIA, 2018. *VIA Engineering Guidelines - Project Description Guideline.* [Online]
Available at:
https://studienet.via.dk/projects/Engineering__project_methodology/General/Guidelines/2018%20Project%20Description%20(Appendix%201)%20VIA%20Engineering%20Guidelines.pdf
[Accessed 28 February 2019].

9 Appendix B – Full UML Diagram



10 Appendix C - User Guide

When system was started, first window will be login. User has 3 buttons *Registration, Login, Guest*.

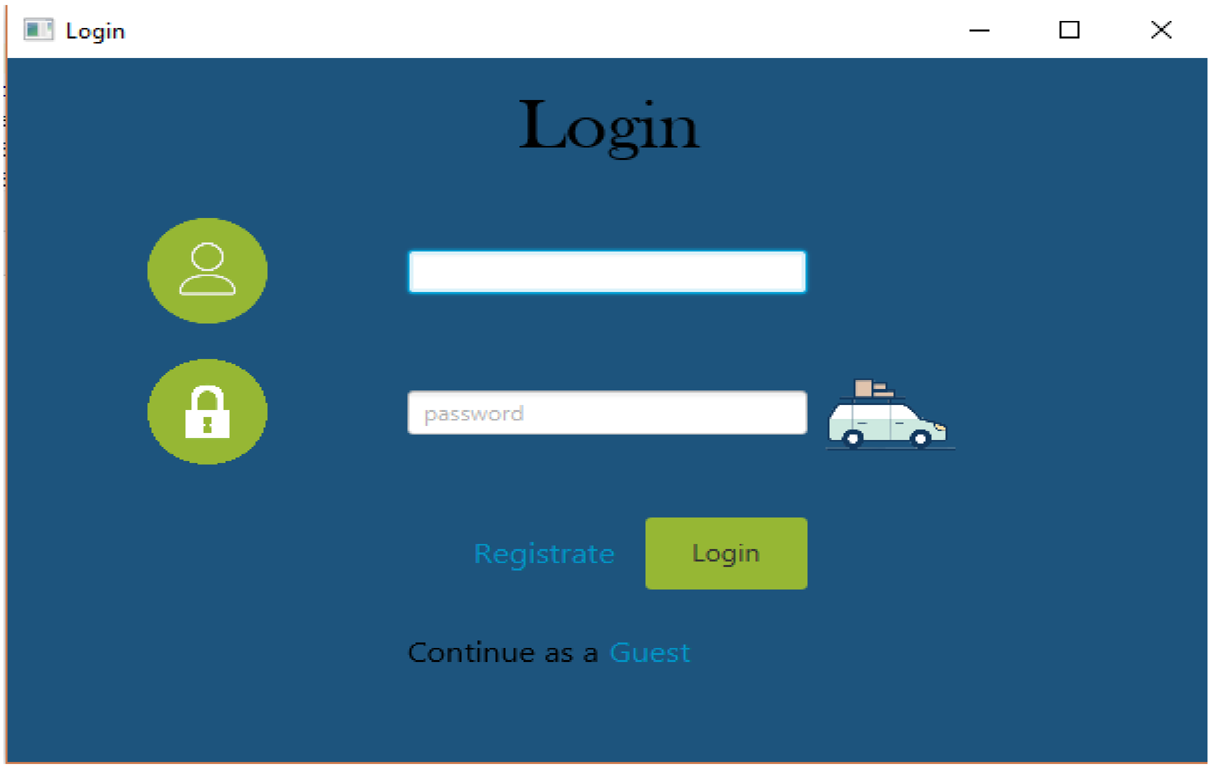


Figure 45 Login View

Registration The user needs to fill in the credentials and if the user will put different mails or passwords or skip something, the system will show an error.

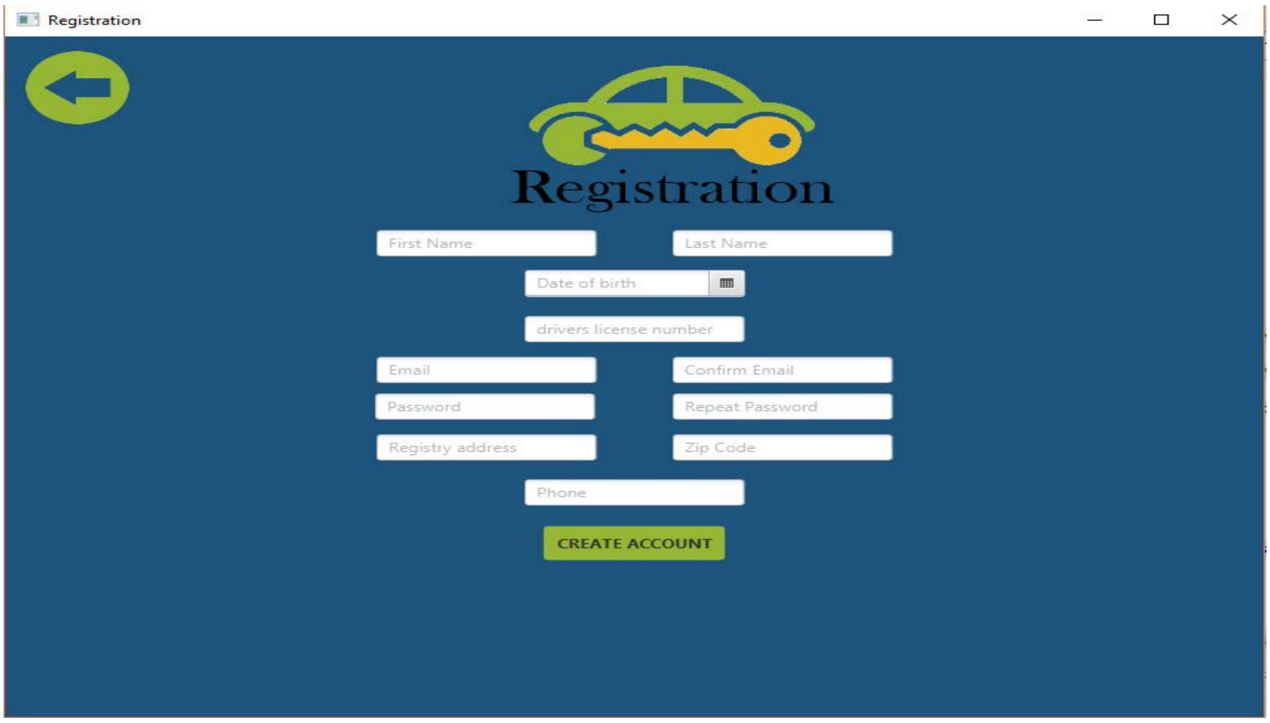
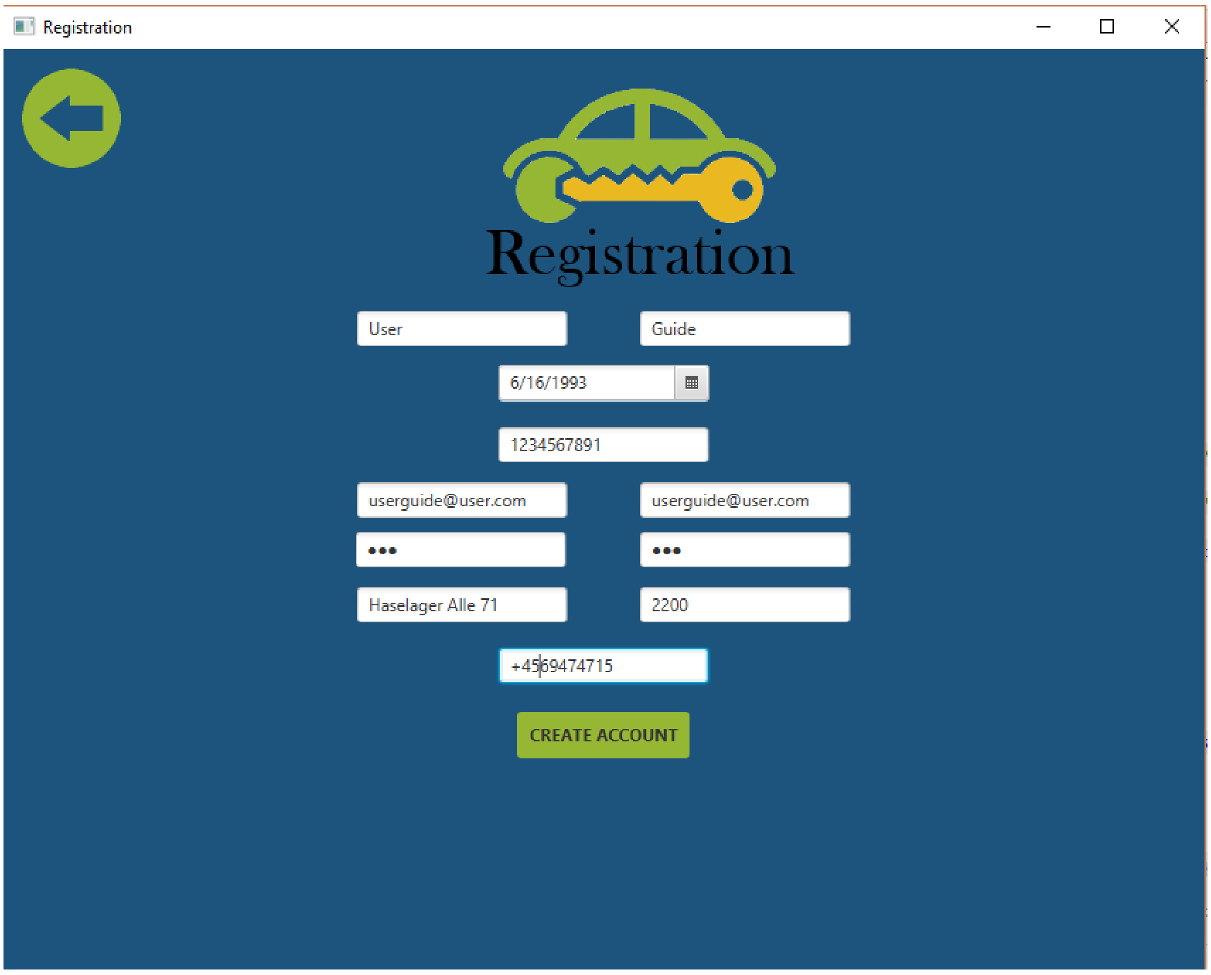


Figure 46 Registration View



User
Guide
6/16/1993
1234567891
userguide@user.com
userguide@user.com
•••
•••
Haselager Alle 71
2200
+4569474715

Figure 47 Registration view with credentials

Then the user should check one more time the credentials and click on button "*Create Account*" system automatically will save the data in database and then will appear a new window "Search car ".

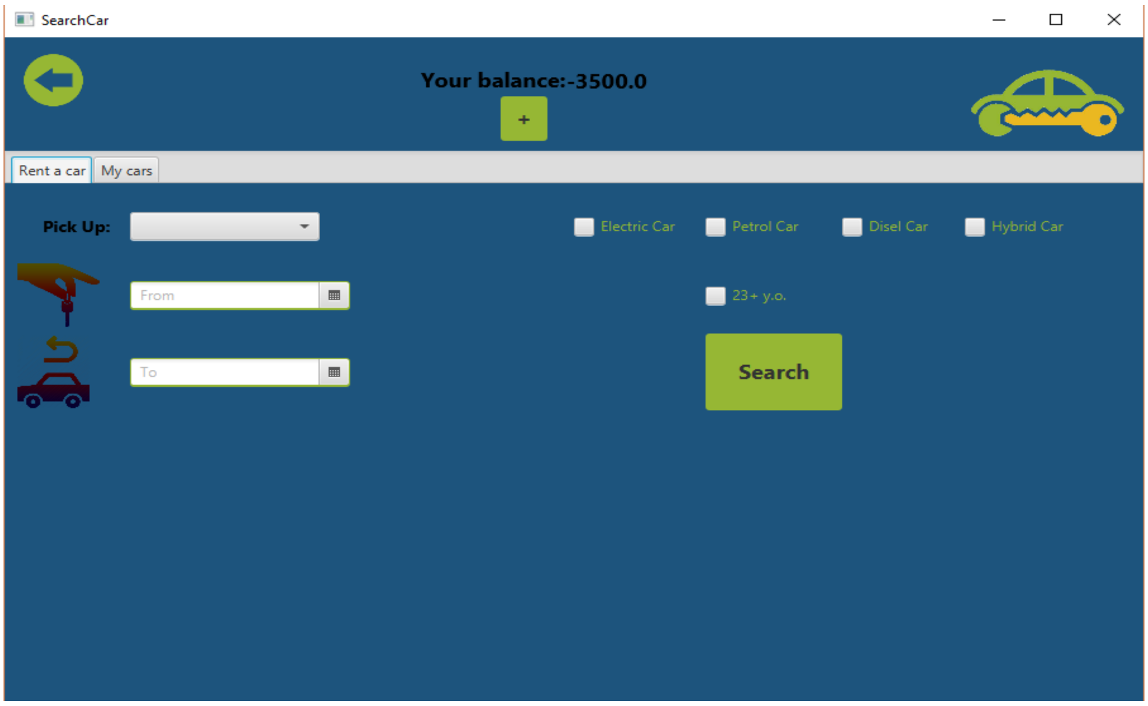
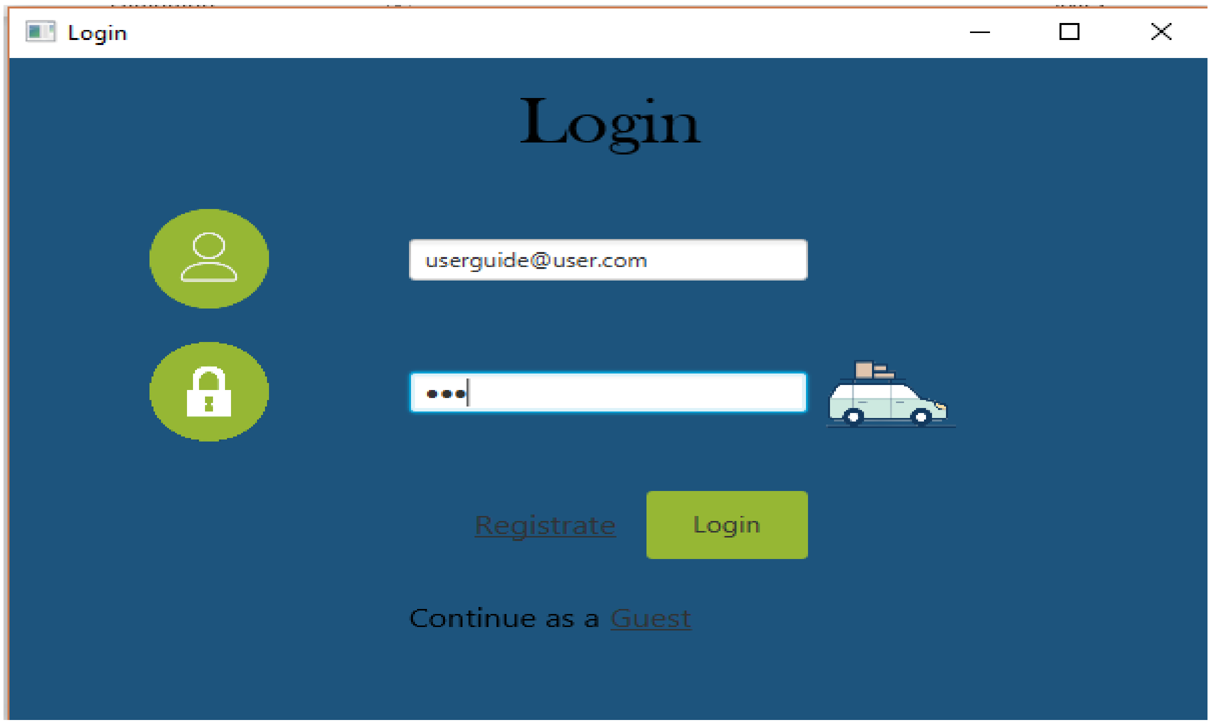


Figure 48 Search Car View

Login as a customer the user should fill in the *mail* and *password* then click "*Login*"



userguide@user.com

Figure 49 Login View Filled out

Now the user ”*userguide@user.dk*” have personal account. The user has the possibilities to adding money in individual wallet. The user could press the “+” button and insert how much money the user wants to put on user balance.

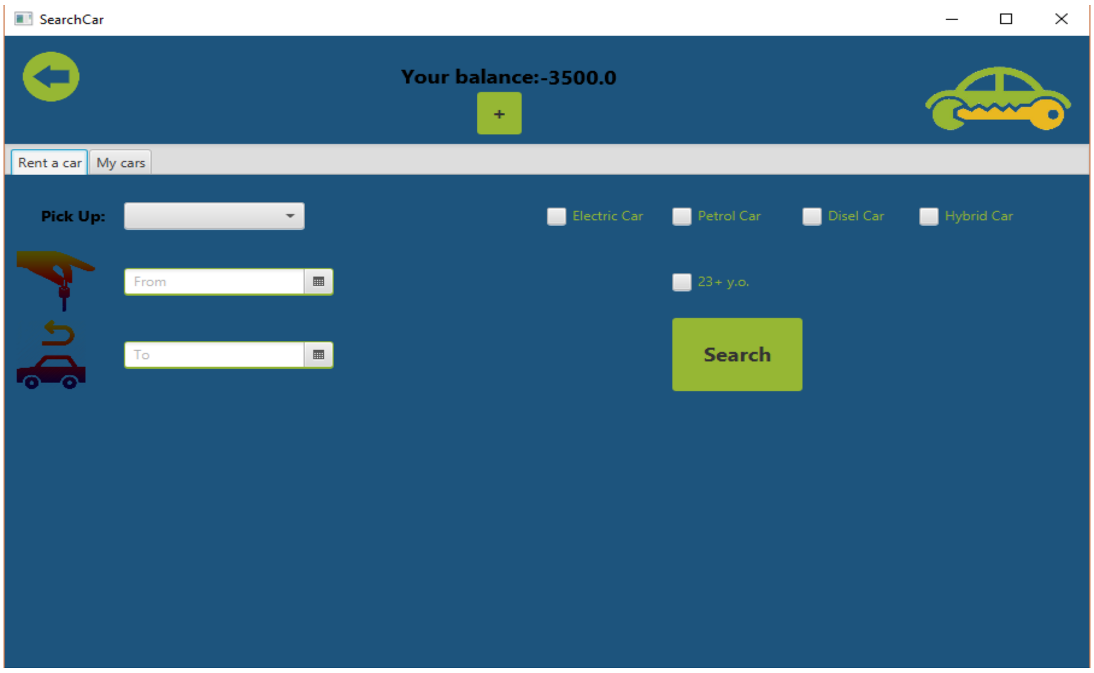


Figure 50 Search Car View

Then click ”*Add*” and system updated the data in database.

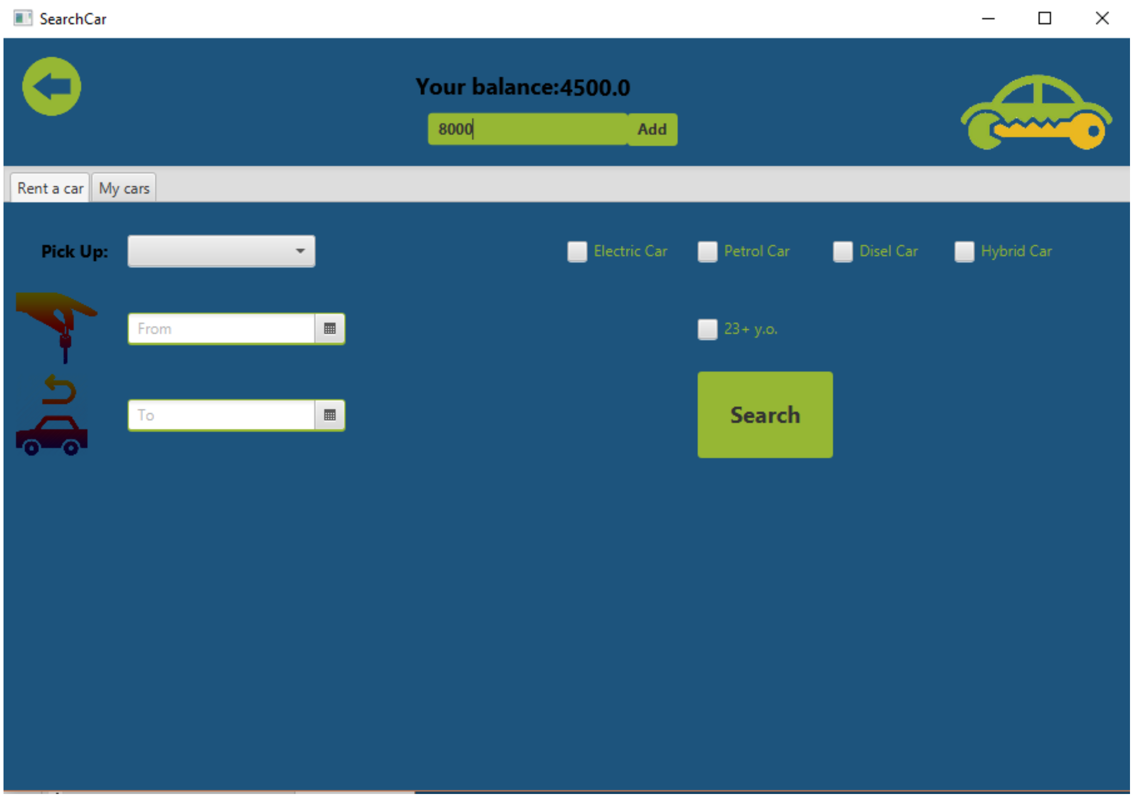


Figure 51 Add money for User

To search a car the customer should fill in the credentials, choose what type of car the user wants and click on “Search” button.

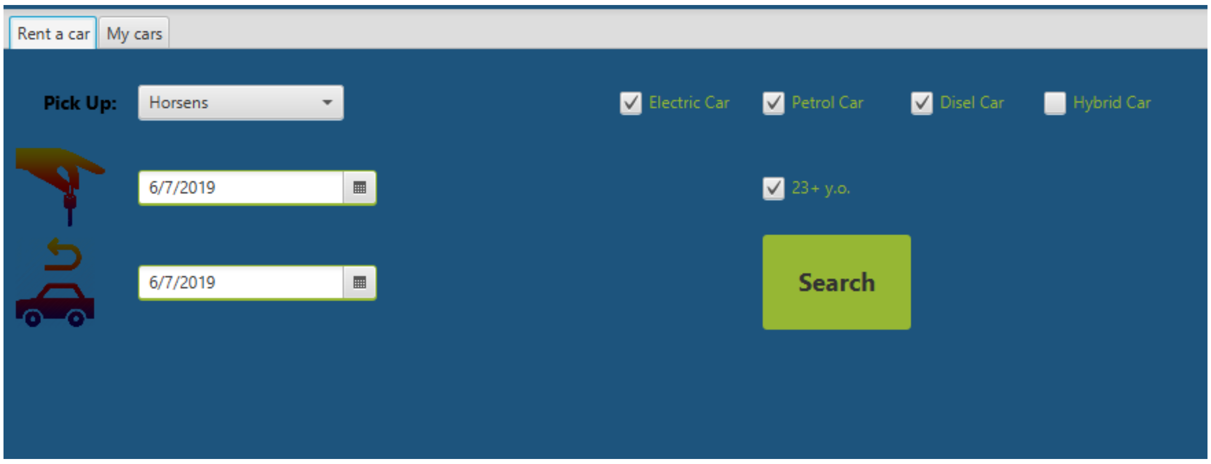
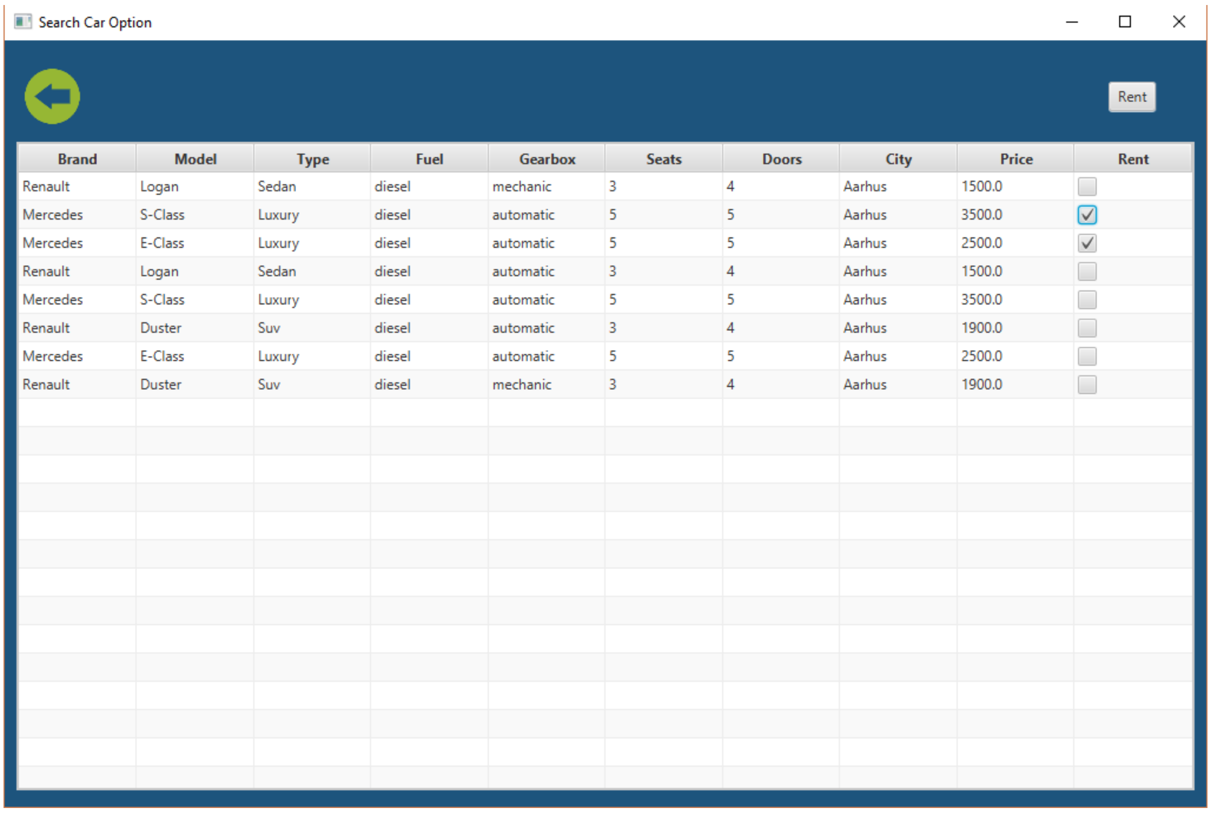


Figure 52 Searching for car

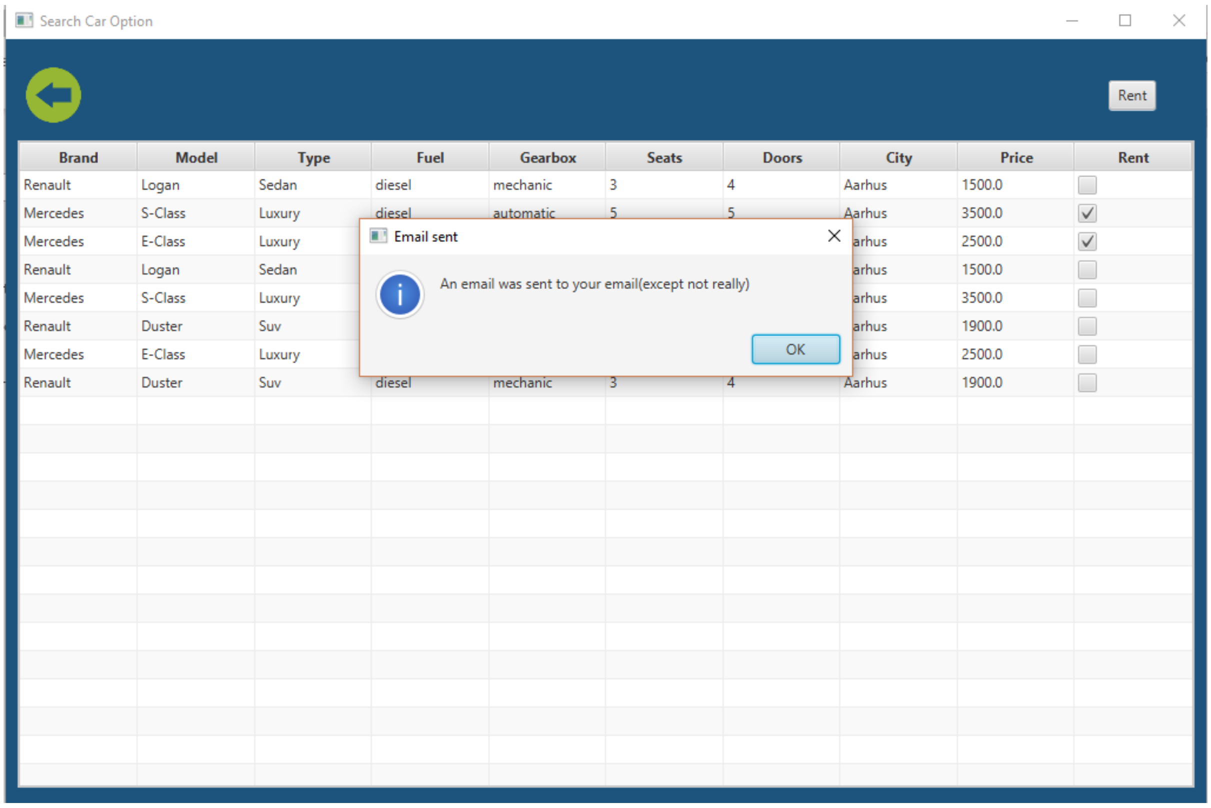
Available cars will appear in a new window, user is able to choose which car the customer wants then click on “Rent” button



Brand	Model	Type	Fuel	Gearbox	Seats	Doors	City	Price	Rent
Renault	Logan	Sedan	diesel	mechanic	3	4	Aarhus	1500.0	☐
Mercedes	S-Class	Luxury	diesel	automatic	5	5	Aarhus	3500.0	✓
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Aarhus	2500.0	✓
Renault	Logan	Sedan	diesel	automatic	3	4	Aarhus	1500.0	☐
Mercedes	S-Class	Luxury	diesel	automatic	5	5	Aarhus	3500.0	☐
Renault	Duster	Suv	diesel	automatic	3	4	Aarhus	1900.0	☐
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Aarhus	2500.0	☐
Renault	Duster	Suv	diesel	mechanic	3	4	Aarhus	1900.0	☐

Figure 53 Renting car

An confirmation window will appear that the system sent an email with booking information about car/s.



Brand	Model	Type	Fuel	Gearbox	Seats	Doors	City	Price	Rent
Renault	Logan	Sedan	diesel	mechanic	3	4	Aarhus	1500.0	☐
Mercedes	S-Class	Luxury	diesel	automatic	5	5	Aarhus	3500.0	✓
Mercedes	E-Class	Luxury					arhus	2500.0	✓
Renault	Logan	Sedan					arhus	1500.0	☐
Mercedes	S-Class	Luxury					arhus	3500.0	☐
Renault	Duster	Suv					arhus	1900.0	☐
Mercedes	E-Class	Luxury					arhus	2500.0	☐
Renault	Duster	Suv	diesel	mechanic	3	4	Aarhus	1900.0	☐

Email sent

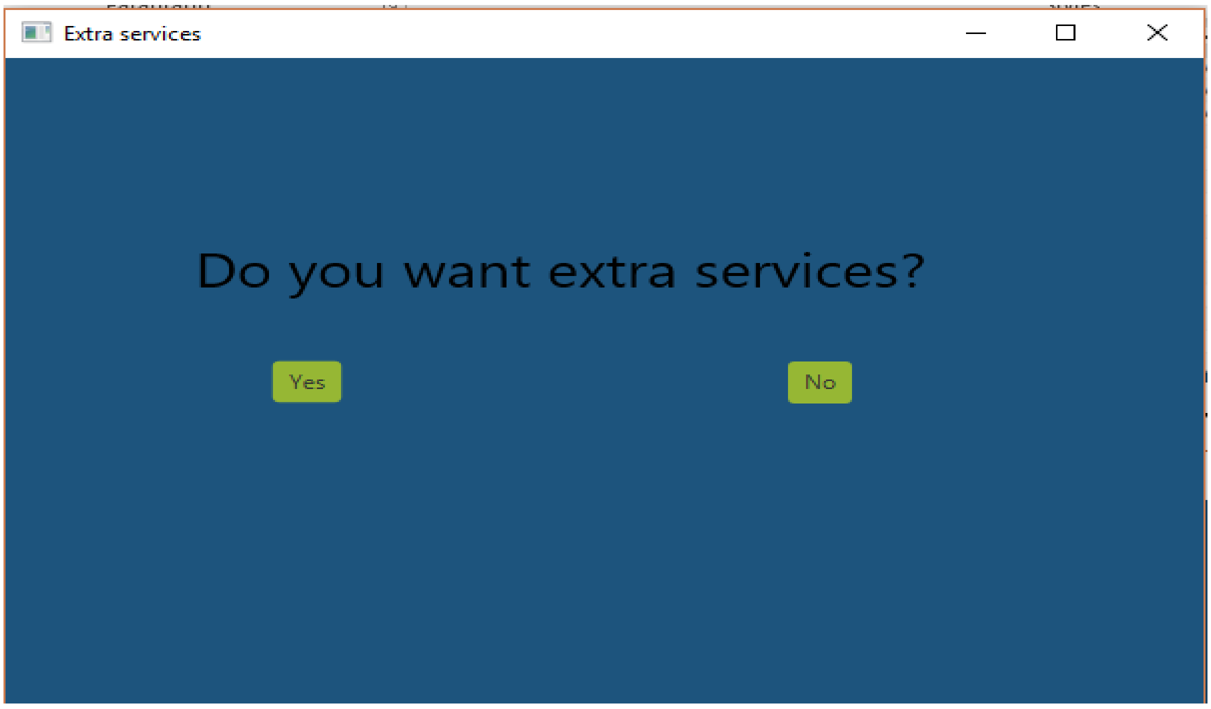
An email was sent to your email(except not really)

OK

Figure 54 Car rented

After pop-up window system will ask if the user wants extra services.

If the user will click NO, the window will close and goes to search a car window.



Do you want extra services?

Figure 55 Extra Services

Now the user is able to choose which extra services the user wishes.

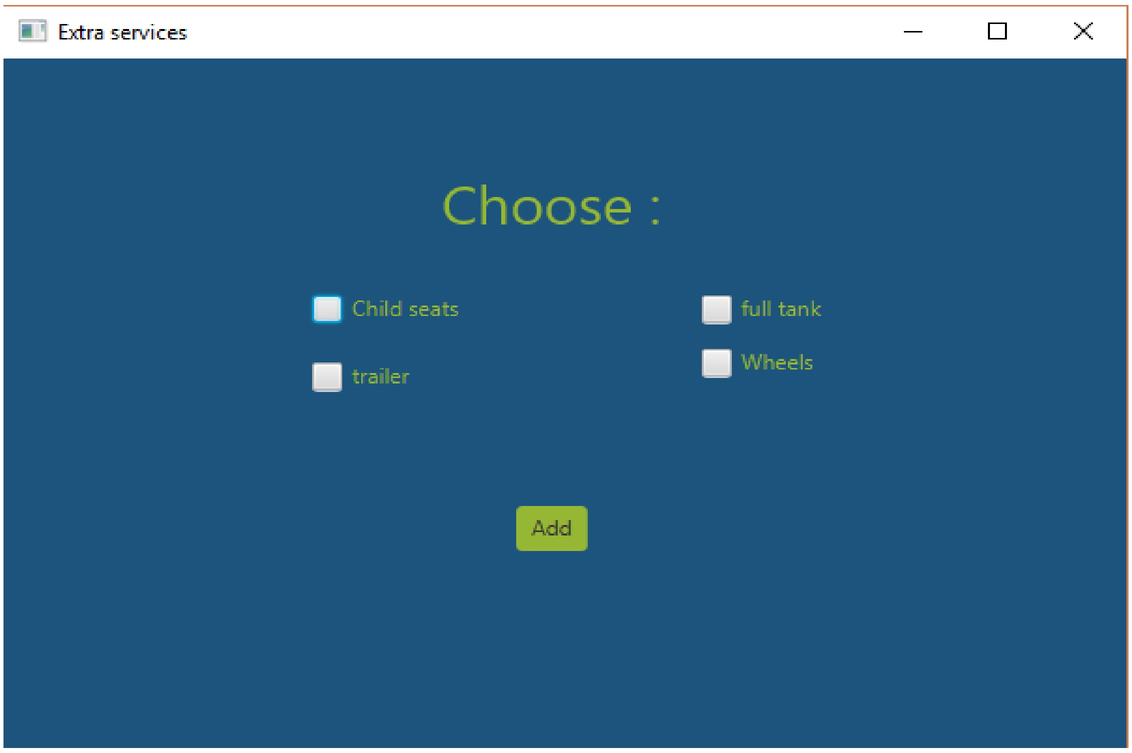
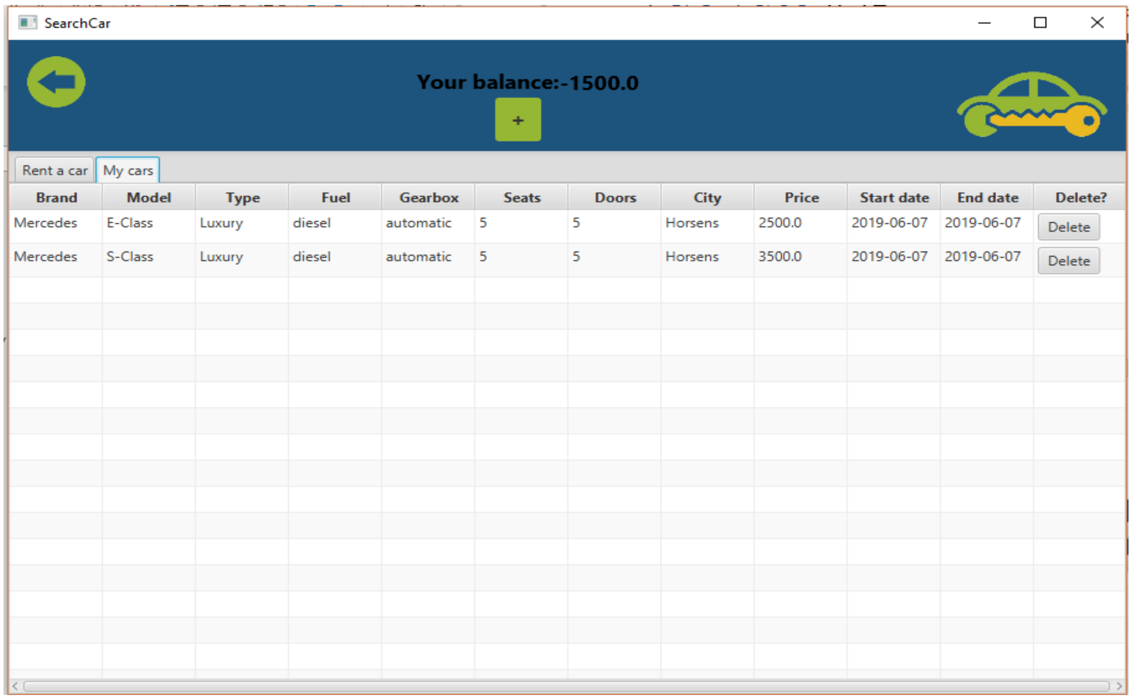


Figure 56 Choose Extra Services

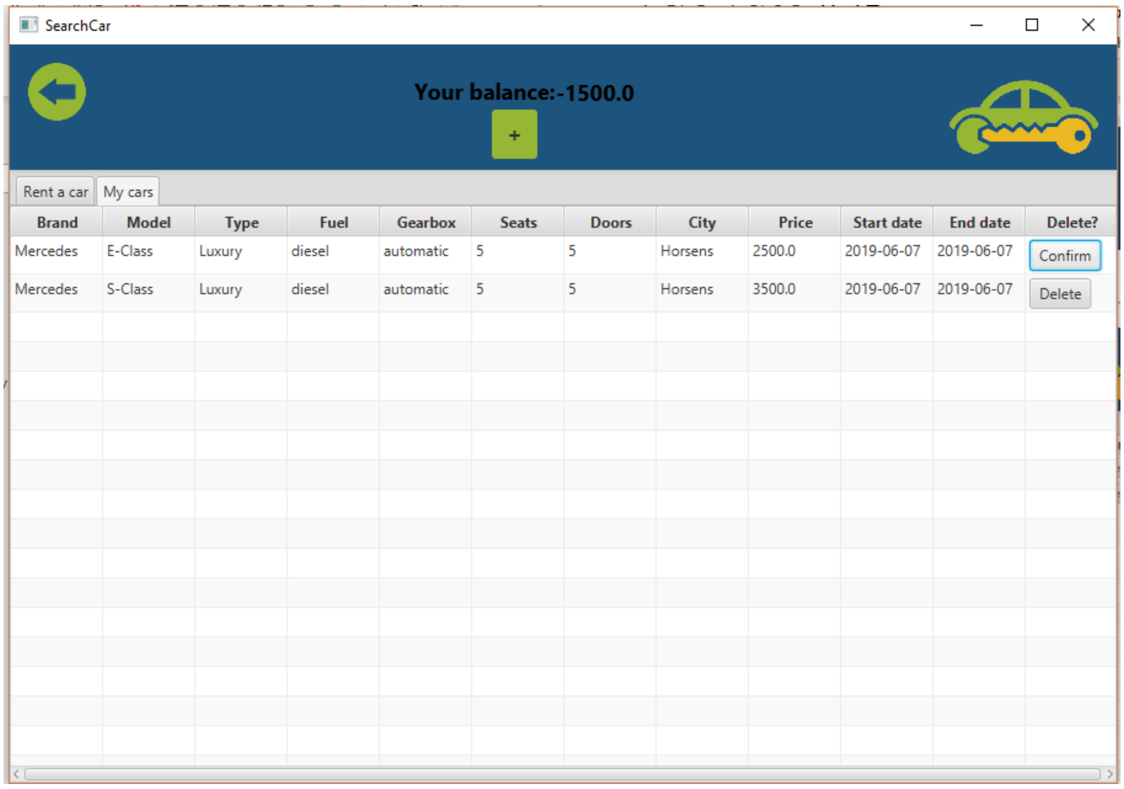
Then the user can see user's cars and if the user needs to delete a car, "*Delete*" button is the solution. One click is not enough the system are waiting the second click to confirm the deleting.



Brand	Model	Type	Fuel	Gearbox	Seats	Doors	City	Price	Start date	End date	Delete?
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Horsens	2500.0	2019-06-07	2019-06-07	Delete
Mercedes	S-Class	Luxury	diesel	automatic	5	5	Horsens	3500.0	2019-06-07	2019-06-07	Delete

Figure 57 Booked cars for User

Second click “*Confirm*”



SearchCar

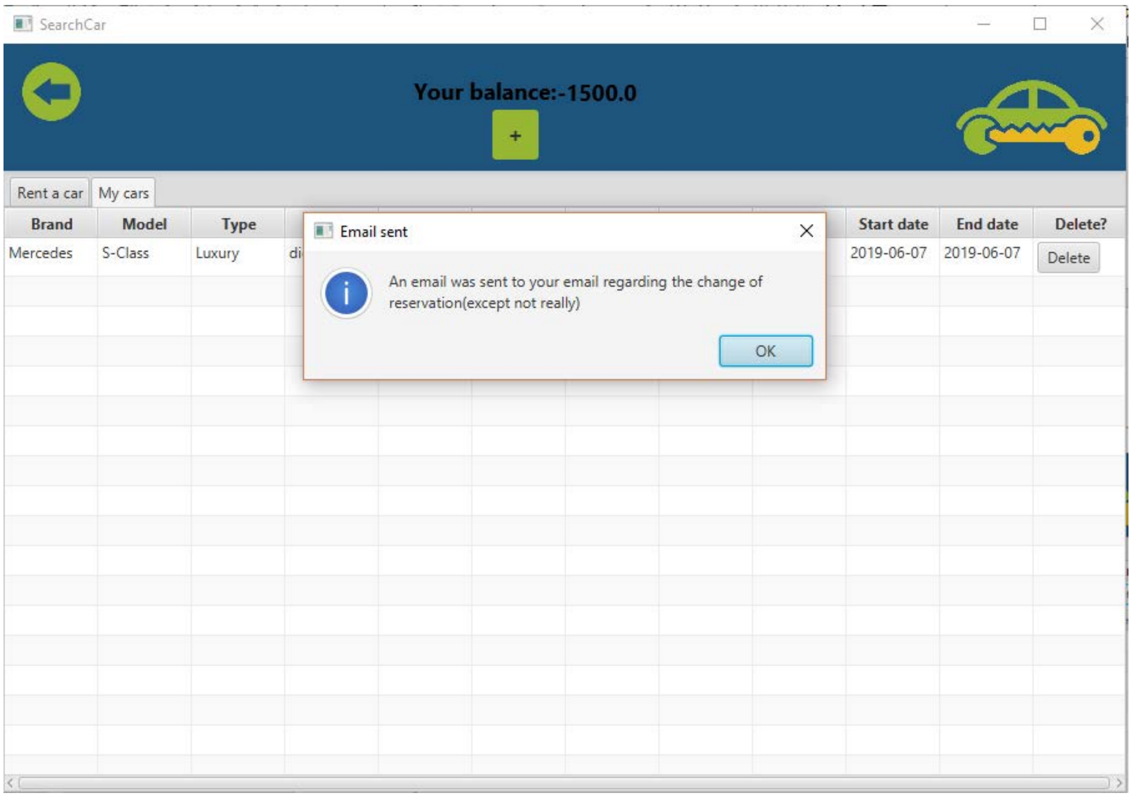
Your balance:-1500.0

Rent a car | My cars

Brand	Model	Type	Fuel	Gearbox	Seats	Doors	City	Price	Start date	End date	Delete?
Mercedes	E-Class	Luxury	diesel	automatic	5	5	Horsens	2500.0	2019-06-07	2019-06-07	Confirm
Mercedes	S-Class	Luxury	diesel	automatic	5	5	Horsens	3500.0	2019-06-07	2019-06-07	Delete

Figure 58 Delete booked car

A new pop-up will appear that the system sent an email that the car was deleted.



SearchCar

Your balance:-1500.0

Rent a car | My cars

Brand	Model	Type	Start date	End date	Delete?
Mercedes	S-Class	Luxury	2019-06-07	2019-06-07	Delete

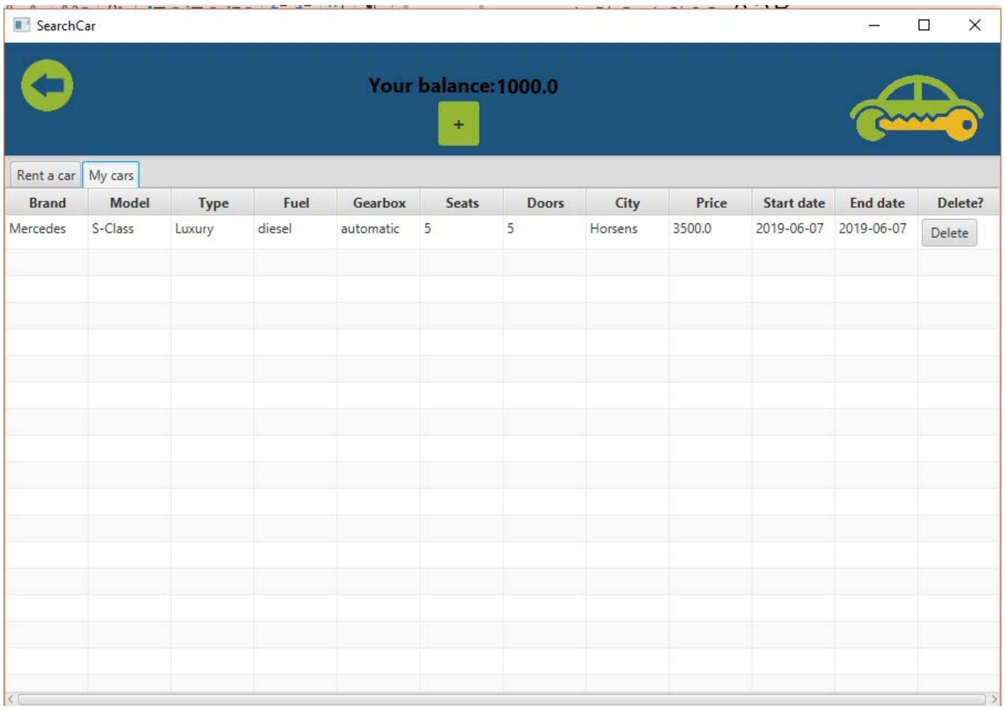
Email sent

An email was sent to your email regarding the change of reservation(except not really)

OK

Figure 59 Pop-up window that car was deleted

Now the user has one car



Brand	Model	Type	Fuel	Gearbox	Seats	Doors	City	Price	Start date	End date	Delete?
Mercedes	S-Class	Luxury	diesel	automatic	5	5	Horsens	3500.0	2019-06-07	2019-06-07	Delete

Figure 60 Updated Table

As a manager

Manager have possibility to login from the same program and to have ability to edit, add, delete cars also the manager can see which cars are booked, available and who booked the car.

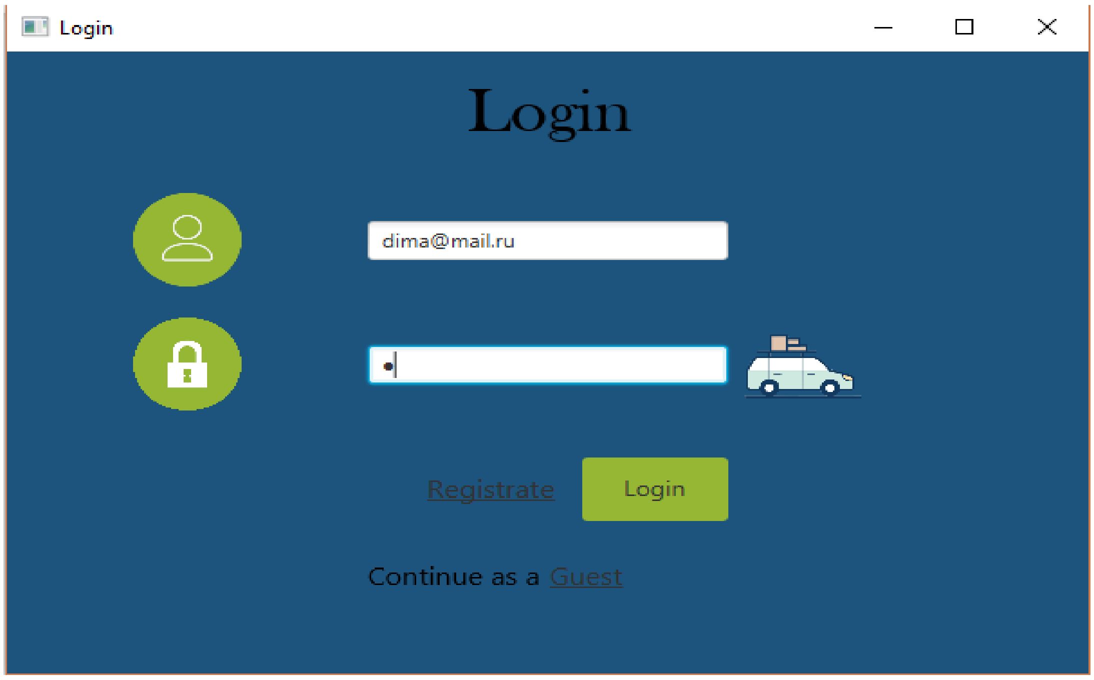
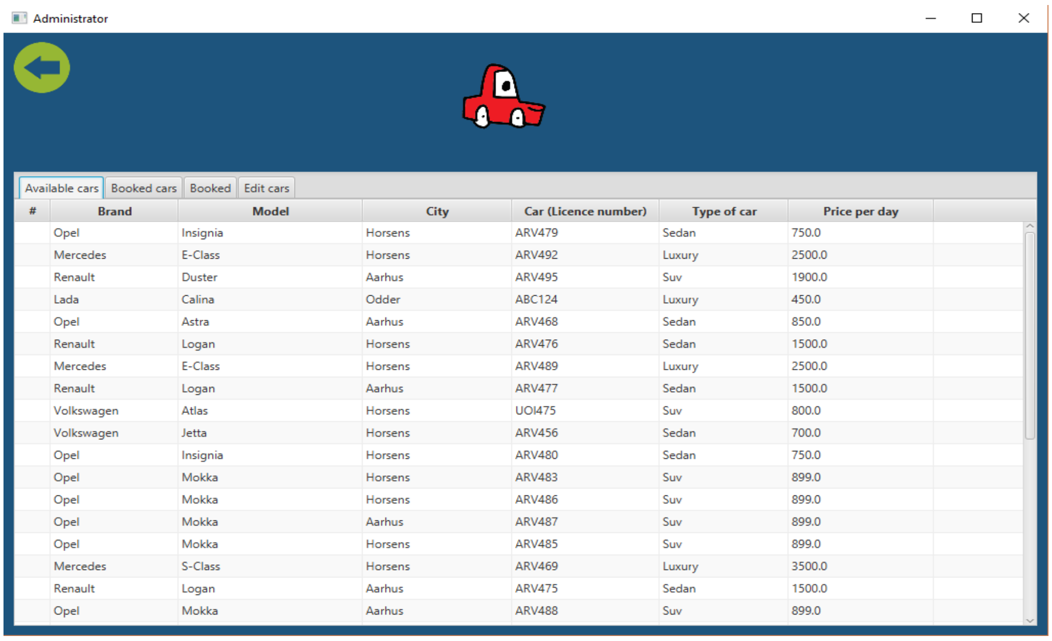


Figure 61 Login as a Manager

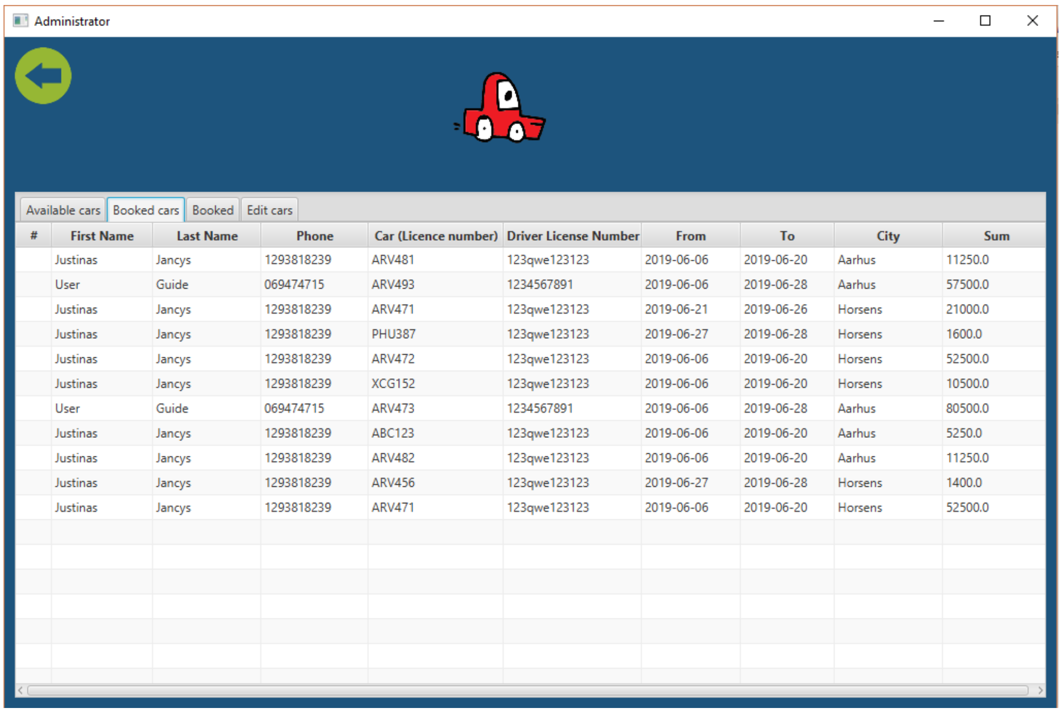
First window as a manager, manager have 4 tabs, “Available cars”, “Booked cars” “Booked” and “Edit cars”.



#	Brand	Model	City	Car (Licence number)	Type of car	Price per day
	Opel	Insignia	Horsens	ARV479	Sedan	750.0
	Mercedes	E-Class	Horsens	ARV492	Luxury	2500.0
	Renault	Duster	Aarhus	ARV495	Suv	1900.0
	Lada	Calina	Odder	ABC124	Luxury	450.0
	Opel	Astra	Aarhus	ARV468	Sedan	850.0
	Renault	Logan	Horsens	ARV476	Sedan	1500.0
	Mercedes	E-Class	Horsens	ARV489	Luxury	2500.0
	Renault	Logan	Aarhus	ARV477	Sedan	1500.0
	Volkswagen	Atlas	Horsens	UOI475	Suv	800.0
	Volkswagen	Jetta	Horsens	ARV456	Sedan	700.0
	Opel	Insignia	Horsens	ARV480	Sedan	750.0
	Opel	Mokka	Horsens	ARV483	Suv	899.0
	Opel	Mokka	Horsens	ARV486	Suv	899.0
	Opel	Mokka	Aarhus	ARV487	Suv	899.0
	Opel	Mokka	Horsens	ARV485	Suv	899.0
	Mercedes	S-Class	Horsens	ARV469	Luxury	3500.0
	Renault	Logan	Aarhus	ARV475	Sedan	1500.0
	Opel	Mokka	Aarhus	ARV488	Suv	899.0

Figure 62 First view as a Manager

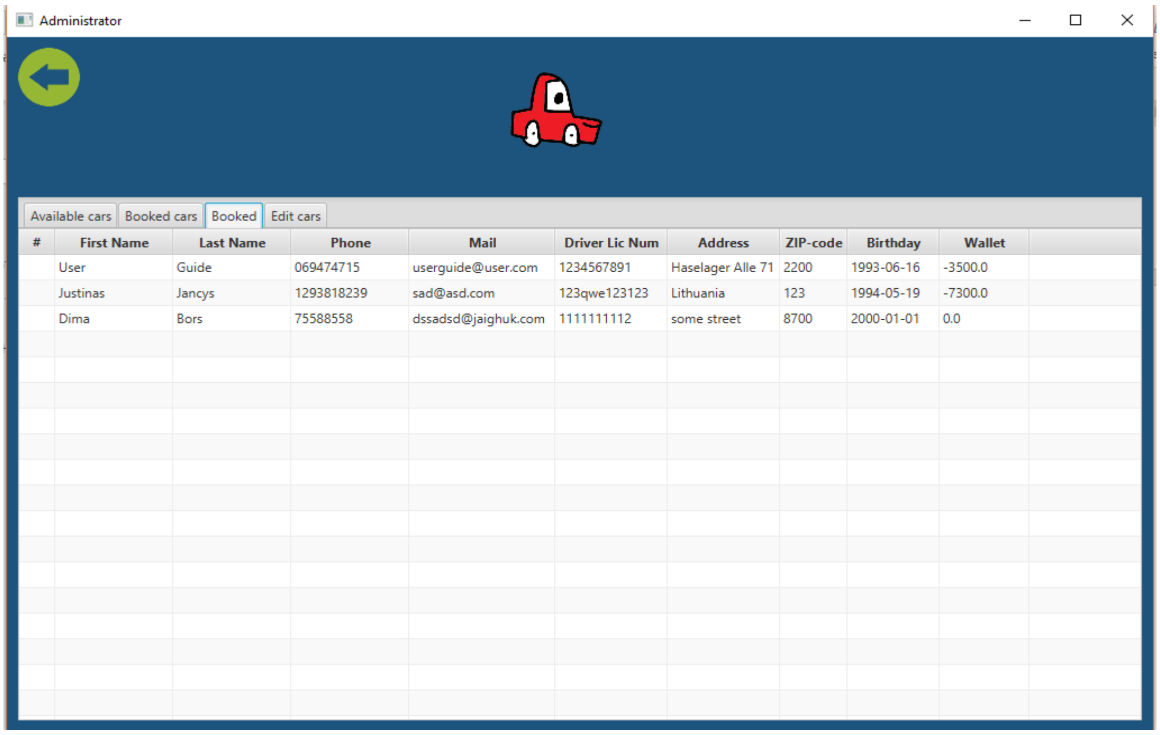
In “*Booked cars*” tab the manager can see how many cars was booked from current day to the future, information about client and car, how many days, which location and sum for all days.



#	First Name	Last Name	Phone	Car (Licence number)	Driver License Number	From	To	City	Sum
	Justinas	Jancys	1293818239	ARV481	123qwe123123	2019-06-06	2019-06-20	Aarhus	11250.0
	User	Guide	069474715	ARV493	1234567891	2019-06-06	2019-06-28	Aarhus	57500.0
	Justinas	Jancys	1293818239	ARV471	123qwe123123	2019-06-21	2019-06-26	Horsens	21000.0
	Justinas	Jancys	1293818239	PHU387	123qwe123123	2019-06-27	2019-06-28	Horsens	1600.0
	Justinas	Jancys	1293818239	ARV472	123qwe123123	2019-06-06	2019-06-20	Horsens	52500.0
	Justinas	Jancys	1293818239	XCG152	123qwe123123	2019-06-06	2019-06-20	Horsens	10500.0
	User	Guide	069474715	ARV473	1234567891	2019-06-06	2019-06-28	Aarhus	80500.0
	Justinas	Jancys	1293818239	ABC123	123qwe123123	2019-06-06	2019-06-20	Aarhus	5250.0
	Justinas	Jancys	1293818239	ARV482	123qwe123123	2019-06-06	2019-06-20	Aarhus	11250.0
	Justinas	Jancys	1293818239	ARV456	123qwe123123	2019-06-27	2019-06-28	Horsens	1400.0
	Justinas	Jancys	1293818239	ARV471	123qwe123123	2019-06-06	2019-06-20	Horsens	52500.0

Figure 63 Booked cars tab

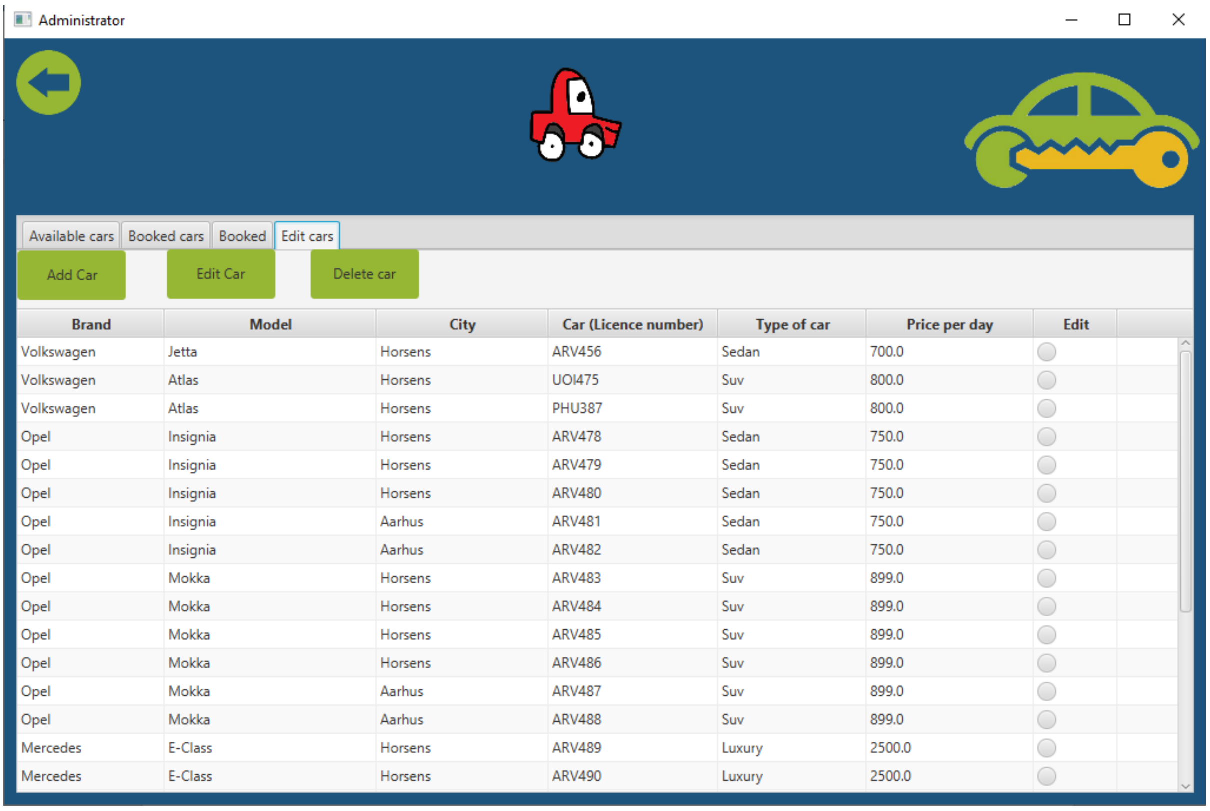
In “Booked” tab the manager can see information about clients, how many clients booked a car minimum one time from the past.



#	First Name	Last Name	Phone	Mail	Driver Lic Num	Address	ZIP-code	Birthday	Wallet
	User	Guide	069474715	userguide@user.com	1234567891	Haselager Alle 71	2200	1993-06-16	-3500.0
	Justinas	Jancys	1293818239	sad@asd.com	123qwe123123	Lithuania	123	1994-05-19	-7300.0
	Dima	Bors	75588558	dssadsd@jaighuk.com	1111111112	some street	8700	2000-01-01	0.0

Figure 64 User that have booked a car

In “Edit Cars” tab the manager is able to add, delete or edit cars.



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Volkswagen	Jetta	Horsens	ARV456	Sedan	700.0	○
Volkswagen	Atlas	Horsens	UOI475	Suv	800.0	○
Volkswagen	Atlas	Horsens	PHU387	Suv	800.0	○
Opel	Insignia	Horsens	ARV478	Sedan	750.0	○
Opel	Insignia	Horsens	ARV479	Sedan	750.0	○
Opel	Insignia	Horsens	ARV480	Sedan	750.0	○
Opel	Insignia	Aarhus	ARV481	Sedan	750.0	○
Opel	Insignia	Aarhus	ARV482	Sedan	750.0	○
Opel	Mokka	Horsens	ARV483	Suv	899.0	○
Opel	Mokka	Horsens	ARV484	Suv	899.0	○
Opel	Mokka	Horsens	ARV485	Suv	899.0	○
Opel	Mokka	Horsens	ARV486	Suv	899.0	○
Opel	Mokka	Aarhus	ARV487	Suv	899.0	○
Opel	Mokka	Aarhus	ARV488	Suv	899.0	○
Mercedes	E-Class	Horsens	ARV489	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV490	Luxury	2500.0	○

Figure 65 Edit cars tab

To add a car the manager needs to press the "*Add*" button. The fill in the credentials. Press a "*Add*" button again and a car will be added to the database.

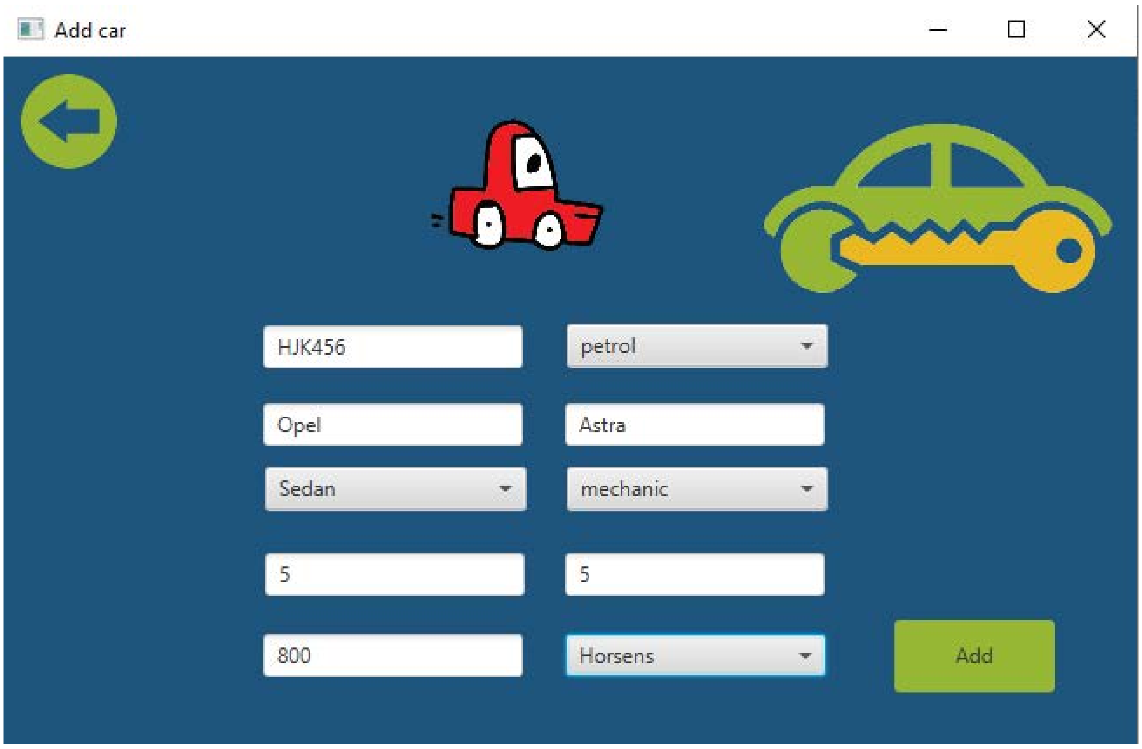
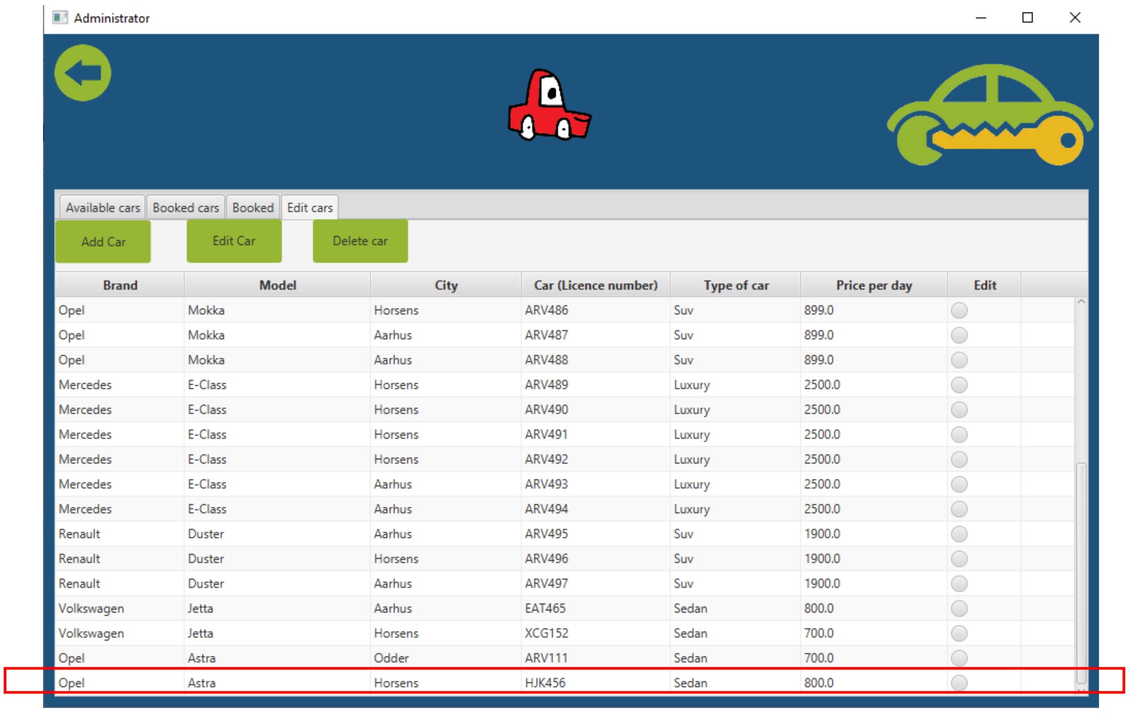


Figure 66 Add a new car

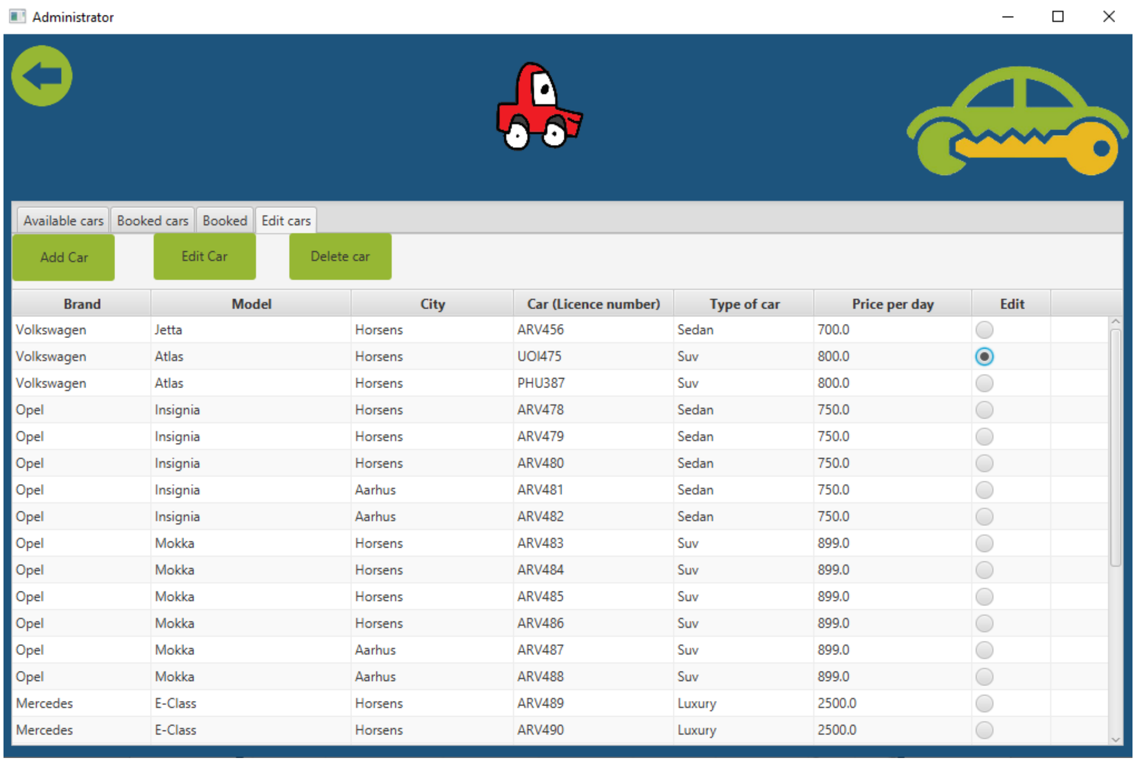
Now the manager can see the updated table with new car in list



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Opel	Mokka	Horsens	ARV486	Suv	899.0	○
Opel	Mokka	Aarhus	ARV487	Suv	899.0	○
Opel	Mokka	Aarhus	ARV488	Suv	899.0	○
Mercedes	E-Class	Horsens	ARV489	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV490	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV491	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV492	Luxury	2500.0	○
Mercedes	E-Class	Aarhus	ARV493	Luxury	2500.0	○
Mercedes	E-Class	Aarhus	ARV494	Luxury	2500.0	○
Renault	Duster	Aarhus	ARV495	Suv	1900.0	○
Renault	Duster	Horsens	ARV496	Suv	1900.0	○
Renault	Duster	Aarhus	ARV497	Suv	1900.0	○
Volkswagen	Jetta	Aarhus	EAT465	Sedan	800.0	○
Volkswagen	Jetta	Horsens	XCG152	Sedan	700.0	○
Opel	Astra	Odder	ARV111	Sedan	700.0	○
Opel	Astra	Horsens	HJK456	Sedan	800.0	○

Figure 67 The table was updated with new car

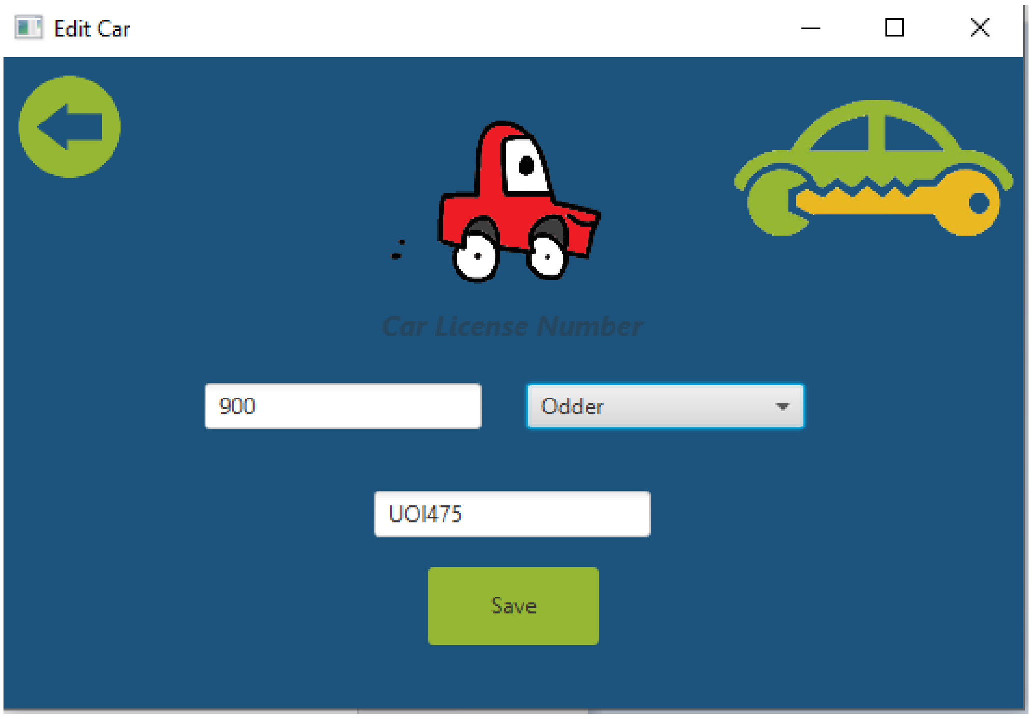
The manager can select a car and edit choose a car click "*Edit car*", then will appear a new window with the old city and the old price.



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Volkswagen	Jetta	Horsens	ARV456	Sedan	700.0	○
Volkswagen	Atlas	Horsens	UOI475	Suv	800.0	◉
Volkswagen	Atlas	Horsens	PHU387	Suv	800.0	○
Opel	Insignia	Horsens	ARV478	Sedan	750.0	○
Opel	Insignia	Horsens	ARV479	Sedan	750.0	○
Opel	Insignia	Horsens	ARV480	Sedan	750.0	○
Opel	Insignia	Aarhus	ARV481	Sedan	750.0	○
Opel	Insignia	Aarhus	ARV482	Sedan	750.0	○
Opel	Mokka	Horsens	ARV483	Suv	899.0	○
Opel	Mokka	Horsens	ARV484	Suv	899.0	○
Opel	Mokka	Horsens	ARV485	Suv	899.0	○
Opel	Mokka	Horsens	ARV486	Suv	899.0	○
Opel	Mokka	Aarhus	ARV487	Suv	899.0	○
Opel	Mokka	Aarhus	ARV488	Suv	899.0	○
Mercedes	E-Class	Horsens	ARV489	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV490	Luxury	2500.0	○

Figure 68 Car selected for edit

Manager will change the price and city and click on "*Save*" button.



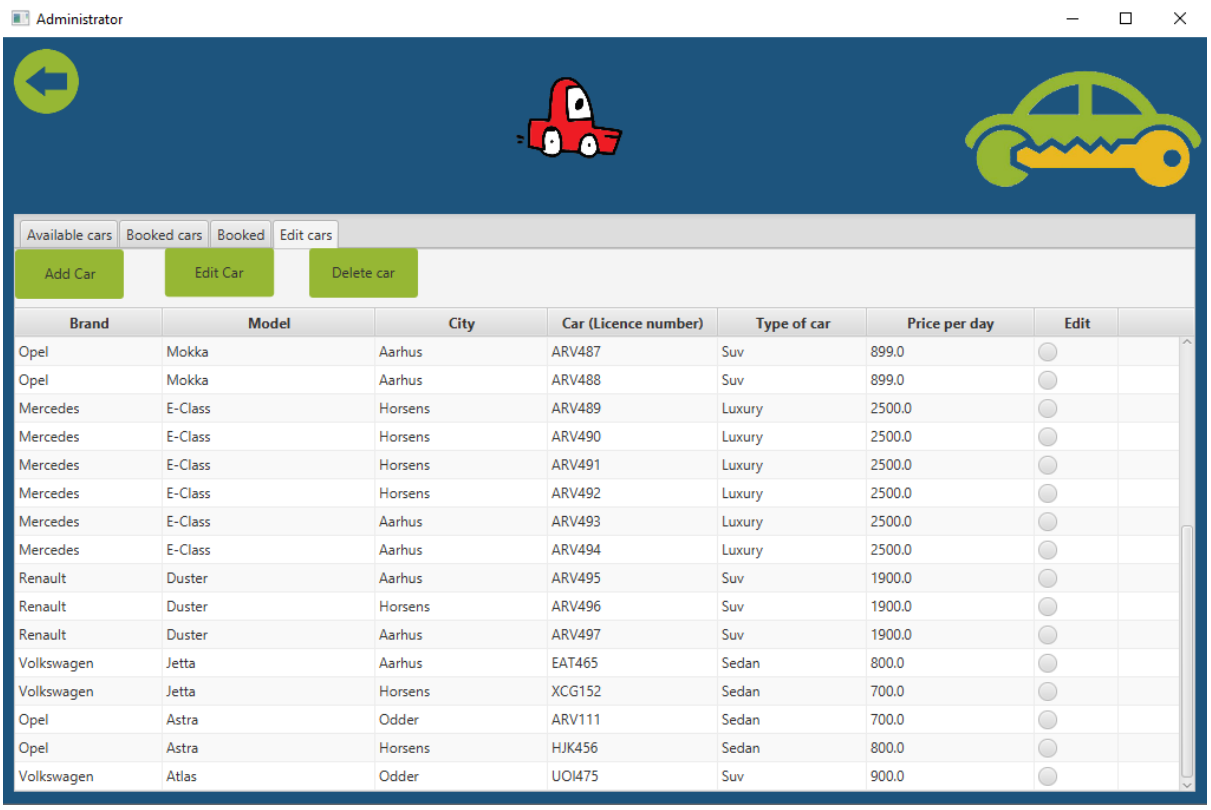
900

Odder

UOI475

Figure 69 Car Edit

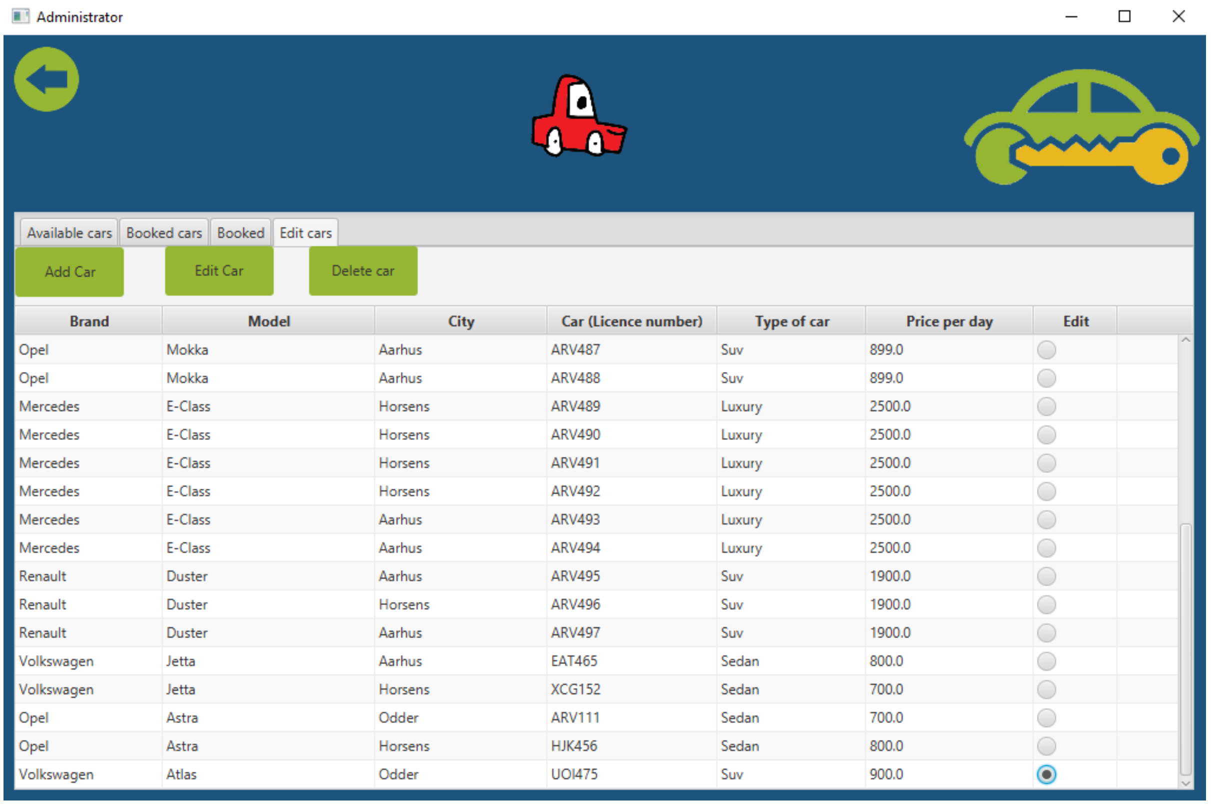
The selected car was edited successfully and added at the bottom of table.



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Opel	Mokka	Aarhus	ARV487	Suv	899.0	○
Opel	Mokka	Aarhus	ARV488	Suv	899.0	○
Mercedes	E-Class	Horsens	ARV489	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV490	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV491	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV492	Luxury	2500.0	○
Mercedes	E-Class	Aarhus	ARV493	Luxury	2500.0	○
Mercedes	E-Class	Aarhus	ARV494	Luxury	2500.0	○
Renault	Duster	Aarhus	ARV495	Suv	1900.0	○
Renault	Duster	Horsens	ARV496	Suv	1900.0	○
Renault	Duster	Aarhus	ARV497	Suv	1900.0	○
Volkswagen	Jetta	Aarhus	EAT465	Sedan	800.0	○
Volkswagen	Jetta	Horsens	XCG152	Sedan	700.0	○
Opel	Astra	Odder	ARV111	Sedan	700.0	○
Opel	Astra	Horsens	HJK456	Sedan	800.0	○
Volkswagen	Atlas	Odder	UOI475	Suv	900.0	○

Figure 70 Updated information about edit car

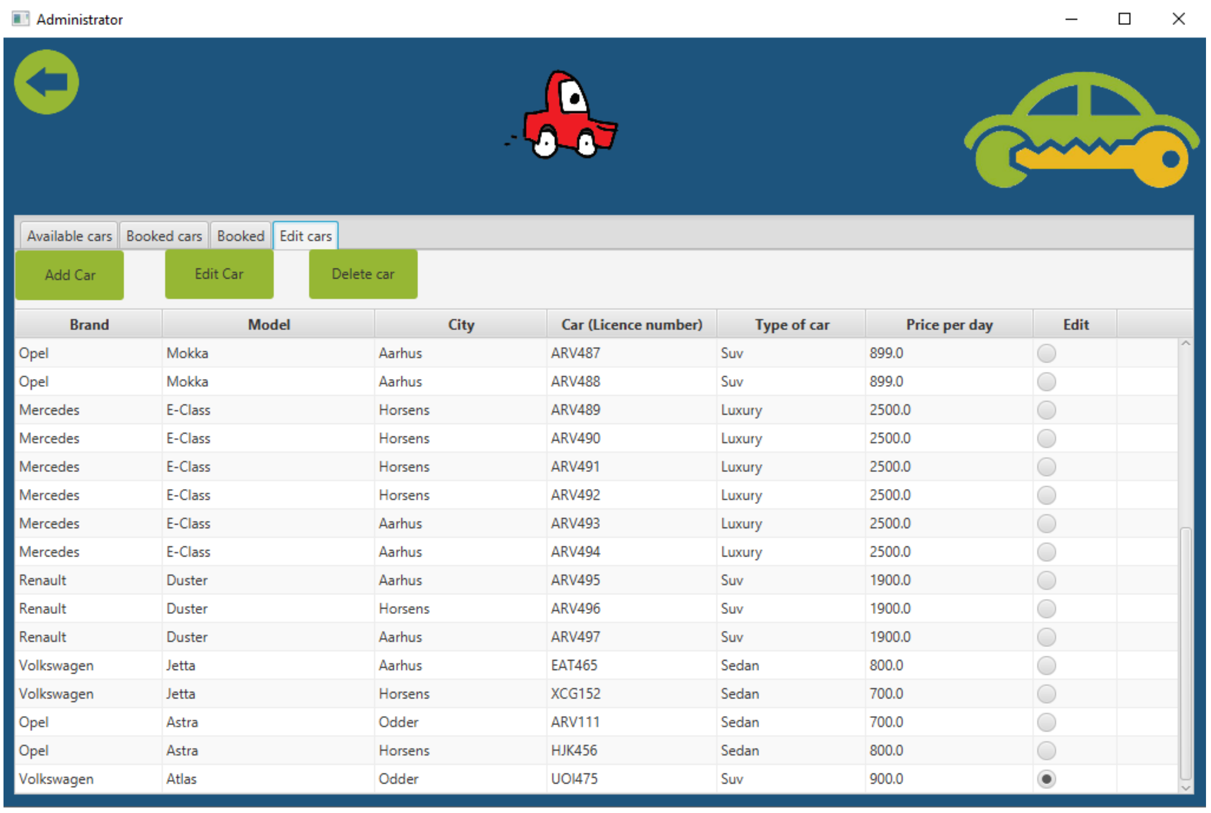
If the manager wants to delete a car then a car should be selected



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Opel	Mokka	Aarhus	ARV487	Suv	899.0	○
Opel	Mokka	Aarhus	ARV488	Suv	899.0	○
Mercedes	E-Class	Horsens	ARV489	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV490	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV491	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV492	Luxury	2500.0	○
Mercedes	E-Class	Aarhus	ARV493	Luxury	2500.0	○
Mercedes	E-Class	Aarhus	ARV494	Luxury	2500.0	○
Renault	Duster	Aarhus	ARV495	Suv	1900.0	○
Renault	Duster	Horsens	ARV496	Suv	1900.0	○
Renault	Duster	Aarhus	ARV497	Suv	1900.0	○
Volkswagen	Jetta	Aarhus	EAT465	Sedan	800.0	○
Volkswagen	Jetta	Horsens	XCG152	Sedan	700.0	○
Opel	Astra	Odder	ARV111	Sedan	700.0	○
Opel	Astra	Horsens	HJK456	Sedan	800.0	○
Volkswagen	Atlas	Odder	UOI475	Suv	900.0	●

Figure 71 Select a car to delete

And press “*Delete*” (Doesn’t work in the last update)



Brand	Model	City	Car (Licence number)	Type of car	Price per day	Edit
Opel	Mokka	Aarhus	ARV487	Suv	899.0	○
Opel	Mokka	Aarhus	ARV488	Suv	899.0	○
Mercedes	E-Class	Horsens	ARV489	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV490	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV491	Luxury	2500.0	○
Mercedes	E-Class	Horsens	ARV492	Luxury	2500.0	○
Mercedes	E-Class	Aarhus	ARV493	Luxury	2500.0	○
Mercedes	E-Class	Aarhus	ARV494	Luxury	2500.0	○
Renault	Duster	Aarhus	ARV495	Suv	1900.0	○
Renault	Duster	Horsens	ARV496	Suv	1900.0	○
Renault	Duster	Aarhus	ARV497	Suv	1900.0	○
Volkswagen	Jetta	Aarhus	EAT465	Sedan	800.0	○
Volkswagen	Jetta	Horsens	XCG152	Sedan	700.0	○
Opel	Astra	Odder	ARV111	Sedan	700.0	○
Opel	Astra	Horsens	HJK456	Sedan	800.0	○
Volkswagen	Atlas	Odder	UOI475	Suv	900.0	●

Figure 72 Updated table after deleted a car