Actor: unregistered user

[image here]

Use case: Register herself/himself on the website

**Description**:

An Unregistered User, that is someone who doesn't have a profile yet, can submit his data, in the website, in the registration page.

After the system has verified and saved all the information, the guest becomes a Registered User, and after he will receive his personal password

**Actors**:

Unregistered User

**Input Condition**:

Null

**Output Condition**:

Unregistered User successfully ends the registration process, and becomes a Registered User. From now on he/she can log in to the application or the website using his/her email address and password, then can access to peculiar functionalities.

**Events flow**:

* Unregistered User on the home page clicks on the “register” button to start the registration process;
* Unregistered User fills all the fields;
* Unregistered User clicks on “DONE” button;
* The system will verify all the inserted data and then will save it in the database;
* The system will generate a password for that new user, which will be sent to him/her.

**Exceptions**:

* The guest has already registered;
* The email inserted is already associated to another user;
* One or more fields are empty;
* At least one of the inserted data are not valid;

All the exceptions cause the application to notify the error to the user with an alert window. The application then come back to the registration form, and the Event Flow will restart from the filling step.

Actor: car !!!!!!!!!!!!!!!A FINE CORSA IL SISTEMA CHIEDE ALLA MACCHINA INFO

[image here]

Use case: Send its position

**Description**:

A Car, can notify the system with its position thanks to his integrated GPS service.

**Actors**:

Car

**Input Condition**:

Request of its position (system) [//ow each 2-5 min]

Incidente??

**Output Condition**:

The system now has the selected car’s position (couple of coordinates).

**Events flow**:

* The System connects to a specific car;
* The System asks for its position;
* Car takes the couple of coordinates with the GPS;
* Car send this information to the system.

**Exceptions**:

* Car send at least one unreal coordinate;

This exception causes the system to notify the car with an error message, and after that the system keeps trying with the position request, and every time it will restart from the asking step.

Use case: Send the level of the battery

**Description**:

A Car, can notify the system with the level of its battery.

**Actors**:

Car

**Input Condition**:

Request of battery level (system) [//ow each 2-5 min]

**Output Condition**:

The system now has percentage of the level of the car’s battery.

**Events flow**:

* The System connects to a specific car;
* The System asks for its battery level;
* Car send this information to the system.

**Exceptions**:

* Car send an unreal value;

This exception causes the system to notify the car with an error message, and after that the system keeps trying with the battery level request, and every time it will restart from the asking step.

Use case: Possible discount for more than one passenger

**Description**:

A Car, can notify the system with the availability of the discount for more than one passenger for at least the half of the rental.

**Actors**:

Car

**Input Condition**:

Request of the possibility of applying a discount (system) [azione singola di conteggio persone]

**Output Condition**:

The system now has the average number of people, who have done the travel. [???]

**Events flow**:

* Il conteggio si azzera ogni nuovo viaggio
* [every time that a person enters or exists from the car, this calculate the average number of people, compared with the time they have spent on the car]
* The System connects to a specific car;
* The System asks for the availability of the discount;
* Car send this information to the system.

**Exceptions**:

* [???]

This exception causes the system to notify the car with an error message, and after that the system keeps trying with the position request, and every time it will restart from the asking step.

Use case: Notify if it’s charging

**Description**:

A Car, notifies the system with a charging message.

**Actors**:

Car

**Input Condition**:

The plug of the power station has been inserted in the car, and the charging phase is starting.

**Output Condition**:

The system receives the message, that contains the power station’s code, too. Therefore, now the system knows if the car is charging, in which station and also if this station is inside a SpecialSafeArea.

**Events flow**:

* The car notice that is in charge;
* It takes the plug information as soon as it starts charging;
* The car connects notify the system with a charging message.

**Exceptions**:

* Car send an unreal value for the plug’s code;

This exception causes the system to notify the car with an error message, and then the car starts this process again (starting from the second phase)

Actor: registered user

[image here]

Use case: Log in

**Description**:

An user already registered can provide his credential (email address and password) to log in to the application, becoming a Registered User and thus gaining some privileges.

**Actors**:

Registered User

**Input Condition**:

The person that wants to log in must be already registered to the system.

**Output Condition**:

The user is now logged in to the webpage or the app as a Registered User and may do lots of actions.

**Events flow**:

* The user fills in the log in form already present in the home page (on the web page or on the mobile application).
* The system verifies the inserted credentials, and if they are correct, let the Registered User to access.

**Exceptions**:

* Credentials are not valid;
* Email address and password don’t match.

In these exception, an alert window will be shown and the access denied, remaining on the home page, and giving the guest the possibility to try again.

Use case: Search for a car

**Description**:

An user, already logged in, can search for a car writing an address or using his GPS position and selecting a radius for the research.

**Actors**:

Registered User

**Input Condition**:

The user must be already logged in. For the GPS position the user must make the research with a device that is able to send the position.

**Output Condition**:

The system

**Events flow**:

* The user fills in the log in form already present in the home page (on the web page or on the mobile application);
* The system search for all the car in that selected area and sends all of the resulting cars to the user;
* The user now sees all the car and the battery level, in the area he/she has.

**Exceptions**:

* User wants to provide the position via GPS, with a device that cannot provide a GPS position;
* User provides an invalid address.

In these cases, an alert window will be shown and the search starts again.

Use case: Reserve a car

**Description**:

A user, after a valid research, can reserve one of the available cars.

**Actors**:

Registered User

**Input Condition**:

The user must have already done a valid research, which have returned at least one car.

**Output Condition**:

The user has an active reservation, that expires in one hour.

**Events flow**:

* User selects one of the available cars and reserves it;
* The system creates a Reservation with the user and the selected car;
* User now have a reservation, that expires in one hour.

**Exceptions**:

* User wants to reserve a car while he/she has already one (underway).

In these case, an alert window will be shown, and returns on the home page.

Use case: Unlock and pick the selected car

**Description**:

A user, while his reservation is valid, can unlock the car and pick it.

**Actors**:

Registered User

**Input Condition**:

A reservation must be valid and the user must be less than 20 meters far from the reserved car.

**Output Condition**:

The system unlocks the car and the user can enter and drive the reserved car.

**Events flow**:

* The user is less than 20 meter far from the car and with his app, push the button to unlock the car
* The system receives the request;
* The system checks the distance between user and car;
* The system unlocks the car and the ride starts;
* The car is now unlocked and the user can enter in it and drive.

**Exceptions**:

* User tries to unlock the car when is more than 20 meters far from the car

In this case, a negation message will be show, telling the user to be closer.

Use case: Set a route on a GPS navigator

**Description**:

A user, whenever he wants, after the Ride is began, can set a route on the car GPS, even if there was a previous planned route (which will be automatically deleted).

**Actors**:

Registered User

**Input Condition**:

Setting a GPS route can be done extending the unlock action.

**Output Condition**:

The user now has a route on GPS navigator.

**Events flow**:

* The user writes an address on the car’s navigator;
* The car computes a route, with the provided data, and shows it on the screen.

**Exceptions**:

* User provides an invalid address.

In these cases, an alert window will be shown and the search starts again.

Use case: Stop the car

**Description**:

Whenever and wherever the user exits from the car, which is locked by the system and the user automatically stops the car.

User stops the car also when he/she push the start/stop button.

In the meantime, the ride still goes on: user will pay for the minute with the stopped car the same as the minutes with the running car.

**Actors**:

Registered User

**Input Condition**:

The action of stopping the car can be done extending the unlock action.

**Output Condition**:

The user’s car is locked [, if he exited; the car is off if has just pushed the button]

**Events flow**:

* The user exits from the car;
* The system locks the car.

**Events flow2**:

* The user pushes the stop button.

Use case: Park and end the rental

**Description**:

A user, whenever he wants, after the Ride is began, can park and end the rental.

**Actors**:

Registered User

**Input Condition**:

Parking and Finishing the rental can be done extending the unlock action.

The user must have previously stopped the car in one of the SafeArea.

**Output Condition**:

The system locks the car and set it available again. It also finishes the user’s ride.

The user from now on, can make a new reservation.

**Events flow**:

* The user stops the car in a Safe Area;
* The user push the button to end the rental;
* The system verify that the car is stopped inside one of the Safe Areas.
* The system stops charging the user;
* At the end the system connects himself with the car to know if there are some applicable discount.

**Exceptions**:

* User tries to end the rental inside the car
* User tries to unlock the car, even though he/she has not stopped the car in a Safe Area

In these exceptions, an alert window will be shown on the app and after the home page appears.

Use case: MONEYSAVINGOPTION

Use case: Drive

**Description**:

A user, whenever he wants, after the Ride is began, can drive and move the car.

**Actors**:

Registered User

**Input Condition**:

Driving the car can be done extending the unlock action.

**Output Condition**:

The user can move the car in the direction he wants.

**Events flow**:

* The user turns on the car’s motor;
* The user start driving.

Use case: Set a route on a GPS navigator

**Description**:

A user, can also recharge the car on a power greed station, always before the ending of the rental.

**Actors**:

Registered User

**Input Condition**:

Recharging the car on a power greed station can be done extending the unlock action.

The ride cannot be terminated when the user start charging the car.

**Output Condition**:

The car is in charge.

**Events flow**:

* The User stops the car;
* The User connect the car with the wire of the power grid station;
* The car start charging;
* User can disconnect the car.

**Exceptions**:

* User tries to charge the car after he/she had pushed the button to end the rental

In this case, the car does not allow to the user to charge, because the system has locked the opening for the plug.

Actor: Employee

[image here]

Use case: change a car’s status

**Description**:

An employee can change manually the status of a car.

**Actors**:

Employee

**Input Condition**:

Some problems appears.

**Output Condition**:

The Car has now a different status.

**Events flow**:

* One problem appear;
* Employee changes the car status manually;
* The system checks the old and the new statuses;
* The car has a new status.

**Exceptions**:

* Employee set a new status that is the same as the last one.

With this exception the system undoes all the operation that the employee has done, letting him start from the begin.

Use case: maintenance

**Description**:

An employee can change manually the status of a car.

**Actors**:

Employee

**Input Condition**:

None.

**Output Condition**:

The Car has been improved, with some maintenance fixings.

**Events flow**:

* Employee can improve the car.

Use case: change a car’s position

**Description**:

An employee can move a care and then change its position.

**Actors**:

Employee

**Input Condition**:

The system asks to him to redistribute some cars.

**Output Condition**:

Moved cars have now a different position.

**Events flow**:

* The system asks to an employee to move some cars;
* The system provides the new positions of the cars;
* The employee moves the cars;
* The employee communicates the end of his/her job;
* The system checks if all the cars are in the right position.

**Exceptions**:

* The employee moves the cars in the wrong position.

With this exception, the system provides again all the car’s positions that are not correct and the process goes on waiting again for a message of a completed job by the employee.