Politecnico di Milano

A.A. 2016–2017

Software Engineering 2: "PowerEnJoy"

*Requirements Analysis and Specification Document*

Pietro Ferretti, Nicole Gervasoni, Danilo Labanca

November 10, 2016

# Contents

# 1 Introduction

## 1.1 Purpose

The purpose of this document is to provide a detailed description and specification of a digital management system for PowerEnJoy, an electric car sharing service. This document will illustrate functional and non-functional requirements of the software to-be, outlining costraints, showing potential user interfaces for the software, and explaining the domain assumptions made. Additionally, at the end of the document we will present an Alloy model to further specificate the world and environment our system will have to manage.

This document is first and foremost a description of the project for all interested stakeholders and a reference for future development, but can also be used as a basis for legally binding agreements.

## 1.2 Scope

The aim of this project is to specify in detail a new digital management software for PowerEnJoy, a car-sharing service that employs electric cars only.

Users must be able to register to PowerEnJoy providing their personal data, driving license and payment information. After registering, users should be able to look for available cars near them or to a specific location. If they find a car that matches their needs, they can reserve the car up to one hour before using it. After approaching a reserved car users should have a way to unlock the car and enter it. While using a car users are charged a certain amount of money depending on distance traveled and time elapsed; while driving the car a display will inform the user of the amount of money they will have to pay at the end of the ride. The system stops charging the user after the car is parked in a safe area. Furthermore the system incentivizes virtuous behaviours by offering several discounts if certain conditions are met (e.g. charging a car at a power grid station).

To accomplish these goals we need to bridge the gap between the world (cars and passengers) and our system (servers, databases, etc.). The users will be able to access the system functionalities through a web and a mobile application, and a dedicated touchscreen installed in every car. A central system will manage the cars and the reservations, and will communicate with the cars through the Internet. The system will follow the cars' movements thanks to the GPS tracker installed in every car. Furthermore all cars will be equipped with a great variety of sensors to check the cars' status at any time.

## 1.3 Definitions, Acronyms, Abbreviations

### 1.3.1 Definitions

### 1.3.2 Acronyms

### 1.3.3 Abbreviations

## 1.4 Reference Documents

- ISO/IEC/IEEE Std. 29148:2011, "Systems and software engineering – Life cycle processes – Requirements engineering"

- Specification document: "Assignments AA 2016-2017.pdf"

## 1.5 Document Overview

# 2 Overall Description

## 2.1 Product Perspective

## 2.2 Product Functions

The PowerEnJoy application will offer many different functionalities; specifically, the goals we want to accomplish are the following:

**[G1]** Guests must be able to register as users by choosing a username and providing their personal data, driving license and payment information. They will receive a password at the email address they specified.

**[G2]** Users must be able to login with the username/email inserted and the password received on registration.

**[G3a]** Users must be able to find the location and battery charge of all the available cars within a certain distance from their current location.

**[G3b]** Users must be able to find the location and battery charge of all available cars around a specified location.

**[G4]** Users must be able to reserve a car for up to one hour before they pick it up. If they don't take the car before the time expires they are charged a fixed fee of 1 EUR.

**[G5]** Users must be able to cancel a reservation if they decide to not actually use the car.

**[G6]** A user that reaches a car reserved by them must have a way to tell the system they're nearby, so that the system unlocks the car and the user may enter.

**[G7]** The system must charge the user who reserved the car from the moment the engine is ignited after unlocking it. The amount charged depends on the distance traveled and time elapsed.

**[G8]** The system must allow the user to see the amount they're being charged through a screen on the car.

**[G9]** The system must stop charging the user as soon as the car is parked in a safe area and the user exits the car notifying the system that they ended their ride.

**[G10]** Users must be able to leave a car without losing the reservation by informing the system that they're only temporarily parking; in this case the system continues on charging the user. The user should be able to end the ride without coming back to the car if they so choose.

**[G11]** The system must lock the car automatically after the user exits the car.

**[G12]** The system must provide a money saving option to the user, proposing a suitable power grid station near the user's final destination. The user will get a discount if they park the car there and plug it in the power grid.

**[G13]** The stations proposed by the system with the money saving option must be chosen in a way that ensures a uniform distribution of cars in the city.

**[G14]** The system must apply a discount of 10% on the last ride if the user took at least two other passengers onto the car.

**[G15]** The system must apply a discount of 20% on the last ride if the car is left with more than 50% of remaining battery charge.

**[G16]** The system must apply a discount of 30% on the last ride if the car is left at special parking areas where they can be recharged, and the user takes care of plugging the car into the power grid.

**[G17]** The system must charge 30% more if the car is left at more than 3 km from the nearest power grid station or with less than 20% remaining battery charge.

**[G18]** Users with outstanding bills cannot have access to the cars and reservations until the bills are paid.

**[G19]** Users with outstanding bills should have a way to pay them.

**[G20]** Available cars should actually be available to the user and in good conditions (i.e. not dirty, damaged, and/or with low battery)

## 2.3 User Characteristics

## 2.4 Constraints

To provide all the functionalities specified in this document, every car must have:

- a reliable GPS tracker

- a mobile Internet connection (for example a 3G or 4G connection)

- sensors to collect data from the engine and the battery

- an electronic switch to lock and unlock the doors

- a weight sensor on each seat

- a touchscreen to offer a dedicated interface to users

- an internal system to control these components and communicate with the central system

- a unique qr code printed on the outside of the car

*and we need a GPS navigator on the car to find power grid stations*

The web application needs an Internet connection to work, for example to connect to the database and find available cars.

The mobile application will need at least an Internet connection to receive data from and communicate with the central system. Access to the device's location is not mandatory, but is needed to provide the "find available cars near me" functionality.

## 2.5 Assumptions and Dependencies

*See section 2.4 for hardware assumptions.*

The system has a predefined list of all the safe areas and their locations.

The system has a predefined list of all the power grid stations and their locations.

The system knows how many plugs are present in each station and how many of them are free in a given moment.

The car's data connection is always working and stable

The car's GPS signal is always available and precise

After unlocking a car users turn on the engine shortly after (i.e. the user doesn't stand in the car idly for long periods of time). We suppose that the user wants to get to their destination as soon as possible.

When the user has ended the reservation or chosen to temporarily park from the car's screen, we assume that the users and all the passengers have exited the car before the last door is closed.

When the weight measured by a seat weight sensor exceeds a certain threshold, it means that a person is present on that seat (and not an object).

Stolen cars are handled by someone else (police, lawyers)

A phone line is available in case of car accidents or damage to the car.

-> insurance, assistance, fines

An external agency is entrusted with the task of charging cars with low or dead batteries.

An external agency is entrusted with the task of regularly cleaning the inside and outside of cars.

Users always report dirty or damaged cars

Users don't let the cars run out of battery before parking in a safe area. (Otherwise they call a phone line and pay a fee)

# 3  Specific Requirements

## 3.1  External Interface Requirements

### 3.1.1  User Interfaces

### 3.1.2  Hardware Interfaces

### 3.1.3  Software Interfaces (API)

### 3.1.4  Communication Interfaces

## 3.2 Functional Requirements

### 3.2.1 Requirements

**[G1]** Guests must be able to register as a user by choosing a username and providing their personal data, driving license and payment information. They will receive a password at the email address they specified.

- [RE.1.1] The system must offer an interface (e.g. a form on a web page or on the mobile application) where the user can enter and submit the data needed for registration: a username of their choice, their personal data (first name, last name, date of birth), a driving license ID or a photo of one, and a payment method.
- [RE.1.2] The system must verify that the data provided by a user on registration is valid.
- [RE.1.3] When a user registers the system must randomly generate a safe password and associate it to the user's account.
- [RE.1.4] After a user is registered the system must send the generated password to the user via email at the email address they provided.

**[G2]** Users must be able to log in with the username/email they submitted on registration and the password they received.

- [RE.2.1] The system must offer an interface (e.g. a form on a web page or on the mobile application) where the user can enter and submit their credentials (username/email and password) to log in.
- [RE.2.2] The system must compare the credentials submitted by the user with the ones saved in the database.
- [RE.2.3] If the credentials submitted on login are valid, the system authenticates the user and allows them to use the available functionalities.
- [RE.2.4] If the credentials submitted on login are not valid, the system denies access to the user and notifies them of the error.

**[G3a]** Users must be able to find the location and battery charge of all the available cars within a certain distance from their current location.

- [RE.3a.1] The system must offer an interface where the user can activate the option to look for nearby available cars if the user's location is available (e.g. thanks to a GPS receiver).
- [RE.3a.2] When the user activates the option to find nearby available cars, the system must get the user's location and look in the database for every available car near them.
- [RE.3a.3] The system must show the results of the search for nearby available cars on a suitable interface (e.g. a map).

**[G3b]** Users must be able to find the location and battery charge of all available cars around a specified location.

- [RE.3b.1] The system must offer an interface where the user can submit an address and search for available cars near that location.
- [RE.3b.2] When the user starts a search for available cars near a certain address, the system must look in the database for every available car close to the address provided by the user.
- [RE.3b.3] The system must show the results of the search for available cars on a suitable interface (e.g. a map).

**[G4]** Users must be able to reserve a car for up to one hour before they pick it up.

- [RE.4.1] The system must offer an interface that shows to the user the cars available for reservation (e.g. the search results specified in RE.3a.3) and allow the user to select a car.
- [RE.4.2] When a user selects a car for reservation the system must mark the car as reserved by that user.
- [RE.4.3] If the car has not been unlocked for an hour after reservation, the system must mark the car as available and charge the user a fixed amount (1 EUR).

**[G5]** Users must be able to cancel a reservation if they decide to not actually use the car.

**[G6]** A user that reaches a car reserved by them must have a way to tell the system they're nearby, so that the system unlocks the car and the user may enter.

- [RE.6.1] The system must offer an interface to unlock the car, with their position or the car's qr code.
- [RE.6.2] If the user is within a certain distance from the car, their device location is available and they choose from the interface to unlock the car with the position, the system must unlock the car.
- [RE.6.3] If the user chooses from the interface to unlock the car with the qr code and the user takes a clear picture of the qr code of the right car, the system must unlock the car.

**[G7]** The system must charge the user who reserved the car from the moment the engine is ignited after unlocking it. The amount charged depends on the distance traveled and time elapsed.

- [RE.7.1] The system must keep track until the end of the reservation of the amount the user has to pay.

- [RE.7.2] The system starts to **charge the user** -> ambiguo, non paga subito the moment the reserved car's engine is ignited after unlocking the car.
- [RE.7.3] The amount charged by the system depends on the time elapsed and the distance traveled by the user, following specific rules.

**[G8]** The system must allow the user to see the amount they're being charged through a screen on the car.

- [RE.7.1] The system must keep track until the end of the reservation of the amount the user has to pay.
- [RE.8.1] After the system has started keeping track of the amount that the user has to pay, the system must show it in a clear way on the car screen while the car is in use and the reservation hasn't ended.

**[G9]** The system must stop charging the user as soon as the car is parked in a safe area and the user exits the car after ending the reservation.

- [RE.7.1] The system must keep track until the end of the reservation of the amount the user has to pay.
- [RE.9.1] The moment a car is parked in a safe area, the system must ask through an interface on the car screen if the user wants to end the reservation or just park temporarily.
- [RE.9.2] If a reserved car has been parked in a safe area and the user has chosen to end the reservation, then after the user has closed the doors the system must carry out the transaction for the amount that the system calculated, with the method of payment specified by the user on registration.

**[G10]** Users must be able to leave a car in a safe area without losing the reservation by informing the system that they're only temporarily parking; in this case the system keeps charging the user.

- [RE.7.1] The system must keep track until the end of the reservation of the amount the user has to pay.
- [RE.9.1] The moment a car is parked in a safe area, the system must ask through an interface on the car screen if the user wants to end the reservation or just park temporarily.
- [RE.10.1] When a car is temporarily parked and locked, the system keeps on charging the user but with different rules than those used in RE.7.3, this time depending only on time elapsed while parked.
- [RE.10.2] After a temporarily parked car is unlocked by the user and the engine is turned on, the system starts charging the user with the rules used in RE.7.3 again.

**[G11]** The system must lock the car automatically after the user exits the car.

- [RE.9.1] The moment a car is parked in a safe area, the system must ask through an interface on the car screen if the user wants to end the reservation or just park temporarily.
- [RE.11.1] The system must lock the car when the last door has been closed after the user chose to end their reservation or park temporarily.

**[G12]** The system must provide a money saving option to the user, proposing a suitable power grid station near the user's final destination. The user will get a discount if they park the car there and plug it in the power grid.

- [RE.12.1] The system must offer an interface where the user can select the money saving option and submit their final destination.
- [RE.12.2] When the user submits their final destination for the money saving option, the system must find a suitable power grid station near the ones close to the destination that have at least one free spot.
- [RE.12.3] The system must show on the car's navigator the location of the power grid station chosen for the money saving option and how to get there.
- [RE.12.4] The system must apply a discount on the amount the user has to pay if they selected the power saving option and they parked in the station the system chose for them.

**[G13]** The stations proposed by the system with the money saving option must be chosen in a way that ensures a uniform distribution of cars in the city.

- [RE.13.1] To choose a station in the process specified in RE.12.2, the system selects among all the stations with free spots within a certain distance of the user's final destination (e.g. a 10-minutes walk) the one with the fewest available cars nearby.

**[G14]** The system must apply a discount of 10% on the last ride if the user took at least two other passengers onto the car.

- [RE.7.1] The system must keep track until the end of the reservation of the amount the user has to pay.
- [RE.14.1] If the weight sensors in the car seats detect at least two passengers (excluding the driver) for at least half of the time the car has been driving, then the system must apply a discount of 10% at the end of the reservation on the amount the user has to pay.

**[G15]** The system must apply a discount of 20% on the last ride if the car is left with more than 50% of remaining battery charge.

- [RE.7.1] The system must keep track until the end of the reservation of the amount the user has to pay.

- [RE.15.1] When the car has been parked in a safe area and the user has chosen to end the reservation, the system must apply a discount of 20% on the amount the user has to pay if the car is left with more than 50% remaining battery charge.

**[G16]** The system must apply a discount of 30% on the last ride if the car is left at special parking areas where they can be recharged, and the user takes care of plugging the car into the power grid.

- [RE.7.1] The system must keep track until the end of the reservation of the amount the user has to pay.
- [RE.16.1] When the car has been parked in a power grid station, has been plugged in the power grid and the user has chosen to end the reservation, the system must apply a discount of 30% on the amount the user has to pay.

**[G17]** The system must charge 30% more if the car is left at more than 3 km from the nearest power grid station or with less than 20% remaining battery charge.

- [RE.7.1] The system must keep track until the end of the reservation of the amount the user has to pay.
- [RE.17.1] When the user has ended the reservation, if the car is parked more than 3 km from the nearest power grid station or has less than 20% remaining battery charge, the system must increase the amount the user has to pay by 30%.

**[G18]** Users with outstanding bills cannot have access to the cars and reservations until the bills are paid.

- [RE.18.1] If a transaction fails the user must be marked as suspended until the outstanding bill is paid.
- [RE.18.2] Suspended users must not be allowed to reserve cars.

**[G19]** Users with outstanding bills should have a way to pay them.

- [RE.19.1] Users must have an option on the application to pay their outstanding bills.
- [RE.19.2] When the user chooses to pay their outstanding bills, if the transaction succeeds they are no longer marked as suspended, otherwise the user is notified and they remain suspended.
- [RE.19.3] Users must have a way to change the method of payment saved on the system through a dedicated option on the application.

**[G20]** Available cars should actually be available to the user and in good conditions (i.e. not dirty, damaged, and/or with low battery)

- [RE.20.1] Users should have a way to report problems with a car they reserved (excessive dirtiness, damage, etc.) through the interface on the car's display.

- [RE.20.2] If a user reports a car for dirtiness or damage, the system will mark it as unavailable, end the user's reservation and notify an employee to solve the problem.

- [RE.20.3] If a car has less than 20% battery charge at the end of a trip, the system will mark the car as unavailable and notify the employees in charge of recharging cars.

- [RE.20.4] If a car is unlocked by force or moves without any reservation, the system will consider it stolen and mark it as unavailable. The system notifies the people in charge of handling theft.

- [RE.20.5] If a car is repaired, cleaned or otherwise fixed, an employee will mark it as available through a dedicated interface.

## 3.3 Use Cases

### 3.3.1 A guest registers to PowerEnJoy.

| | |
|---|---|
| Actor | Guest |
| Goal | [G1] |
| Preconditions | The guest is not already registered |
| Execution Flow | |

1. The guest is on the home page and clicks on the "register" button to start the registration process.

2. The guest chooses between "Sign up with Google", "Sign up with Facebook" or inserting the data manually.

3. In the case of manually inserting, the guest fills in at least all mandatory fields with the required information (first name, last name, username, email address, DOB).

4. In the other cases, the guest connects to Google or Facebook and gives the system permission to take the data from there.

5. The guest uploads a photo of his driving license or inserts the information manually.

6. The guest inserts the number of the credit card and the corresponding CVV.

7. The system verifies the correctness of the inserted data.

8. The guest gives their consent to the processing of their personal data by checking a box.

9. The guest clicks on the "confirm" button and submits the data.

10. The system generates a password and sends it to the user's email address.

11. The system saves the data in the DB.

12. The system informs the user of the successful registration and redirects the user to the profile management page.

| Postconditions | The guest successfully completes the registration process and become a user. From now on he or she can log in to the application using his credentials and use PowerEnJoy. |
|---|---|
| Exceptions | |

1. The guest is already registered.

2. The guest inserts invalid information.

3. The guest inserts a username used by another user.

4. The guest inserts an email address used by another user.

5. The guest doesn't give their consent to the processing of personal data.

Each exception is handled by warning the guest of the problem and letting them insert the data again.

### 3.3.2 A user logs in to the PowerEnJoy application.

| Actor | User |
|---|---|
| Goal | [G2] |
| Preconditions | The user must be registered in the system. |
| Execution Flow | |

1. The guest opens the PowerEnJoy application and presses on the login button.

2. The guest inserts the username or email and password received during registration and submits them.

3. The system checks the credentials submitted by the user.

4. The user is authenticated and redirected to the page where he can look for a car.

| Postconditions | The user is now logged in and can use all the functionalities of the system. |
|---|---|
| Exceptions | |

1. The guest submits invalid credentials.

### 3.3.3 A user searches for available cars near his position.

| | |
|---|---|
| Actor | User |
| Goal | [G3a] |
| Preconditions | The user is logged in to the system and the GPS is active. |
| Execution Flow | |

1. The user presses the button to be localized on the map.

2. The system receives the user's position and searches the DB for all the available cars close to that position.

3. The application shows on a map all the cars that were found.

4. The user navigates on the map to choose a car.

5. The user selects a car he or her is interested in and sees its battery charge.

| | |
|---|---|
| Postconditions | The user can see all the available cars and their battery charge. |
| Exceptions | |

1. There aren't any available cars, in this case the system will suggest to search in a different location.

### 3.3.4   A user searches for available cars in a specific position.

| | |
|---|---|
| Actor | User |
| Goal | [G3b] |
| Preconditions | The user is logged in to the system |
| Execution Flow | |

1. The user presses the search bar to insert a location.

2. The user inserts an address or a name of a place

3. The system receives the user's input.

4. The system interprets the address or place's name and converts it into a position.

5. The system checks in the DB all the available cars close to that position.

6. The application shows on a map all the cars that were found.

7. The user navigates on the map to choose a car.

8. The user selects a car he or her is interested in and sees its battery charge.

| | |
|---|---|
| Postconditions | The user can see all the available cars and their battery charge. |
| Exceptions | |

1. The address inserted by the user doesn't exist.

2. There aren't any available cars, in this case the system will suggest to search in a different location.

### 3.3.5  A user reserves a car.

| | |
|---|---|
| Actor | User |
| Goal | [G4] |
| Preconditions | The user is logged in and there is at least an available car. |
| Execution Flow | |

1. The user selects a car on the map.

2. The system shows to the user the battery remaining charge.

3. The user confirms the selected car as their choice for reservation.

| | |
|---|---|
| Postconditions | The car is reserved for the user for an hour. |
| Exceptions | |

1. The car is reserved by an another user before the user confirms the reservation.

2. The user is suspended by the system. In this case, the application doesn't let them reserve the car and reminds them that they have to pay the pending bill(s).

### 3.3.6  A user cancels a reservation for a car.

| | |
|---|---|
| Actor | User |
| Goal | [G4] |
| Preconditions | The user is logged in, has reserved a car and hasn't unlocked the car yet. |
| Execution Flow | 1. The user goes to the main page of the application. 2. The user presses the "Cancel reservation" button. 3. The user confirms that they want to cancel the reservation. 4. The system marks the car as available again. |
| Postconditions | The car is available and the user has no pending reservation. |
| Exceptions | 1. The reservation has already expired. The system charges the user a fee of 1 EUR and cancels the reservation. |

### 3.3.7 A user unlocks the car with the QR code printed on the car.

| | |
|---|---|
| Actor | User |
| Goal | [G6] |
| Preconditions | The user is logged in and close to the car he reserved. |
| Execution Flow | |

1. The user presses on the camera button, takes a picture of the QR code printed on the car and submits it to the system.

2. The system identifies the car with the QR code and checks the reservation.

3. The system enables the button to unlock the car on the application.

4. The user presses the button.

5. The system unlocks the car.

| | |
|---|---|
| Postconditions | The car unlocked and ready to be started. |
| Exceptions | |

1. The user sends a QR code of a car he didn't reserve.

### 3.3.8 A user unlocks the car using his position.

| | |
|---|---|
| Actor | User |
| Goal | [G6] |
| Preconditions | The user is close to the car he reserved and has the localization activated. |
| Execution Flow | |

1. The user presses on the localization button and sends his position to the system.

2. The system checks if the user's position is close to the car's.

3. The system enables the button to unlock the car on the application.

4. The user presses the button.

5. The system unlocks the car.

| | |
|---|---|
| Postconditions | The car is unlocked and ready to started. |
| Exceptions | |

1. The user is too far from the car he reserved

### 3.3.9   A user ends a ride.

| | |
|---|---|
| Actor | User |
| Goal | [G9] |
| Preconditions | The user is using the car. |
| Execution Flow | |

1. If the user is driving he or her stops the car and turns the engine off.

2. The system checks if the car's position corresponds to a safe area.

3. The display shows to the user two options:

   **a)** to end the ride;

   **b)** to park the car temporarily.

4. The user presses the button a).

5. The user exits the car and closes the doors.

6. The system locks the car.

7. The system stops charging the user.

8. The system sets the car as available.

9. The system carries out the payment transaction.

| | |
|---|---|
| Postconditions | The car is stopped in a safe area, available and ready to be reserved again. |
| Exceptions | |

- The car is not parked in a safe area. In this case the system tells the user the closest safe areas to park in.

- The user drives away instead of exiting the car. In this case the process is canceled.

### 3.3.10  A user parks the car without ending the ride.

| | |
|---|---|
| Actor | User |
| Goal | [G10] |
| Preconditions | The user is using the car. |
| Execution Flow | |

1. If the user is driving he or she stops the car and turns the engine off.

2. The system checks if the car's position corresponds to a safe area.

3. The display shows to the user two options:

   **a)** to end the ride;

   **b)** to park the car temporarily.

4. The user presses the button b).

5. The user exits the car and closes the doors.

6. The system locks the car automatically.

| | |
|---|---|
| Postconditions | The car is stopped in a safe area, ready to be unlocked by the same user. |
| Exceptions | |

- The car is not parked in a safe area. In this case the system tells the user the closest safe areas to park in.

- The user drives away instead of exiting the car. In this case the stop is canceled.

### 3.3.11 The system suggests to the user a PGS to park the car at and save money.

| Actor | System, User |
|---|---|
| Goal | [G12] |
| Preconditions | The user is using the car. |

**Execution Flow**

1. The user inserts the destination on the car's navigator.

2. The user chooses the option "save money!"

3. The system calculates the best solution, so it searches the nearest PGS to the destination keeping in mind a uniform distribution of the cars.

4. The system suggests the PGS to the user through the display.

5. The user drives to the PGS, parks the car and turns the engine off.

6. The user plugs the car in the power grid.

7. The user chooses from the car's display to end the ride.

8. The system detects that the car's position corresponds to the right PGS and that the car is plugged in.

9. The system applies a discount on the amount the user has to pay.

| Postconditions | The car is ready to be locked, transaction hasn't happened yet |

**Exceptions**

1. There isn't any available PGS near the user's destination.

2. The user doesn't park the car at the PGS suggested by the system.

3. The user doesn't plug the car in the power grid.

4. The user drives away after choosing to end the ride. The discount is canceled.

### 3.3.12   Payment

| | |
|---|---|
| Actor | System, Payment API |
| Goal | [G9], [G19] |
| Preconditions | The user ended a ride and has a pending charge. |
| Execution Flow | |

1. The system starts the transaction.

2. The system sends its credentials to the Payment API.

3. The system sends the user's account data to the API.

4. The system sends the amount to receive to the API.

5. The API confirms the payment.

6. The system closes the transaction.

| | |
|---|---|
| Postconditions | The bill is paid |
| Exceptions | |

1. The Payment API is not available. In this case, the system marks the bill as "waiting" and retries later.

2. The transaction is rejected. In this case, the system suspends the user until he pays the pending bill.

### 3.3.13   User pays a pending bill.

| | |
|---|---|
| Actor | User, System, Payment API |
| Goal | [G18] |
| Preconditions | The user is logged in, has a pending bill and is suspended. |
| Execution Flow | |

1. The user goes to his profile page.

2. The user selects the pending bill.

3. The user selects the payment method.

4. The user confirms the transaction.

5. The system carries out the payment as in the "Payment" use case.

6. The API confirms the payment.

7. The system cancels the suspension.

| | |
|---|---|
| Postconditions | The bill is paid and the user is no longer suspended. |
| Exceptions | |

1. The Payment API is not available. In this case, the system marks the bill as "waiting" and retries later.

2. The transaction is rejected. In this case, the user is notified of the problem and remains suspended.

### 3.3.14 A user reports a problem to the system.

| | |
|---|---|
| Actor | User, System |
| Goal | [G20] |
| Preconditions | The user is logged in and has reserved a car |
| Execution Flow | |

1. The user goes to the reporting page of the application.

2. The user selects the type of problem the car has (dirty, damaged, ...).

3. The user confirms and submits the data to the system.

4. The system checks the data inserted by the user.

5. The system sets the car as unavailable.

6. The system ends the user's reservation and ride.

| | |
|---|---|
| Postconditions | The car is marked as unavailable and the user has no reservation. |
| Exceptions | |

1. The user inserts wrong data.

## 3.4  Performance Requirements

## 3.5  Design Constraints

## 3.6  Software System Attributes

### 3.6.1  Reliability

### 3.6.2  Availability

### 3.6.3  Security

### 3.6.4  Maintainability

### 3.6.5  Portability

# 4 Appendix

## 4.1 Alloy Model

## 4.2 Software and tools used

## 4.3 Hours of work

# 5 Revisions

## 5.1 Changelog