POLITECNICO
MILANO 1863

eMall System

e-Mobility for All

RASD

Requirement Analysis and Specification Document

Version 3.0 - 22/12/2022

Alireza Yahyanejad - 10886993

Mohammad Hosein Behzadifard – 10880732

Alireza Azadi - 10888648

Contents

Figures

Tables

1. INTRODUCTION

1.1. Purpose

In the past, humans have been highly dependent on fossil fuels. With the advancement of science and technology, today humanity has been able to use renewable energy in various ways, including as fuel for cars. As a result, it prevents the warming of the earth and the rise of the water level, which is one of the causes of the carbon dioxide gas released from cars due to the consumption of fossil fuels.
Electric mobility (e-Mobility) is proposed to reduce carbon dioxide. Regarding e-Mobility, several charging stations have been created so that cars can be charged according to their battery capacity. There are several charging sockets in each charging station. There may be different types of charging sockets so that different cars can be charged. Finally, drivers pay an amount of money for charging their cars.
This document focuses on the Requirements Analysis and Specification Document (RASD) of the system and describes the main goals, the domain assumptions, the scenarios which may happen, the use cases, the list of functional and non-functional requirements which system should fulfill, and finally the diagrams to visualize the interactions between components and performance of the system.

1.1.1 Goals

Table 1 Goals

Goals	Description
G1	eMSP allows end user to know about the charging stations nearby, their cost, any special offer they have
G2	eMSP allows end user to book a charge in a specific charging station for a certain timeframe
G3	eMSP allows end user to start the charging process at a certain station
G4	eMSP allows end user to be notified when the charging process is finished
G5	eMSP allows end user to pay for the obtained service
G6	eMSP allows end user to sort the available stations
G7	CPMS allows CPO to know the location of a charging station

G8	CPMS allows CPO to know the external status of a charging station
G9	CPMS allows CPO to decide from which DSO to acquire energy
G10	CPMS informs CPO to dynamically decide where to get energy for charging
G11	CPMS allows CPO to acquire by the DSOs information about the current price of energy
G12	CPMS allows CPO to know the internal status of a charging station
G13	CPMS allows CPO viewing the charging process to infer when the battery is full
G14	CPMS allows CPO to know the time to start charging a vehicle according to the amount of power supplied by the socket
G15	CPMS allows CPO sets current price of energy for paying of services

1.2 Scope

To charge electric cars, an application has been defined through which the end user can do the following:

- First, the user must log in to her/his account. Then activate the GPS of her/his smartphone to find the nearest charging station.
- After selecting the nearest charging station suggested by the application, the user books a time.
- Through the notification of the user's smartphone, the end of the charging process will be shown to her/him.
- Through the application, the amount to be paid and the time when the battery was charged is shown to the user.
- Finally, the user connects to the banking portal through the application and pays the cost of charging the electric car.

Also, there are CPOs who can either own and operate a set of charging stations or use them for third parties. For managing charge stations, they install and use CPMSs.

1.2.1 World Phenomena

Table 2 World Phenomena

World Phenomena	Description
WP1	The end user comes to charge station with her/his car
WP2	The end user finds out that her/his car needs charging
WP3	The end user plugs in her/his car to the charger socket
WP4	The end user leaves the charging station after payment is successful

1.2.2 Shared Phenomena

Table 3 Shared Phenomena

Shared phenomena	Description	Control
SP1	End user searches for nearby charging stations	World
SP2	eMSP shows the nearby stations by default	System
SP3	End user sorts the available charging stations	World
SP4	End user books a charge in a specific charging station	World
SP5	End user pays the money for services	World
SP6	End user will be notified when the charging process is finished by eMSP	System
SP7	User can see any special offer in eMSP	System
SP8	Charging stations will be suggested by eMSP to user	System

SP9	CPO knows the location of charging station through CPMS	System
SP10	CPO knows the "external" status of charging station through CPMS	System
SP11	CPO know the “internal” status of a charging station through CPMS	System
SP12	CPO acquire by the DSOs information about the current price of energy through CPMS	System
SP13	CPO decides from which DSO to acquire energy	World
SP14	CPMS will dynamically decide where to get energy for charging	System
SP15	eMSP provides a QR code for booking conformation	System
SP16	End user can check charging process of her/his car	World

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

Table 4 Definitions

Definition	Description
Notification	A message shown to the user by system when he/she must be notified about something (ex: the end of the charging process will be shown to him/her).
External Status	Number of charging sockets available, their type such as slow/fast/rapid, their cost, if all sockets of a certain type are occupied, the estimated amount of time until the first socket of that type is freed

Internal Status	Amount of energy available in its batteries, if any, number of vehicles being charged and, for each charging vehicle, amount of power absorbed, and time left to the end of the charge
Time Frame	A specified period in which something occurs or is planned to take place
Valid QR code	A QR code is defined valid in the 5 minutes after end user wants to pay for the services

1.3.2 Acronyms

Table 5 Acronyms

Acronyms	Description
CPO	Charging Point Operator
CPMS	Charge point Management System
DSO	Distribution System Operator
eMSP	e-Mobility Service Provider
eMall	e-Mobility for All
GPS	Global Positioning System
API	Application Programming Interface

1.3.3 Abbreviations

Table 6 Abbreviations

Abbreviations	Description
G	Goal
R	Requirement
C	Component
WP	World phenomena
SP	Shared phenomena

1.4 Revision History

Table 7 Revision History

Version	Date	Modification
1.0	11/02/2022	First version
2.0	16/02/2022	Update all sections
3.0	22/02/2022	• Goals updated • Class diagram updated

		• Alloy code improved • Section 5 updated

1.5 Reference Documents

- Specification Document: “Assignment RDD AY 2022-2023.pdf”
- Course slides
- https://evroaming.org/app/uploads/2021/11/OCPI-2.2.1.pdf

1.6 Document Structure

- Section 1

Overview of the purpose of the project and defining the scope of the system. Describe the specifications such as the definitions, acronyms, abbreviations, revision history, and references. As well as introducing the goals, world and shared phenomena of the software.

- Section 2

Defining the main scenarios and then explaining the main features in the software by class diagram and state charts. In user characteristics, the types of actors that use the application are explained. The product function subsection defined the functionalities of the application. In the end, the domain assumptions are defined.

- Section 3

The main part of the project which introduces interface requirements such as user interface, hardware interface, software interface, and communication interfaces. Presenting the functional requirements that are shown by use case diagrams and sequence diagrams. Then requirements are mapped to use cases.

- Section 4

Using Alloy language for analyzing the system and brief comments for clarifying the Alloy codes.

- Section 5

Shows how much time is spent by each member of the group.

- Section 6

Contains the references.

2. OVERALL DESCRIPTION

2.1. Product perspective

2.1.1. Scenarios

- **End user wants to pay the money for service**

  After Giulia's EV had been charged, she received a notification in her phone in the eMSP app to be notified about the completion of her car charging process. She can pay the cost on her phone through her debit/credit card online or she can pay the cost at the charging station through the charging point which is NFC card reader and cash machine or QR which includes a link to the payment gateway. Hence Giulia has already charged her profile credit, and just paid the invoice through the app.

- **End user wants to book a charge in charging station**

  The user Bertil has a business meeting at 4 pm in another city near his hometown but his EV battery charge is not enough for such a distance, and it is crucial for him to be present on time. So, he decided to book a charging point to avoid being in the charging point line and wasting his time. Therefore, he opens the app and finds his destination location to get a list of proper charging points on his way to the meeting and books a charging point among them for a specific time.

- **End user wants to find the nearest charging station**

  Alex has an electric car. He knows there is an application named eMSP which besides all the features, shows all charging stations. So, he can find the nearest one. He already signs in and chooses his car model so when he goes to the charging station section in the application and turns on his mobile phone GPS, the map shows his location on the map and all charging stations. He can filter all the stations by price, other user ratings, and travel distance, and he can see all the new offers that CPOs provided for him.

- **End user looks at the notification to indicate the completion of charging**

  John is sitting in the coffee shop near the charging station while his car is plugged in. He can see the status of his car's battery, battery percentage, the speed of charging, and the remaining time to get fully charged. When his car gets fully charged the eMSP mobile application sends him a notification "your car is ready to pick up" so he goes to the charging station to use his car. Also, he can set a limit in the application for battery charging, When the battery percent reaches that the mobile application shows the notification.

- **End user views suggestions**

Rachel is CPO of a big company, so she doesn't have enough time to Handle her plans and she is very busy. When she got into her car, she sees that her car didn't have enough battery, so she opens the mobile application and goes to the suggestion section. The application asks for her location and her calendar permissions, she accepts that, and she enables her phone GPS and Bluetooth. The application gets her car's battery status by Bluetooth, and it sends her current location and her today plans to eMSP servers, after a few seconds the application shows some suggestions that relied on her available time and her location.

- **CPO suggests charging stations**

  Peter has an EV and wants to find a proper charging point to charge his car battery. He opens his eMSP app and creates his profile. He enters the type of charging socket, battery capacity, and car model. Then he clicks on the charging point suggestion button and waits for the app to show a list of suggested charging stations around him. Meanwhile, the app sends all the information plus the location of Peter's car through the API to the CPO. Since the CPO knows about all the charging stations through the CPMS, try to find the nearest charging point with the minimum cost of charging price and with a socket that matches Peter's EV charging socket. The CPO prepares a list of charging point locations ordered descending by the calculated score for each charging point and returns them as a response to the request. App then retrieves this list and shows them in a list and also on the map.

- **CPO finds the location of charging station**

  Mike is a CPO. He signs in to the CPMS dashboard so he can see the status of all charging stations such as booked one, free one, in use, and the energy consumption of the charging station. In the manage charging station section, he can search for all charging stations by their name, and address.

- **CPO knows the external status of charging station**

  Federico is a CPO, and he wants to know the external status of charging stations such as the sockets availability, types of them, and so on. He can go through his admin panel and see all the statistics about the CPMS. All this information is gathered and sent to CPO through the CPMS.

- **CPO knows the internal status of a charging station**

  Monica is a CPO at the Viale Romagna street station. All the charging ports are in use. She goes to the CMPS dashboard and started looking for information about that station. She can see all the cars models and their current battery percentage, the ports' output voltage, how much costs users must pay, the remaining time to cars get fully charged, how much energy each port consumes to generate electricity, each port temperature, ports which are booked, the cost of energy they should provide and how much power each car absorb.

- **CPO gets information about the current price of energy**

  Alessandro opened the eMSP app and chose a charging point to book it for one hour later. He entered his battery status in the app. Through the defined API, the application sends all the information including the time of booking, battery status, and the charging point id to the CPMS. CPO gets the charging station battery status and gets the status of the grid (whether the booked time is during peak electricity usage or not) through the DSO to decide whether to use the station battery for charging or use the grid. Then it can calculate the cost which has been received from the DSO through the CPMS and turn it back to the user. Then based on the price, he can choose among the suggested or available charging points.

2.1.2 Class Diagram

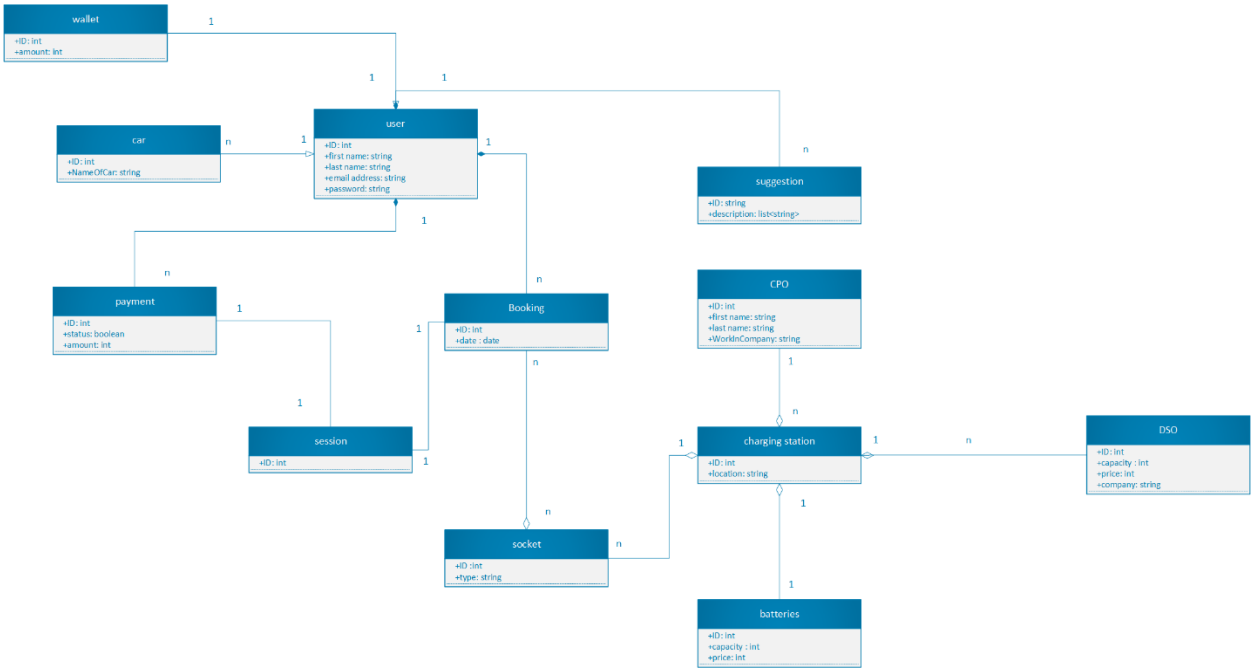

Figure 1 Class Diagram

2.1.3 Statecharts

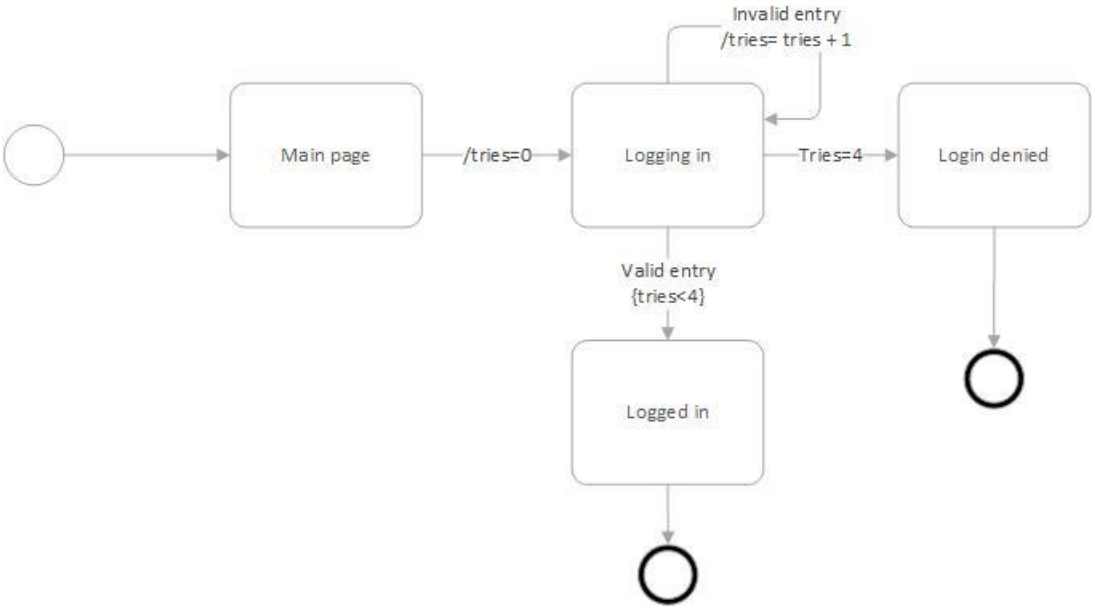

Figure 2 Statechart for login

Regarding figure 2, the user goes to the main page and tries to log in. If she/he inserts her/his information wrong for 4 times, then the user must wait until the login page appears to her/him again. Else the user logged in.

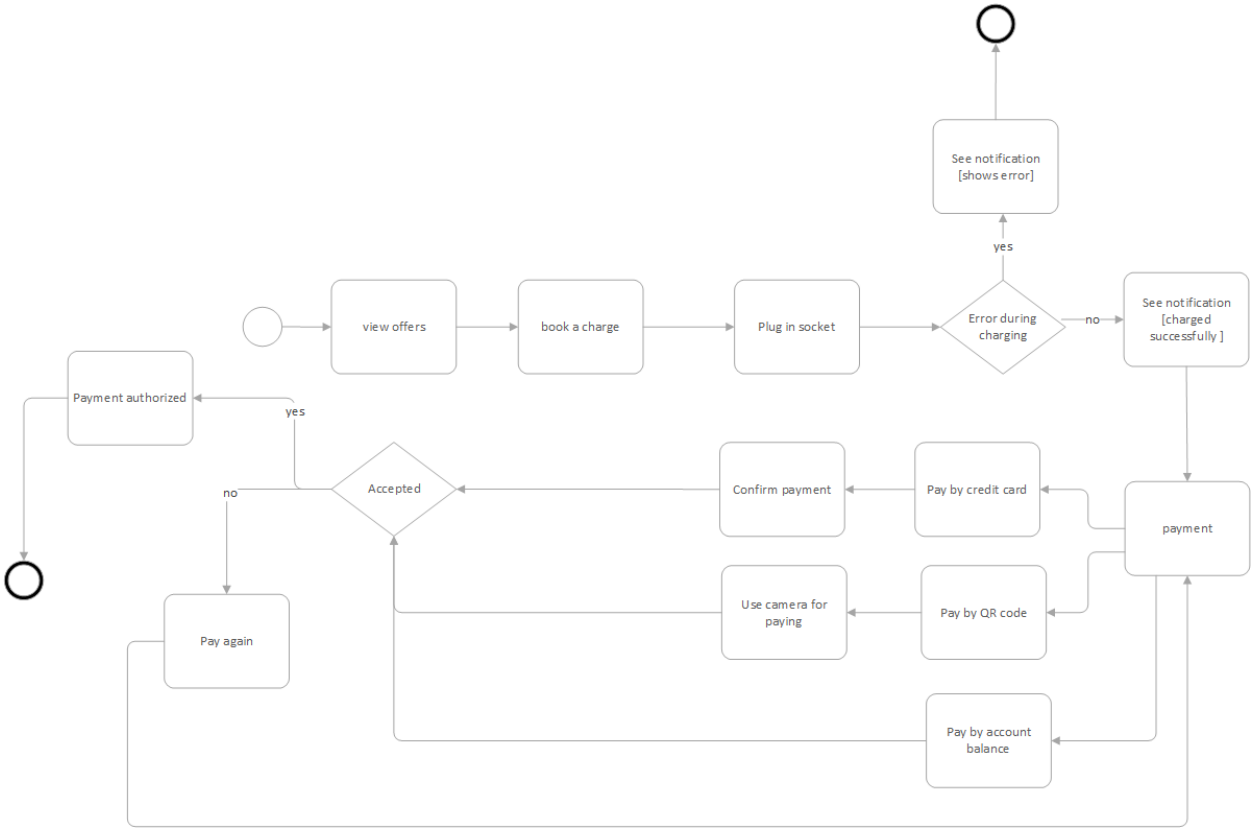

Figure 3 Statechart for session

About figure 3, the user can view offers and suggestions, then she/he can book a charge and plug in the socket. After that, if the charging process is successful or unsuccessful, the user can see the notification about it. If the charging process is successful, the user must pay for the services. The user can pay by QR code, credit card, or her/his account balance. During the payment process, if the user cannot pay, she/he must try again. Else the payment is authorized.

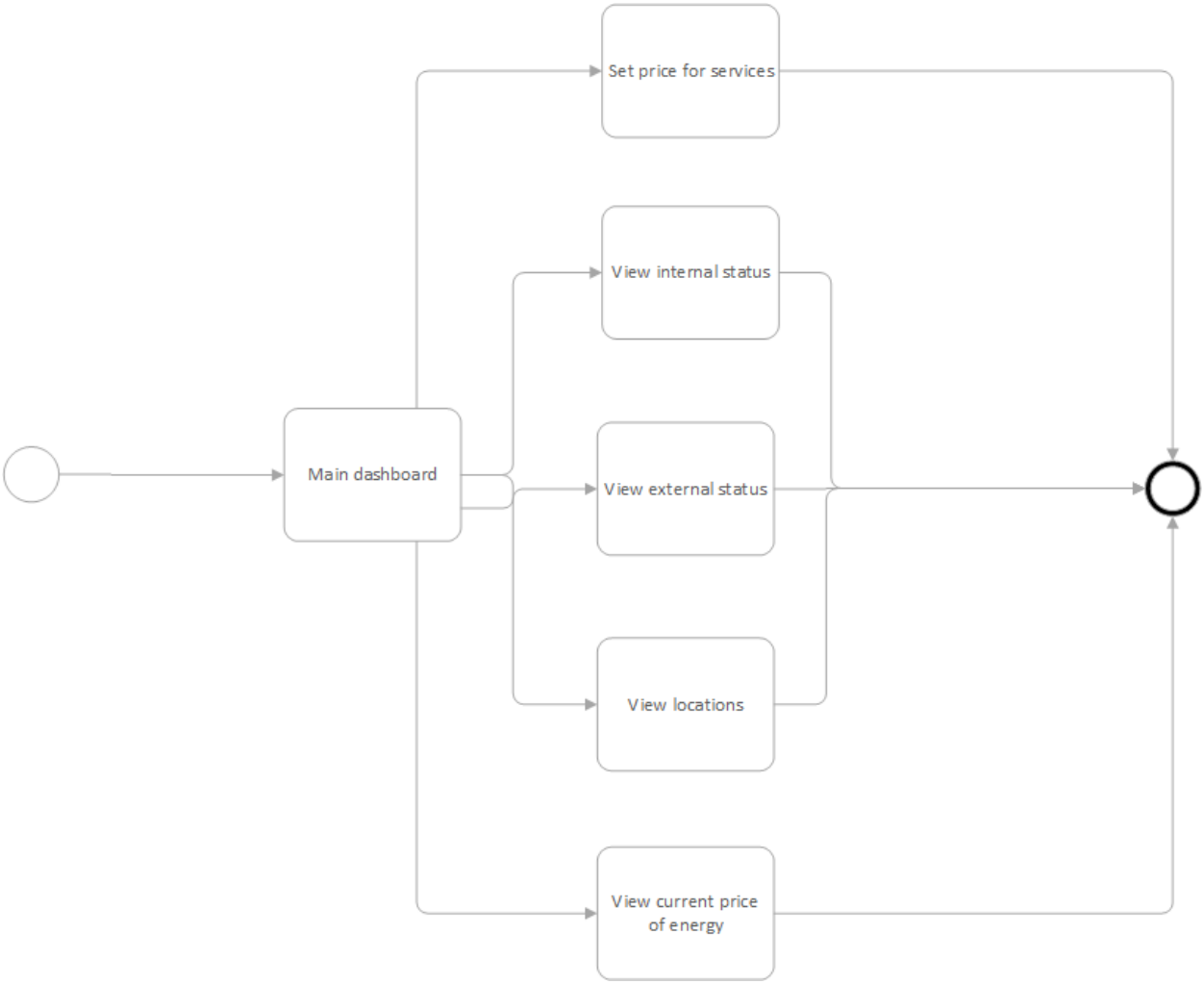

Figure 4 statechart for CPO

Regarding figure 4, CPO can do some work on her/his dashboard main page. The CPO can set the price for services, view internal status, view external status, view locations, and view the current price of energy.

2.2. Product functions

Here are described most functions of the systems. The less important ones are mentioned in other sections as well.

- **Service payment methods**

One of the important features of eMall application is having a payment function. Therefore, developers provide multiple methods for payment. Users can pay for the charge with Apple pay or Google pay application as a third-party application or just use their credit card, or they can save some money in their eMSP account and pay automatically with that money. Also, there are some more methods for payment like contactless cards.

- **Book a charge in a specific charging station**

  The most important thing that users care about is finding the best charging station and booking a port for a specific time. Users can see all the stations and sort all of them by different parameters. After all of this, the user can select one charging station and book one place. When they want to go to booked charging port, one scanner read their car plaque number, or drivers show their application barcode to the barcode scanner then CPMS shows a message to the CPO.

- **Create a notification when the charging process is finished**

  It is one of the basic features of the app for users. As respects, as the charging process is time-consuming and long, the system must notify the user about the status of the charging and in this case, notify the user when the charging is finished. Notification allows the end-user (EV owner) to use her/his time meanwhile his/her car is charging and be notified to take his/her car from CP whenever the charging process has been completed. This notification will be sent to the user through the eMSP app installed on the user's mobile phone.

- **Getting energy for charging**

  One of the major functionalities of the CPMS is to decide how an electric car should be charged. An electric car can be charged in three ways: 1) station battery 2) DSO or 3) A mix thereof according to availability and cost. So, CPO can see how an electric machine is charging. This decision is dynamic with CPMS.

2.3. User characteristics

The following two actors are considered in the system.

- **End user**
  A person who has an electric car and wants to charge her/his car at a charging station through eMSP.

- **CPO**
  CPOs are the owner of some charging points. In other words, their job is to install and maintain charge stations in charging stations.

2.4. Assumptions, dependencies and constraints

Domain assumptions are the facts that we assume to be true in the world.

Table 8 Domain Assumption

Domain Assumption	Description
D1	User and no other organizations/existing systems handles the insertion of her/his information
D2	User must have an electric car
D3	User must arrive at charging station at the time that she/he booked
D4	CPO must know the exact current price of energy from DSOs
D5	Each CPO works only for a specific company
D6	CPO gives different offers for each user
D7	CPO sets current price for services
D8	DSOs' prices must be updated
D9	Voltage of each charging point must be appropriate
D10	Each charging point must have a battery and DSOs for charging

3. SPECIFIC REQUIREMENTS

3.1. External Interface Requirements

3.1.1 User Interfaces

The user interface of eMSP is an application on smartphones that will be used for users. Most parts of this application are developed for the users.
Also, because CPO can do some work like select a DSO for charging and see the information, she/he uses a website.

3.1.2 Hardware Interfaces

All the hardware that these applications require is:

- Users must use a smartphone that can connect to the internet and can use GPS services, access their calendars, and Bluetooth.
- CPOs must use a web browser to work with CPMSs.

3.1.3 Software Interfaces

The systems use Map as an external interface:

- eMSP provides a public API for the user to find her/his position and find the best way for attending to the nearest charging station.
- CPOs can access all locations of charging stations to check the status of each charging station.

3.1.4 Communication Interfaces

- Each eMSP communicates with CPMS of each CPO thorough API. This communicate will happen because data in eMSP put from information that CPMS has.
- Each CPMS has communication with DSOs.
- eMSP application can get access to the calendar of the user and her/his navigation system.

3.2. Functional Requirements

Table 9 Functional Requirements

Requirement	Description
R1	The eMSP must allow an unregistered end user to register
R2	After an end user fills all the blanks on registration page correctly, the system must send an email to her/him, in order to confirm her/his email
R3	The eMSP must allow a logged-out end user to login
R4	The CPMS must allow CPO to login
R5	The CPMS must allow CPO to view the list of charging stations with their statuses
R6	The eMSP must notify the user when the charging process is finished through notification
R7	The eMSP must access to the GPS of the end user smartphone for suggesting charging stations
R8	The CPMS must allow a logged-out CPO to login
R9	The eMSP must notify the user when there is an error during charging process through notification
R10	The eMSP must allow end user to view her/his information
R11	The eMSP must allow end user to enter her/his information
R12	The CPMS can connect to several eMSPs through API
R13	The CPMS must connect to each charging points
R14	The eMSP must allow end user to pay for the service
R15	The eMSP must create QR code if end user wants to pay by QR code
R16	The eMSP must allow end user to book a charging point

R17	The eMSP must access to the end user's calendar
R18	The eMSP must suggest charging points depending on the status of the battery
R19	The CPMS must show external status to the CPO
R20	The CPMS must show internal status to the CPO
R21	The CPMS must show locations of charging stations
R22	The CPMS must allow CPO to insert the current price for services
R23	The eMSP must show the receipt after end user paid for the services
R24	When the car is charging, the eMSP must show the exact status of the battery to the end user
R25	When the end user is on the main page, eMSP must shows the list of all charging stations in order
R26	eMSP must shows the prices of each charging stations to the end user in main page
R27	eMSP must allow end user to select the way of payment
R28	eMSP must show the ways of payments to the end user
R29	After an end user inserted all the personal information related to her/him, the eMSP must allow her/him to confirm
R30	eMSP must allow the end user to edit information about her/his credit card
R31	After an end user inserted all the information related to her/him credit card, the eMSP must allow her/him to confirm
R32	eMSP must allow end user to cancel the time she/he booked
R33	The CPMS must show available DSOs to the CPO
R34	When end user wants to see available charging stations, the eMSP must show special offers that CPO considered for the end user
R35	After an end user books a charging point, the eMSP must send a notification to her/him to inform her/him
R36	eMSP must allow the end user to edit her/his information
R37	eMSP must allow the end user to see the status of her/his car battery
R38	The CPMS must allow CPO to view the current price of energy through DSOs
R39	The eMSP must show available charging stations to the user
R40	The eMSP must show the charging stations' status to the user
R41	The CPMS must allow CPO to select DSOs to acquire energy
R42	The CPMS must show how a vehicle is charging to the CPO
R43	The CPMS must allow the CPO to insert offers for end users

3.2.1. Use Case Diagrams

- CPO

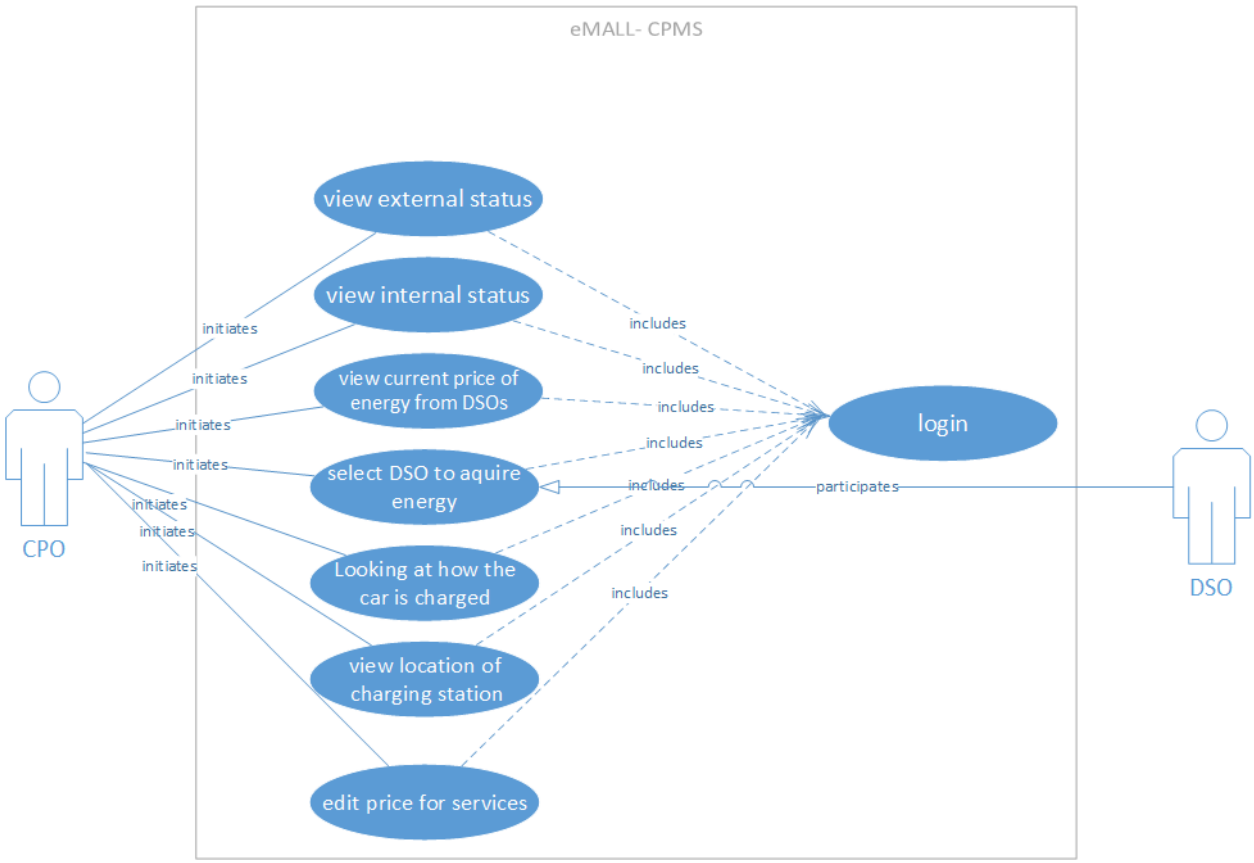

Figure 5 Use Case Diagram for CPO

- Registered user

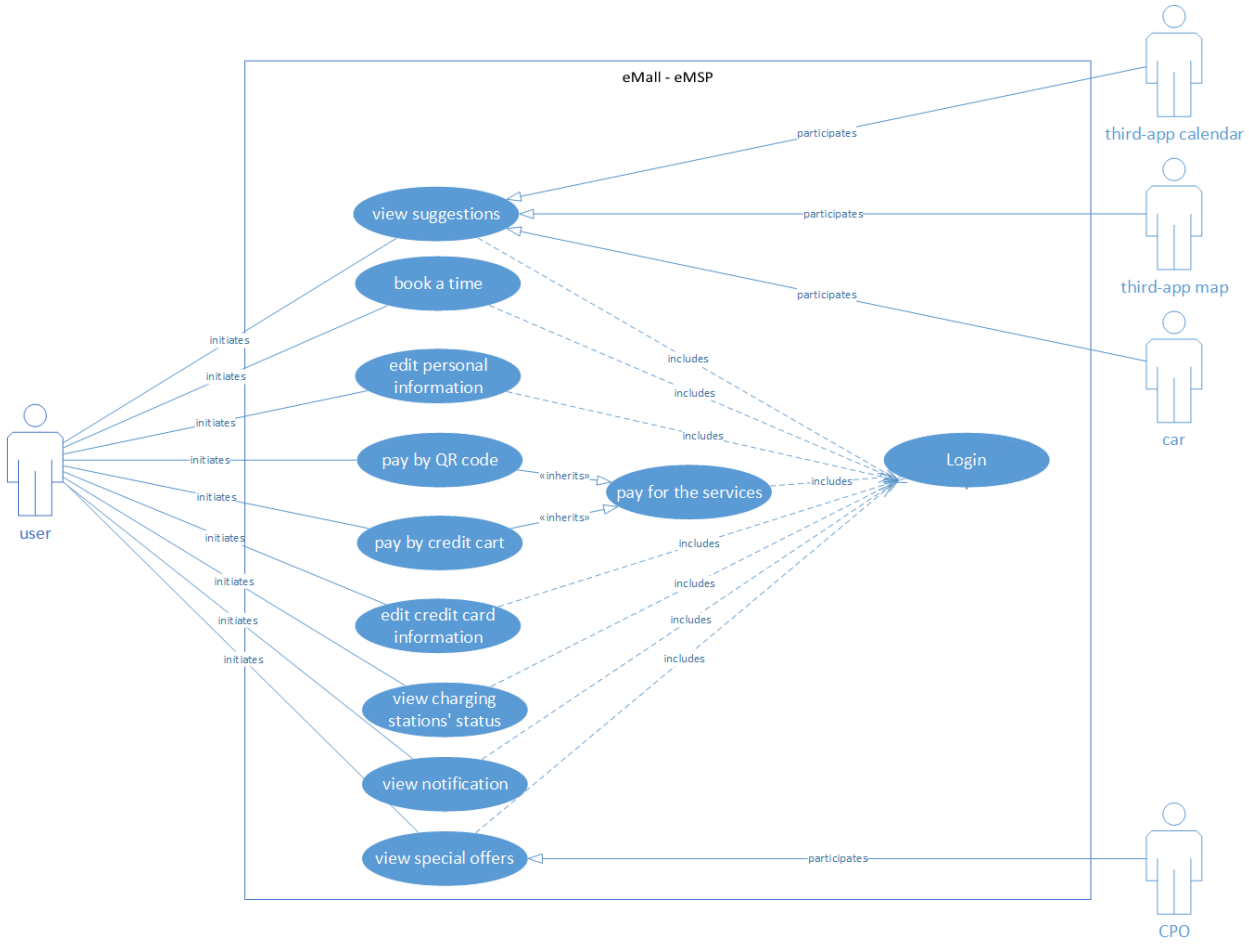

Figure 6 Use Case Diagram for End User

- Unregistered user

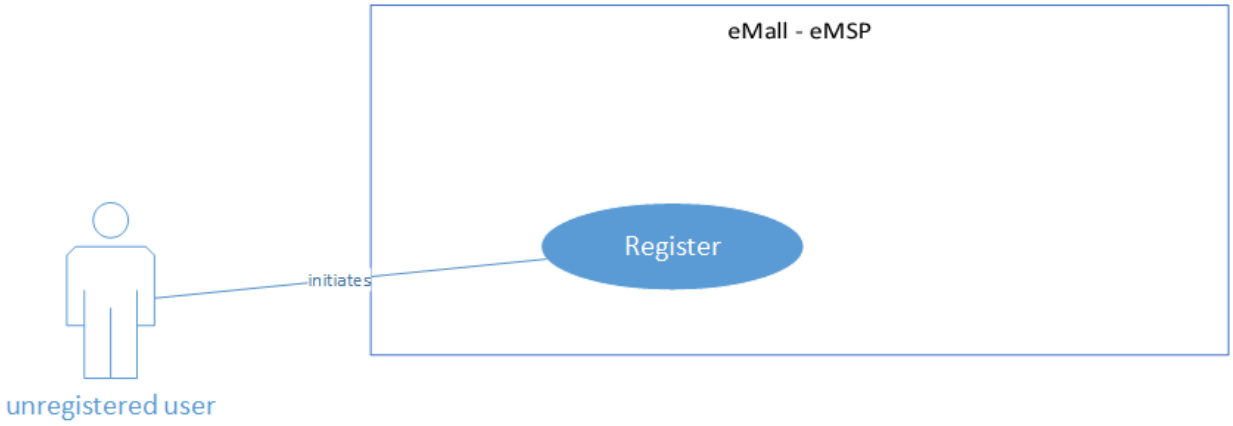

Figure 7 Use Case Diagram for unregistered user

3.2.2. Use Cases

- View external status

Table 10 use case for CPO - View external status

Name	View External Status
Actor	CPO
Entry Conditions	The CPO has already Logged to the CPMS dashboard in successfully.
Events Flow	1- Click on the Charging Station lists on the panel. 2- Dashboard shows a list of this CPO charging stations. 3- CPO selects a Charging Stations. 4- Dashboard shows all charging port and their type. 5- CPO can see all external status of charging station in top of the page. 6- CPO can see all external status of the port by clicking on the port.
Exit Conditions	Information is presented successfully.
Exceptions	• Charging Stations get disconnected from the network and not be able to send the Charging Point status. • The CPMS was down and not able to send the list of Charging Stations to the CPO panel.

- View internal status

Table 11 use case for CPO - View internal status

Name	View Internal Status
Actor	CPO
Entry Conditions	The CPO has already Logged to the CPMS dashboard in successfully.
Events Flow	1. Click on the Charging Station lists on the panel. 2. Dashboard shows a list of this CPO charging stations. 3. CPO select a Charging Stations. 4. Dashboard shows all charging port and their type. 5. CPO can see all internal status of charging station in top of the page. 6. CPO can see all internal status of the port by clicking on the port.
Exit Conditions	Information is presented successfully.
Exceptions	• Charging Stations get disconnected from the network and not be able to send the Charging Points status.

	• The CPMS was down and not able to send the list of Charging Stations to the CPO panel.

- Edit current price of energy

Table 12 use case for CPO - Edit current price of energy

Name	Edit current price of energy
Actor	CPO
Entry Conditions	He/she access to the (admin panel). The CPO has already Logged in successfully.
Events Flow	1. CPO open the panel. 2. From dashboard, choose price settings. 3. Edit current price and save the settings.
Exit Conditions	Price gets updated successfully
Exceptions	• The price was not valid. E.g., negative value • eMSP was not able to update the database. (Database error, infrastructure problem and so on)

- Select DSO to acquire energy

Table 13 use case for CPO - Select DSO to acquire energy

Name	Select DSO to acquire energy
Actor	CPO
Entry Conditions	The CPO has already Logged to the CPMS dashboard in successfully.
Events Flow	1. CPO go to DSO section in the CPMS dashboard. 2. CPO can see all of DSOs' name and their prices. 3. CPO can request them by clicking on forward button to send a request. 4. After that DSO accept CPO request, CPMS shows a notification that DSO accept your request.
Exit Conditions	DSO accept to provide energy for CPO and the CPO update the eMSP database successfully.
Exceptions	1. DSO is not available. 2. DSO does not accept to provide energy for CPO. 3. eMSP was not able to update the database. (Database error, infrastructure problem and so on)

- Select kind of getting energy

Table 14 use case for CPO - Select kind of getting energy

Name	Select kind of getting energy
Actor	CPO
Entry Conditions	The CPO has already Logged to the CPMS dashboard in successfully.
Events Flow	1. CPO go to list of charging point. 2. Dashboard shows list of CPOs all charging stations. 3. CPO by clicking on internal status of specific charging point can see the whole internal status of charging point. 4. CPO can change the power source of one socket from battery to DSOs energy. 5. CPMS can handle this situation when battery is drained and automatically change to DSO energy. 6. CPO can set preferred Energy in top of the page in the dropdown. This causes the fact that CPMS can automatically use DSO after that use batteries or reverse.
Exit Conditions	1. DSO accept to provide energy for CPO if the DSO has been selected. 2. CPO update the eMSP database successfully.
Exceptions	1. DSO is not available. 2. DSO does not accept to provide energy for CPO. 3. eMSP was not able to update the database.

- View location of charging station

Table 15 use case for CPO - View location of charging station

Name	View location of a charging station
Actor	CPO
Entry Conditions	The CPO has already Logged to the CPMS dashboard in successfully.
Events Flow	1. CPO go to list of charging point. 2. Dashboard shows list of CPOs all charging stations. 3. CPO click on eye icon in front of any item of the list. 4. Dashboard shows the location of the charging station near to Internal Status button External Status button.
Exit Conditions	Charging Station information, including the location be shown successfully.

Exceptions	1. Sub systems including CPMS and eMSP not be available. 2. The eMSP database was not responding (for instance, because of the heavy load)

- Register

Table 16 use case for user - Register

Name	Register
Actor	End user
Entry Conditions	End user opens the application
Events Flow	1. End user selects to register. 2. End user fills all the blank fields. 3. End user clicks on register button. 4. The system sends an email to his/her. 5. End user confirms his/her email. 6. The system allows him/her to login.
Exit Conditions	The end user is registered in the app.
Exceptions	1. End user doesn't check the policies checkbox. 2. End user doesn't fill all the fields. In this case, application shows warning. 3. the email address already exists in the system. In this case, application warns him/her to login.

- Login

Table 17 use case for user - Login

Name	Login
Actor	End user
Entry Conditions	1. End user opens the application. 2. She/He already registered.
Events Flow	1. End user selects login button. 2. End user enters email addresses as username password to login. 3. End user clicks on login button.
Exit Conditions	The system allows the user to login.
Exceptions	1. End user enters the wrong username. 2. End user enters the wrong password. In both cases, system warns him/her that user with this data not found.

- View special offers

Table 18 use case for user - View special offers

Name	View special offers
Actor	End user
Entry Conditions	1. End user opens the application. 2. He/she already signed in.
Events Flow	1. End user clicks on today's offers button. 2. Application shows a page that contains list of all the offers for the user. 3. End user can click on one offer and accept it.
Exit Conditions	Accept an offer or decline all offers.
Exceptions	He/she doesn't login to the application. In this case, user must login to the application to see offers.

- Book a time

Table 19 use case for user - Book a time

Name	Book a time
Actor	End user
Entry Conditions	1. End user has already Logged in successfully. 2. In the application, user go to charging stations section.
Events Flow	1. End user clicks on Plus icon to save new booking. 2. End user selects time period what he/she want to book. 3. The application shows all charging station that have available port for end user car. 4. End user selects one charging station that he/she want. 5. The application asks for what time and how long he/she wants to book.
Exit Conditions	End user clicks on confirm/cancel button.
Exceptions	End user enters wrong time period.

- Edit personal information

Table 20 use case for user - Edit personal information

Name	Edit personal information
Actor	End user
Entry Conditions	1. End user has already logged in successfully. 2. In the application, user go to edit personal data section.
Events Flow	1. End user clicks on profile icon. 2. Application shows the information about user. 3. End user can change email address, his/her car model, his/her profile photo.
Exit Conditions	End user clicks on save/cancel button.
Exceptions	1. End user enters invalid email address. 2. End user selects image that the volume is more than 5MB.

- Pay by QR code

Table 21 use case for user - Pay by QR code

Name	Pay by QR code.
Actor	End user
Entry Conditions	1. End user has already Logged in successfully. 2. Application shows a notification that charging is finish.
Events Flow	1. There is one LCD on that port which user put in the car for charging, after the charging finished it shows the QR code. 2. End user scans this QR code with his/her application. 3. Application asks for payment method. 4. End user can select to pay with his/her saved card, third-party application like google pay or apple pay or pay with money that saved to his/her account.
Exit Conditions	Payment is successful.
Exceptions	1. End user selects to pay with her/his account money but there isn't enough money. 2. End user doesn't have enough money in his/her bank account.

- Pay by credit card

Table 22 use case for user - Pay by credit card

Name	Pay by credit card
Actor	End user
Entry Conditions	1. End user has already Logged in successfully. 2. Application shows a notification that charging is finish.
Events Flow	1. End user clicks on notification. 2. Application opens in car status page. 3. End user sees payment alert about payment in top of application. 4. End user clicks on alert. 5. Application opens in payment method page. 6. End user clicks on credit card. 7. Application sends request to eMSP. 8. eMSP processes the request and send response to mobile application of user about deducting money.
Exit Conditions	Payment is successful or error occur in payment.
Exceptions	End user doesn't have enough money in his/her bank account.

- Edit credit card information

Table 23 use case for user - Edit credit card information

Name	Edit credit card information
Actor	End user
Entry Conditions	1. End user has already Logged in successfully. 2. End user goes to edit payment method section in the application.
Events Flow	1. Application shows the list of all payment method such as all credit card and account balance. 2. End user can enter new credit card for his/her payment by pressing on Plus icon or edit credit card by click on pen icon. 3. End user can edit his/her credit card information like card number or card name and expire date.
Exit Conditions	End user clicks on Save/Cancel button. If user click on save, he/she must verify his/her card information.

Exceptions	1. End user enters invalid card number, expire date or card name. 2. End user doesn't verify his/her card account.

- End User Get Some Suggestion

Table 24 use case for user - End User Get Some Suggestion

Name	End User Get Some Suggestion
Actor	End User
Entry Conditions	1. End user has already Logged in successfully. 2. End user goes to suggestion section.
Events Flow	1. Give location permission to application. 2. Give calendar permission to application. 3. Enable phones GPS. 4. Enable phones Bluetooth. 5. After a few seconds, Application shows some suggestion to user.
Exit Conditions	End user confirms or decline one/all suggestion.
Exceptions	End user doesn't give permission to application.

3.2.3. Sequence Diagrams

- CPO views location of charging station

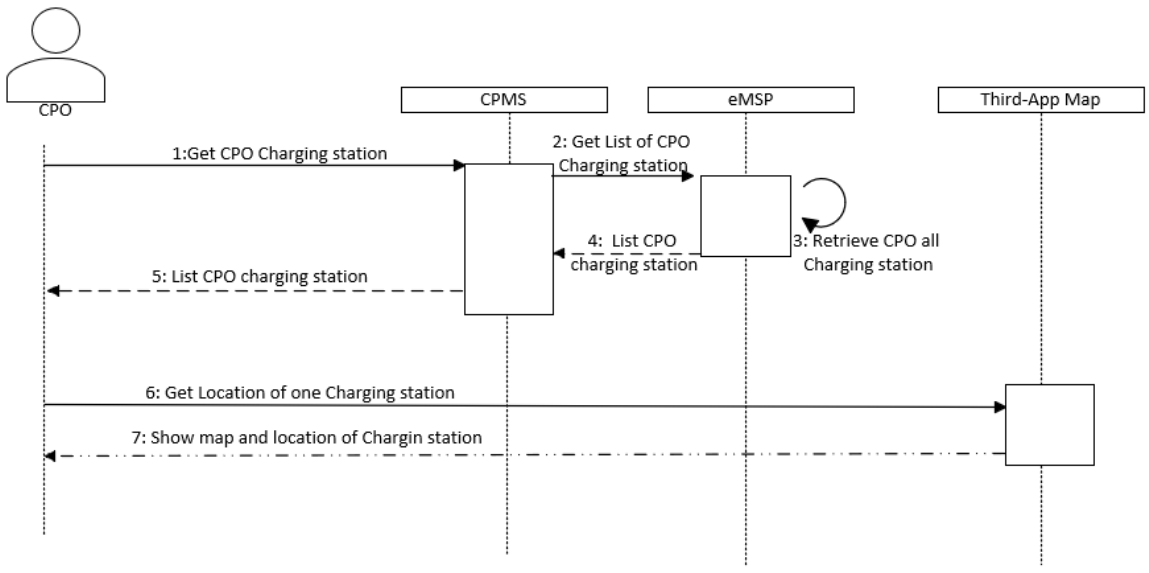

Figure 8 CPO views location of charging station

- Select DSO to aquire energy

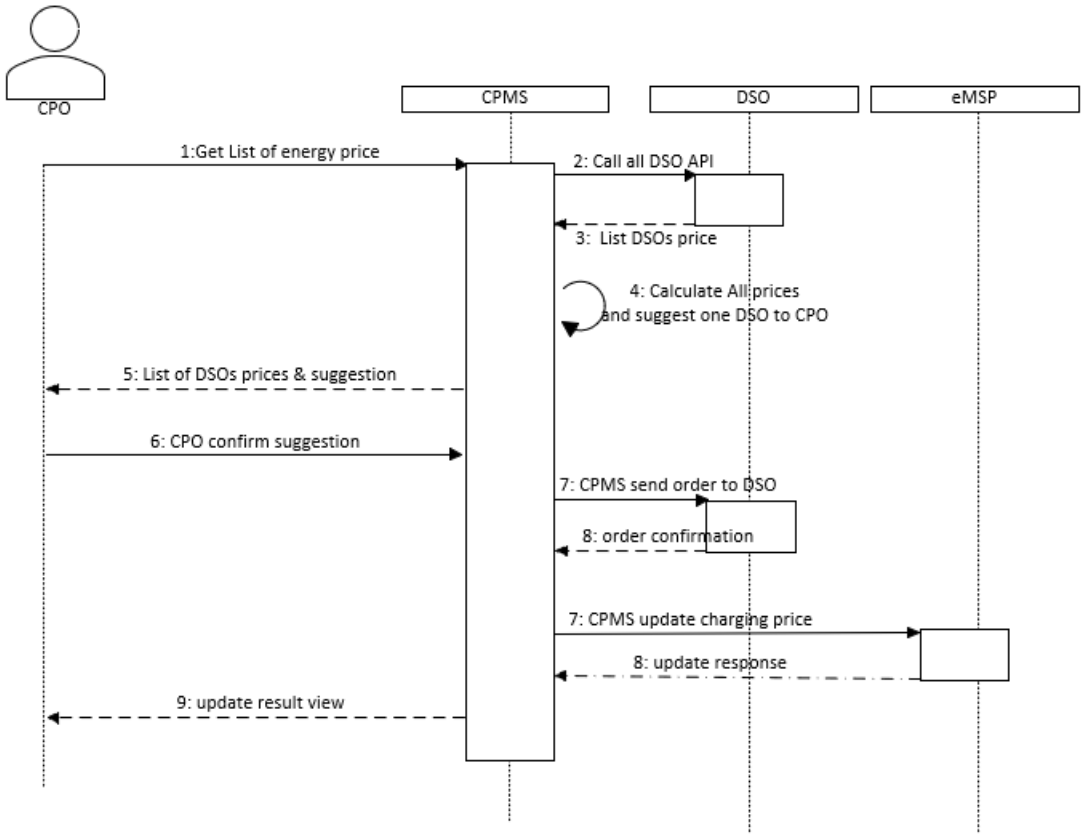

Figure 9 Select DSO to acquire energy

- CPO sets price of service

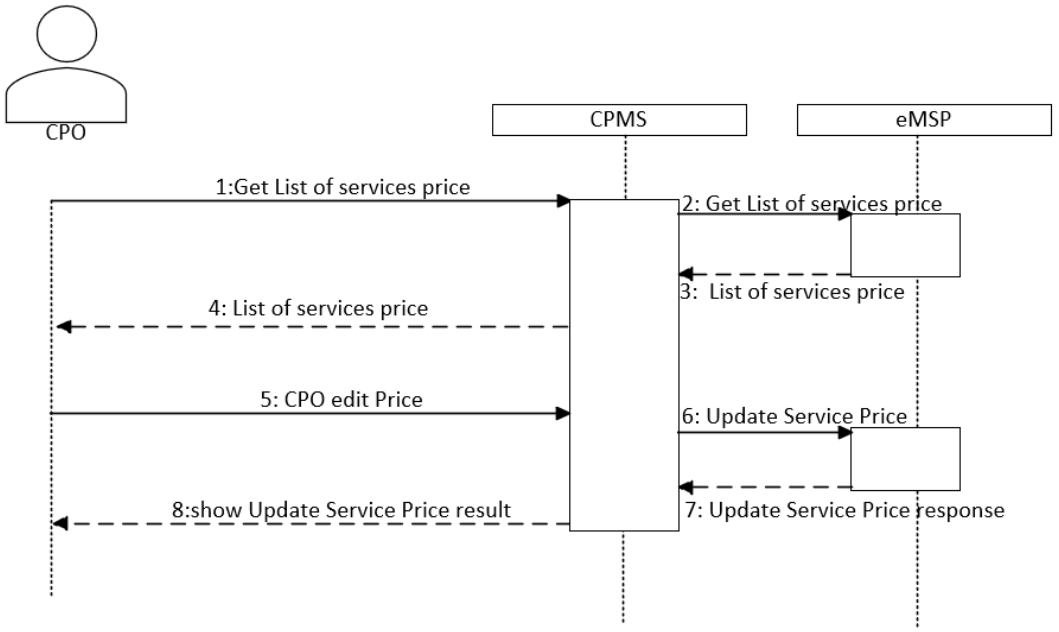

Figure 10 CPO sets price of service

- CPO views external status

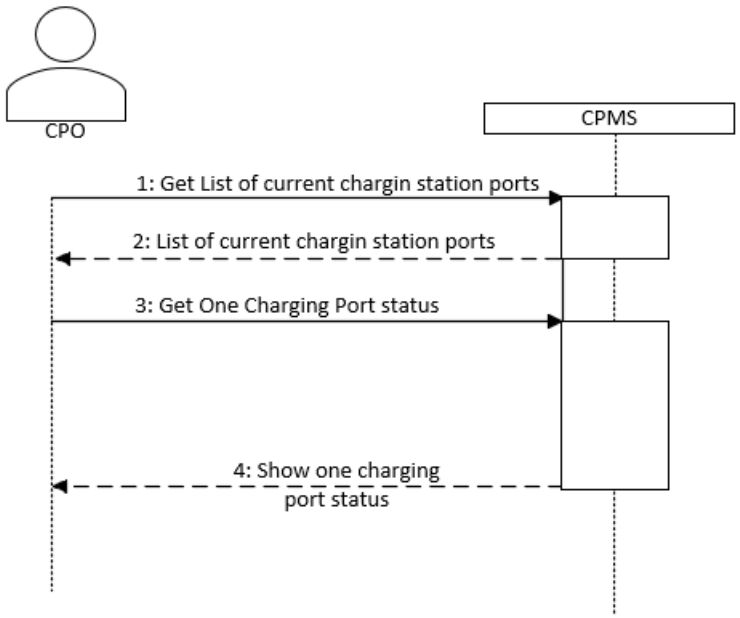

Figure 11 CPO views external status

- CPO views Internal status

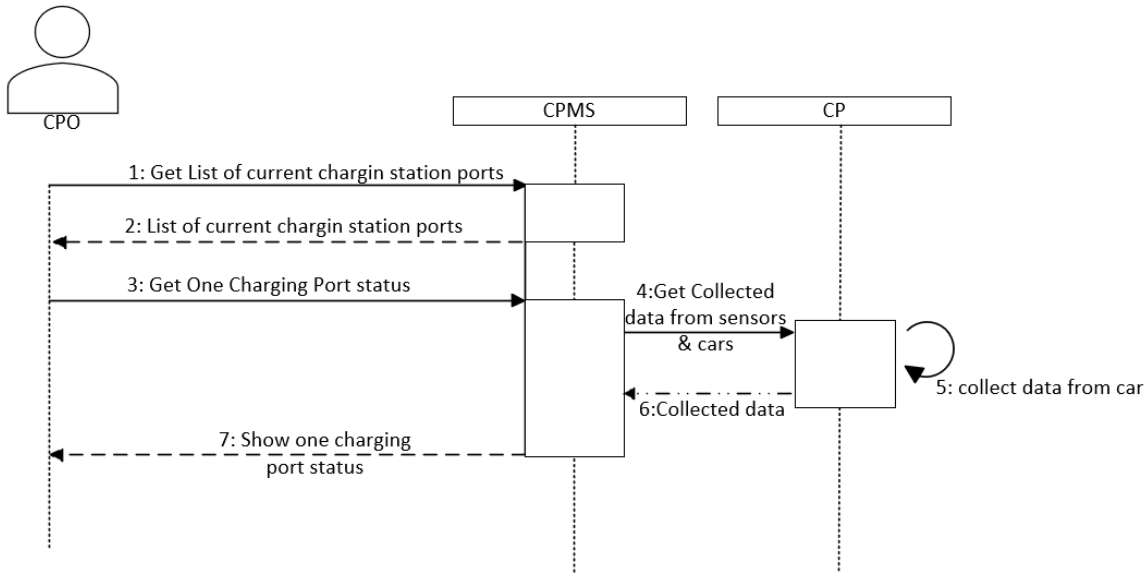

Figure 12 CPO views Internal status

- End User gets seggestions

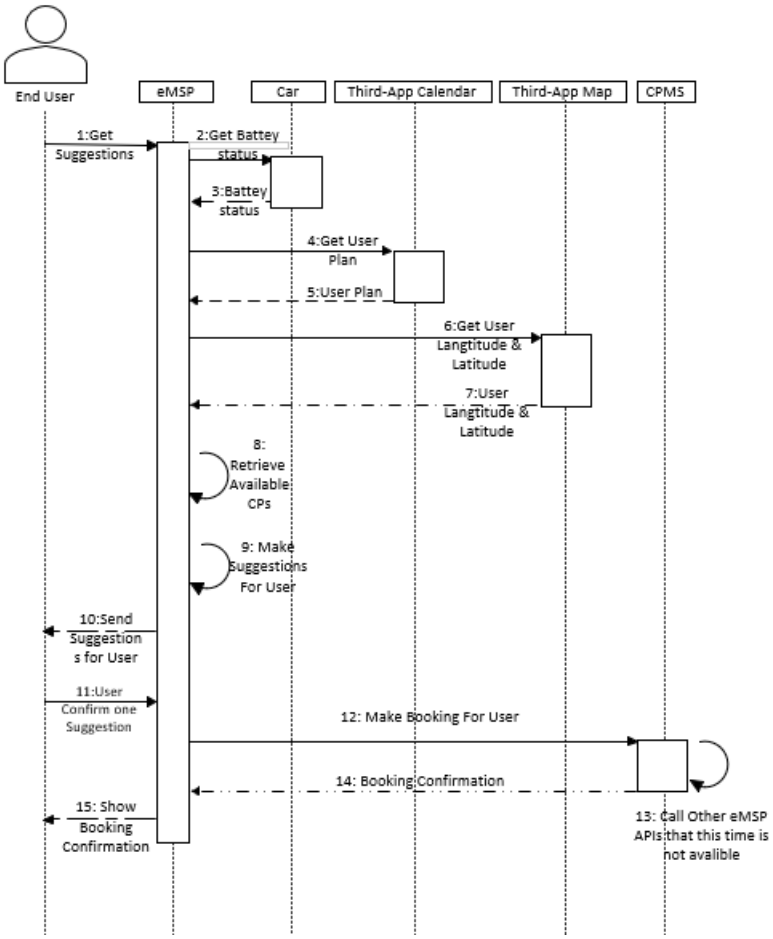

Figure 13 End User gets suggestion

- End User pays by credit card

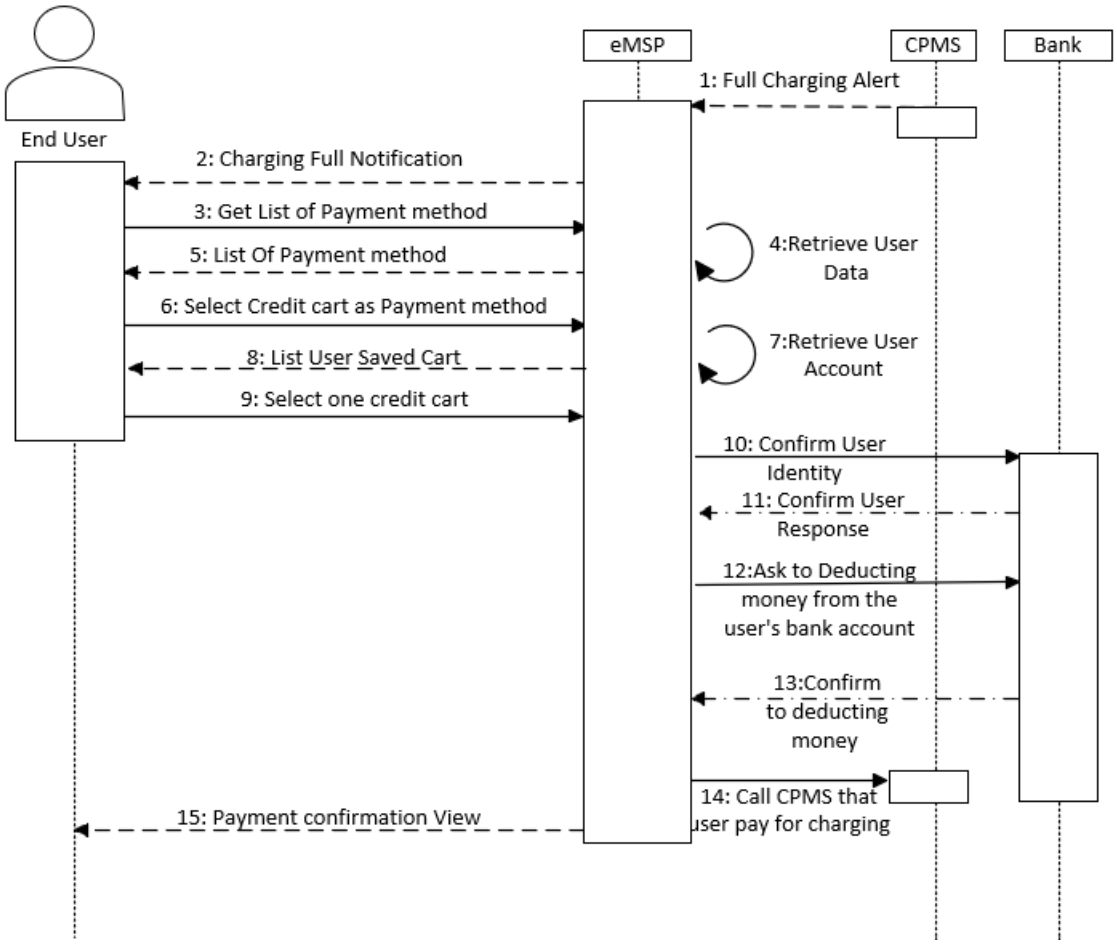

Figure 14 End User pays by credit card

- CPO gives offers

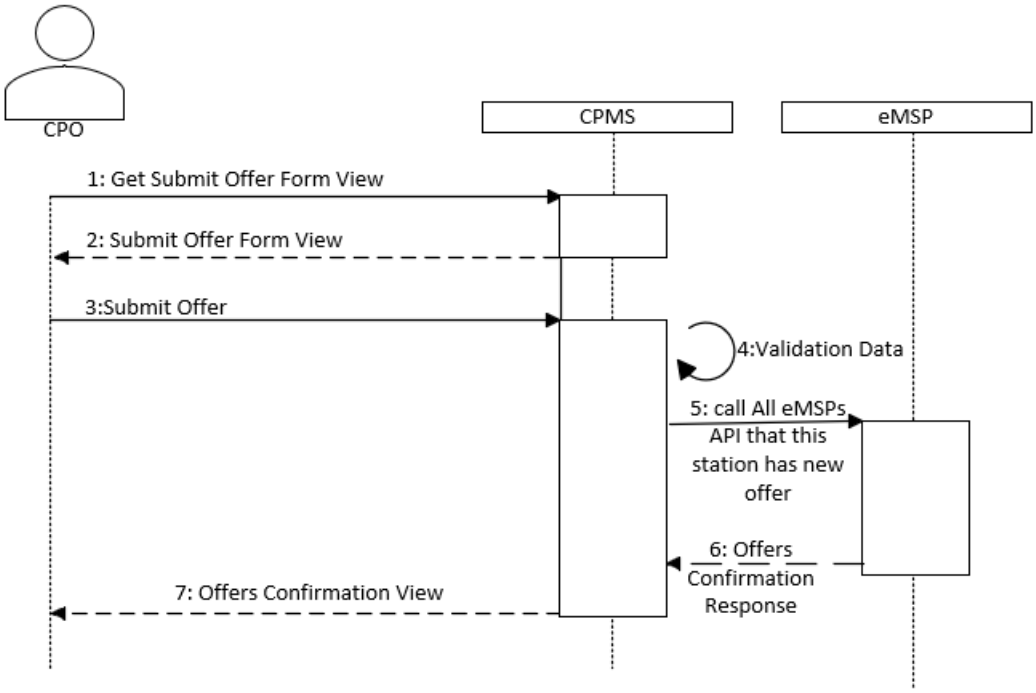

Figure 15 CPO gives offers

- End user pays by account balance

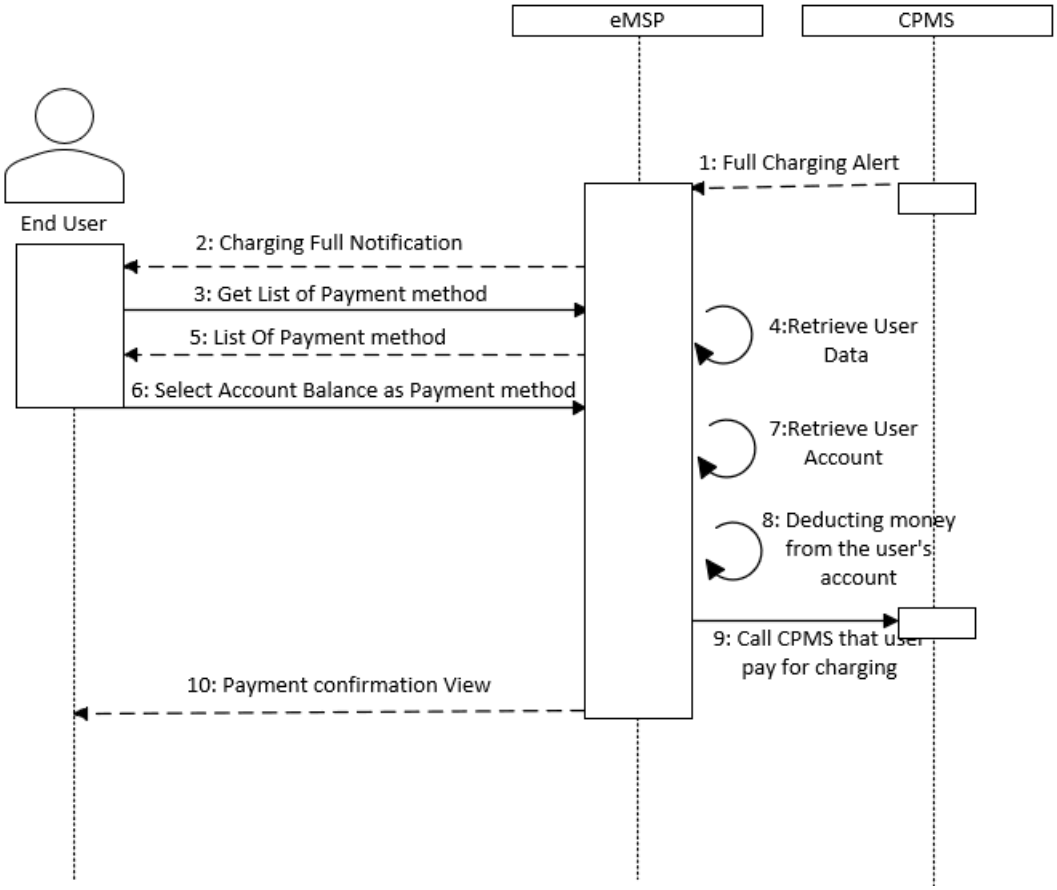

Figure 16 End User pays by account balance

- End User booking

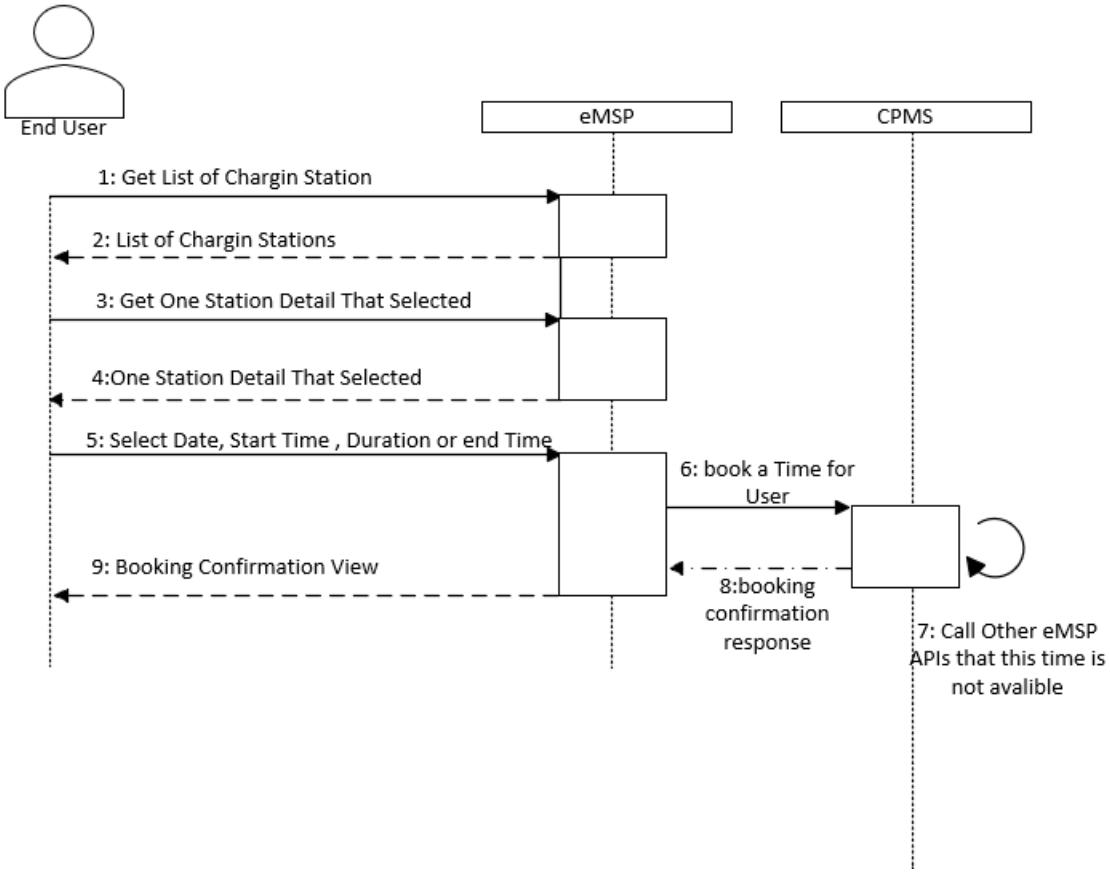

Figure 17 End User booking

- End User login

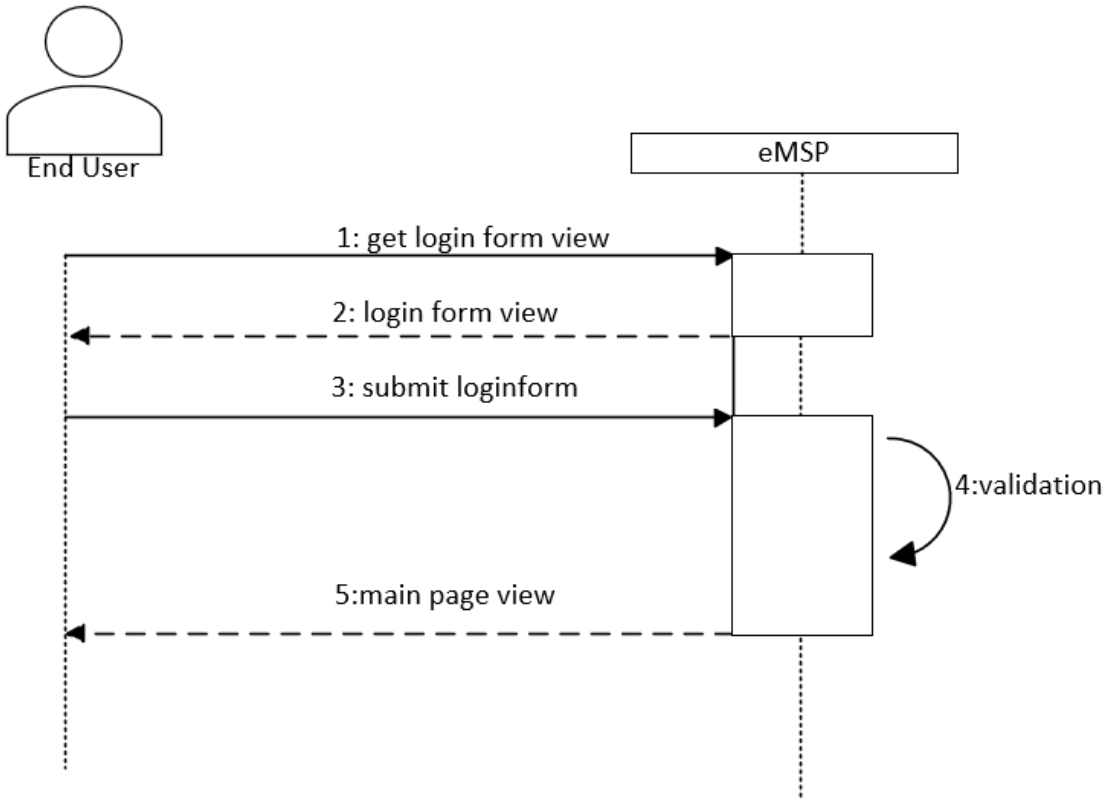

Figure 18 End User login

- End User registration

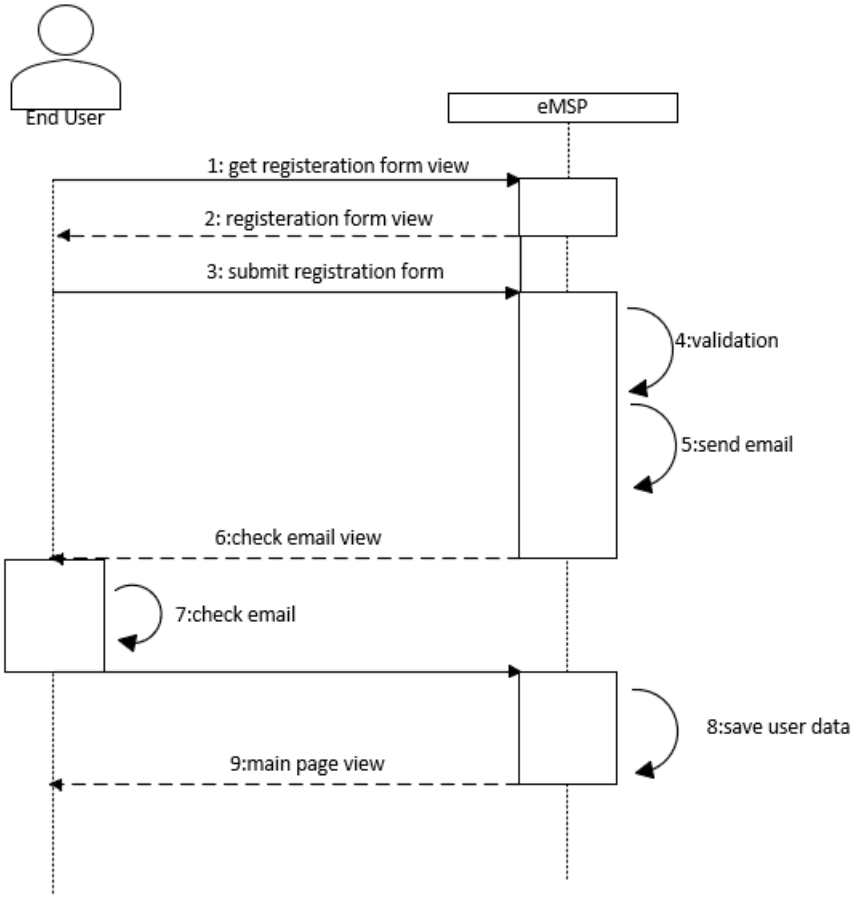

Figure 19 End User registration

- CPO gets price form DSOs

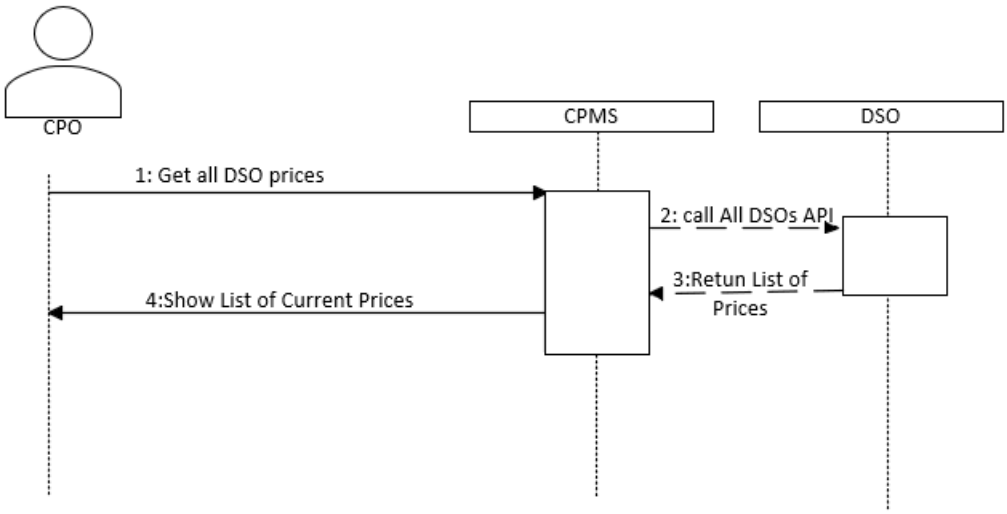

Figure 20 CPO gets price form DSOs

3.2.4. Mapping on Requirements

Table 25 Mapping on Requirements

Use case	Requirements
view external status	R4, R5, R8, R19
view internal status	R4, R5,R8, R20
login (CPO)	R4, R8
edit price for services	R22, R38
select DSO to acquire energy	R33, R41
looking at how the car is charged	R33, R42
view location of charging station	R21
register	R1, R2, R30, R31, R36
login (user)	R3
view suggestions	R7, R18, R34
book a time	R16, R32, R35
edit personal information	R29, R36
pay by QR code	R14, R15, R23, R27, R28
pay by credit card	R14, R23, R27, R28, R30, R31
edit credit card information	R30, R31
view charging station's status	R39, R40
view notification	R6, R9, R35
view special offers	R43 , R34
view location of charging station	R21

3.3 Performance Requirements

- The system must be able to serve a great number of users simultaneously.
- The system must guarantee correct responses.
- The system must be able to send a response to a query less than 3 seconds since it has been received.
- The system must be available 99% of the time.

3.4. Design Constraints

3.4.1 Standards compliance

- The eMSP must manage the data retrieved from the users with respect to privacy laws.
- The eMSP must require the customer's permission to retrieve data regarding the position.
- The CPMS must manage data with respect to privacy laws.

3.4.2 Hardware limitations

- The web browser or the smartphone which the user and the CPO are using must have the ability to connect to the internet and use GPS services.
- The smartphone which the end-user has must have the ability to scan a QR code.
- All systems must have enough memory to run the application.

3.4.3 Any other constraint

- The information which is related to the battery status of the user's car must be accurate 99% of the time.
- All battery charging prices must be updated periodically and displayed to the user through the eMSP.

3.5. Software System Attributes

3.5.1 Reliability

The system must be able to run continuously without any interrupts. The reliability of the system depends on the services of the system and should be up 99% of the time. This means the MTTR, or downtime should be 365 days per year. In order to guarantee this time of downtime the system must have an appropriate infrastructure with a full backup system located in different offices that replicates the core services for covering the general failure of the main system.

3.5.2 Availability

Like other kinds of charging stations that are available 24 hours a day, these kinds of charging stations must be like them. So, our system needs high availability. It should be noted that CPOs work in two shifts, morning and night, and the system is always available for end users.

3.5.3 Security

The provided information by CPOs is not sensitive, thus we don’t need high security for the CPMS as software. But regarding eMSP, because our system keeps the information about the credit card of the user and account balances, we need high security for eMSP.
Generally, for preventing some problems, we need to encrypt stored data and the password of the user’s hashed before storing.

3.5.4 Maintainability

The software must be written in python and others like Django, PostGIS, Redis, PostgreSQL, and other technologies for developing application. Codes must be written with good standards. The occurred

problems must be detected easily and solved simply. The problem in one component must have not interrupted other components.

3.5.5 Portability

The eMSP must be designed simply and implemented on different smartphone platforms. This software runs on different platforms and must support Android and iOS operating systems for mobile devices. Also, the CPMS must be run in a web application through all browsers like Chrome, Microsoft Edge, and Firefox.

4. FORMAL ANALYSIS USING ALLOY

4.1. Code

The main goal of our Alloy analysis is to validate the model and see some samples of our eMall system. For showing the result, we assume a world that exist one end user, one CPO, 3 cars, and more than 2 sockets.

```
////segments
sig Firstname{}
sig Lastname{}
sig Email{}
sig Password{}
sig NameOfCar{}
sig Amount{}
sig Company{}
sig Capacity{}
sig CompanyName{}
sig Username{}
sig Wallet{}

sig EndUser {
    firstname: one Firstname,
    lastname: one Lastname,
    email: one Email,
    password: one Password,
    own : set Car,
    receive: set suggestion,
    book : set Booking,
    pay : some Payment,
    accountBalance: one Wallet
}{
    #own > 1
```

```
}

sig Car{
    nameOfCar: one NameOfCar,
}

sig suggestion{}

sig Payment{
    amount: one Amount,
    status: one Bool,
}

sig Booking{
    date: one Date,
    socket : one Socket
}

sig session{
    haveBook: one Booking,
    havePayment: one Payment
}

sig Date{
    date: one TypeOfDate
}

sig Socket{
    type: one Type
}

sig ChargingStation{
    location: one Location,
    manageBy: some CPO,
    battery: one Battery,
    dso: some DSO,
    socket: some Socket
}{
    #socket > 0
}

sig Battery{
    price: one Int,
    capacity: one Capacity
}{
```

```
    #price >0
    #capacity > 0
}

sig DSO{
    price: one Int,
    capacity: one Capacity,
    companyName: one CompanyName,

}{
    #price != 0
    #capacity != 0
}

enum Type{
    slow,
    fast,
    rapid
}

enum TypeOfDate{
    Sunday,
    Monday,
    Tuesday,
    Wednesday,
    Thursday,
    Friday,
    Saturday
}

sig CPO{
    firstname: one Firstname,
    lastname: one Lastname,
    username: one Username,
    password: one Password,
    company: one Company
}

sig Location{
    latitude: one Int,
    longitude: one Int
}{
    latitude >= -5 and latitude =< 5
    longitude >= -5 and longitude =< 5
}
```

```
abstract sig Bool{}
one sig TRUE extends Bool{}
one sig FALSE extends Bool{}

////facts

//each email always must be associated with only one user
fact EmailAssociationEndUser{
    all e : Email | one u : EndUser | e in u.email
}

//each Username always must be associated with only one CPO
fact UsernameAssociationCPO{
    all e : Username | one u : CPO | e in u.username
}

//emails of each user are different
fact EmailOfEndUser{
    no disjoint u1,u2: EndUser | u1.email = u2.email
}

//each charging station has its own sokcets
fact SocketsInChargingStation{
    all disjoint c1,c2 : ChargingStation | c1.socket != c2.socket
}

//each ChargingStation always must be associated with some socket
fact ChargingStationAndSockets{
    all c : ChargingStation | some s : Socket | s in c.socket
}

//Username of each CPO is different
fact UsernameOfCPO{
    no disjoint u1,u2: CPO | u1.username = u2.username
}

// each user must have some car
fact CarsOfEndUser{
    all u : EndUser | some c : Car | c in u.own
}

//the locations of each charging station must be different
fact LocationOfChargingStation{
    all disjoint c1,c2 : ChargingStation| c1.location != c2.location
```

```
}

// latitudes and longitudes of all charging stations must be different
fact latitudesAndlongitudes{
    all disjoint l1,l2: Location |
    l1.latitude = l2.latitude => l1.longitude != l2.longitude
}

//each loaction must have one latitude
fact LatitudeAndLocation{
    all l : Location | one a : Int | a in l.latitude
}

//each loaction must have one longitude
fact LatitudeAndLocation{
    all l : Location | one a : Int | a in l.longitude
}

//each socket must have only one type
fact TypeOfSocket{
    all t : Type | one s : Socket | t in s.type

}

//each CPO must work only for one company
fact CompanyForCPO{
    all a : CPO | one c : Company | c in a.company
}

// in each charging station some CPOs work
fact ChargingStationCPOs{
    all c : ChargingStation | some a : CPO | a in c.manageBy
}

// each socket can book by 2 different bookings in 2 different date
fact SocketForBooking{
    all disjoint b1,b2 : Booking | b1.socket = b2.socket and b1.date != b2.date
}

//each DSO must have one price
fact DSOAndPrice{
    all d : DSO | one p : Int | p in d.price
}

//each battery must have one price
```

```
fact BatteryAndPrice{
    all b : Battery | one p : Int | p in b.price
}
//each payment must have one status
fact paymentAndStatus{
    all p : Payment | one s : Bool | s in p.status
}

// in each session, there are only one payment
fact SessionForPaymentAndBooknig{
    all s : session | one p : Payment | p in s.havePayment
}

// in each session, there are only one booking
fact SessionForPaymentAndBooknig{
    all s : session | one b : Booking | b in s.haveBook
}

// each user can have several payments
fact SeveralPaymentForEndUser{
    all u : EndUser | some p : Payment | p in u.pay
}

//each user can have some suggestion
fact SuggetionForEndUser{
    all u : EndUser | some s : suggestion | s in u.receive
}

//each charging station can have only one battery
fact BatteriesInChargingStation{
    all c : ChargingStation | one b : Battery | b in c.battery
}
//the battery of each charging stations are different
fact ChargingStationForBatteries{
    all disjoint c1,c2 : ChargingStation | c1.battery != c2.battery
}

//each cahrging station can have several DSOs
fact BatteriesInChargingStation{
    all c : ChargingStation | some d : DSO | d in c.dso
}

//each DSO is only for one charging station
fact ChargingStationAndDSO{
    all disjoint c1,c2 : ChargingStation | c1.dso != c2.dso
```

```
}

//each Booking must have only one date
fact DateOfBooking{
    all b : Booking | one d : Date | d in b.date
}

//each DSO must have only one companyName
fact CompanyNameOfDSO{
    all d : DSO | one n : CompanyName | n in d.companyName
}

//each DSO must have only one capacity
fact CapacityOfDSO{
    all d : DSO | one c : Capacity | c in d.capacity
}

//each Battery must have only one capacity
fact CapacityOfDSO{
    all b : Battery | one c : Capacity | c in b.capacity
}

//each payment must have one amount
fact amountInDSO{
    all p : Payment | one a : Amount | a in p.amount
}

//each car must have only one name
fact NameOfCar{
    all c : Car | one n : NameOfCar | n in c.nameOfCar
}

//each user must have different payment
fact PaymentOFEndUser{
     all disjoint u1,u2: EndUser | u1.pay != u2.pay
}

//each session has different booking
fact SessionForBooking{
    all disjoint s1,s2: session | s1.haveBook != s2.haveBook
}

//each session has different payment
fact SessionForBooking{
    all disjoint s1,s2: session | s1.havePayment != s2.havePayment
```

```
}

//cars of each user must be different
fact CarsOfEndUsers{
    all disjoint u1,u2 : EndUser | u1.own != u2.own
}

//each wallet always must be associated with only one user
fact WalletOnlyForEndUser{
   all w : Wallet | one u : EndUser | w in u.accountBalance
}

//wallets of each end user must be different
fact WalletsOfEndUsers{
    no disjoint u1,u2: EndUser | u1.accountBalance = u2.accountBalance
}

//each user must have some booking
fact BookForEndUser{
    all u : EndUser | some b : Booking | b in u.book
 }

//each user must have separated booking
fact SeperatedBookingForEndUsers{
    no disjoint u1,u2: EndUser | u1.book = u2.book
}

//predicates
pred AddSuggestion[u1,u2: EndUser, s: suggestion]{
    u2.receive = u1.receive + s
}

pred CreateBooking[u1,u2: EndUser, b: Booking]{
    u2.book = u1.book + b
}

pred AddPayment[u1,u2: EndUser, p: Payment]{
    u2.pay = u1.pay + p
}

pred world1{
    #EndUser =1
    #CPO=1
    #Car=3
    #Socket >2
```

```
}

run world1 for 5
```

4.2. Results

```
Executing "Run world1 for 5"
  Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20 Mod
  17890 vars. 1325 primary vars. 37514 clauses. 130ms.
  Instance found. Predicate is consistent. 80ms.
```

Figure 21 result for world 1

4.3. Generated Instances

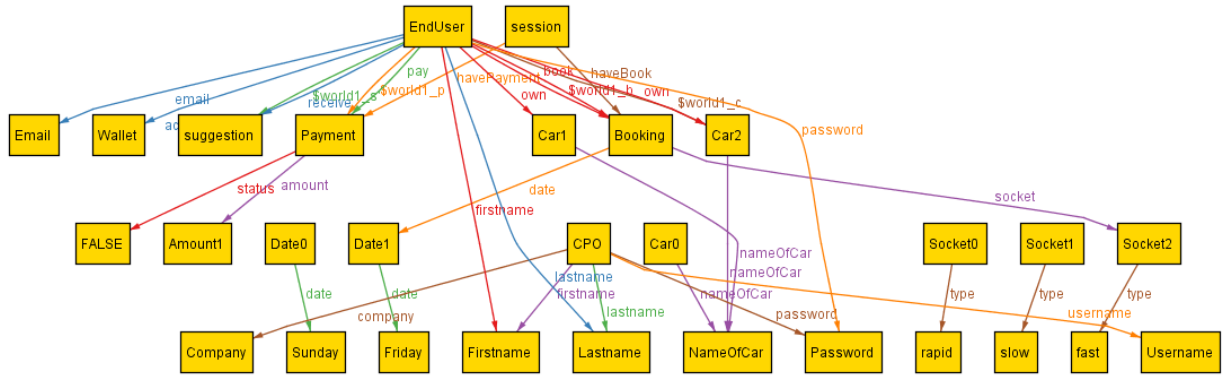

Figure 22 generated instance 1 for world 1

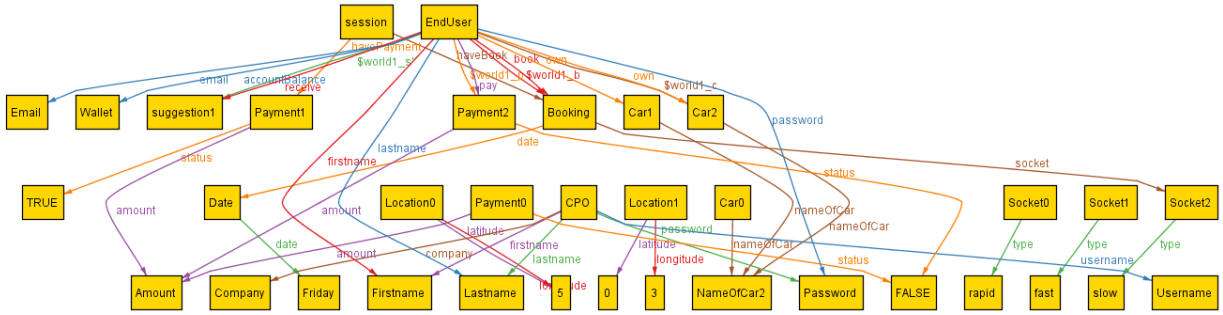

Figure 23 generated instance 2 for world 1

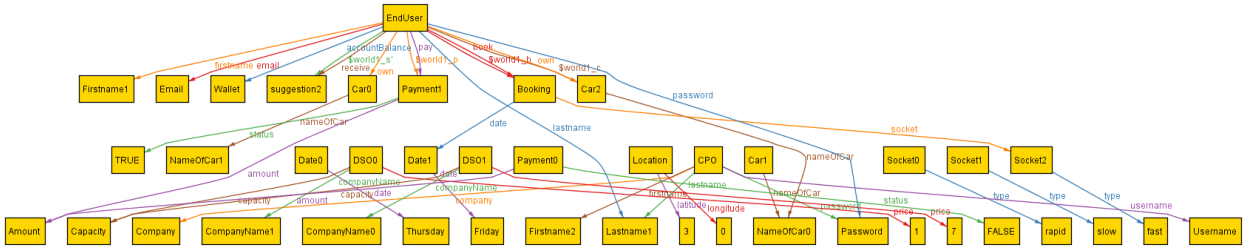

Figure 24 generated instance 3 for world 1

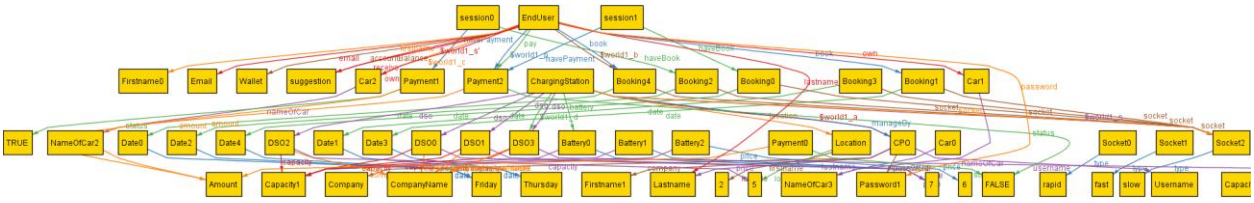

Figure 25 generated instance 4 for world 1

5. EFFORT SPENT

5.0.1

Table 26 Effort spent – Alireza Yahyanejad

Task	Time Spent
Introduction	6h
Overall description	9.5h
Specific requirements	13h
Formal analysis	6h
Reasoning	9h
Total	43.5h

5.0.2

Table 27 Effort spent - Mohammad Hosein Behzadifard

Task	Time Spent
Introduction	2h
Overall description	3h
Specific requirements	4h
Formal analysis	6h
Reasoning	5h
Total	20h

5.0.3

Table 28 Effort spent – Alireza Azadi

task	Time Spent
Introduction	1h
Overall description	2h
Specific requirements	5h
Formal analysis	4h
Reasoning	4h
Total	16h

6. REFERENCES

- Specification Document: “Assignment RDD AY 2022-2023.pdf”
- Course slides
- https://evroaming.org/app/uploads/2021/11/OCPI-2.2.1.pdf