* ~~CPO-view external status~~
* ~~CPO-view internal status~~
* ~~CPO-Edit current price of energy~~
* ~~CPO-Select DSO to acquire energy~~
* CPO-Select Kind of getting energy(There is no suitable component for this)
* ~~CPO-View location of a charging station~~
* ~~End user-Register~~
* ~~End user-login~~
* End user-View special offers 🡪 (There is no suitable component for this)
* ~~End user-book a time~~
* ~~End user-Edit personal information~~
* End user- pay by QR code(similar as pay by credit card)
* ~~End user-pay by credit cart~~
* End user- Edit credit card information(similar as edit personal information)
* ~~End user- Get some suggestion~~

2.4 *Runtime view*

In the following sequence diagrams, we discuss the detailed interactions between user and each

component. The main interactions between user and system have been already discussed in

RASD. In order to keep the diagrams readable, we just visualize the main flow of the events and

don’t consider the exceptions.

* End user-Register

End user fills the registration fields by his/her information through the “UserMobileApp” or component. Then, the information are sent to the “Authentication” component.

In this step, we should check whether there is another user with same email. So, the

“Authentication” component sends the entered email to the “Model” component. Here, a

temporary model with entered email is created. Then, a query is sent to database to check the

possibility of existing user with same email. The answer of database is encrypted. So, it is passed

to the “DataEncrypt” component to be decrypted. If the retrieved data from database is null, it

means there is no user with entered email, so the “Authentication” component creates a new

user with the inserted information, plus a flag equal to false and pushes it to the database.

Then, an email is sent to the registering user with a confirmation by “Mail” component. When

the customer clicks on the confirmation link, he/she will be redirect to a web page. Finally, a

query will be sent to database to update this user flag to true.

* End user-Login

The user enters the email and password through the “UserMobileApp” or component. Then, his/her information are sent to the “Authentication” component. In this step,

we should check whether the user has been registered. So, the “Authentication” component

sends the entered email to the “Model” component. Here, a temporary model with entered

email is created. Then, a query is sent to database to check the possibility of existing user with

same email. The answer of database is encrypted. So, it is passed to the “DataEncrypt”

component to be decrypted. If the retrieved data from database isn’t null, it means the user

with entered email exists. Then, the “Authentication” component should check the correctness

of entered password. In case the check is positive the retrieved user is returned, otherwise a null

value is returned.

* End user-View special offers

The user has already logged and now he/she is interacting with “UserMobileApp” component. First, the user go to charging stations section through “UserMobileApp” component and enter her/his time frame for charging. Then press on search button, So request forwards to the “Booking” component. This component send a request to the server asking for all Charging station with Suitable and free charging port for user time frame in the database. The answer of database is encrypted. So, it is passed to the “DataEncrypt” component to be decrypted.

The End user selects a Charging station and his/her request forwards to the “Booking” component and another request asking for book a time frame in a specific charging station to the server. After saving data in DBMS, “Booking” component send a request to “GooleCalendarService” to save a booking in user calendar.

* End user-Edit personal information

The user has already logged and now he/she is interacting with “UserMobileApp” component. First, the user go to edit personal data section through “UserMobileApp” component and “UserMobileApp” component send a request to server to retrieve end user information. Then, User can edit his/her personal information like first name, last name and his/her car model. After that, end user press on save button. So, new request forwards to the “Edit Information” component. This component send a request to the server asking for replace all new data of user to replace with the old ones. The answer of database is encrypted. So, it is passed to the “DataEncrypt” component to be decrypted.

* End user-pay by credit cart

The user has already logged and now he/she is interacting with “UserMobileApp” component. Through, End user see full charging notification. First, the user click on notification so he/she can see his/her car status. In the top of status page, can click on payment. So, “UserMobileApp” component send a request to server to retrieve user payment methods like account balance and credit card. End user click on credit card. Through the Bank API in the “Pay” component, the money was deducting from his/her bank account but data is encrypted so with “DataEncrypt” component data is decrypted and show the result to user.

* End user- Get some suggestion

The user has already logged and now he/she is interacting with “UserMobileApp” component.

First, the user go to Suggestion section through “UserMobileApp” component and “UserMobileApp” component send a request to “Suggestion” component. “Suggestion” component needs to know about user cars status like battery percentage, user current location and user calendar. So, “UserMobileApp” component get user current location via “GoogleMapService” component and get car status via Bluetooth or API and send these data to ”Suggestion” component. This component retrieves user calendar with “GoogleCalendarService” component. Then send a request to “Searching” component to find charging stations which are suitable for user cars and with his/her time frame that finds out with “GoogleCalendarService”.

After that, “Suggestion” component make list of suggestion for user and send it to “UserMobileApp” component. User confirm one suggestion. So, “Suggestion” component send a request to “Booking” component to save a booking for user and set a calendar appointment with “GoogleCalendarService” component.

* CPO-view external, internal status and location of charging stations

The CPO has already logged and now he/she is interacting with “**CPOWebApp**” component.

First, the CPO go to My Charging station section, “**View Location**” component send a request to “Model” component and this component send a query to “DBMS” component to find all the charging stations of CPO with their location then return the list to “**CPOWebApp**” component. The CPO can see all external status and charging process with “View External Status” component and “Charging Process Component”. Also, the CPO can see all internal status through “View Internal Status” component. All data are encrypt, So with “DataEncrypt” component we try to decrypt it.

* CPO-Select DSO to acquire energy & Edit Price of energy

The CPO has already logged and now he/she is interacting with “**CPOWebApp**” component.

If CPO want to edit price of energy or to know the latest price for energy, must send a request through “Set Service Fee” component. This component call another component which name is “DSO” component. “DSO” component call the all DSO’s API to get latest update of price and return it to “CPOWebApp”. CPO can add some profit to one DSO price and save it. In this case, “Set Service Fee” send a request to “DBMS” component to update price of energy also send a request to DSO to acquire the energy with “Select DSO” component.