0.1 Reason on implementing a Web Client

As the application has its target user in the administrative world, it was decided that a web will be created alongside the mobile application that will make the usage of our services easier to our customers in administrative environments. It has been decided that the mobile app can be really useful for doing little tramits, for chatting with some customers, for quick searches of products/services and enterprises...
But it has been felt that in order to make it more usable and accessible for administrative porposes a web client should also be created.

Furthermore, the implementation of the web client shined some light on how to communicate with the backend and has lead us to iterate a little bit on the API endpoints that were implemented on the last sprint. This is nice because now on the third sprint the integration of the mobile app with the backend will be a lot more straightforward.

0.2 React

For implementing this Web Client React has been used, which is a library built in javascript for creating User Interfaces. To be more precise React was used because it gives the opportunity of working with Hooks. Hooks are functions that comprehend inside them inmutable components with independent states, wich makes the code a lot more clean and structured and simplifies the amount of chaos that has to be dealt with usually when it's programmed with vanilla javascript.

Also React has a huge community, and it has a lot of libraries that have been key for making a lot more easier the programming task. Some examples can be:

- react-router-doom
- react-modal
- redux-react-session
- react-bootstrap
- ...

0.3 Web Client Architecture

By now This Web Client is still in a development phase, so for now the default testing server that React provides to us is being used. Although, a nginx+docker is expected to be used when deploying our application to a production enviroment.

Right now the application is able to fully connect with the backend, beeing able to Register, SignUp and interacting with all the different features that our application provides.

0.4 Redux

In this part of the project, redux has also been used. In fact a react library called redux-react-session that builds a store to maintain the state of our session is being used. Also it creates the correspondig Session Cookie to maintain the value of the authorization token provided by our backend. This way, every time a request that needs authorization has to be made, it will be possible to pass the value of the token stored in the browser cookie within the request .