##introduction to software part

This project contains not only hardware engineering part but also a big part of software engineering and programming. Software development related to this project can be separated in several parts. First and most obvious is programming of microcontrollers ESP-32 and Arduino Mega. But there are also other important parts of software mainly web and mobile applications used for telemetry, testing of individual components, testing of different scenarios and finally application that is meant to be used as the final tool of controlling and flying the airplane.

#Mobile and web applications

Since all the data exchanged between the pilot (on the ground) and the airplane (in the air) goes through Wi-Fi connection and uses WebSocket protocol the most natural form of visualizing and generating that data was through creation of web applications. Later it was enhanced, and all these web applications got their mobile application counterparts. It was logical step forward since not all testing takes place in the laboratory and in front of the computer. Also piloting application needed to be portable from the very beginning since it is much more suitable to control the aircraft by gliding the fingers over the tablet or phone screen than by holding the bulky laptop.

Web application were developed by using JavaScript and HTML5 with several frameworks and libraries like jQuery and Bootstrap, and all the networking features were utilized by using the existing Web API.

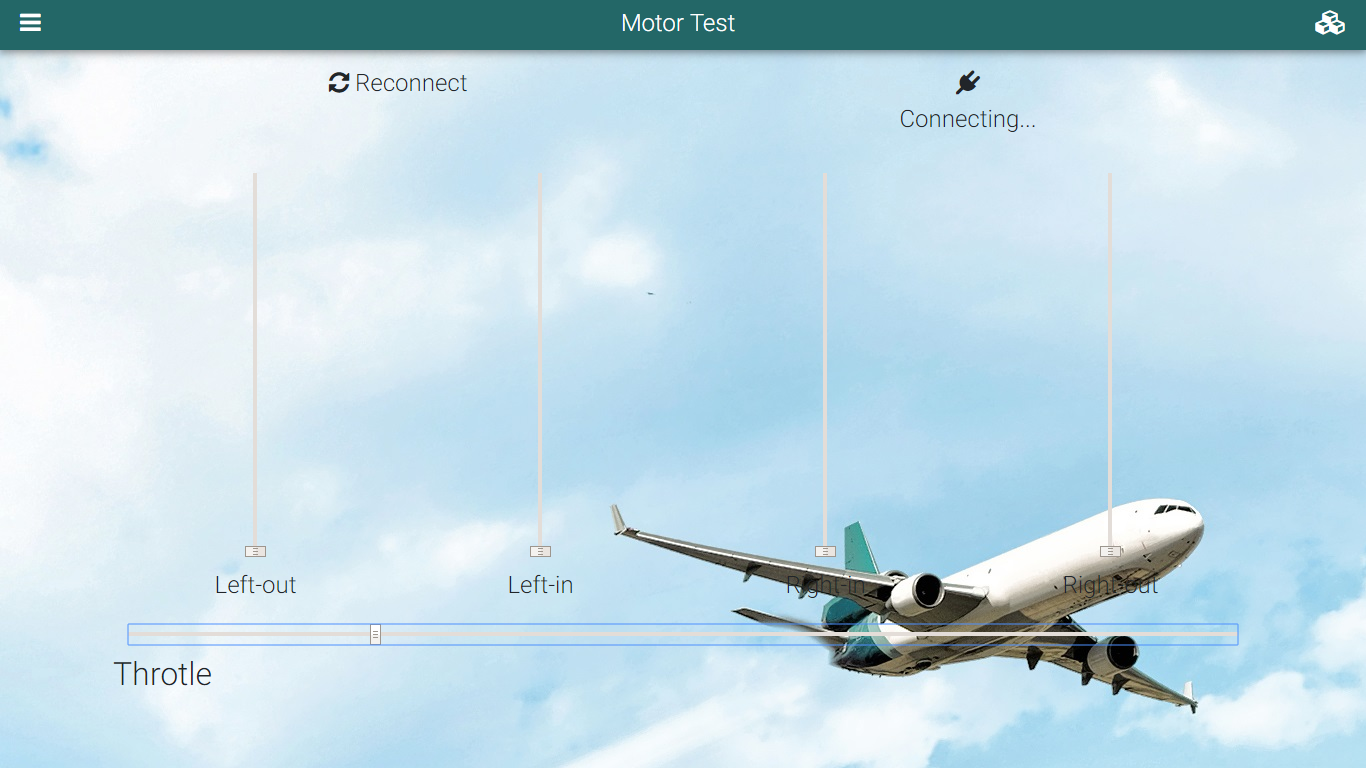
Mobile applications are developed by reusing majority of the code needed for the web applications and combining it with Apache Cordova framework and its plug-ins.

#testing and Telemetrics application

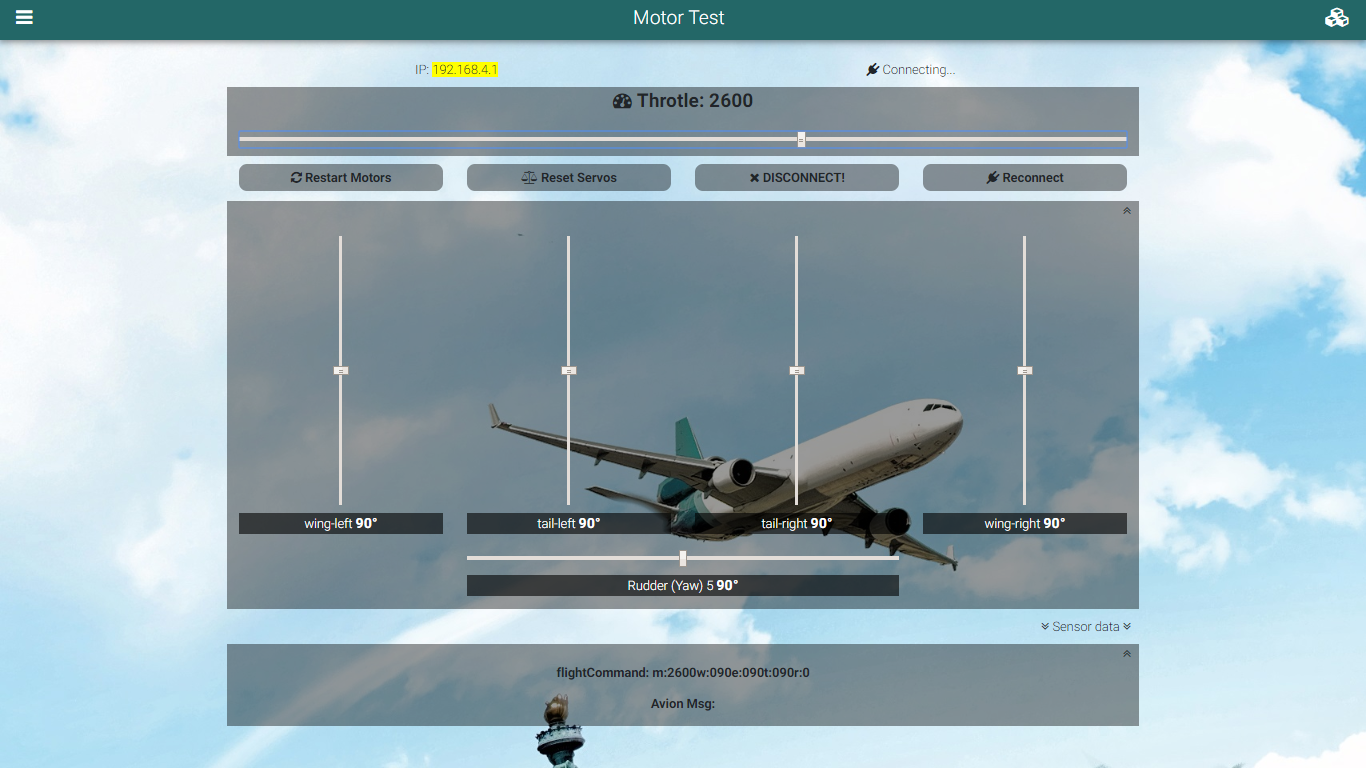
first application that we developed was for testing purposes. It went through several iterations since every new component added to the aircraft needed to be represented and tested through this application.

In the first version it was used to test the stability of the connection between the user and the aircraft (this is addressed again in communication section).

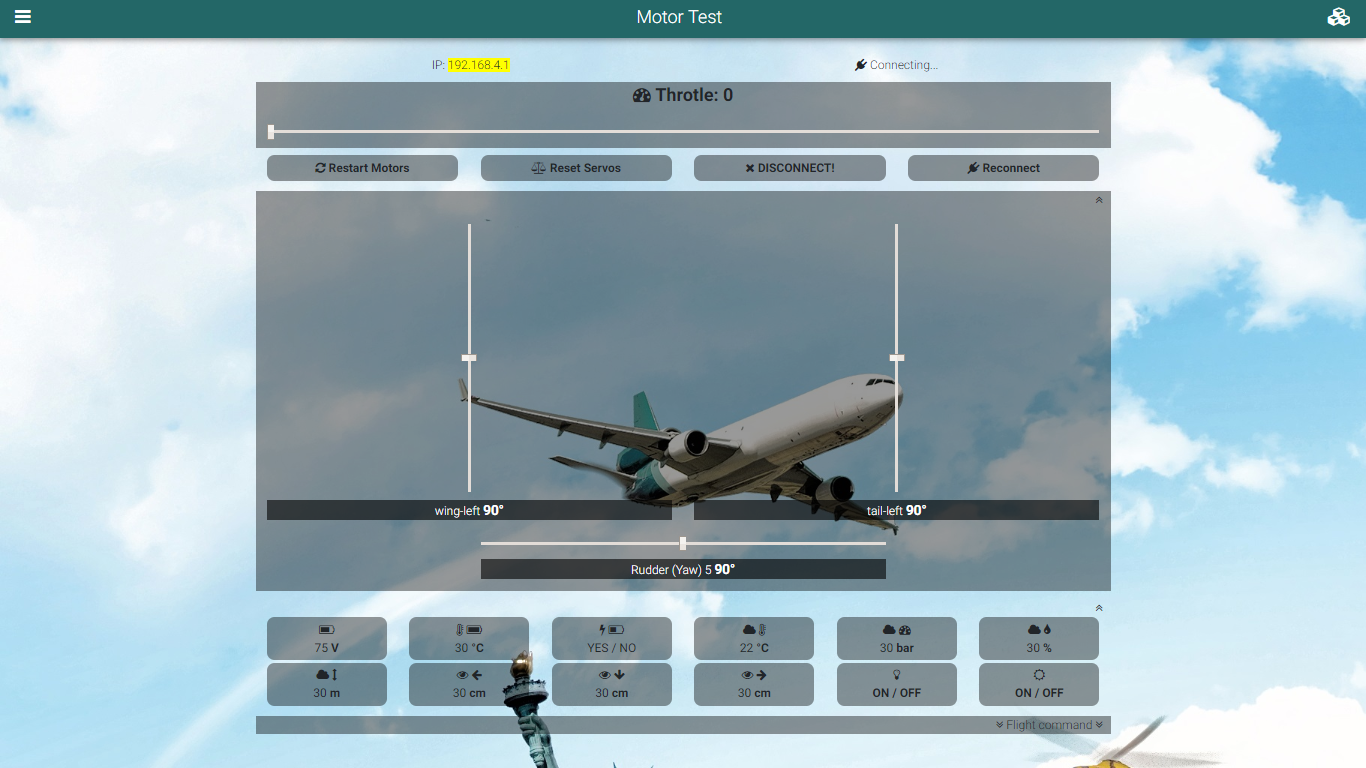
Next, we added the user interface needed to test and calibrate the motors, and servo motors.



As we added more sensors, we also needed the way of monitoring the data coming in from the sensors, so we improved the application once more to be able to interpret the strings of data received from the airplane and visualize it in understandable way.



Finally, we worked on improving the speed and reliability of the connection and optimized the amount of data that must be sent from the application to the aircraft. It also brought slight redesign to the application.



#pilot application

Pilot application inherits all the features from the testing application, but some data is hidden from the user since it is not flight critical and it would only obstruct the screen real estate. The main difference is in the GUI design since the pilot application resembles the common design of RC Airplane joysticks.

