Presentazione:

Intro:

CatalogServer:

The Catalog.json file as the aim of device register and …., it communicates with the other component through REST protocol. There are three main section. The section with the general information about the application, for example the IP address in which the catalog is reachable through REST, the base URL of the topic, the Broker and the port of the application for the MQTT communication paradigm, the Telegram token for the bot of the application, and general information about the ThingSpeak.   
The section of Device List with the information of the specific device: name, ID, user associated, type of measure, unit, communication paradigm used (e.g. topic), value sampled and timestamp of the last update, that was foundamental to understand if the Device does not work or is temporary disconnected.   
Last section of UserList that contain the information of the user: the main feature of the battery specific for each vehicle and so for each user with the aim to compute the autonomy of the vehicle and the percentage of battery necessary connected to the km necessary, chatID to connect the USER to the specific information about his application on the TelegramBot, the ChannelID to connect channel of thingspeak and USR, the Device connected with the USR, and the Agenda of the USR with the number of kilometers necessary to the user to the specific appointment and the day in which is.

The Catalog communicate with all the other actors in the platform exploiting REST communication.

The GET method was used to retrieve information about the application, showing a part of the catalog. For example the URL AllUsers shows all the UserList, or a specific USER, the catalog..

The PUT methods was used to update information, for example the sensors use the PUT method to publish the value sample and the timestamp in which was sampled and this is done during all the Demo by the sensors (PARTE DI VIDEO CHE DIMOSTRA). The other method are used by the user to update the information both of the battery feature and of the Agenda section.

Remove – Update – BatterInfo (VIDEO)

The POST method was used to add information on the Catalog, for example add a new device, a new user, or insert the ChatID of the USER. The main feature is the possibility to add an appointment on the Agenda. This feature could be tested by PostMan that simulate a web interface, but also by Telegram through the BOT. Later was shown how.

USER (VIDEO)

DeviceConnector:

Control strategy: is the core of the application, as the aim to receive data from sensor and Catalog, and choose if the battery charger must be on or off. This control strategy is done each 30 seconds, and is done according to this logic. The main control is about the USER request, the user through the User Awareness interface could choose if want the actuator ON, OFF or leave the choise to the control strategy logic. This was done with 3 digital buttons, each of this associated with a value of a variable that is a flag.

Flag=0, actuator OFF

Flag equal to 1, actuator ON

Flag equal to 2, control strategy choise

The control strategy work in this way:

First check is done to understand if the vehicle is in the garage. We use a digital button in order to say that if the digital button, working as presence sensor, is press the machine is in the garage and so we suppose that is plugged to the charging station, if not, is not in the garage, so is not plugged and the control strategy put the charger OFF. Then was checked if the energy is produced by the photo panels, so the home is full of green energy and this energy was used to charge the car, so put ON the charger. The next check is on the percentage of battery needed for the USR during this day compare with the percentage of the battery available. This percentage needed is compute using the information about the feature of the battery, the number of total kilometers that the USER has to do in this day, a plus of 25% to stay in a safe zone, and a plus of the 20% according to the ambient temperature and so the necessity to use the heater or the conditioning.

The percentage of battery necessary is computed according to the info of the Battery feature. If the percentage of kilometers that the user has to do in this day is greater than the max autonomy that the battery could provide, was send an alert to the user through the TelegramBot, otherwise was send the value of percentage battery necessary to the control strategy.

State Control: is an application that collect all the Alert SMS and send them to the user throurh Telegram. The alert are connected to the presence of the vehicle, The percentage of battery is too low, for example if the presence sensor does not work or the control strategy has a problem the USER could activate manually the charger, if the km of the day are too high so could be preferrable if the user stop in a charging station during the trip, and if the temperature battery info is too high, indeed the state control put the charger OFF until the temperature decrease above the normale.

In this demo simulate the case of 4 USER, with USER ID not sorted, and with 3 different scenario. USER 1 is in a location where there is a sunny day so the quantity of the photon is very high and is able to produce enough energy, so independent of the other factor the control strategy put the charger ON.

In the case of the USER 2 the percentage of battery necessary according to the Agenda of this day is more than the one available, so actuator ON, and percentage of battery increasing.   
  
In the case of the USER 4, the percentage of battery necessary according to the Agenda of this day is less than the one available, so actuator OFF and percentage of battery decreasing very slow.

This could be visible through ThingSpeak and Node-red that use a graphical interface to represent the scenario and make the user awareness about the working of the application

Node red allow also the user to interact with the application, with the button to set the charger manually or by control strategy.

ThingSpeak – Data Analisys:

TelegramBot: