G.H. 101

MAKE YOUR CITY GREEN

Team members

Berti Marco	s315819
Brozzo Doda Umberto	s314562
Dettori Giovanni	s315017
Macario Davide	s315054

The idea - overview

Our goal is to help people take care of their plants without worrying about watering them or how much light they need through an user friendly platform and remote monitoring

Retrieving and collecting data from remote sensors

Analysis of parameters for optimal plant growth

Analysis of environment parameters for weather prediction

Remote access to greenhouse consumption and needs

Communication protocols used

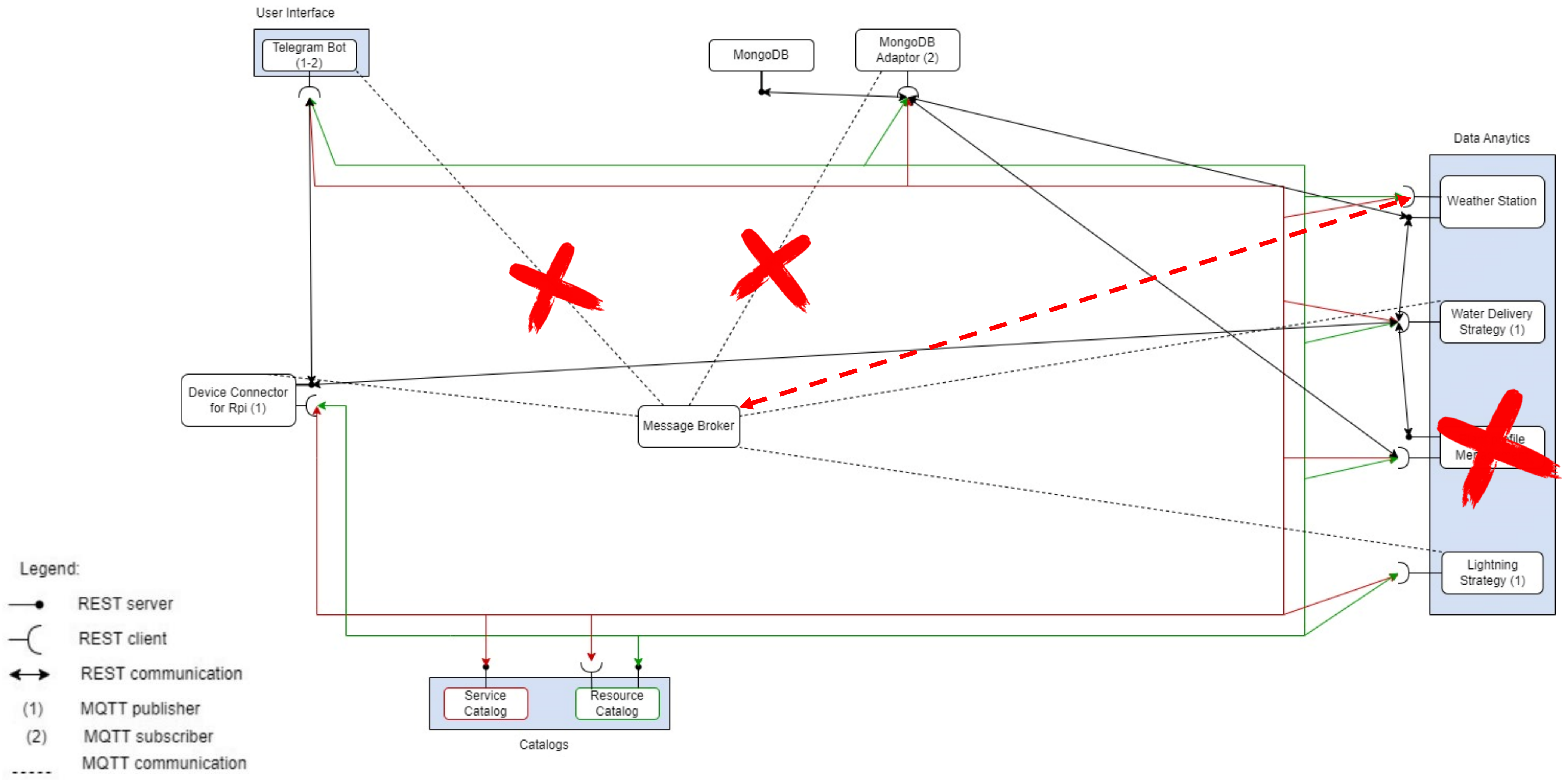
REST Web Service

- It provides synchronous communications among the microservices using CRUD methods .
- It exposes addressable resources through URI

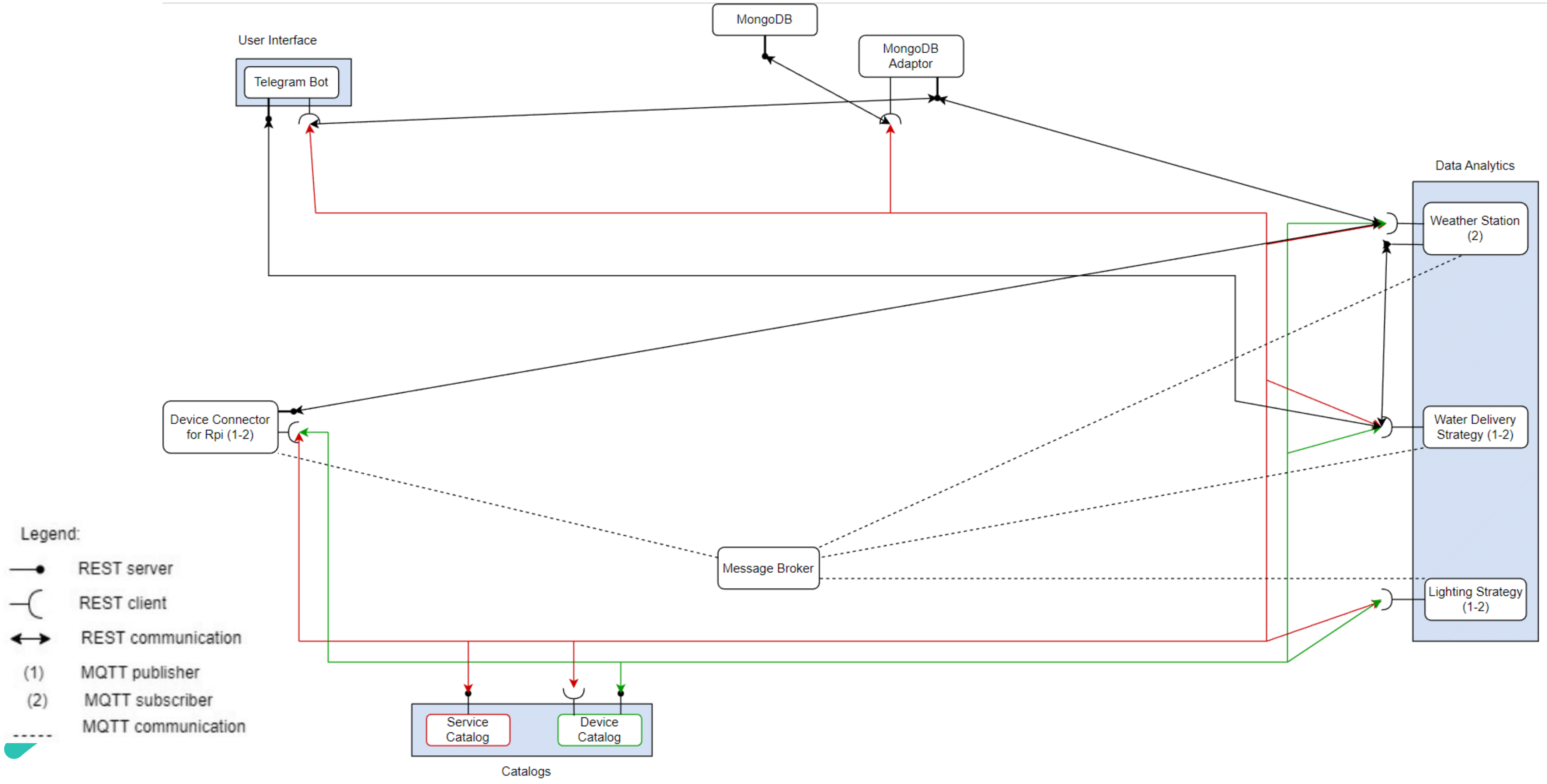
MQTT

- It provides asynchronous communications among publishers and subscribers using a message broker.
- message exchange is based on topics

Use Case Diagram - Proposal



Use Case Diagram - Modified



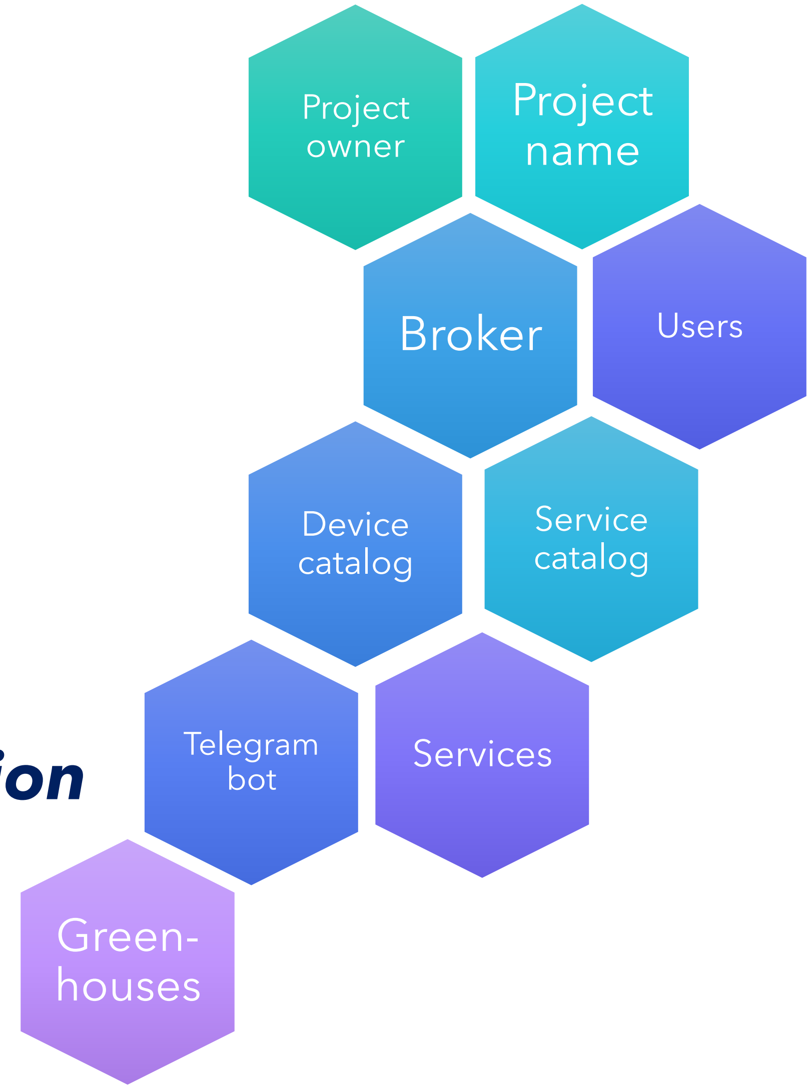
Microservices

Step by step

Service catalog

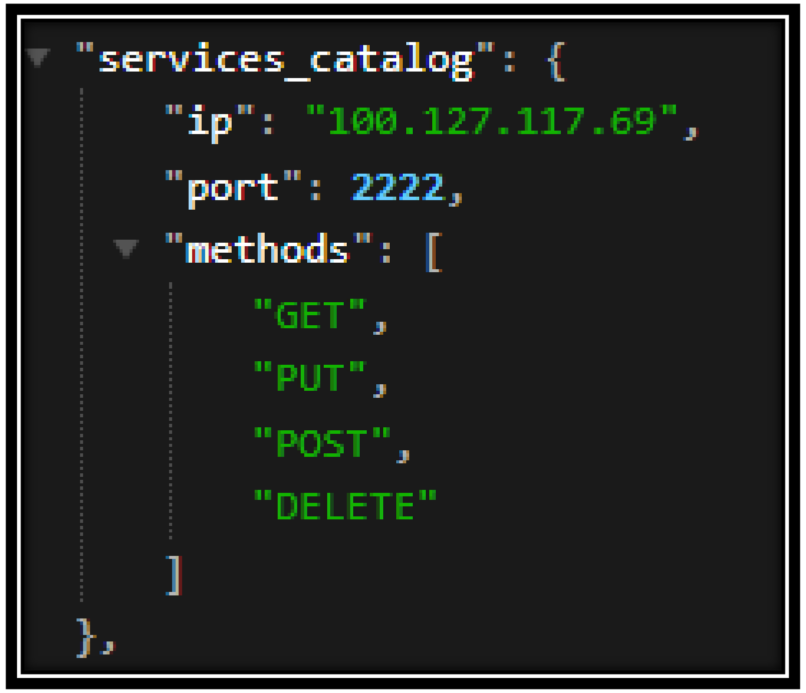
It is the service registry for the application. It allows different application entities to obtain a list of all available services and how to interface with them (i.e., the needed APIs) and it provides the means for service advertisement.

Stored information



Service catalog

Messages exchange with this element happens over REST Web Services with JSON files and each application element must retrieve information upon start up. The Service catalog must be known by every microservice in advance.



```
"services_catalog": {
    "ip": "100.127.117.69",
    "port": 2222,
    "methods": [
        "GET",
        "PUT",
        "POST",
        "DELETE"
    ]
},
```

Service Catalog

GET

It retrieves the information.

- Project info
- Broker
- Telegram bot
- Device catalog
- Users / user (*given ID*)
- Greenhouses / greenhouse (*given ID*)
- Services / service (*given ID*)

POST

It inserts the information.

- Device catalog
- User
- Greenhouse
- Service

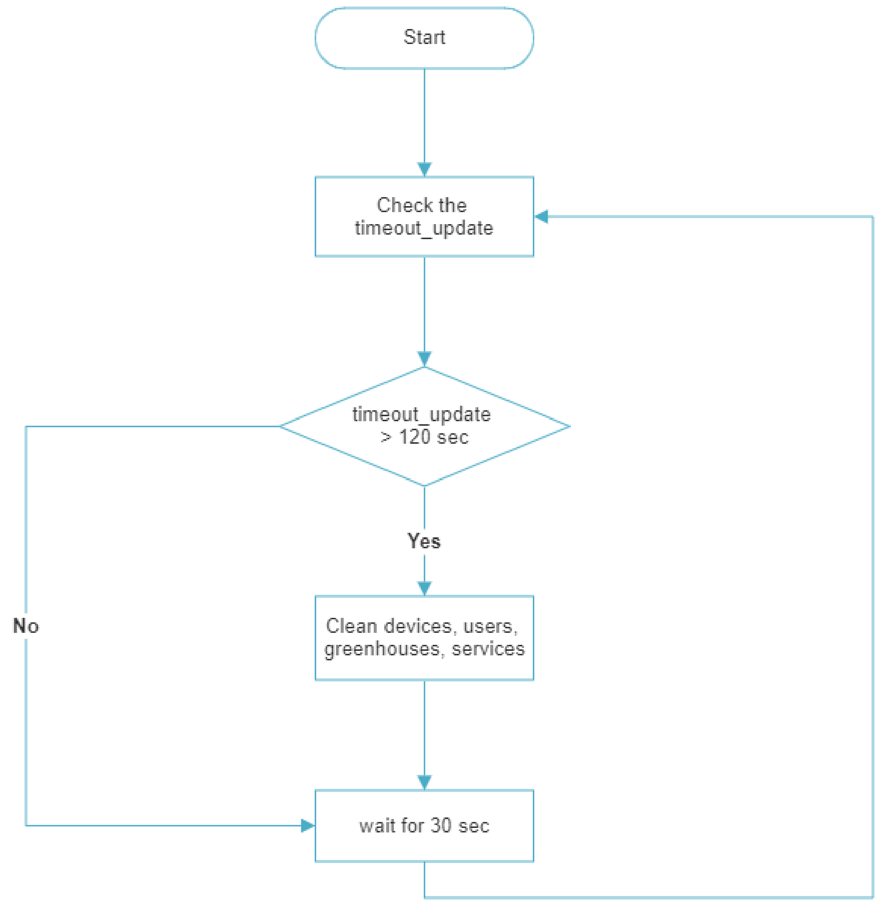
PUT

It updates information.

- Device catalog
- User
- Greenhouse
- Service

Service catalog
Main loop

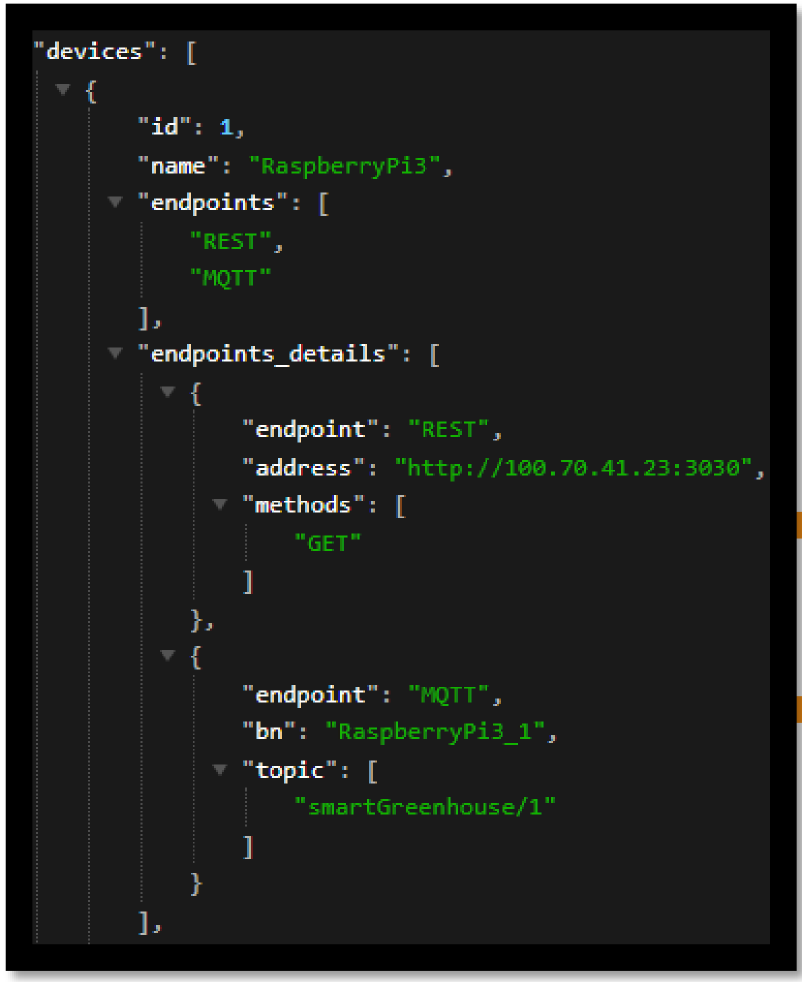
The catalog keeps updated the list of users, greenhouses, services and devices. It controls which are still connected and removes those who are not available anymore.



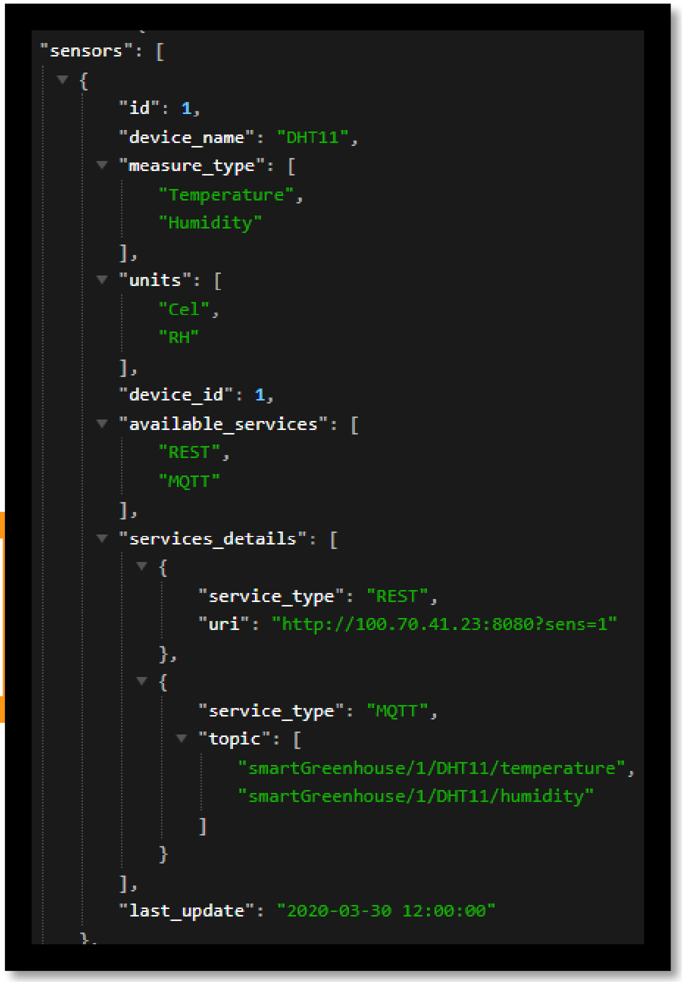
Device Catalog

It is the device registry for the application, providing a list of all available devices and the resources they expose.

Each device exposes all sensors that a micro-service can use.



```
"devices": [
  {
      "id": 1,
      "name": "RaspberryPi3",
      "endpoints": [
          "REST",
          "MQTT"
      ],
      "endpoints_details": [
          {
              "endpoint": "REST",
              "address": "http://100.70.41.23:3030",
              "methods": [
                  "GET"
              ]
          },
          {
              "endpoint": "MQTT",
              "bn": "RaspberryPi3_1",
              "topic": [
                  "smartGreenhouse/1"
              ]
          }
      ],
```



```
"sensors": [
  {
      "id": 1,
      "device_name": "DHT11",
      "measure_type": [
          "Temperature",
          "Humidity"
      ],
      "units": [
          "Cel",
          "RH"
      ],
      "device_id": 1,
      "available_services": [
          "REST",
          "MQTT"
      ],
      "services_details": [
          {
              "service_type": "REST",
              "uri": "http://100.70.41.23:8080?sens=1"
          },
          {
              "service_type": "MQTT",
              "topic": [
                  "smartGreenhouse/1/DHT11/temperature",
                  "smartGreenhouse/1/DHT11/humidity"
              ]
          }
      ],
      "last_update": "2020-03-30 12:00:00"
  },
```

Device Catalog

GET

It retrieves the information.

- Devices / device (*given ID*)
- Devices / device (*given name*)

POST

It inserts the information.

- Device

PUT

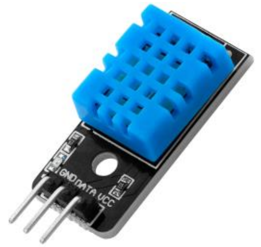
It updates information.

- Device

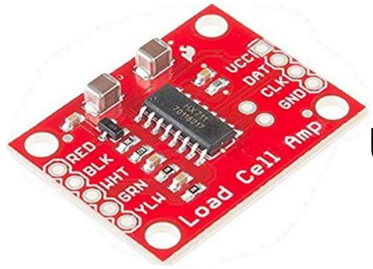
Raspberry Pi connector

It is the Device Connector referring to a specific greenhouse. Each one is equipped with 5 sensors:

BMP180
Used to measure the temperature and pressure

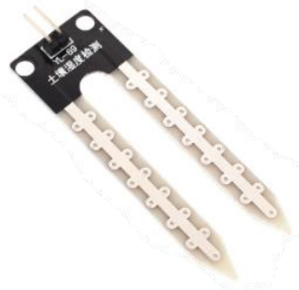


DHT11
Used to measure the temperature and humidity



HX711
Used to measure the tank weight

BH1750
Used to measure the light intensity



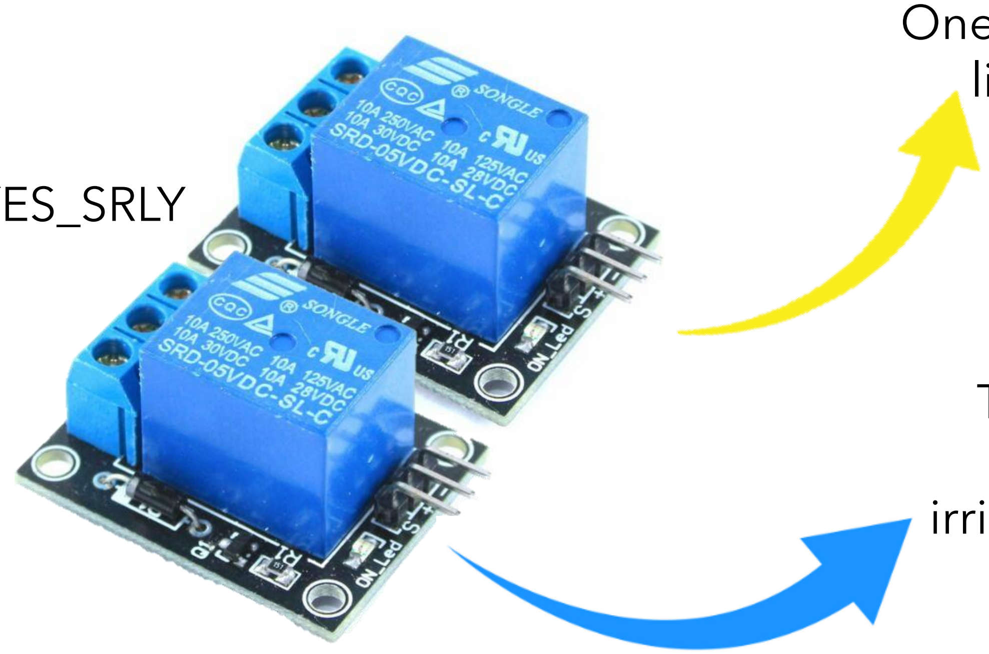
YL-69 (chirp)
Used to measure the soil moisture

Raspberry Pi connector

It is the Device Connector referring to a specific greenhouse. Each one is equipped with 5 sensors.

And 2 actuators:

Two KEYES_SRLY



One to control the light system

The other to control the irrigation system

Raspberry Pi connector

The Raspberry Pi receives as input the information related to the plant the user wants to grow.

Information is exchanged both over:

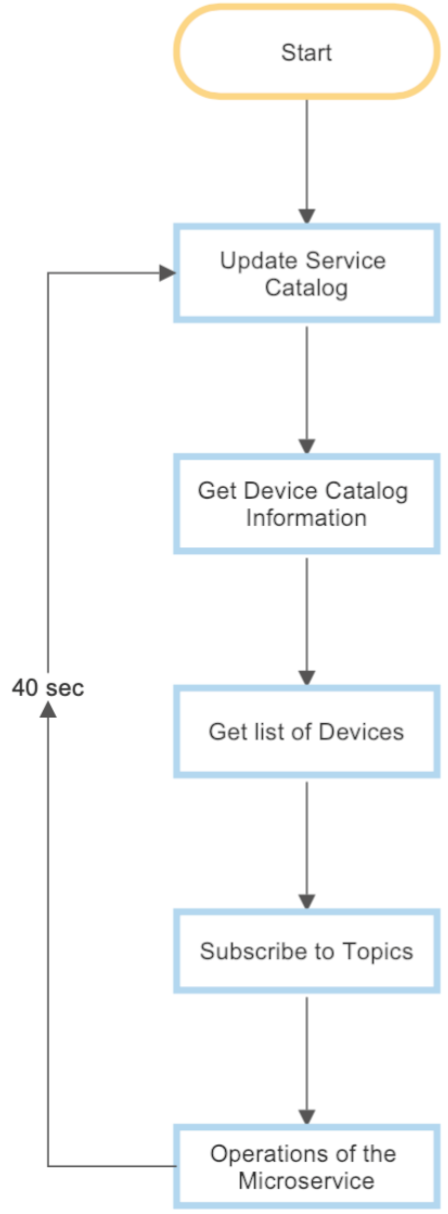
- REST APIs, for instantaneous readings requested by the user though the GET.
- MQTT: every 10 minutes it receives the measures from sensors and forwards them to the proper strategy.

  - ***Lightening strategy***
  - ***Water delivery strategy***
  - ***Weather station strategy***

Main loop of the strategies

In the initialization (start), strategies register to the service catalog.

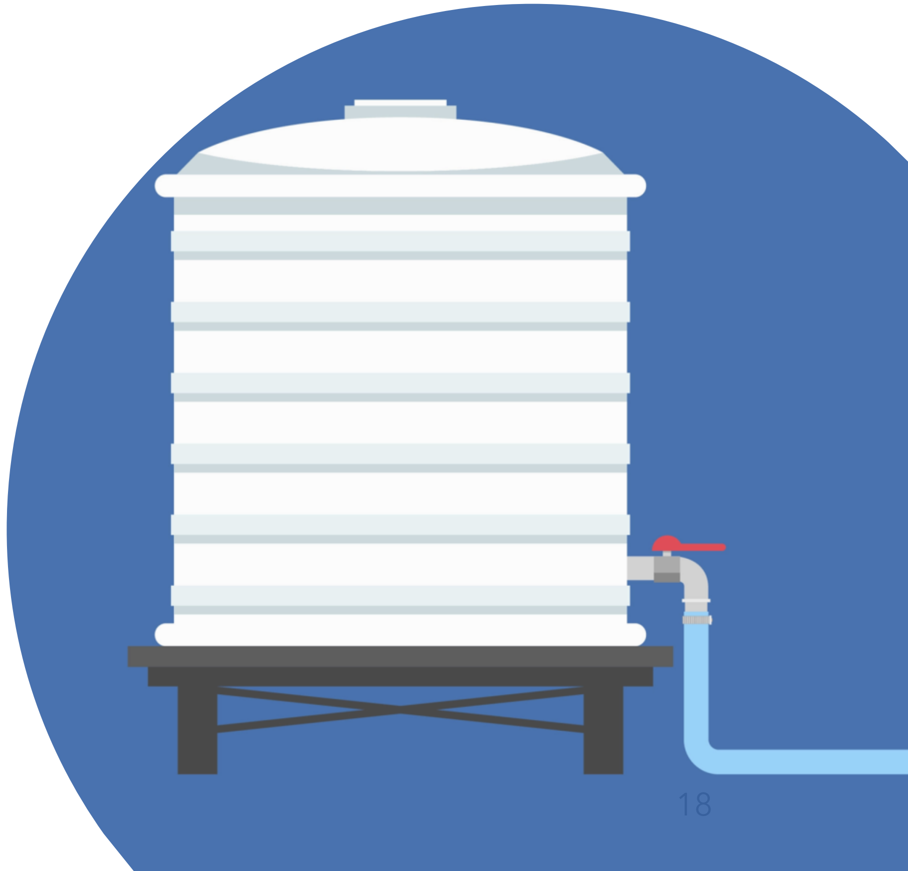
Then, the main loop is started and every 40 seconds the strategy updates its information to the service catalog to prevent the timeout from expiring.



Water delivery strategy

It is a microservice for the application which is able to read data from sensors (water tank level and soil moisture) and decides if it is necessary to water the plant.

- Water tank level: based on quantity of water in the tank and on the weather (it will rain or not tomorrow) it can send a message to the user to manually fill the tank or to let the rain fill it
- Soil moisture: by comparing the sensor value with the water needs of the plant, it can decide to water the plant or not through an actuator.



Lighting strategy

It is a micro service for the application which is able to read data from a sensor (light intensity) and decides if it necessary to switch on/off the light

- Light intensity: by comparing the sensor value with the light needs of the plant during the day, it can decide to light the plant.

Weather station

It is a microservice for the application which is able to read data from the sensors (temperature, pressure and air-humidity) and provide to the user (via Telegram) the current weather.

Information is exchanged both over:

- *GET:*
  - *Current weather: used to inform the user via Telegram about the current weather.*
  - *Will it rain: used by the water delivery strategy to know if the day after will rain.*
- MQTT: *used to receive sensors values and fit them to perform the machine learning algorithm*

Weather station

Additionally, it calls the ***weather prediction*** to retrieve weather (rain) forecasting respect the present and the next day.

Using past data retrieved from *Meteo.it* we trained the Quadratic discriminant model to have the prediction. At the and we obtain an accuracy of 83% for the present day and 77% for the next day. These two output are good considering that the common accuracy of weather foreign is around 89%.

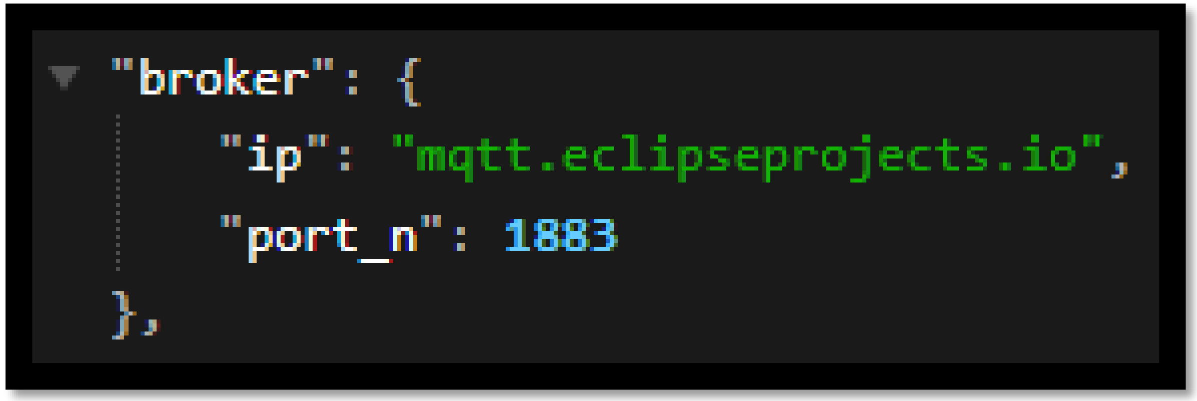
For each day we have used:

```
['TMEDIA','TMIN','TMAX','UMIDITA','PRESSIONESLM','STAGIONE','FENOMENI']
```

Where 'FENOMENI' gives the output (rain presence or not) and 'STAGIONE' has been used to improve the performance.

Message broker

It is a server which enables asynchronous communication by storing messages exchanged via MQTT under different topics. It is the key element in asynchronous communication.



```
"broker": {
    "ip": "mqtt.eclipseprojects.io",
    "port_n": 1883
},
```

mongoDB®

It is a non relational database that exchanges JSON documents.

It is a service used for managing data, exchanged over REST Web Services. It is used to store measured data coming from the weather prediction strategy and a big collections of 2759 plants.

The plant's database has been retrieved from a bigger online database. Thanks to data cleaning operations we have corrected and rearranged the records.

MongoDB adaptor

It makes possible the communication between the database and the other microservices.
In particular, it talks through REST with:

TELEGRAM BOT: to retrieve the information of a plant and chose the best ones based on user preferences (e.g.: size, country, temperature,…).

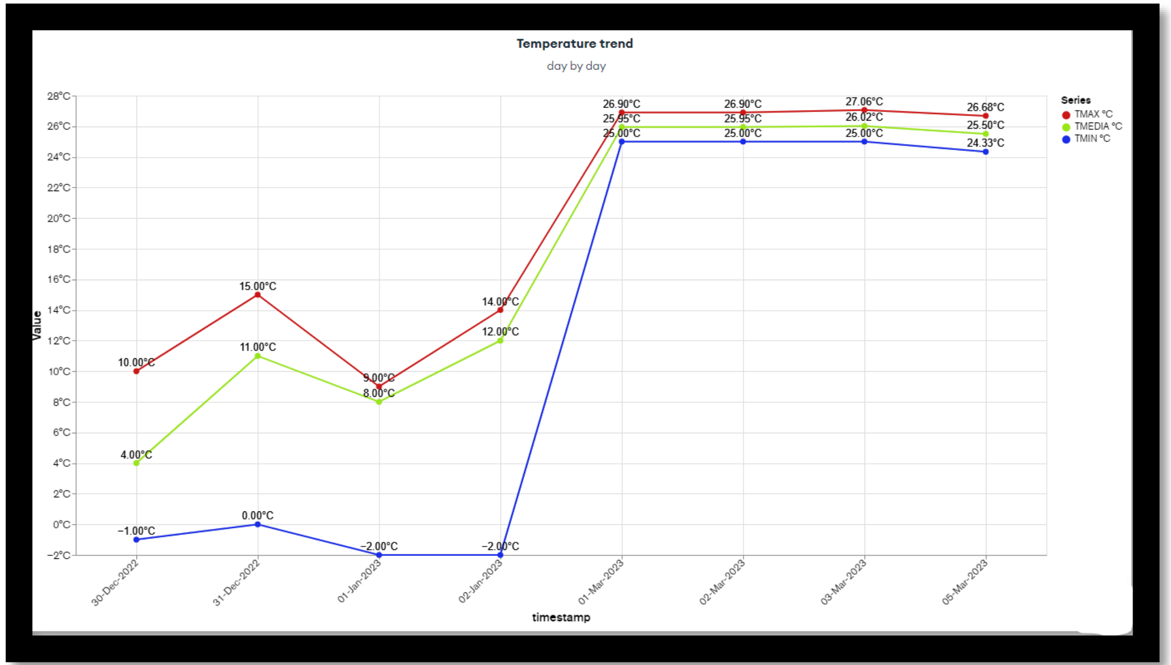
WEATHER STATION: to memorize the statistics of all days providing an historical trace of the behaviour of the greenhouse. In this way the user can control the wealth of the plant.

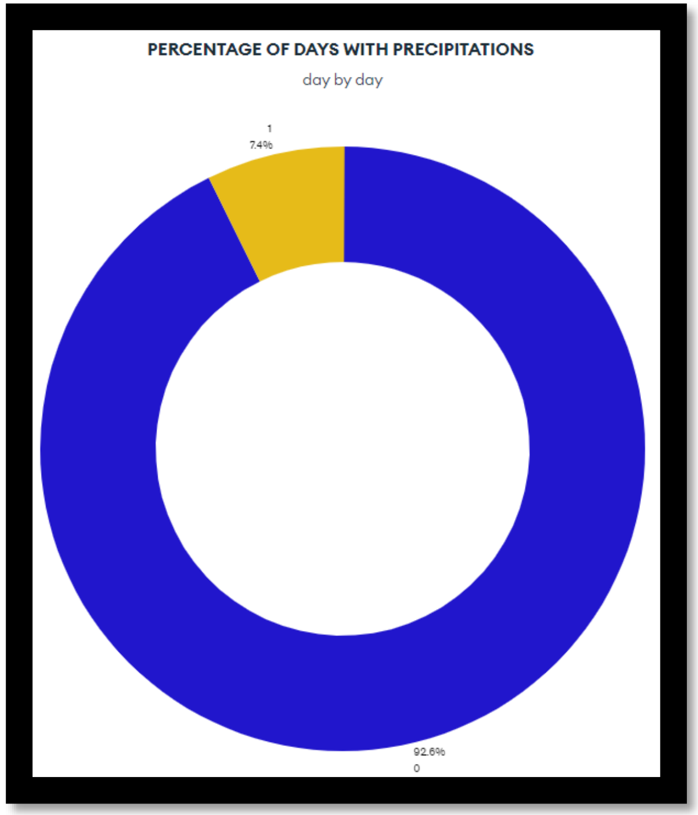
MongoDB charts

This powerful platform lets to visualize data in a intuitive way.

In our application we have given the possibility to the user to retrieve four graphs through the URL of a dynamic image.

We provide the temperature, the percentage of days with precipitations, humidity and pressure.





Telegram Bot

The user interface is provided by the telegram bot. The user just need to press the “START” button to access the application and receive the list of command he/she can perform.

Telegram Bot

At this point the user chose one of the following commands on chat:

/addUser: to add a new user

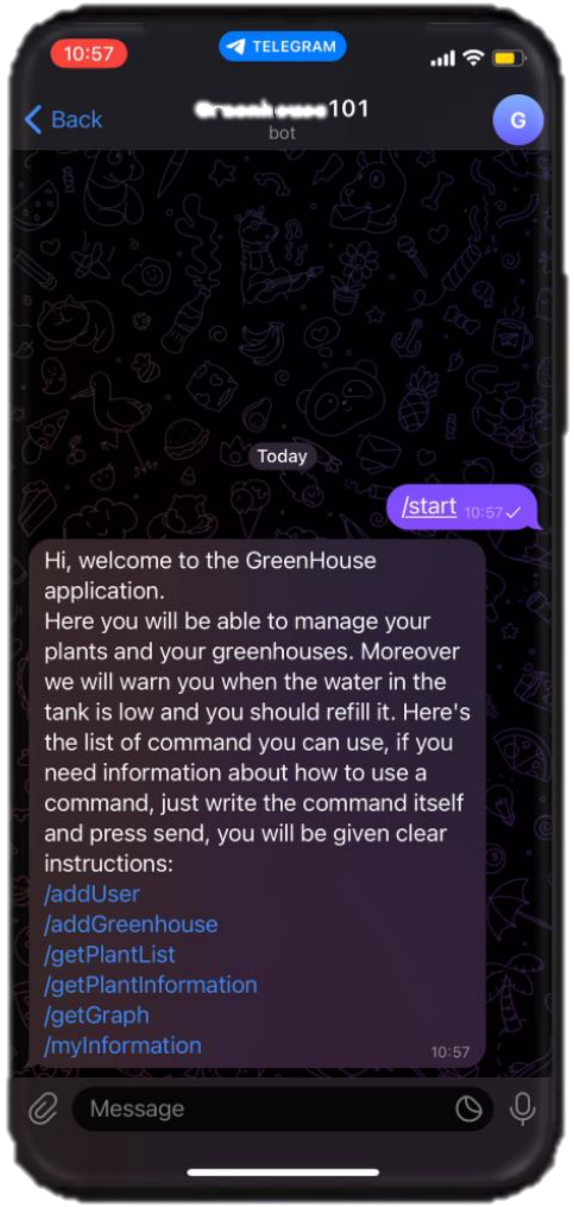
/addGreenhouse: to add a new greenhouse

/getPlantList: to get a list of plants matching a need of the plant

/getPlantInformation: to get the information of a plant

/getGraph: to get graphs about weather and temperature

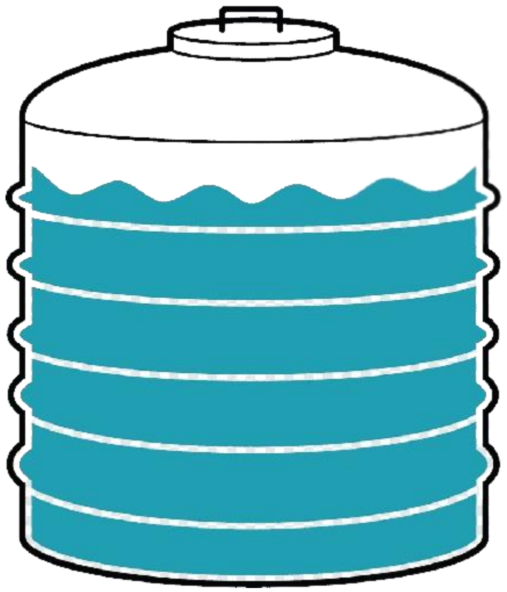
/myInformation –> to get user's data



Hi, welcome to the GreenHouse application.
Here you will be able to manage your plants and your greenhouses. Moreover we will warn you when the water in the tank is low and you should refill it. Here's the list of command you can use, if you need information about how to use a command, just write the command itself and press send, you will be given clear instructions:
/addUser
/addGreenhouse
/getPlantList
/getPlantInformation
/getGraph
/myInformation
10:57

Telegram Bot

It implements a REST interface, in particular:



POST: provides an opportunity for the water delivery strategy to contact the user to alert him that the tank level is low.
In this way he can refill it.

Thanks for your attention

Gruppo 17