

PURPLE IOT

PROJECT PRESENTATION

Jeljeli Hamza
jeljelihamza@gmail.com

INDEX

1. What is Purple IoT ?
2. Architecture
3. Exploring Purple IoT
4. Concepts
5. Used Technologies

WHAT IS PURPLE IOT ?

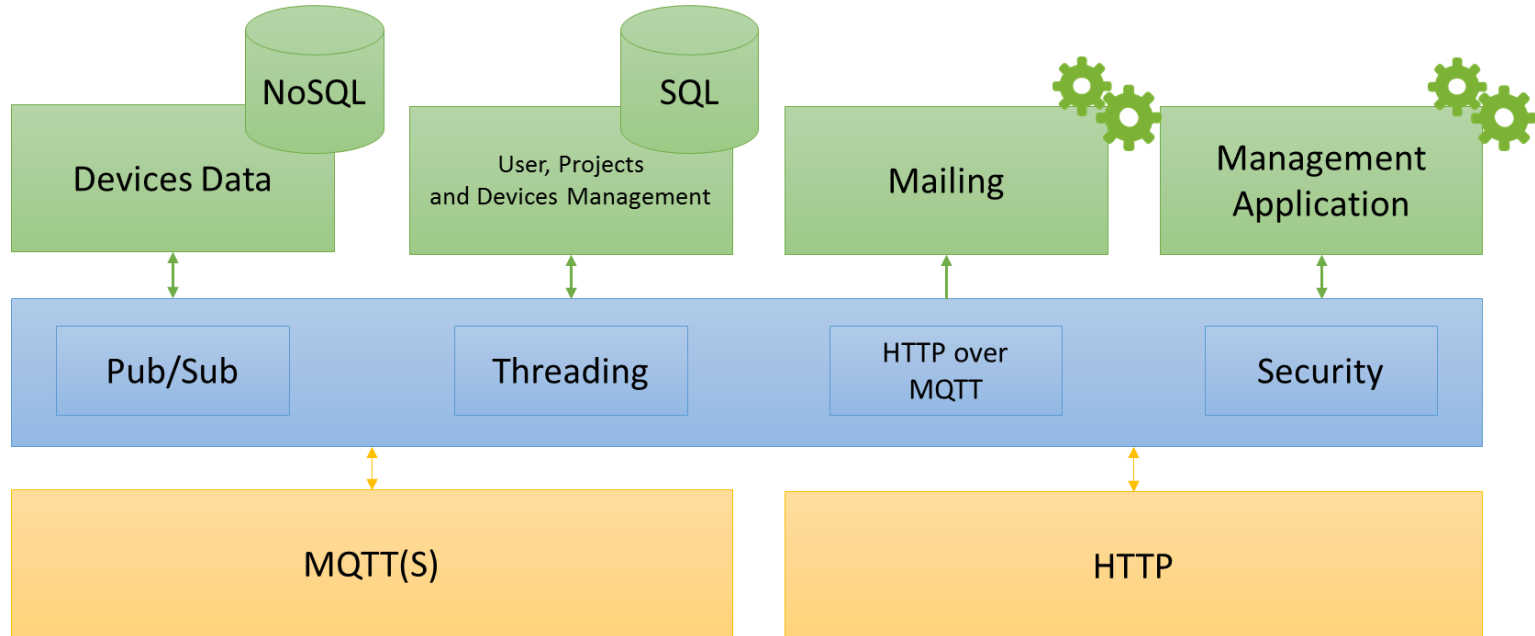
❖ **Purple IoT** is a Software as a service (SaaS) who provides a set of tools for Internet of things (IoT) data exchange through MQTT/HTTP.

❖ **Purple IoT** offers :

- ❖ Connectivity interfaces : to collect data from devices (sensors, actuators ..) through MQTT & HTTP protocols.
- ❖ Projects, Gateways & Devices Management
- ❖ Data monitoring : to monitor data collected from devices in real time.
- ❖ Data storage and viewing : Webservices with authentication OR using API keys.

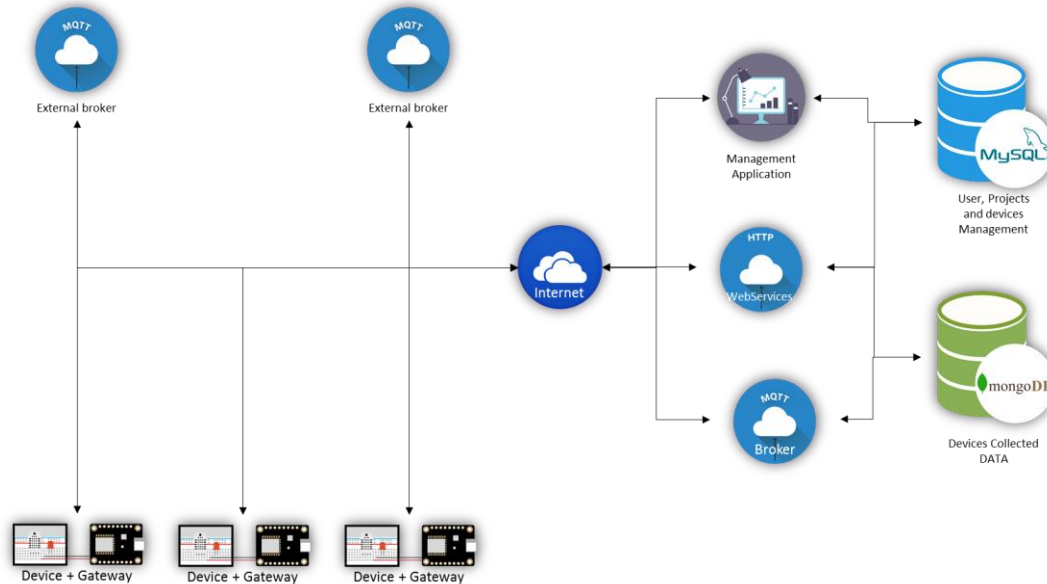
ARCHITECTURE

LOGICAL ARCHITECTURE




ARCHITECTURE

PHYSICAL ARCHITECTURE



EXPLORING PURPLE IOT




Hello! let's get started
Sign in to continue.

hamzajeljel

SIGN IN

[Forgot password?](#)

Don't have an account? [Create](#)



New here?
Signing up is easy. It only takes a few steps.

Username

Name

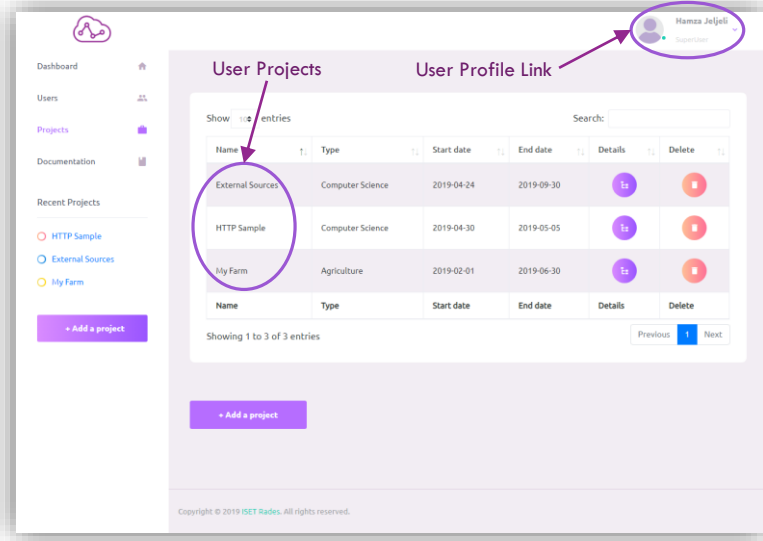
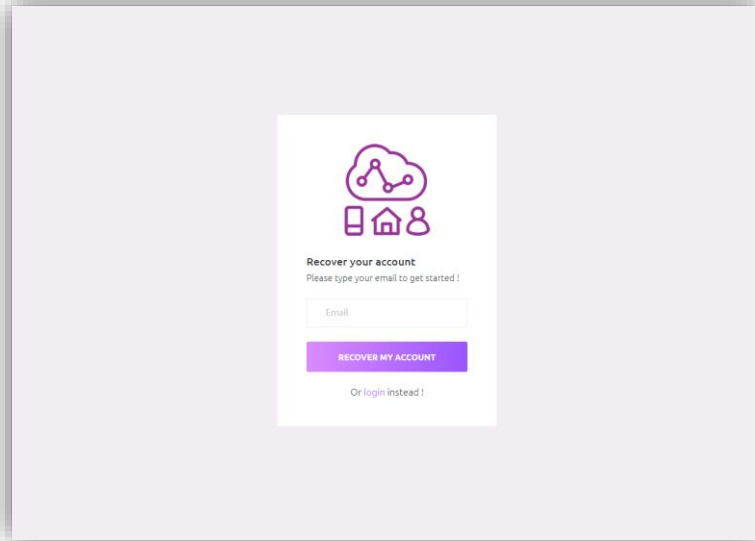
Email

☐ I agree to all Terms & Conditions

SIGN UP

Already have an account? [Login](#)

EXPLORING PURPLE IOT



EXPLORING PURPLE IOT

The screenshot shows the 'My Farm' project details page. The left sidebar contains a navigation menu with 'Dashboard', 'Users', 'Projects', and 'Documentation'. Below this is a 'Recent Projects' section with links to 'HTTP Sample', 'External Sources', and 'My Farm'. A purple 'Add a project' button is at the bottom of the sidebar. The main content area has a purple header with a briefcase icon and the title 'My Farm'. Below this are two sections: 'Project Details' and 'Gateway Details'. The 'Project Details' section includes fields for 'Project Name' (My Farm), 'Project Type' (Agriculture), 'Begin Date' (01/02/2019), and 'End Date' (30/06/2019). The 'Gateway Details' section includes fields for 'Gateway Tag' (ESP8266-CLIENT1056), 'Contained Entity' (Please choose (Optional) ...), and a checkbox for 'This gateway is a client to another Platform'. Below these are fields for 'IP Address' and 'Port'.

Dashboard

Users

Projects

Documentation

Recent Projects

- HTTP Sample
- External Sources
- My Farm

+ Add a project

My Farm

Project Details

Project Name: My Farm

Project Type: Agriculture

Begin Date: 01/02/2019

End Date: 30/06/2019

Gateway Details

Gateway Tag: ESP8266-CLIENT1056

Contained Entity: Please choose (Optional) ...

☐ This gateway is a client to another Platform

IP Address: IP Address

Port: Port

The screenshot shows the 'Devices Management' section. The left sidebar is the same as the previous screenshot. The main content area has a purple header with a briefcase icon and the title 'Devices Management'. Below this are two sections: 'Entity' and 'Devices listing'. The 'Entity' section includes a dropdown for 'Please choose (Optional) ...', a checkbox for 'This gateway is a client to another Platform', and fields for 'IP Address', 'Port', 'Authorization Username', 'Authorization Password', 'MQTT Topic', and 'Protocol'. The 'Devices listing' section includes a table with columns 'Device ID', 'Device Type', and 'Actions'. The table lists two devices: 'SENSOR_LhDxW248SPYdg' and 'SENSOR_xFFcA9QfQDB59Rl', both of type 'Sensor'. Each device has two action buttons: a blue square and a red circle.

Entity

Please choose (Optional) ...

☐ This gateway is a client to another Platform

IP Address: IP Address

Port: Port

Authorization Username: Authorization Username

Authorization Password: Authorization Password

MQTT Topic: MQTT Topic

Protocol: MQTT

Devices listing

Device ID	Device Type	Actions
SENSOR_LhDxW248SPYdg	Sensor	
SENSOR_xFFcA9QfQDB59Rl	Sensor	

EXPLORING PURPLE IOT

The screenshot shows the Purple IoT dashboard interface. On the left is a sidebar with navigation links: Dashboard, Users, Projects, Documentation, and Recent Projects. The main area displays the details for a device named 'SENSOR_LhDxIW248StPYdg'. A large purple antenna icon is on the left. To its right, a table lists device attributes: Device ID (SENSOR_LhDxIW248StPYdg), Device Type (Sensor), Minimum Value (0), Maximum Value (100), Location Type (Indoor), and Coordinates (0,0). Below this, there's a 'Manage API Keys' section with a search bar and a table showing 1 entry.

Device ID
SENSOR_LhDxIW248StPYdg

Device Type
Sensor

Minimum Value
0

Maximum Value
100

Location Type
Indoor

Coordinates
0,0
0,0
0,0
0,0

This screenshot shows the 'Acquired data' section of the Purple IoT dashboard. It features a table with columns for Device, Key, and Delete. The table contains one entry for the device 'SENSOR_LhDxIW248StPYdg' with a long alphanumeric key. Below the table, a line graph titled 'Acquired data' shows the data values over time. The y-axis represents values from 0 to 100, and the x-axis shows timestamps from 09/02/2019 09:58:21 to 09/02/2019 09:58:59. The data points are all at a value of 0. A purple bracket on the left of the graph is labeled 'Data Acquisition visualisation'.

Device	Key	Delete
SENSOR_LhDxIW248StPYdg	2D7x9sRzkAnBK6nQ63t4uHtdhFOTeOQuaTU9O49Lq7LUFUcqlg4nmDXXUaEVOldS	

Showing 1 to 1 of 1 entries

Previous 1 Next

Acquired data

SENSOR_LhDxIW248StPYdg Legend

Values

0 10 20 30 40 50 60 70 80 90 100

09/02/2019 09:58:21 09/02/2019 09:58:37 09/02/2019 09:58:48 09/02/2019 09:58:59

CONCEPTS

- ❖ A **User account** is the main key in the whole application. It enables the user to log in to the administration dashboard, authenticate to the MQTT or MQTTS Broker, to view data acquired from the devices through the Web service or to publish data through the MQTT over HTTP bridge.
- ❖ A User account is identified by :
 - ❖ Username
 - ❖ Profile Type (Can be promoted as administrator by a SuperUser)
 - ❖ Password (The SHA-256 Fingerprint of the password is stored in the database not the password in plain itself for security purposes)
 - ❖ Account Status : Blocked/Not Blocked
 - ❖ Set of additional User Informations : Name, Email Address, Phone Number, etc ...

CONCEPTS

- ❖ A **User account** may have one or many **Projects**.
- ❖ A **project** is created by a User which contains the declarations of a set of **gateways** (theoretically) but in the implementation it is only limited in one gateway.
- ❖ After the expiration of the project, every request received from the gateways will be rejected.
- ❖ A Project is identified by:
 - ❖ Name
 - ❖ Project Type : IT Field for example, Agriculture, etc ..
 - ❖ Begin Date
 - ❖ End Date

CONCEPTS

- ❖ A **Gateway** contains the configuration settings of a physical entity. Such settings could be IP Address, Port, Tags, etc ...
- ❖ A Gateway contains the declaration of a set of devices (Sensors, Actuators).
- ❖ A Gateway is essentially identified by:
 - ❖ Tag
 - ❖ Contained Entity : for managing Entities inside each other
 - ❖ Used data exchange protocol : MQTT, MQTTS or HTTP
- ❖ Other informations might be :
 - ❖ isClient : To indicate that this gateway is declared to listen to another broker.
 - ❖ IP Address & Port : Settings used for listening to another broker (In Base64).
 - ❖ MQTT Authentication : Settings used for listening to another broker.
 - ❖ MQTT Topic : Settings used for listening to another broker and Used for HTTP over MQTT Publishing (In Base64).

CONCEPTS

- ❖ The Data sent through the gateway must be in JSON format.
- ❖ **GID** : GatewayID (the broker can add this field if it wasn't found in the message).
- ❖ **SID** : The Device ID (required)
- ❖ **TD**: TimeDate (the broker can add this field if it wasn't found in the message).
- ❖ **VAL** : The value sent by the Device (required)

```
{  
  "GID": "ESP8266-CLIENT1056",  
  "VAL": 25.5,  
  "SID": "SENSOR_LhDx1W248StPYdg"  
}
```

Note : Keys must be written in UPPERCASE.

CONCEPTS

- ❖ A **Device** is the declaration of a **Physical Device** sending data through a gateway to the software.
- ❖ A Device is identified by :
 - ❖ Device ID : Unique
 - ❖ Minimum Value
 - ❖ Maximum Value
 - ❖ Device Position : Using X,Y,Z and also T (Time) Axis
 - ❖ Position Type : Static or Relative to the Gateway

CONCEPTS

❖ **Data** are organized in two databases :

- ❖ A Relational Database : Which contains user, projects, gateways and devices declarations and informations.
- ❖ A Non Relational Database : Which contains Data received from devices. Data are stored in collections. Each collection is named by a Username.

CONCEPTS

❖ Viewing the Data* acquired from the devices or gateways is guaranteed using Webservices :

Link	Method	Description
/WS/SensorsData/publish/	POST	Publish message as MQTT from HTTP.
/WS/SensorsData/get/{device}/	GET	Get the last value received from a device .
/WS/SensorsData/get/{device}/all/	GET	Dumps all the values received from a device .
/WS/SensorsData/get/{device}/{count}/	GET	Get a certain number of values received from a device .
/WS/GatewayData/get/{gatewayTag}/	GET	Get the last value received from a gateway** .
/WS/GatewayData/get/{gatewayTag}/all/	GET	Dumps all the values received from a gateway** .
/WS/GatewayData/get/{gatewayTag}/{count}	GET	Get a certain number of values received from a gateway** .

* : Requires authentication using the administration dashboard credentials.

** : Gateway that listen to an External Broker.

CONCEPTS

❖ Viewing the Data using generated API keys is also made possible through the Webservices :

Link	Method	Description
/WS/api/SensorsData/get/{ device }/{apikey}/	GET	Get the last value received from a device .
/WS/api/SensorsData/get/{ device }/all/{apikey}/	GET	Dumps all the values received from a device .
/WS/api/SensorsData/get/{ device }/{count}/{apikey}/	GET	Get a certain number of values received from a device .
/WS/api/GatewayData/get/{ gatewayTag }/{apikey}/	GET	Get the last value received from a gateway *.
/WS/api/GatewayData/get/{ gatewayTag }/all/{apikey}/	GET	Dumps all the values received from a gateway *.
/WS/api/GatewayData/get/{ gatewayTag }/{count}/{apikey}/	GET	Get a certain number of values received from a gateway *.

* : Gateway that listen to an External Broker.

CONCEPTS

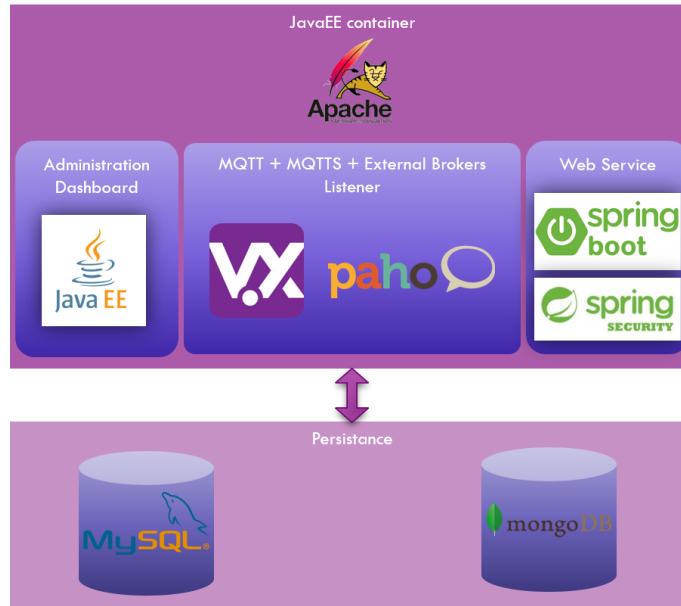
```
{  
  uuid: "39b7cfc9-fca9-49f5-803c-b65e809246d0",  
  gid: "ESP8266-CLIENT1056",  
  val: 25.5,  
  mqtt_TOPIC: "/home/hamza_room/temperature",  
  td: "2019-05-02T07:58:59.000+0000",  
  sid: "SENSOR_LhDxlW248StPYdg"  
}
```

❖ The response from the Webservice is in JSON format and it contains the following fields :

- ❖ UUID : Unique identifier for the message.
- ❖ GID : The Gateway ID
- ❖ VAL : The Value sent by the device
- ❖ SID : The Device ID
- ❖ MQTT_TOPIC : MQTT Topic which the value arrived from.
- ❖ TD: Time and Date of when the value was captured/Saved.

USED TECHNOLOGIES

- ❖ The application was mainly built using **Java** programming language.
- ❖ **Hibernate** framework was used for persistence.
- ❖ **Vert.x** framework was used for building a custom implementation of an MQTT Broker.
- ❖ External brokers listeners are using the **Paho** MQTT Client framework.
- ❖ Webservices are developed using **Spring Boot** and **Spring Security** Frameworks.
- ❖ **MySQL** was for user, projects, gateways and devices declarations and informations.
- ❖ **MongoDB** was used for storing data received from devices.

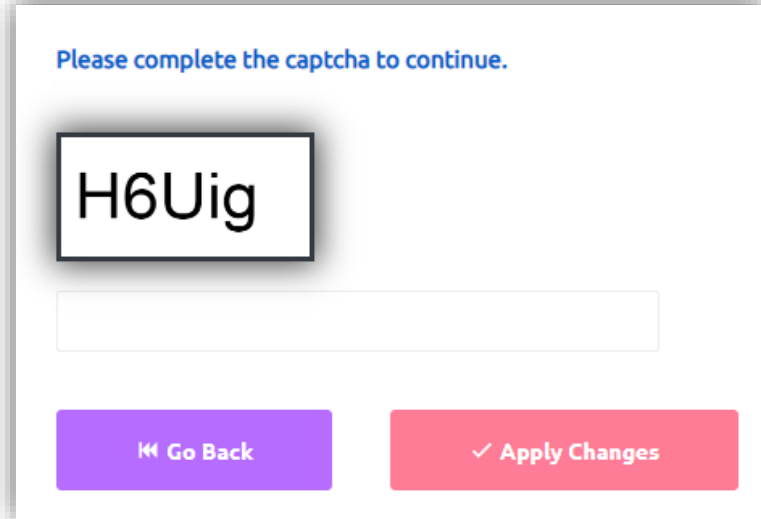


USED TECHNOLOGIES

❖ There was also some frameworks developed specially for this project :

❖ **Captcha.js** : is a simple captcha verification that works on the client side. It simply generates an image from a random string. The random string is never stored in the browser's cache but it's SHA-2 fingerprint is stored.

Source : <https://github.com/hamzajeljeli95/Captcha.js>



Please complete the captcha to continue.

H6Uig

Go Back Apply Changes