Technical Report - Project specifications

Blueberry Pi

Course: IES - Introdução à Engenharia de Software

Date: Aveiro, 05/12/2021

Students:
100055: Afonso Campos
98452: Dinis Lei
93343: Isabel Rosário
98599: Miguel Ferreira

Project abstract: BlueberryPi is a platform used for the management of blueberry plantations, specialized in auxiliating with precision agriculture practices. The platform provides the users with relevant information about the plants' health, the weather conditions, the daily production, the site's storage practices, among others.

Table of contents:

1 Introduction

Precision Agriculture, as its name suggests, is a science that requires a lot of precision and information. It's needed to manage the plantations very closely, checking the vital signs and the weather conditions.

Based on this issue, and to facilitate the productor's life, we decided to develop an app that monitorizes blueberry plantations, checking the plants and weather conditions.

We'll use Software Engineering practices, dividing our team by roles, and using collaborative technologies like GitHub, used to store our project in a repository, and ZenHub , to track the tasks for each member and requirements of the project).

2 Product concept

Vision statement

As the introduction briefly explained, the application will be conceived with the objective of being used by practitioners of blueberry precision agriculture. It will allow this type of farmers to have an organized overview of how their crops are reacting to various weather and soil conditions as well as their own blueberry storage practices.

Additionally, it is worth noting that we originally intended to use Kafka as a message broker between the API and the data generation code. However, that proved to be exceedingly complex for the assignment at hand. We opted, instead, for the RabbitMQ software.

Personas

Jerónimo - Jerónimo is a manager and scientist responsible for overseeing the company's blueberry plantations. He needs to monitor and respond to any plant health related emergencies. His work also involves studying the various factors relevant to the growth and maintenance of the blueberry bushes, their overtime variations and how these affect the plantation's blueberry production.

Main scenarios

Monitoring soil water tension

After various reports of dry leaves in the blueberry bushes of the Guarda plantation, Jerónimo is called to investigate possible causes. He accesses the BlueberryPi platform in order to monitor the latest water tension levels in the plantation. Should he verify that the levels have been above those expected he may then issue a report and recommend increasing the irrigation frequency of the bushes.

Monitoring general irregularities

Jerónimo accesses the BlueberryPi platform to verify any irregularities in Viseu's blueberry greenhouse. By looking at the "Alerts" tab, he notices a "Low store humidity" notification. This alert matches an isolated report from a client, claiming that boxes had been shipped with blueberries that had gone bad. Jerónimo issues a report recommending increasing humidity levels in Viseu's blueberry stores.

Studying soil pH effect in blueberry production
Jerónimo is conducting research on how pH affects blueberry bush productivity and health. He needs to regularly extract data on Minho's experimental plantations. Using the BlueberryPi platform, he can easily check the current as well as previous pH levels of the soil, as well as net blueberry production and the average water consumption.

3 Architecture notebook

Key requirements and constraints

We chose to do virtual data generation instead of actually implementing the sensors because it is a project that requires a lot of different data and in specific conditions, so it would be very difficult to physically generate the needed data.

If we were working with physical sensors, we'd have to carefully study the conditions on which the sensors would be used because many of the plantations will be exposed to the variation of the weather and the sensors are very sensitive to some of those conditions.

There are some key requirements and system constraints that have a significant bearing on the architecture. They are:

- The *Python* script will have to generate specific data periodically, simulating physical sensors.
- The *Message Broker* will have to separate the messages and send different data to different topics.
- The system will only be accessible by the plantation staff, needing a login service to ensure privacy.
- The OLTP database will have to be always up and running to execute all the operations the system needs.
- The OLAP database will have to be ready to store the data and retrieve big pieces of data for specific purposes.

Architectural view

The **Web App** will be developed in *Javascript React.* This presentation layer will interact with the API.

The **Data Generation** will be developed in *Python.* There will be a *Python* script that will simulate a real sensor and generate all kinds of data that we need for our project.

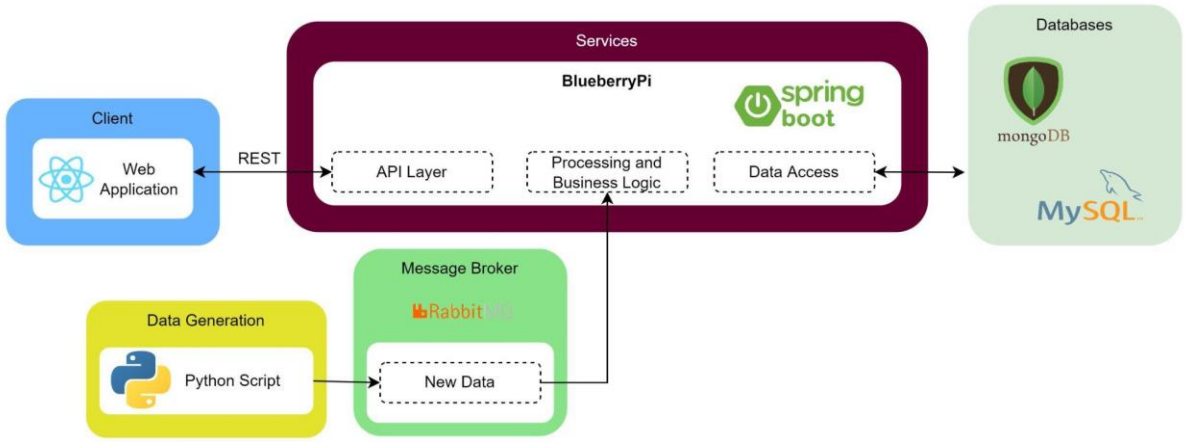
The **Message Broker** will be implemented with *RabbitMQ*, a lightweight service that allows publish-subscribe messages.

The **API,** the **Processing & Business Logic**, and the **Data Access** will all be implemented in *Spring Boot*. The **API** layer will follow a RESTful style which will allow data searches to

present on the visual interface (**Web App**).

The **Processing & Business Logic** layer will be responsible to receive and process the generated data sent from the **Data Generation** layer. This data will be sent through the **Data Access** layer to the respective database. This layer will also be responsible to retrieve the data that the system needs to process in a given time.

The **Long-Term Storage** (Databases) will consist on two different databases. The first one will be an OLTP (using **MySQL**) and an OLAP (using **MongoDB**). The MySQL one will have the data used for main transactions that require relational storage. The MongoDB one will store a history of the collected data and will be accessed mainly for getting statistic data.



Module interactions

→ The **Processing and Business Logic** from *BlueberryPi* service subscribes to a new topic in the **Message Broker**.

→ The **Data Generation** module will generate data and send it to a new (or existing) topic in the Message Broker.

→ The **Message Broker** will send the generated data to the subscriber (in this case the **Processing and Business Logic**)

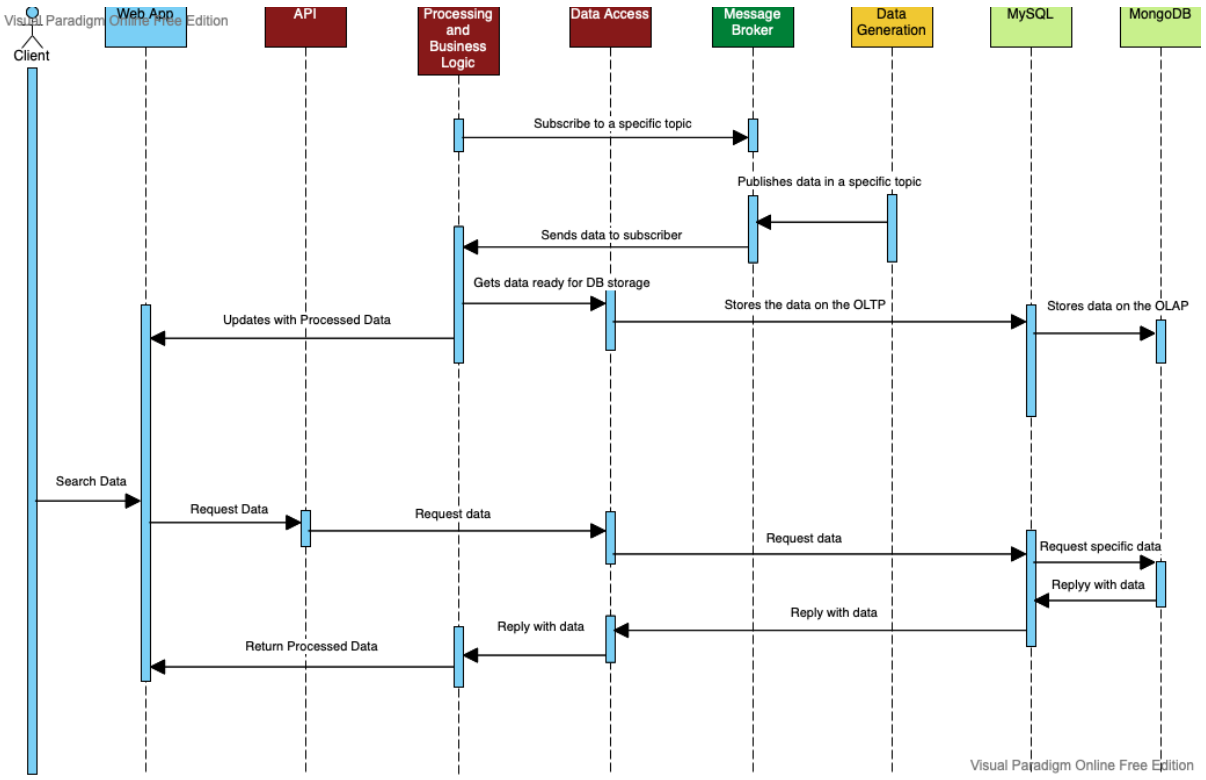
→ The *BlueberryPi* service will consume this data and process it, on the **Processing and Business Logic**, and send this data to the database (**MySQL**) through the **Data Access**. The data will also be stored in **MongoDB**.

→ The **Processing and Business Logic** from *BlueberryPi* service will update the **Web App** with the new processed data.

→ The **client** will want to search data so he makes a request to the **Web App**.

→ The **Web App** will make an **API** request and the **Data Access** will request data from the **MySQL** database. If the data is not present in this database (if it is ancient data for example), data will be retrieved from the **MongoDB** database.

→ The retrieved data will then be processed by the **Processing and Business Logic** and returned to the **Web App**.



4 Information perspective

<which concepts will be managed in this domain? How are they related?>

<use a logical model (UML classes) to explain the concepts of the domain and their attributes>

5 References and resources

Useful resources for building the prototype:

- React: https://reactjs.org/
- canvasJS: https://canvasjs.com/
- React Router: https://reactrouter.com/