Technical Report - **Product specification**

Beach Control-Sistema de gestão e monitorização de praias

Course:	IES - Introdução à Engenharia de Software
Date:	Aveiro, 07/10/2024
Students:	André Alves 113962 Bruno Tavares 113372 Francisco Pinto 113763 Diogo Costa 112714
Project abstract:	Our product focuses on getting data of a public beach, such as: Water temperature, existence of pollutants, wind direction and speed, waves height, tide, temperature, precipitation(mm/h), cloud cover (%), UV index.

Table of contents:

1 Introduction

2 Product concept
- Vision statement
- Personas
- Main scenarios

3 Architecture notebook
- Key requirements and constrains
- Architetural view
- Module interactions

4 Information perspetive

5 References and resources

1 Introduction

The **BeachControl**, part of the Integrated Engineering Systems (IES) course, is designed to provide comprehensive management and monitoring of public beaches. This system combines real-time data collection on environmental conditions, such as water temperature and pollution levels, with analytical tools that help interpret this data.

By doing so, the system distinguishes itself from existing solutions by prioritizing user experience and providing useful, data-driven insights. This empowers beachgoers, lifeguards, and environmental researchers to make more informed and effective decisions regarding safety and well-being at the beach.

This report details the product concept, personas, scenarios, and user requirements that shape the development of the BeachControl ensuring it meets the needs of all users while promoting environmental stewardship and public safety.

2 Product concept

Vision statement

Our system will be used for all the people that go to the beach, from the ones who just want to have a nice fun day at the beach to the ones who must work there. By offering a system that provides real-time data on water quality, cloud coverage, weather conditions, pollutants, and beach safety we ensure that the beachgoers and lifeguards are updated with information about the beach, helping decision-making. The platform allows researchers and environmental agencies to collect, analyze, and act on historical data related to pollution trends, and weather conditions. It even can be used to help the administrators set up emergency protocols with real time warnings, reducing response time in case of emergencies. We gather requirements by looking at existing websites that already provide a similar experience as our product, also we talked to some lifeguards and asked about features that they would like to have that would make their job easier, and one was warning all the beachgoers about an event., making it the key feature that differentiate our system from the others.

Personas and Scenarios

Persona: Beachgoer (Sara, 28)

Description: Sara is a young professional who enjoys spending her weekends at the beach with friends and family. She prioritizes safety and comfort, relying on weather and environmental data to plan her visits, ensuring a relaxing and enjoyable day.

Goals: Sara wants to make sure the beach is safe for swimming and comfortable for outdoor activities. She likes to avoid surprises, such as rain or extreme weather, and plans her beach trips based on current conditions.

Needs: She requires up-to-date information on water temperature, wind direction and speed, air temperature, precipitation levels, cloud cover, and the UV index. This helps her decide the best time to visit and how to prepare, ensuring a safe and pleasant experience at the beach.

Persona: Lifeguard (Carlos, 30)

Description: Carlos is a lifeguard with over 5 years of experience, responsible for the safety of beachgoers. He monitors weather and water conditions to ensure a safe environment, staying alert for any potential risks that could affect swimmers and beach visitors.

Goals: Carlos needs to maintain a safe beach environment by keeping track of conditions that might pose risks, such as strong winds, high waves, or dangerous tides. He aims to respond quickly to emergencies and prevent hazardous situations from arising.

Needs: Carlos requires real-time data on wind direction and speed, wave height, tides, water temperature, and UV index. This helps him assess risks, decide when to raise flags, and determine whether swimming is safe. Alerts for sudden changes in weather or water conditions are crucial for his role in ensuring public safety.

Persona: Environmental Researcher (Ana, 40)

Description: Ana is an environmental researcher working for a coastal conservation organization. She studies environmental patterns, focusing on pollution levels and the overall health of coastal areas. Her work involves analyzing long-term data to understand changes in beach ecosystems. b

Goals: Ana's main goal is to track pollutant levels, water quality, and environmental changes over time to ensure the beaches remain safe for both wildlife and human visitors. She aims to identify trends in pollution and other environmental factors to propose measures for coastal preservation.

Needs: Ana requires detailed, historical data on pollutants, water temperature, precipitation, wind direction and speed, and UV index. She uses this information to monitor environmental health, assess risks to wildlife, and study the impact of human activities on the beach ecosystem. Access to long-term data and trends is essential for her research and reporting.

Persona: Administrator (David, 45)

Description: David is an experienced administrator responsible for overseeing the management of the beach data system. He ensures that all data collected is accurate, up-to-date, and accessible to users, including beachgoers, lifeguards, and researchers. With a background in environmental management, David is passionate about promoting public safety and environmental awareness.

Goals: David aims to maintain the integrity and accuracy of the data presented on the platform. He wants to ensure that all user needs are met and managing user access

Needs: David requires efficient tools to manage data entry, updates, and user feedback

Scenarios:

Planning a Safe Beach Day

Sara is planning a beach day with her friends for the weekend. She checks the website early in the morning to see the water temperature, wind speed, and UV index. The UV index is very high, and the wind speed is stronger than usual, so Sara decides to delay the trip until the afternoon when conditions are more favorable.

Ensuring Safety on the Beach

Carlos, on duty as a lifeguard, notices that the conditions at the beach have changed significantly due to increasing wind speeds and rising tides. To ensure the safety of beachgoers, he needs to update the beach safety flag on the official website. Carlos quickly accesses the management system to change the flag color to red, indicating that swimming is prohibited due to dangerous conditions.

Study on the Contamination of Beaches in Lisbon

Description: Ana is conducting a study on the contamination of beaches in Lisbon, specifically focusing on the effects of a recent discharge from a factory into the coastal waters. She uses the app to access data on the presence of certain bacteria in various beaches in the region. Ana analyzes the water samples collected and compares contamination levels across different beaches.

Product requirements (User stories)

Epic: Lifeguard account

User Story 1: Lifeguard Login
Priority: High

As a lifeguard,

I want to log in to the system

so that I can access essential information and manage safety protocols on the beach.

Acceptance Criteria:

- The lifeguard should be able to access the login page and enter their registered email and password.
- The system should validate the credentials and allow access to the Lifeguard dashboard if they are correct.
- If the login fails, the system should display an appropriate error message.

Epic: User Interaction

User Story 2: Report data errors

Priority: Low

As a user,

I should be able to report some inconsistencies in the data provided

so that the administrator can check if everything is accurate.

Acceptance Criteria:

- A button labeled "Report Data Error" should be visible
- Clicking the "Report Data Error" button opens a form allowing the user to describe the error.
- Upon successful submission, the user receives a confirmation message.
- The system should send the report error to the administrator

Epic: User Interaction

User story 3: Update the flag color

Priority: High

As a lifeguard,

I need to be able to update the color of the flag as the day goes by and conditions change,

so that the users can have real-time updated information.

Acceptance criteria:

- The system allows real-time updates of the flag;
- The update is confirmed with a success message;
- Updates are stored with timestamps;

Epic: User Interaction

User story 4: Create beach´s warnings

Priority: Medium

As a lifeguard,

I need to be able to create warnings to reflect possible problems at the beach,

so that I can ensure that everyone who is and intends to go to the beach is safe.

Acceptance criteria:

- The system allows real-time updates of the warnings;
- The system allows to warn the users about dangers and risks on the beach;
- The update is confirmed with a success message;

Epic: Beach's information monitoring

User Story 5: Real-Time beach's information access

Priority: High

As a beachgoer,

I want to access real-time information of the conditions

so that I can see if the water temperature and cloud cover is favorable for an enjoyable day on the beach.

Acceptance Criteria:

- The user should be able to select a beach;
- The information about the selected beach should be displayed, reflecting the real-time changes as they occur;

Epic: Beach's data

User Story 6: Download previous beach's data

Priority: Medium

As an environmental researcher,

I want to be able to download previous data of a beach

so that I can cross data for a research paper that I am working on.

Acceptance criteria:

- The user should be able to select a beach,
- The system should allow the download of a beach data
- The system should show a success message

Epic: Administrative control

User story 7: Lifeguard management

Priority: Medium

As an administrator of the system,

I want to see the logs of the warnings made by the lifeguards

so that I can ensure that all warnings are valid.

Acceptance criteria:

- The system should have an administration panel to see the information.
- The system should show the warnings made by lifeguards
- The administration must be able to delete warnings.

Epic: Lifeguard account

User Story 8: Lifeguard account creation.

Priority: Medium

As an administrator of the system,

I want to create accounts for lifeguards

so that I can ensure only qualified personnel have access to the system.

Acceptance criteria:

- The administrator should be able to access the user management section of the admin dashboard.
- The administrator must fill out a form with the lifeguard's information (name, contact, beach location, etc.).
- The administrator should be able to view the status of created accounts to ensure they are activated.

Epic: Administrative control

User Story 9: Add beach to the system.
Priority: Medium

As an Administrator,

I want to create a new Beach,

so that the system can attract more users.

Acceptance criteria:

- The administrator should be able to access the Beach management section of the admin dashboard.
- The administrator must fill out a form with the Beach´s information.
- The administrator should be able to delete a Beach.
- The administrator should be able to update the information about the Beach.

Epic: Administrative control

User Story 10: Add Sensor to Beach
Priority: Medium

As an Administrator,

I want to add a sensor to a Beach,

so that the system can show more information about the beach.

Acceptance criteria:

- The administrator should be able to access the Sensor management section of the admin dashboard.
- The administrator must fill out a form with the Sensor´s information.

Epic: Administrative control

User Story 11: manage Beach Sensor
Priority: Medium

As an Administrator,

I want to manage the sensors of a Beach,

so that the system can create reliable information.

Acceptance criteria:

- The administrator should be able to access the Sensor management section of the admin dashboard;
- The administrator must be able to see a list of beach sensors;
- The administration should be able to see the status of a sensor;
- The administration can delete of “fix” the sensor;

Epic: Administrative control

User Story 12: Data error Management
Priority: High

As an Administrator,

I need to be able to see data report errors made by users,

so that I can ensure that the data that is shown is correct.

Acceptance criteria:

- The administrator should be able to access the Data reports management section of the admin dashboard;
- The administrator must be able to see a list of data reports;
- The administration can see details of reports to get more information.

Epic: User Interaction

User Story 13: Beachgoer favorites
Priority: High

As a Beachgoer,

I need to be able add a beach to my favorites,

so that I can access the information of the beach i want more easily.

Acceptance criteria:

- The Beachgoer must be able to add a beach to their favorites;
- The Beachgoer muss be able to access a list of their favorites beach´s;
- The administration can see details of reports to get more information.

3 Architecture notebook

Key requirements and constrains

Our system will have a large volume of data, as we have multiple sensors on multiple

beaches generating all sorts of data, so our DB should handle high write load. Furthermore, it should support backups, so data is not lost in case of failure.

It should support a system to handle the constant flow of sensor data from the backend to the front end. All of this should have low latency, so data changes are reflected immediately on the front end. It too should be configured to handle failure without data loss.

As we support login and registration the user information should be handled safely. Also, our API should be secured and protected, using authentication and rate-limiting protocols, minding that the admin and lifeguard must have privileges and API endpoint exclusive to them so they can handle specific events.

For quality of life, it should support moving historical data to a different storage to maintain performance as our sensor should collect data 24/7, 365 days a year.

The front-end interface should remain responsive even when processing large datasets. The application must be accessible from the web browser.

Architetural view

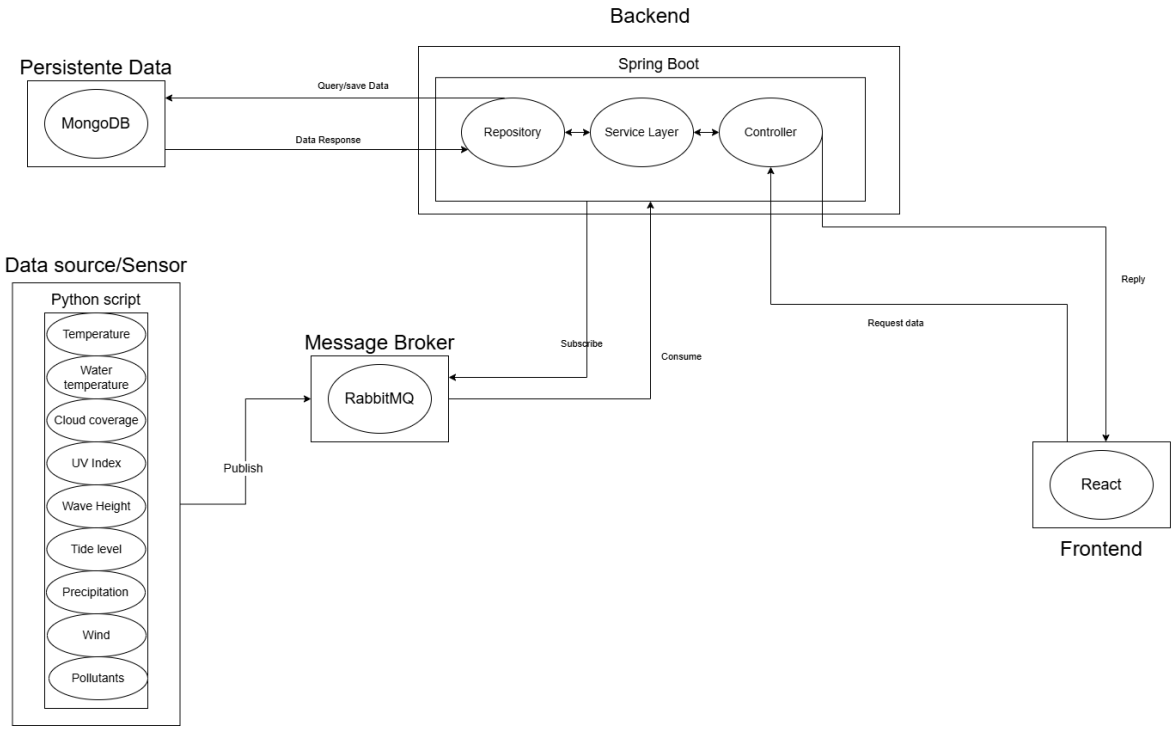
The data generation will be done by python as it offers libraries for mathematical operations, helping create realistic data for the temperature, cloud coverage etc. It can then be sent to our message broker.

The streaming data will be handled by RabbitMQ, it supports asynchronous messaging, allowing decoupling of data generation and data processing. And it can handle high rate of sensor data messages. It also provides reliability, ensuring that no data is lost in transit.

For backend we will use spring boot as it allows us to define REST controllers making it easy to handle HTTP requests, integrating spring security to handle user authentication. It can also process incoming sensor data using spring AMQP and interact with both databases though Spring Data MongoDB and Spring Data JPA.

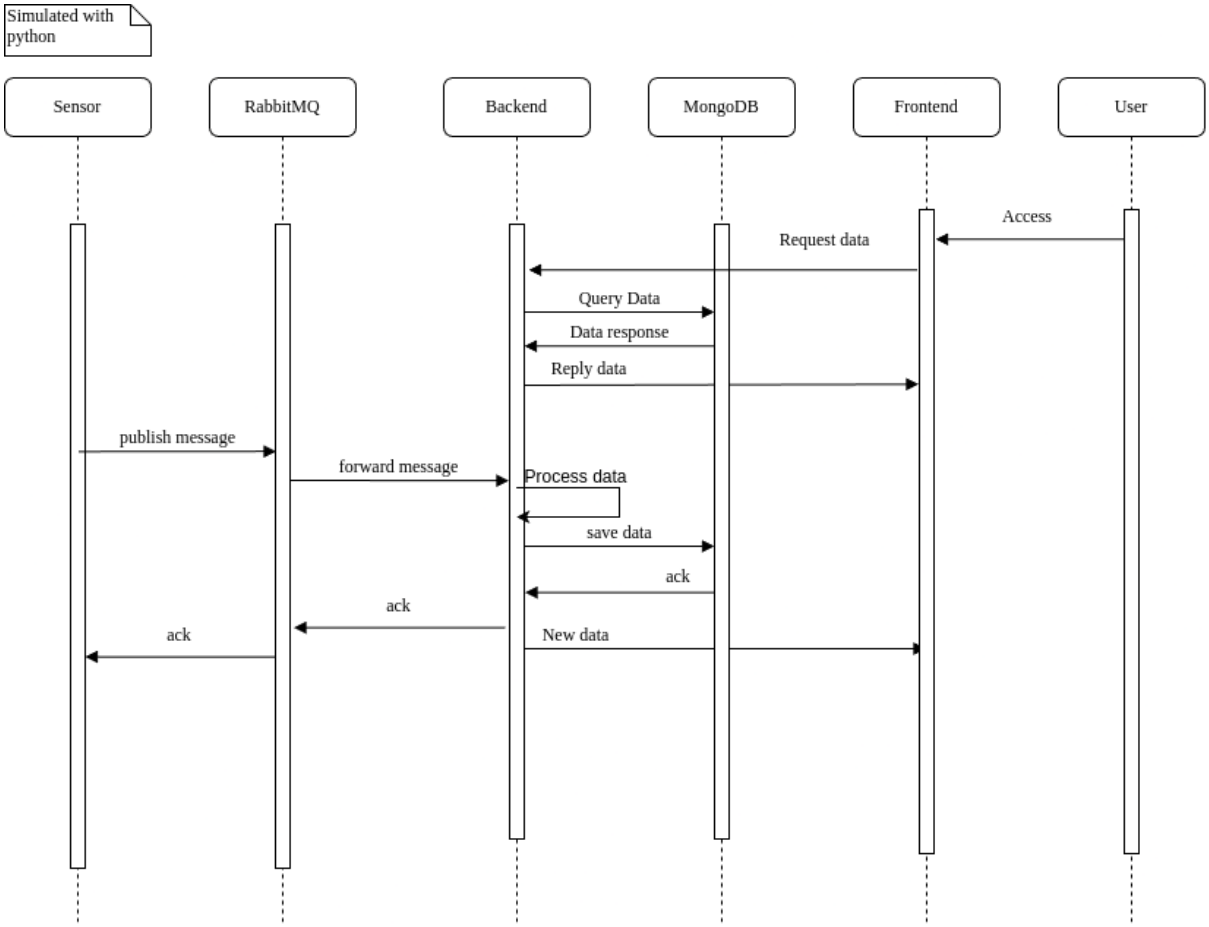
The data will be stored in a MongoDB database as it is a document-based it allows changes in the structure as new sensors to collect new data can be created. It also handles high volume write operations efficiently, as well as an excellent querying and aggregation feature, on top of that sensitive user information will be hashed using Bcrypt.

ReactJS will be used for frontend as it allows to create reusable UI elements for displaying graphs with Chart.js. With http request to query the API and get the necessary information.

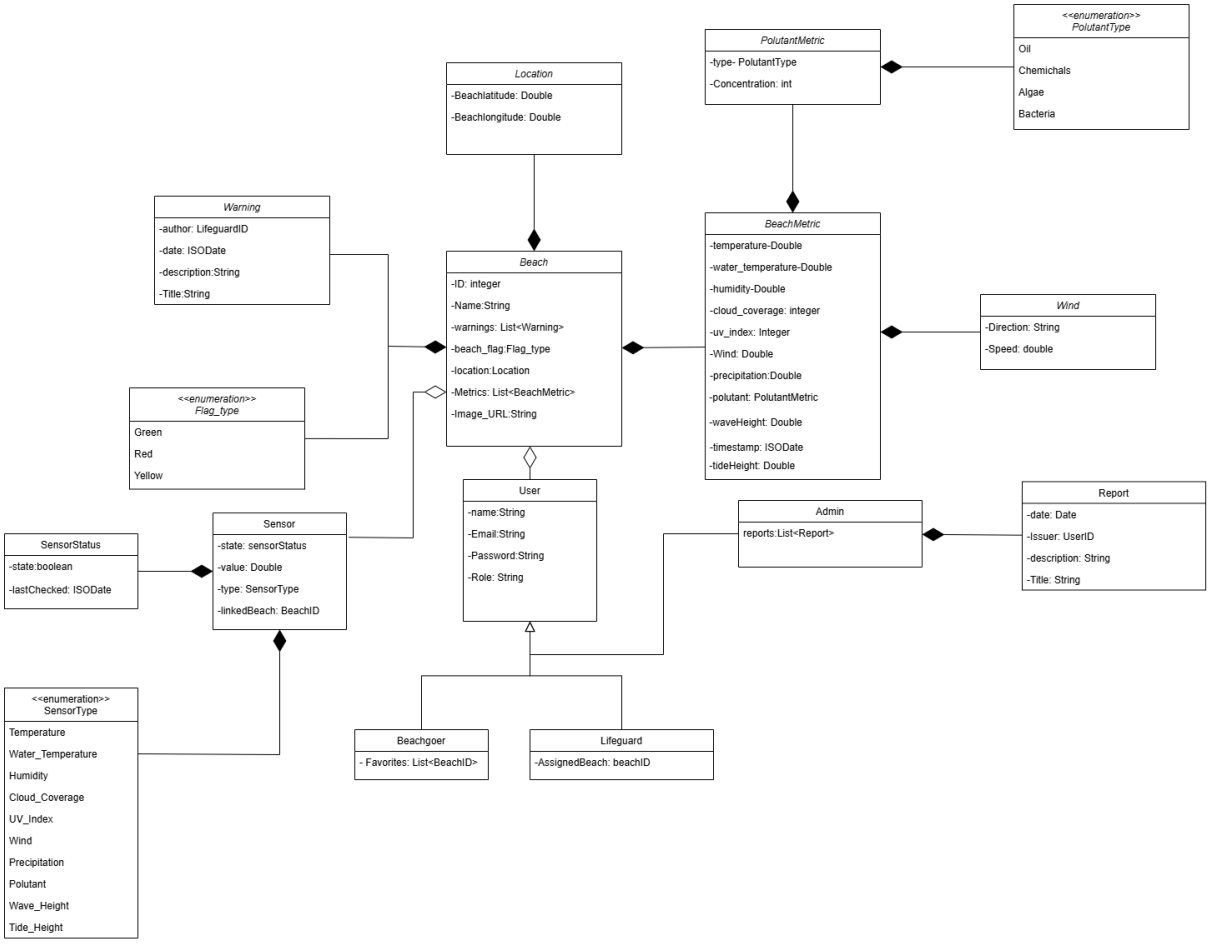


Module interactions

The front end request data to the back end, it is then queried from our MongoDB. After the back end gets the data, it processes the data as necessary and sends it to the front end updating its U display. When new data is generated, it is published to the RabbitMQ that notifies the backend, processing and saving to the MongoDB and sending the new data to the front end.



4 Information perspective



Beach- This object represents the main purpose of the project because it contains the various datapoints that will be created using the data generator. It will have the information that the various users of Beach Control will need to complete their tasks.

The beach entity has an ID, a name, a Beach_Metric object that contains information of beach conditions, a list of warning created by the lifeguards, a Beach_flag object that represents the overall quality of beach and the location of the beach.

BeachMetric- This object has the data that will be generated in relation to beach conditions. The BeachMetric entity contains the temperature, the water temperature, the cloud coverage, the humidity, the uv_index, the wind speed and the precipitation at any given time.

PolutantMetric-This object is like the BeachMetric object, but it only has the information referent to the presence of types of pollutants and their concentrations in the sea.

PolutantMetrictype-Types of pollutants that the sensor measure**.**

Warning- This object referents to the warnings that a given Lifeguard can give about a specific problem that is occurring at the beach that cannot be integrated into the data points like presence of “caravelas”. This entity contains an author that was responsible for the creation of the warning, the beginning date of the warning and a possible ending date if it is necessary, a description of the warning and an id to keep track of them.

Flag_type-This object contains the possible types of flags the beach has like green if the conditions are good, red if they are bad and some others.

User-The user are the various persons that will interact with Beach Control, they can be one of three types: Beachgoer,Lifeguard,Admin itch one will interact with the system differently. The way to differentiate between the Beachgoer and a Lifeguard is the email that is used to login to the website, the Beachgoer has a standard email address while the Lifeguard has an email address that was created by the website admin.

Beachgoer- This user wants to get information about one or more beaches, and they can have a favorites list that contains the beaches he is most interested in.

Lifeguard- This user wants to access our website to update the status of the flag type of the beach and create warnings about possible issues that are occurring.

Admin- He manages the website, creating accounts for lifeguards and responding to possible issues that are reported by users. These issues can be related to data errors Lifeguards warnings or sensor malfunctions.

Reports- This object represents issues users have with the website. This entity contains a Report_type to specify the problem that the user has, a date to know when this report was issued, an issuer the user that created his report and a description to explain the problem.

Sensor- A sensor that exports its data to the system.

SensorStatus- To know if the sensor is working or not and the date that it was last checked

SensorType- The various types of sensors that collect the beach information.

5 References and resources

<document the key components (e.g.: libraries, web services) or key references (e.g.: blog post) used that were really helpful and certainly would help other students pursuing a similar work>

https://aws.amazon.com/pt/event-driven-architecture/

https://spring.io/projects/spring-amqp#overview