Technical Report - **Product specification**

eDuca

Course:	IES - Introdução à Engenharia de Software
Date:	Aveiro, 26/10/2023
Students:	107647:Diogo Silva 107660:Miguel Cruzeiro 107476:Rafael Vilaça 104807:Tiago Quintas
Project abstract:	The Multi-School Student Management Platform is a system designed to enhance the management of multiple schools' student-related activities. It provides a centralized hub for students, teachers, and school administrators to efficiently manage and access student information, academic records, communication, and more. The key concept revolves around improving the educational experience for all stakeholders while ensuring data security, scalability, and user-friendliness.

Table of contents:

1 Introduction

This report presents the concept and vision for a web-based solution designed to revolutionize the way schools handle academic data.

The objective of this project is to create a platform that facilitates the management of grades, notes, exam dates, and related information for both teachers and students in a school environment.

Additionally, it will promote smooth communication between teachers and students.

2 Product concept

Vision statement

This system will be used for:

- Allowing teachers to efficiently input and publish grades, notes, and exam dates.
- Enabling students to access their academic information and track their progress.
- Simplifying the communication process between teachers and students regarding academic matters.
- Unify various schools within a single platform, promoting cooperation and improving the accessibility of academic data across diverse educational institutions.

Personas and Scenarios

Persona 1:

Name: Sara

Age:17

Profession:Student

Sara is an interested student and always wants to be up to date with messages that teachers can send.

She likes to plan her studies and for that she needs some way of knowing when there are tests or meetings.

Worried about her average to enter university, she likes to keep her grades from all her subjects.

Persona 2:

Name:João

Age:49

Profession:Math Teacher

The teacher wants to publish the grades for his subject.

He also wants to warn students about the content that will be covered in next week's test.

As he is very busy, you need some way to know which days there are meetings and which days it is possible to book a test.

Persona 3:

Name:Carlos

Age:35

Profession:Platform Administrator

As the administrator of the platform he can add new schools,delete schools from the system and update them.

Persona 4:

Name:Ana

Age:47

Profession.School Administrator

As the school administrator, when new students arrive, she adds them to the system.

To compare with other schools, it has access to the grades and statistics of students from the respective school.

Product requirements (User stories)

User Registration and Profiles:

This epic focuses on user registration and profile management.

User Stories:

- As a student/teacher, I want to edit my profile.
- As a school administrator, I want to manage and verify user profiles within my school.

School Management:

This epic revolves around school management, including adding, removing, and updating schools in the platform.

User Stories:

- As a platform administrator, I want to add a new school to the platform with its details.
- As a school administrator, I want to update my school's information.
- As a platform administrator, I want to remove a school from the platform.

Student Enrollment and Records:

This epic deals with student enrollment, record management, and academic progress tracking.

User Stories:

- As a school administrator, I want to enroll students in my school and input their basic information.
- As a teacher, I want to update student records with grades and attendance information.

Attendance and Class Scheduling:

This epic covers class scheduling, marking attendance, and managing class sessions.

User Stories:

- As a school administrator, I want to create and manage class schedules.
- As a teacher, I want to mark student attendance for each class.
- As a student, I want to view my class schedule and attendance history.

Communication and Notifications:

This epic focuses on communication between schools, teachers, students, and parents.

User Stories:

- As a teacher, I want to send notifications and updates to students.
- As a student, I want to receive notifications about my academic performance and school events.
- As a school administrator, I want to broadcast important announcements to my school's community.

Report Generation and Analytics:

This epic involves generating reports and analytics to evaluate student and school performance.

User Stories:

- As a school administrator, I want access to analytics and reports on student performance and school metrics.
- As a teacher, I want to generate individual and class-level performance reports.
- As a student, I want to view my academic performance reports.

3 Architecture notebook

Key requirements and constrains

The system should be able to generate and consume data automatically, simulating what would happen in a functioning school.

The system cannot allow third-party access to confidential data, requiring an authentication system.

It should be ensured that the data generated by the simulator is not mixed within the message broker with the received data.

The users should be able to have access to the WebApp at any moment.

Due to the interactions users can perform, the Python script needs to consume the updated new data to ensure the congruence of the generated data.

For example, when the student leaves a class it doesn't make sense to keep generating data to the student.

The system administrator should be the only one able to manage all the school administrators (create,update and delete) and shouldnt be the only one able to update the schools but be the only one able to delete,create them.

The school administrator should be the only one able to manage all accounts (create,update and delete (teacher and student accounts)).

The system administrator parameters should be name,password.

The school administrator parameters should be name,password.

The student account parameters should be name, email, phone number, birthday, address, school, city, Attendance Record, Grades/Classes, Attendance Record and password.

The teacher account parameters should be name, email, phone number, birthday, address, school, city, Subject(s) Taught, Teaching Schedule and password.

The student account has a read only general use (the students can check their information and can only update some of their personal information (contact number,email,password and address are allowed, name and b-date etc aren't).

The teacher account has the same limitations as the student (in personal data update and information check) but has some write privileges.

The teacher should be able to have access to where each class is, mark student attendance and give student grades for each class.

The school admin account must have access to the information of every student and teacher.

Architectural view

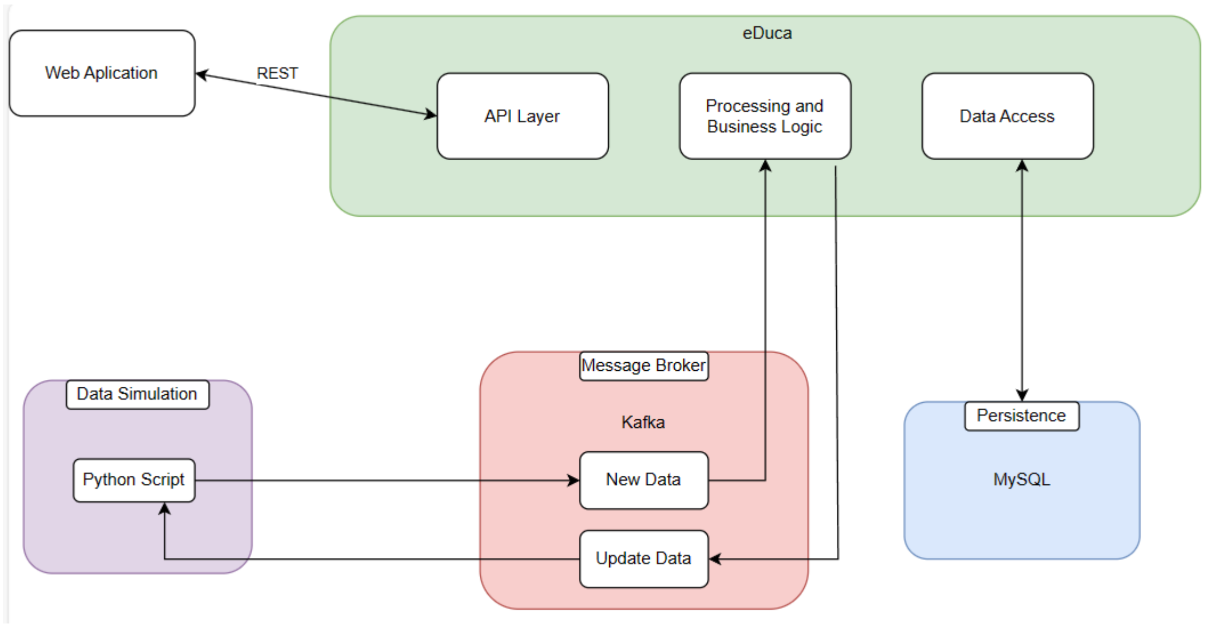
Web Application: To build the web application, we choose HTML and JavaScript due to their advantages in developing dynamic applications and because of our knowledge and experience with these technologies.

eDuca: We will build a service based on Spring Boot. This service will consist of a REST API layer, enabling the retrieval of various system data to be presented in the web application. Additionally, there is the data model layer, facilitating the mapping of data stored in the database to Java objects for processing in the business logic layer.

Simulation of Data: Some of the data shown by the web application will be generated by a Python script, specifically the grades for students. The teacher will send a list of students for whom they want grades through a message broker.

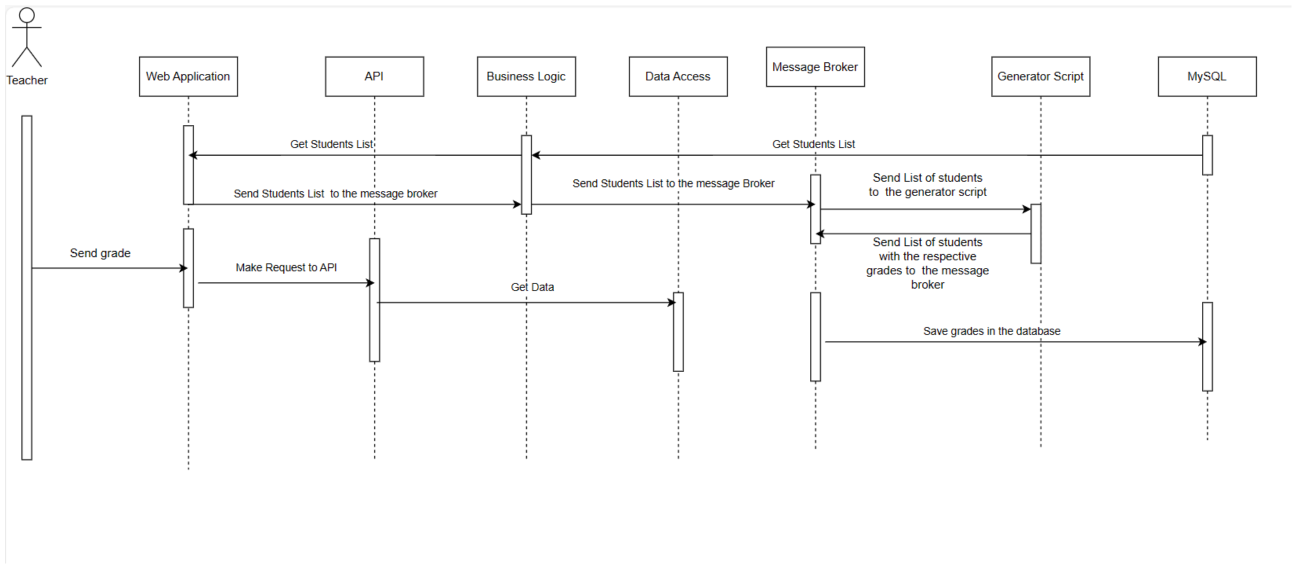
Message Broker: We chose Kafka as the message broker because it is a technology with which we are familiar. The teacher is expected to send a list of students for whom they want grades through the message broker. The message broker will then forward this information to the Python script responsible for generating the grades. Once all grades are generated, the Python script will send pairs of (student, grade) back to the teacher through the message broker.

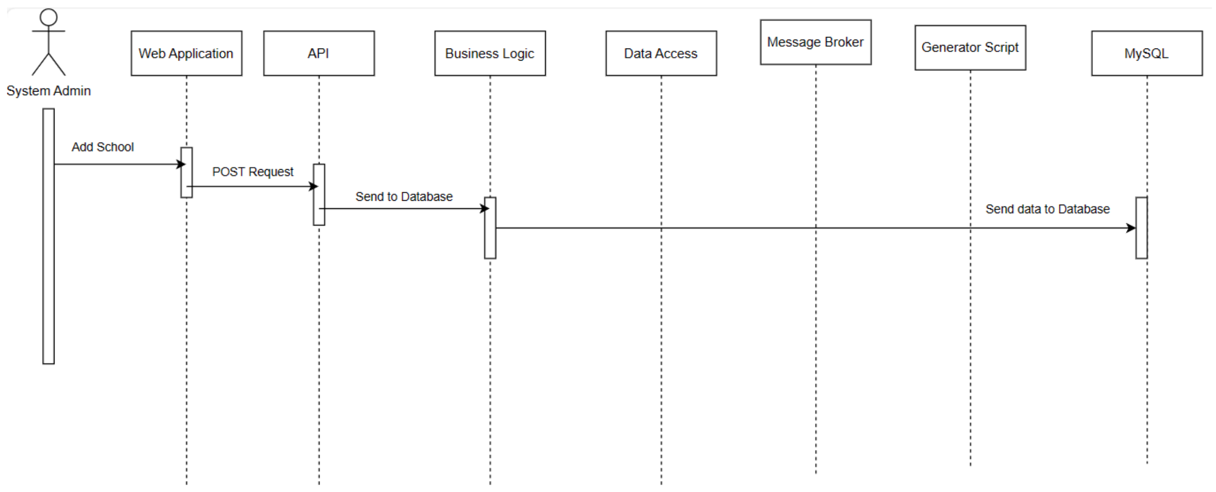
Databases: We will use MySQL to store the most important data of the system. Additionally, we will utilize MongoDB to store information such as notifications that students receive.



Module interactions

- During an evaluation, a list of students stored in the database is sent by the Spring Boot application to a Kafka topic.
- The generator script consumes the data from the topic, generates grades for these students, and sends the updated data back.
- The client, through the web application, views all available information by making a request to the REST API.
- Through the REST API, the requested data is sent, with access to the database and passing through the business logic.
- The system administrator, through the web application, adds a new school, and this information is sent to the API via a POST request.
- Upon receiving the request, the information is updated in the database.


Teacher
Web Application
API
Business Logic
Data Access
Message Broker
Generator Script
MySQL
Get Students List
Get Students List
Send Students List to the message broker
Send Students List to the message Broker
Send List of students to the generator script
Send List of students with the respective grades to the message broker
Send grade
Make Request to API
Get Data
Save grades in the database



4 Information perspetive

<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

<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>