**Handover Document**

***It’s a Match – Group 1***



**Author(s):**

Hristo Hristov

Liam Vaessen

Okley Julio Andreotti

Shanessa Kostaman

**Teacher(s):**

Bartosz Paszkowski

Jeffry Cornelissen

Ruben Steins

**Fontys University of Applied Sciences**

**2023**

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Details | Author |
| 0000 | 14-06-2023 | Setup Document | Shanessa |

Table of contents

[Document History 2](#_Toc137672377)

[Abbreviations and Definitions 2](#_Toc137672378)

[Introduction 4](#_Toc137672379)

[Structure Breakdown 5](#_Toc137672380)

[Project Deliverables 7](#_Toc137672381)

[Team Members 7](#_Toc137672382)

# Introduction

This semester we are working on the challenge **It’s a Match**. This project is actually a huge project that divided into 3 different group that’s responsible on the work division. 1st group is open journalism this group is responsible for the containers, they’re the one that’s going to receive the message from the user and then they’re going to pass the message or request to our group It’s a Match and after we’re done with processing the request we’re going to send it back to the open journalism to give it back to the user.

To be more specific our part is to help the journalists make use of the system without a lot of ‘hassle’ (installation/configuration). This communication from journalist to AI should be “human”-like interaction. This means translating commands, interpret working order and aggregating results (may even be asynchronous). AI-tools themselves should not be serving any UI components. One might see it as a AI workforce. This project is a success if it is transferable, maintainable and extensible after the students involved will have left.

# Structure Breakdown

To get to know more details about how things happen to our project we’re going to explain here:

A picture containing text, screenshot, diagram, font

Description automatically generatedHere in this project, we have 3 services (1 services to process the message and the rest 2 is a service for the AI Adaptor) and 2 message bus (external message bus, RabbitMQ and internal message bus, Kafka).

The first service will be a **MessageBus Service,** this service play role in handling query requests from an external message bus that is shared with Open Journalism. When a query arrives from the external message bus, our task is to scan the incoming query and determine to which topic it belongs so we know to which Kafka topic it belongs to. To accomplish this, we employ specific keywords that assist in categorizing the queries effectively.

Within our internal message bus, we create multiple Kafka topics aligned with the identified query topics. This step ensures that each topic has its dedicated space to receive and process relevant queries. To accomplish this, we employ the keyword associated with a specific topic to route the query to the corresponding Kafka topic.

Beside that, this service also play role in the seamless handling of responses generated by the AI through our internal message bus. As soon as a response is received from the internal message bus, this service takes immediate action and passes it along to the external message bus, which is shared with Open Journalism.

By swiftly relaying the AI's response, we ensure that the valuable insights and information generated by the AI are efficiently communicated to the Open Journalism platform. This enables journalists and users to access the latest and most relevant information in a timely manner.

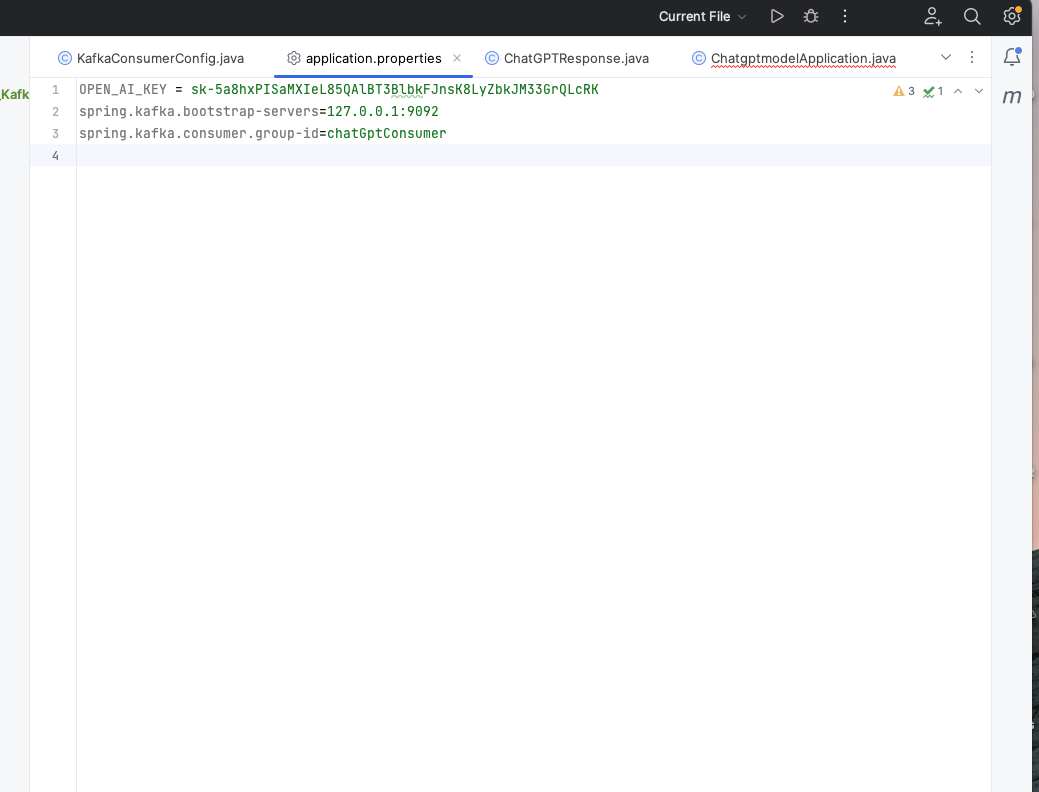
In essence, this service acts as a reliable conduit, bridging the gap between the AI's responses and the wider Open Journalism ecosystem, ensuring a smooth flow of information and facilitating effective collaboration.

**Note: we use up stash for Kafka, to have the access please let Shanessa know so she can add you into the account.**

**For rabbit MQ its hosted-on azure, u can go to this link: 20.126.244.61:15672 username and password is guest.**

The third service will be a ChatGPT Service

This. Service provides CHatGPtT’s message bot service through a Spring Java application. The service takes a request from the user and sends an API call to the chatGPT service and gets a response. As soon as the response is received an event is fired in Kafka which sends the response to the message bus to be consumed by the various subscribers.

At the moment this service works fine, but the API key in application.properties needs to be updated with a licenced one since the API key found there has run out of license. I have attached a screenshot for easier unserstanding.

**Vision AI Service**

We also implemented a visionAI service. This service makes use of Google VisionAI to detect information from images. The way this service is implemented is when an image is uploaded and queried in the service, the controller get web information on the picture mainly:

* **Web Entities:** Find Matching pictures
* **Get Labels**: Guess the label or description of the pictures
* **Get PAges:** Gets pages with images that match the uploaded image

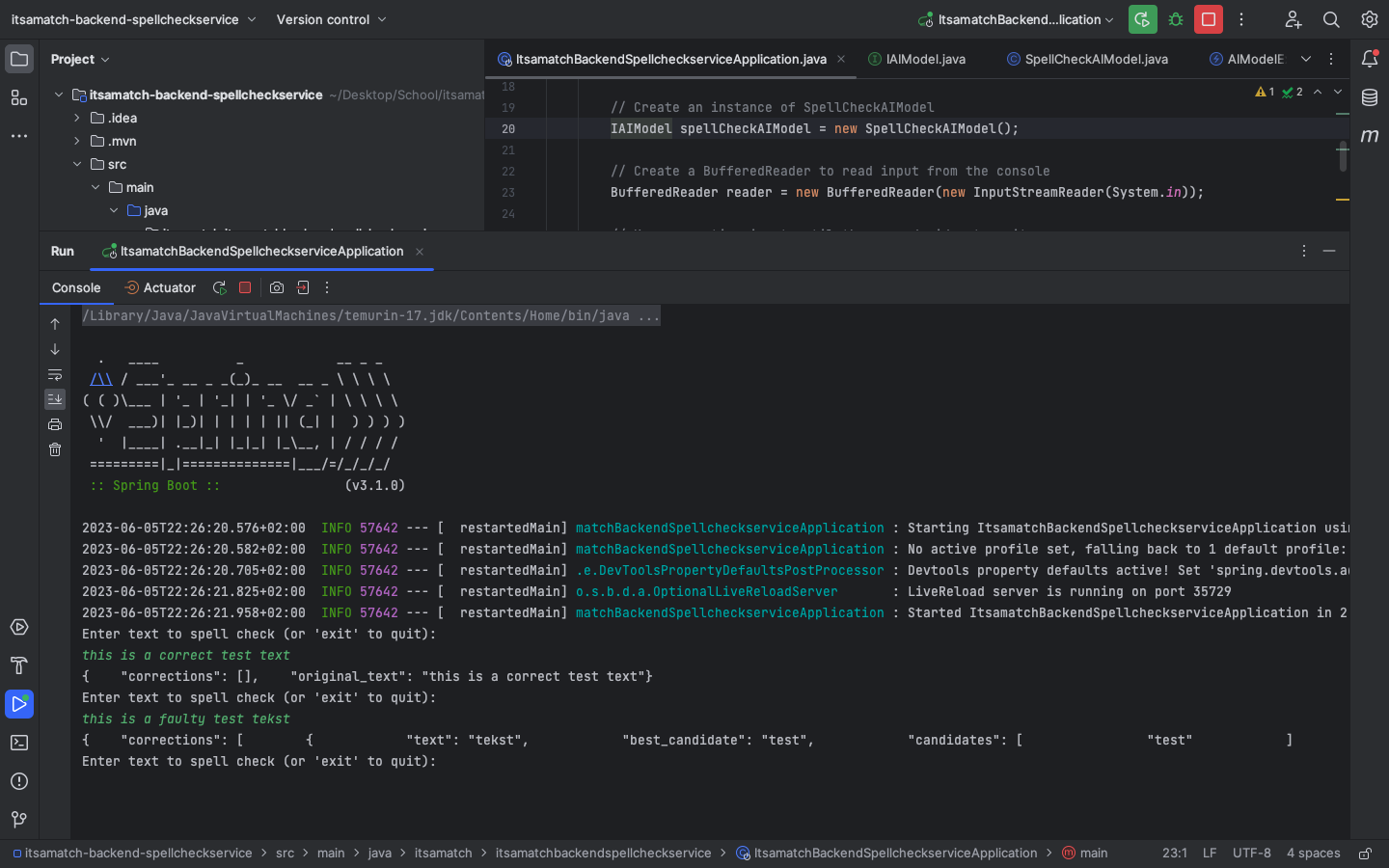
After these values are returned they are published on a KAfka messaging bus to the various subscribers so the response data can be returned to the appropriate service. This service also needs to be updated with new API tokens and account details to be able to make couls to the visionAI service. These tokens and details should be updated in the VisionApplication.java.

**Spell Checker Service**

This service is a spell checker service. It’s a simple model that accepts texts as an output, and returns an object with the following:

* The original text
* Spelling mistakes
* Recommended changes

After these values are returned they are published on a Kafka messaging bus to the various subscribers so the response data can be returned to the appropriate service. This service also needs to be updated with new API tokens and account details to be able to make calls to the visionAI service. These tokens and details should be updated in the SpellCheckAIModel.java file. See image below for results.



After we’re done talking about all the services, now we’re going to talk about the message buses. The external message bus is Rabbit MQ and it’s split between 2 queues, the request queue and the response queue. The internal message bus is Kafka and it’s split into 2 Kafka topics, General Kafka topic is for general queries and Spell-checking Kafka topic is specifically for the topic that’s related with the spell checking.

# Project Deliverables

This is what you will get from our group

|  |
| --- |
| 1. Code 2. Project Plan 3. Research Document 4. Requirement Document 5. Test Plan 6. SAD Document 7. Diagram 8. Mock-Up 9. Prototype (Without code) |

# Team Members

If you have any questions regarding the project itself don’t hesitate to ask and contact us bellow is the contact information of our group member.

*Table 05: Team Members*

|  |  |  |
| --- | --- | --- |
| Name + Phone + e-mail | Abbr. | Role/tasks |
| Name: Hristo Hristov  Phone: +31613699994 e-Mail: [h.hristov-ab@student.fontys.nl](mailto:h.hristov-ab@student.fontys.nl) | Hristo | Team Member |
| Name: Liam Vaessen  Phone: +316-20458083 e-Mail: [liam.vaessen@student.fontys.nl](mailto:liam.vaessen@student.fontys.nl) | Liam | Team Member |
| Name: Okley Julio Andreotti  Phone: +316- e-Mail: [j.okley@student.fontys.nl](mailto:j.okley@student.fontys.nl) | Andy | Team Member |
| Name: Shanessa Kostaman  Phone: +31657854265 e-Mail: [genoveva.shanessakostaman@student.fontys.nl](mailto:genoveva.shanessakostaman@student.fontys.nl) | Shanessa | Team Member |