University of Pisa

SCUOLA DI INGEGNERIA

Corso di Laurea in Artificial Intelligence and Data Engineering

Data mining project documentation

Candidati

Giacomo Mantovani

Stefano Poleggi

Relatori

Prof. Francesco Marcelloni

Prof. Pietro Ducange

Anno Accademico 2019–2020

Contents

1 Introduction **1**

2 Analysis and workflow **3**
2.1 Requirements 3
2.1.1 Functional requirement 3
2.1.2 Non-functional requirements 3
2.2 Use Cases 4
2.2.1 Use Cases Description 4
2.3 Analysis of entities 5

3 Design **6**
3.1 Database Choice 6
3.2 Software architecture 6

1 Introduction

The <**Name of the app**> application offers a real-time sentiment analysis service. When the application starts, the user can perform a sentiment analysis by searching for a topic in the search bar and the application will start extracting tweets analyze them. The result is printed into a graph which shows the number of tweet analyzed so far and the semtiment rating. The graph keeps updating until the user clicks the stop button or when the application is closed. Moreover the application provides some charts to check the results obtained.

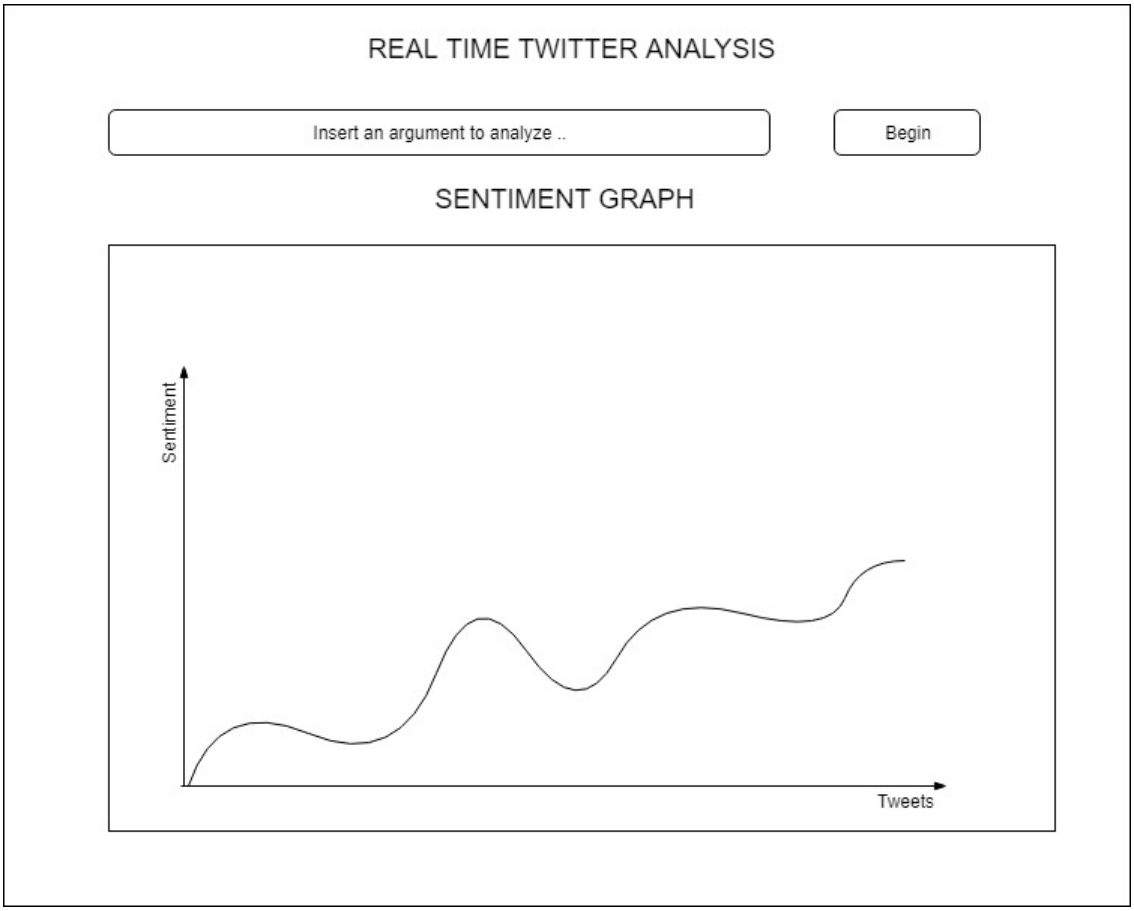

Figure 1: Home Page Mockup

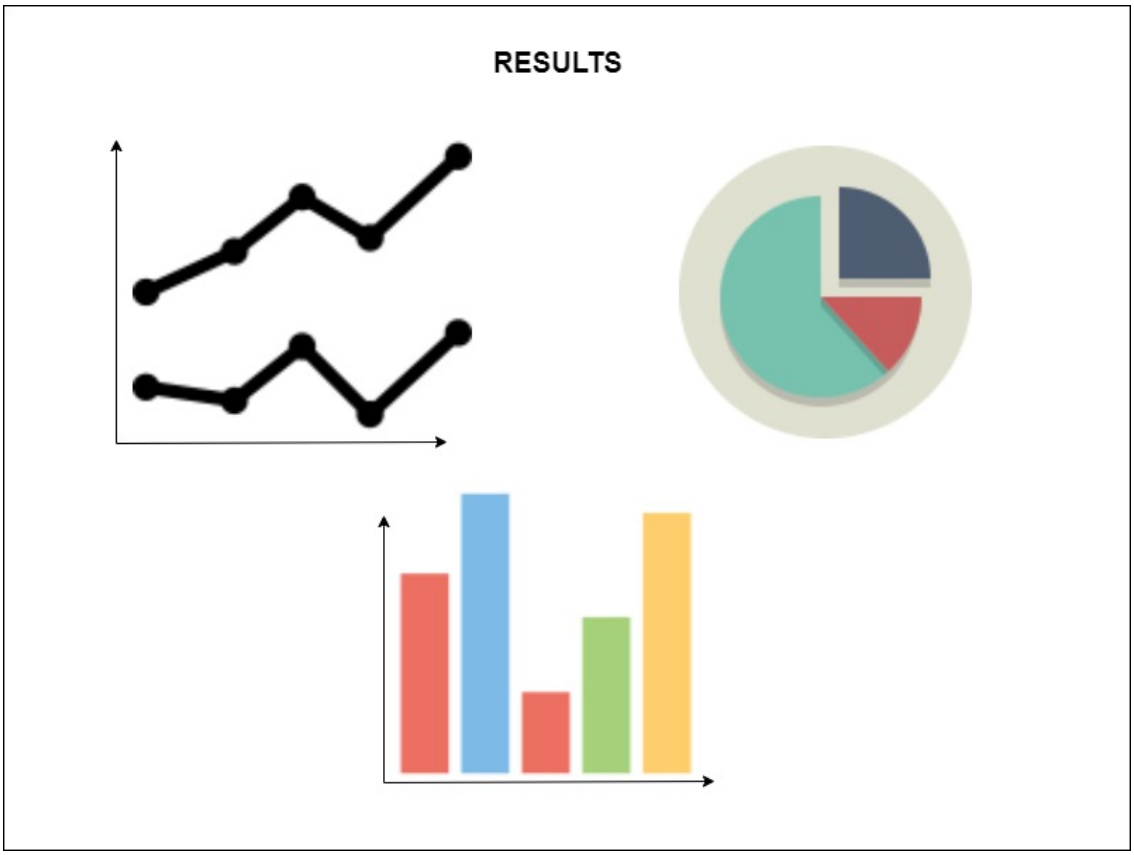

Figure 2: Results Page Mockup

2 Analysis and workflow

2.1 Requirements

2.1.1 Functional requirement

The system has to allow the user to carry out basic functions such as:

- To select a topic.
- To retrive the sentiment analysis of the selected topic.

The system has to perform the following operations:

- Real-time fetching of tweets of a specified topic.
- Perform a sentiment analysis of the tweets and obtain the sentiment (positive, negative or neutral).
- When a negative tweet trend is recognized send a notification.
- Perform a visual analysis by filling charts

2.1.2 Non-functional requirements

- Usability, ease of use and intuitiveness of the application by the user.
- The system should provide access to the database with a few seconds of latency.

2.2 Use Cases

Actors

- User: this actor represents a user of the application

2.2.1 Use Cases Description

Event	UseCase	Actor(s)	Description
Log in, Log out	Login, Logout	Admin, User	The user logs in/out the application.
Display all the Films	Browse, Find, Display Films	User, Admin	The user chooses that he wants to view the list of Films. The system browses the data on the db and returns them on the interface.
View Statistics	View Top Rated Films, View Top Productions, View Top Film-maker Countries, View Most Active Users	Admin	The Admin clicks on the button to view the statistics. The system browses on the db the informations used in the calculation and display the result.
Add a film	Add Film	Admin	The admin submits the Film information. The system updates the db and the interface.
Update a film	Update Film	Admin	The admin selects the film and commits the new informations. The system updates the db and the interface.
Delete a film	Delete Film	Admin	The admin selects the film and submits the delete. The system updates the db and the interface.
View the film informations	Select Film, Display Film Info	User, Admin	The user selects the film. The system shows the film informations on the interface.
Vote a film	Vote Film	User, Admin	The user submits the vote on a selected film. The system updates the db and the interface.

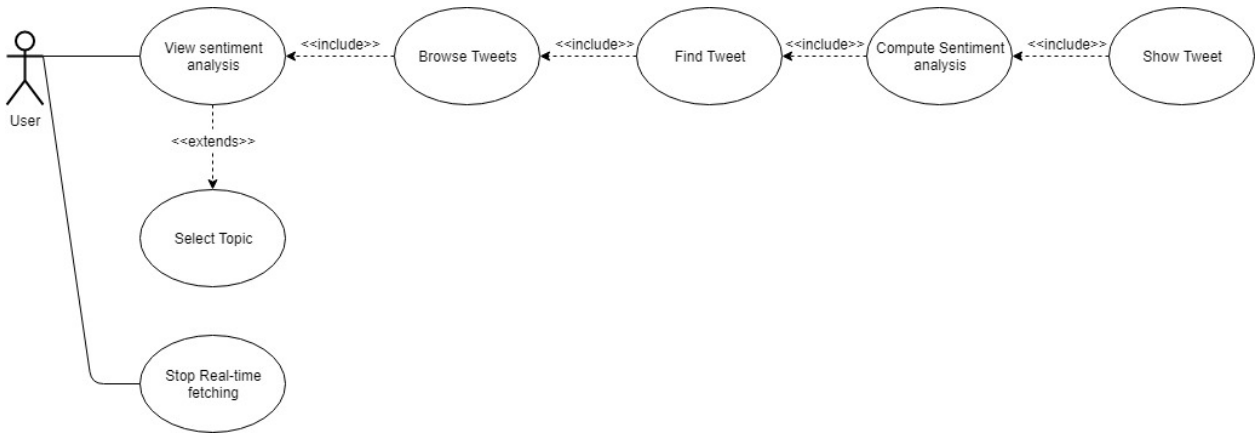

Figure 3: Use cases diagram

2.3 Analysis of entities

This diagram represents the main entities of the application and the relations between them.

Figure 4: UML analysis diagram

3 Design

3.1 Database Choice

After the analysis phase, which has been carried out so far, we start with the design of the <**Name of the app**> application. We decide to use MongoDB as data support. Its document-based structure is very useful for the large amount of data that we need to maintain and access, as well as its high scalability, qualities that we do not find in a relational database.

3.2 Software architecture

The application is designed over 3 different layers, see figure 5:

- Front-end
- Middleware
- Back-end

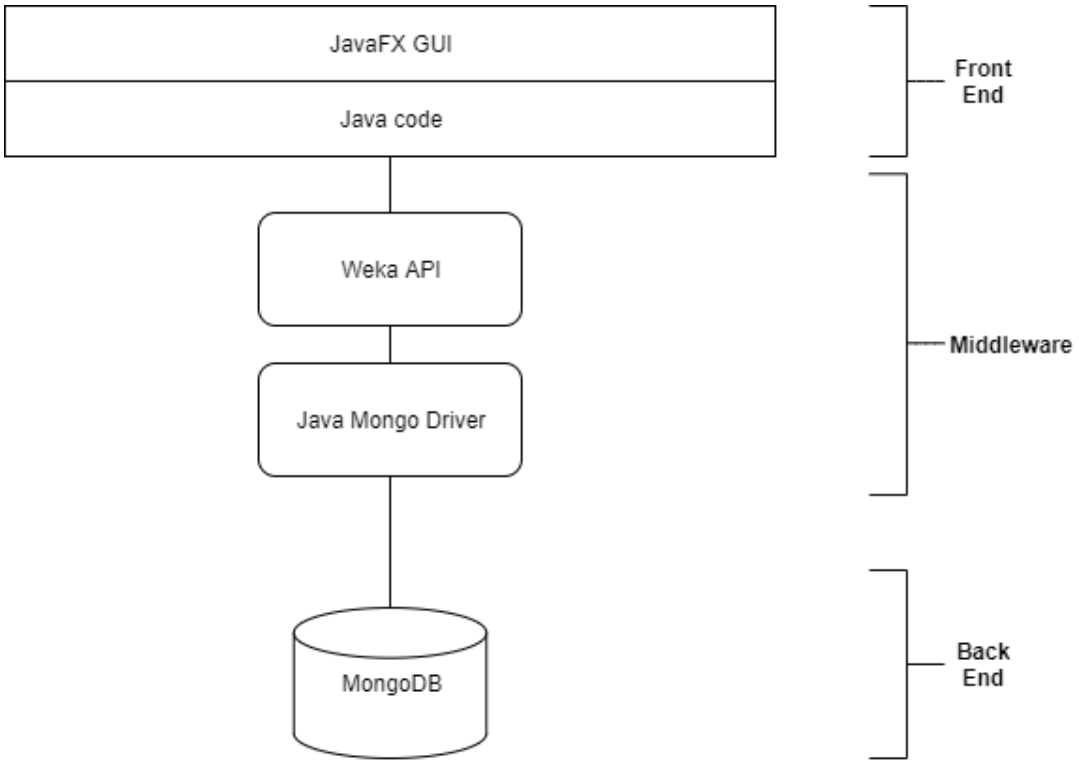

Figure 5: Software architecture diagram