



Cairo University
Faculty of Computers and Artificial intelligence
Department of Computer Sciences



SentiSnap: Sentiment Analysis Tool for Social Media Reviews

Supervised by

Dr. Cherry Ahmed

TA. Menna Allah Youssef

Implemented by

20200589	Nada Alaa Eldeen Ali
20200617	Hady Abdullah Hafez
20200532	Mariam Hany Gamal
20200464	Mohamed Essam Eldeen Mohamed
20200609	Nouran Ahmed Abdelaziz

Graduation Project

Academic Year 2023 - 2024

Final Year Documentation

Table of Contents

Chapter 1: Introduction	6
1.1 Motivation	6
1.2 Problem definition	6
1.3 Project Objective (Suggested Solution)	7
1.4 Gantt chart of project time plan	8
1.5 Project development methodology	9
1.6 The used tools in the project (SW and HW)	10
1.7 Report Organization	11
Chapter 2: Related work	11
2.1 Closest Examples	11
2.2 Main Differences	13
Chapter 3: System Analysis	14
3.1 Project specification	14
3.1.1 Functional requirement	14
3.1.2 Non-functional requirement	15
3.2 Use case Diagram	16
Chapter 4: System Design	17
4.1 System Component Diagram	17
4.2 System Class Diagram	18
4.3 Sequence Diagrams	19
4.4 Project ERD	21
4.5 Machine Learning Architecture	21
4.6 System GUI Design	22
Chapter 5: Implementation and Testing	26
5.1 Implementation	26
5.1.1 Frontend	26
5.1.2 Backend	26
5.1.3 Dataset	27
5.1.4 Data preparation and Preprocessing	28
5.1.5 Machine Learning Models and Algorithms	33
5.1.6 Facebook and Instagram Scrapers	42

5.2 Testing	44
5.2.1 System Testing	44
5.2.2 Integration Testing	48
References	51

List Of Figures

Figure 1.. Gantt Chart	8
Figure 2.. Monkey Learn's analysis example	12
Figure 3.. Usecase Diagram	16
Figure 4.. Component Diagram	17
Figure 5.. Class Diagram.....	18
Figure 6.. Get Analysis for Post Sequence Diagram	19
Figure 7..Show Top10 in specific category sequence diagram.....	19
Figure 8..Search for specific page name sequence diagram	20
Figure 9.. View User History Sequence Diagram	20
Figure 10.. ERD	21
Figure 11.. Machine Learning Architecture	21
Figure 12.. Home Page GUI	22
Figure 13.. Analysis Page GUI.....	22
Figure 14.. Top10 Page GUI	23
Figure 15.. Posts Page GUI	23
Figure 16.. History Page of Text GUI	24
Figure 17.. History Page of URLs GUI	24
Figure 18.. Sign Up Page GUI.....	25
Figure 19.. Sign In Page GUI	25
Figure 20.. Sample of Arabic 100k Reviews Dataset	28
Figure 21.. Bar Chart of Dataset Polarity Count.....	28
Figure 22.. Arabic Cleaning Function.....	30
Figure 23.. Text Preprocessing.....	32

Figure 24.. Document and Word Embedding	33
Figure 25.. Logistic Regression Model	34
Figure 26.. classification report of Logistic regression model.....	35
Figure 27.. SVM Model.....	35
Figure 28.. Classification report of SVM model	36
Figure 29.. Decision Tree Model	36
Figure 30.. Classification report of Decision Tree Model	37
Figure 31.. KNN Model.....	37
Figure 32.. Classification report of KNN Model with k = 7.....	38
Figure 33.. Classification report of KNN Model with k = 10.....	38
Figure 34.. Random Forest Model.....	39
Figure 35.. Classification report of Random Forest model	39
Figure 36.. Multi-Layer Perceptron Model	40
Figure 37.. Classification report of MLP Model.....	40
Figure 38.. LSTM Model.....	41
Figure 39.. RNN Model Accuracy.....	42
Figure 40.. LSTM Model Accuracy.....	42
Figure 41.. Scraped Comments and Caption of the Instagram Post	43
Figure 42.. Instagram Post	43
Figure 43.. Facebook Post.....	43
Figure 44.. Scraped Comments and Caption of the Facebook Post.....	43
Figure 45.. Testing when a user enters Arabic text	44
Figure 46.. Testing when user enters Non-Arabic Text	44
Figure 47.. Analyzing Facebook or Instagram URL	45
Figure 48.. Positive Comments Scraped	45
Figure 49.. Negative Comments Scraped.....	46
Figure 50.. Testing when user enters Invalid URL.....	46
Figure 51.. Testing Link of a Post in a Private Account	47
Figure 52.. Testing Search for a Word.....	47
Figure 53.. Testing Sharing a Post without a Title	48
Figure 54.. Register API when register successfully	48
Figure 55.. Register API when email exists	49
Figure 56.. Signin API when user enters wrong email or pass.....	49
Figure 57.. Signin API when the user enters a correct email and pass.....	50
Figure 58.. Search API	50

List Of Tables

Table 1.. Tools needed for Implementing SentiSnap	7
---	---

List Of Abbreviations

LSTM	Long short-term memory
ORM	object-relational mapper
ANN	Artificial Neural Network
RNN	Recurrent Neural Network
HARD	Hotel Arabic Reviews Dataset
BRAD	Books Reviews in Arabic Dataset
ISRISemmer	Information Sciences Institute semmer

Chapter 1: Introduction

The main area of this project is the integration of sentiment analysis and machine learning. Sentiment analysis, also known as opinion mining, is a natural language processing (NLP) task that involves determining the sentiment or emotional tone expressed in a piece of text. It is widely used in various applications, such as social media monitoring, customer feedback analysis, and market research. Machine learning techniques are commonly employed for sentiment analysis due to their ability to automatically learn patterns and relationships from data.

1.1 Motivation

This project is inspired by the growing importance of understanding people's feelings and opinions online, especially in Arabic reviews. Nowadays, what people say online is a big deal, and it influences what others think and decide. However, it's not always easy to figure out how people feel in these online discussions.

Our main goal is to create a tool that helps users, including online brands, influencers, and companies, better understand the sentiments expressed in Arabic reviews on different social media platforms, particularly Instagram and Facebook. Ultimately, we hope that this tool will empower brands, influencers, and companies to improve their products, content, or any related aspects based on the sentiments expressed in the reviews, leading to continuous enhancement and better connection with their audience.

1.2 Problem definition

In the ever-growing world of social media, platforms like Instagram and Facebook have become prominent venues where users share their experiences and opinions on various topics, spanning numerous categories. Both individuals and businesses within these categories heavily rely on customer feedback and public sentiment to gauge the reception of their products or services. However, analyzing the vast volume of reviews and comments manually is not only time-consuming but also impractical, often leading to delays in understanding consumer sentiment and making informed decisions.

There is a significant need for a specialized tool designed to perform sentiment analysis on social media reviews, particularly tailored for platforms like Instagram and Facebook.

1.3 Project Objective (Suggested Solution)

The primary objective of SentiSnap is to develop an efficient and user-friendly sentiment analysis tool designed to handle the specific challenges of analyzing Arabic social media comments and text. This tool aims to provide individuals and businesses with a quick and reliable means of understanding public sentiment without the need for time-consuming manual analysis.

This project will use Natural Language Processing (NLP) techniques to analyze the sentiment of the comments on posts or the sentiment of a given text. This project will help a lot of different online groups like companies, influencers, and brands.

They all benefit by understanding how people feel on social media reviews. For companies and brands, it means they can make better products and improve how they talk to their customers. Influencers can use it to know what their followers feel, so they can create content that people like. It'll also benefit the general people as they know the sentiments on the products they want to buy, or to know if the influencer they're following is reliable or not.

Our website takes the link to analyze its comments, first scrape all the comments, then categorize it to positive comments and negative comments in a friendly and visually appealing interface in a few minutes, saving the time of the users by showing them the positive and negative, making it more readable instead of reading all the comments to understand.

Implementing the SentiSnap project needs tools like:

HTML, CSS for frontend	Bootstrap framework
JavaScript programming language	React JS framework for frontend
Python programming language	Regular Expression (regex)
PHP Laravel for backend	Postman for API testing
NLTK library	Instaloader Library
facebook_scraper Library	sklearn Library

Table 1.. Tools needed for Implementing SentiSnap

1.4 Gantt chart of project time plan

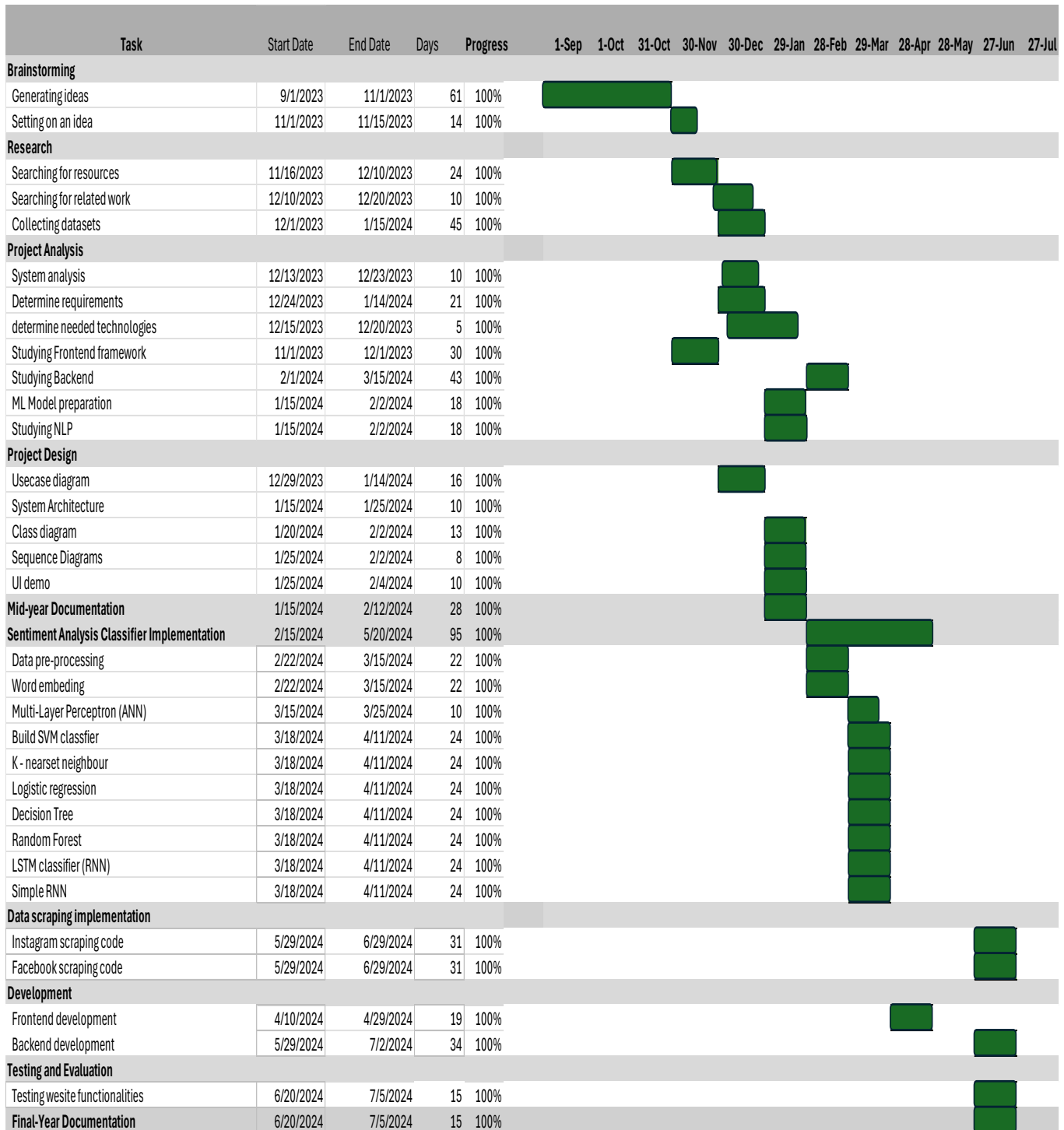


Figure 1.. Gantt Chart

1.5 Project development methodology

In this project, we used the waterfall methodology, it's the basic software development lifecycle (SDLC). The waterfall methodology is a sequential approach to software development. It follows a set order of phases where each phase must be completed before the next one begins. Each phase has defined deliverables and milestones, making it easier to track progress. So, this methodology helped our project in setting deadlines and meeting them. The phases of the waterfall methodology are:

- **Requirements Gathering:**

In this phase, we defined what the requirements of the project will be, and who is going to use it. We gathered the functional and non-functional requirements of the project, analyzed the needs for sentiment analysis in Arabic reviews on Instagram and Facebook, and developed detailed specifications for what is needed for frontend, backend, and machine learning.

- **System Design:**

In this phase, we take the gathered requirements from the first phase and create a detailed system architecture and design documents based on it. Example: create UML diagrams, design the database schema, and plan the machine learning model architecture for sentiment analysis.

- **Implementation:**

The implementation phase involves the coding and development of the website, based on what's given from the previous phase. We developed the frontend, backend, machine learning model for sentiment analysis, including data preprocessing and training the model with collected datasets, then integrating them.

- **Testing:**

Once the implementation phase is completed, we conducted thorough testing of the website to ensure that it meets all the requirements and functions as intended. The sentiment analysis classifier is tested extensively to ensure its accuracy and reliability.

By following these phases, the SentiSnap project can be developed in a structured and organized manner, ensuring all aspects are thoroughly planned, executed, and maintained.

1.6 The used tools in the project (SW and HW)

Frontend:

- HTML, CSS, Bootstrap
- Java Script
- React js
- Figma for designing the pages
- VSCode development environment
- ColorHunt website for help determining Color
- Postman for testing API

Backend:

- PHP
- Laravel
- MySQL

Machine Learning:

- Python programming language: NumPy, Pandas, Sklearn, Matplotlib for visualization, nltk, pyarabic for data preprocessing.
- Arabic 100k Reviews: The dataset combines reviews from hotels, books, movies, products and a few airlines to train sentiment analysis classifier based on Arabic reviews with has two classes (Negative and Positive).
- Google Collab environment for models training.

Commet Scraper:

- Python library:
 - o Facebook-scraper library to scrape Facebook public pages.
 - o Instaloader library to scrape Instagram public pages.

1.7 Report Organization

The rest of the document discusses the different phases and characteristics of the project.

Chapter Two: we're going to talk about related works, compare these related works with our project, and determine user problems with them, to try to fix these problems in our project.

Chapter Three: in this chapter, we're going to analyze our system, defining the project specifications, including functional requirements, according to what we concluded in related work research, non-functional requirements that defines the quality attributes, system performance, and constraints of our SentiSnap project, and the usecase diagram.

Chapter Four: system design, this chapter is more focused on the UML diagrams that describe the relation between the classes and component of the system, and how the data is stored in the database.

- Class Diagram
- Component Diagram
- Sequence Diagrams
- Entity Relation Diagram
- Machine Learning Architecture
- GUI design

Chapter Five: Implementation and Testing, in this chapter we're going to discuss our project in detail; we're going to discuss our front-end, back-end, the machine learning models, and the scraping of instagram and facebook's posts.

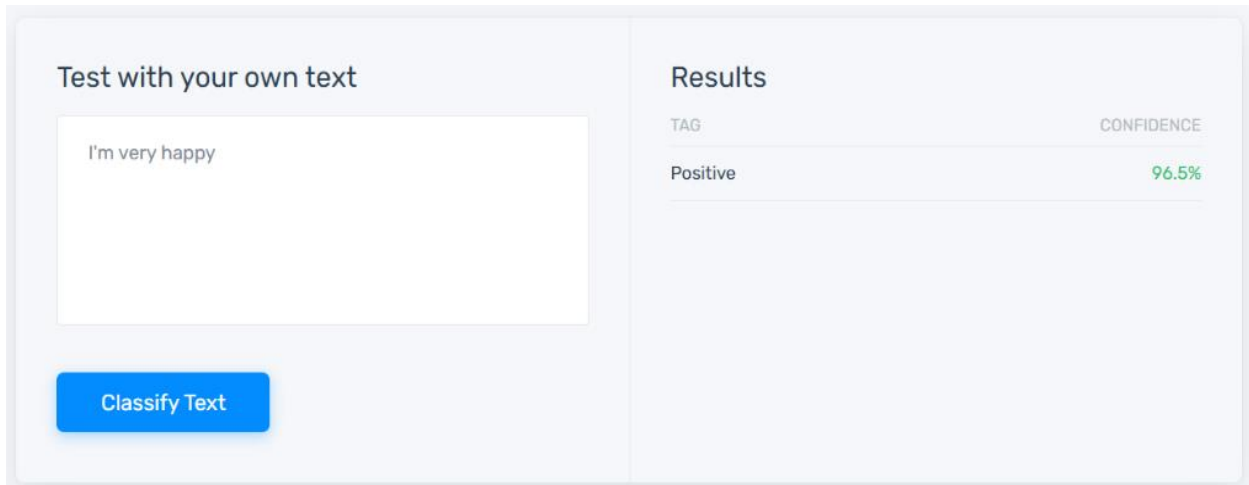
And test our SentiSnap project system, and test the integration of each API.

Chapter 2: Related work

2.1 Closest Examples

- **Monkey Learn's sentiment analysis**
 - **Purpose:** This tool analyzes English comments and classifies them into three categories: positive, negative, and neutral.
 - **Application:** It is used for software products, app reviews, and regular sentences.

- **Technology:** Uses machine learning to process and categorize text data.
- **Scope:** Focused on English language reviews across various platforms and products.



TAG	CONFIDENCE
Positive	96.5%

Figure 2.. Monkey Learn's analysis example

- **Talkwalker sentiment analysis**

- **Purpose:** Collects sentimental information from reviews on online articles, social media platforms, or survey responses.
- **Features:** Detects basic forms of sarcasm and helps teams immediately react to relevant posts.
- **Technology:** Utilizes artificial intelligence and natural language processing to analyze the tone and emotion of posts.
- **Scope:** Broad application across different content types and platforms.

- **Sentisum sentiment analysis**

- **Purpose:** Uses customer feedback data to identify bottlenecks in products and shape the product roadmap.
- **Features:** Automatically tags and categorizes feedback using NLP, identifying positive and negative sentiments and their underlying drivers.
- **Technology:** Focuses on customer feedback to improve product quality and business productivity.

- **Scope:** Targeted towards customer feedback data for product improvement.

2.2 Main Differences

In this section, we outline the primary distinctions between SentiSnap and other sentiment analysis tools, highlighting SentiSnap's unique features and capabilities.

SentiSnap vs. Monkey Learn:

- **Language Focus:**
 - **SentiSnap:** Specializes in Arabic reviews.
 - **Monkey Learn:** Primarily handles English reviews.
- **Platform Scope:**
 - **SentiSnap:** Specifically targets social media platforms, particularly Instagram and Facebook.
 - **Monkey Learn:** Applies to various platforms and product reviews without a specific focus on social media.

SentiSnap vs. Talkwalker:

- **Emotion Analysis:**
 - **SentiSnap:** Focuses on the sentiment of comments (positive or negative).
 - **Talkwalker:** Analyzes tone and emotion, providing a broader emotional understanding.

SentiSnap vs. Sentisum:

- **Feedback Scope:**
 - **SentiSnap:** Focuses on understanding public sentiment to help brands and influencers.
 - **Sentisum:** Specifically targets customer feedback for product improvement.
- **Technological Approach:**
 - **SentiSnap:** Uses NLP techniques to analyze social media comments, highlighting the most frequent positive words.
 - **Sentisum:** Employs NLP to tag and categorize feedback, focusing on the underlying drivers of sentiments.

Unique Features of SentiSnap:

- **Arabic Language Processing:** Specializes in analyzing Arabic social media reviews, a niche that is often underrepresented in existing tools.
- **Positive and Negative Comments Analysis:** Extracts and displays all the positive comments and negative comments in a visually appealing manner, saving users' time and making sentiment trends easily understandable.
- **User Interface:** Designed to be user-friendly and visually appealing, catering to both technical and non-technical users.

SentiSnap offers a unique and specialized tool for Arabic social media sentiment analysis, setting it apart from existing solutions like Monkey Learn, Talkwalker, and Sentisum. By focusing on the specific challenges of analyzing Arabic comments on platforms like Instagram and Facebook, SentiSnap provides a tailored, user-friendly interface that highlights key sentiment trends quickly and effectively, ensuring valuable insights for brands, influencers, and general users.

Chapter 3: System Analysis

3.1 Project specification

3.1.1 Functional requirement

User can:

1. Sign-up.
2. Sign-in.
3. Logout.
4. Enter a link in the provided input.
5. Choose the category of the link content.
6. Enter a text in the provided input to get its analysis.
7. View the percentage of positive and negative percentage of the provided Arabic text.
8. View the percentage of positive and negative Arabic reviews in the provided link.
9. View the positive comments on the provided link
10. View the negative comments on the provided link
11. Show history of text or URLs.
12. Show top 10 links of each category.

13. Search by page name or any word.
14. Share analysis.

3.1.2 Non-functional requirement

- **Usability:**
 - The system should be easy to use and navigate, with clear instructions and an intuitive interface.
- **Reliability:**
 - The sentiment analysis accuracy should be at least 80%.
 - If the system goes down, it should be able to recover within minutes.
- **Performance:**
 - The system processes should be done in a few seconds.
 - The system should be able to handle many users and files without experiencing delays.
- **Scalability:**
 - The system should handle concurrent requests from multiple users.
 - The system can perform well with small or large amounts of text.
- **Maintenance:**
 - Updates or changes to user account functionalities shouldn't impact the overall system stability, as the system structure follows the SOLID principles.

3.2 Use case Diagram

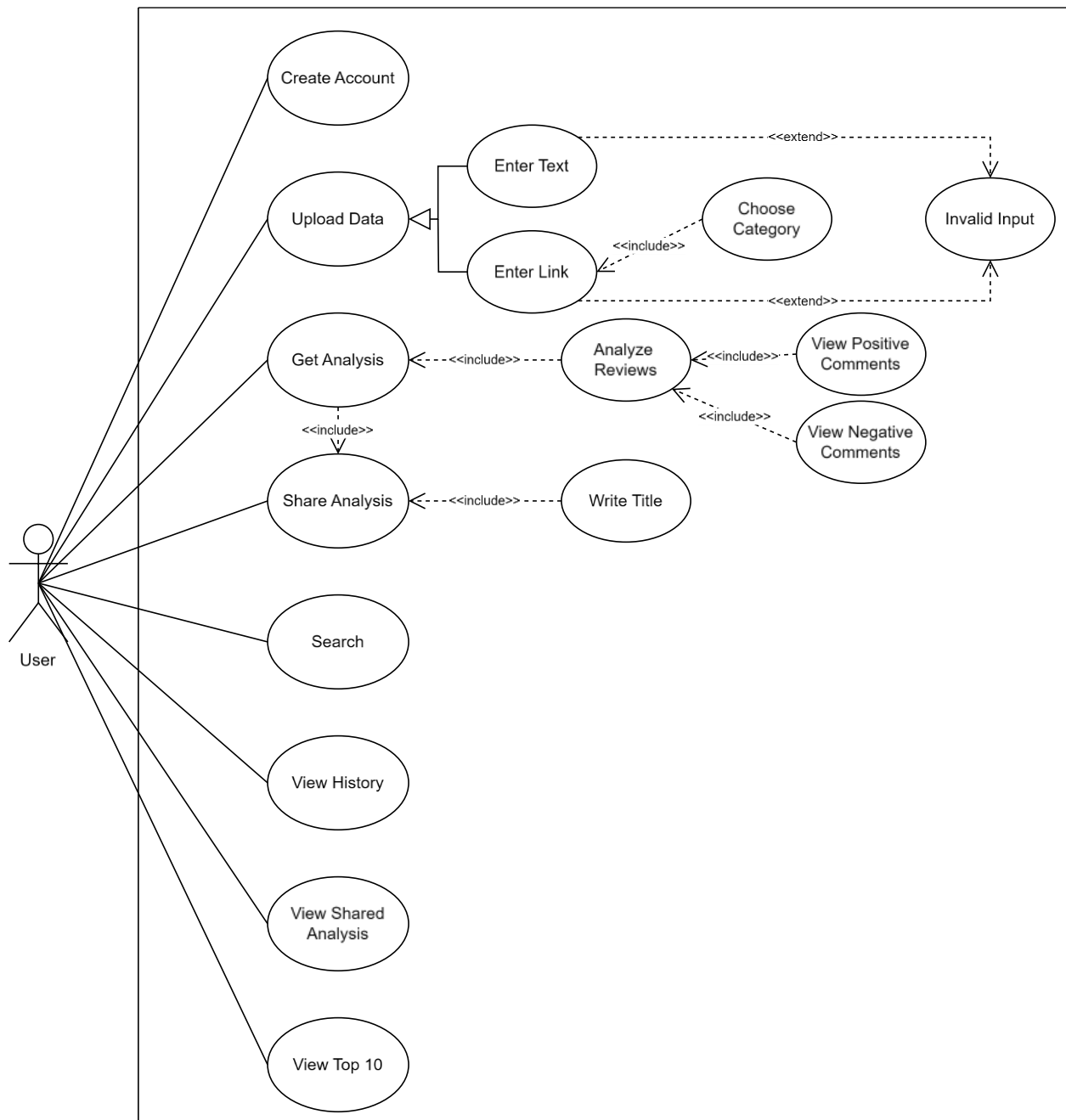


Figure 3.. Usecase Diagram

Chapter 4: System Design

4.1 System Component Diagram

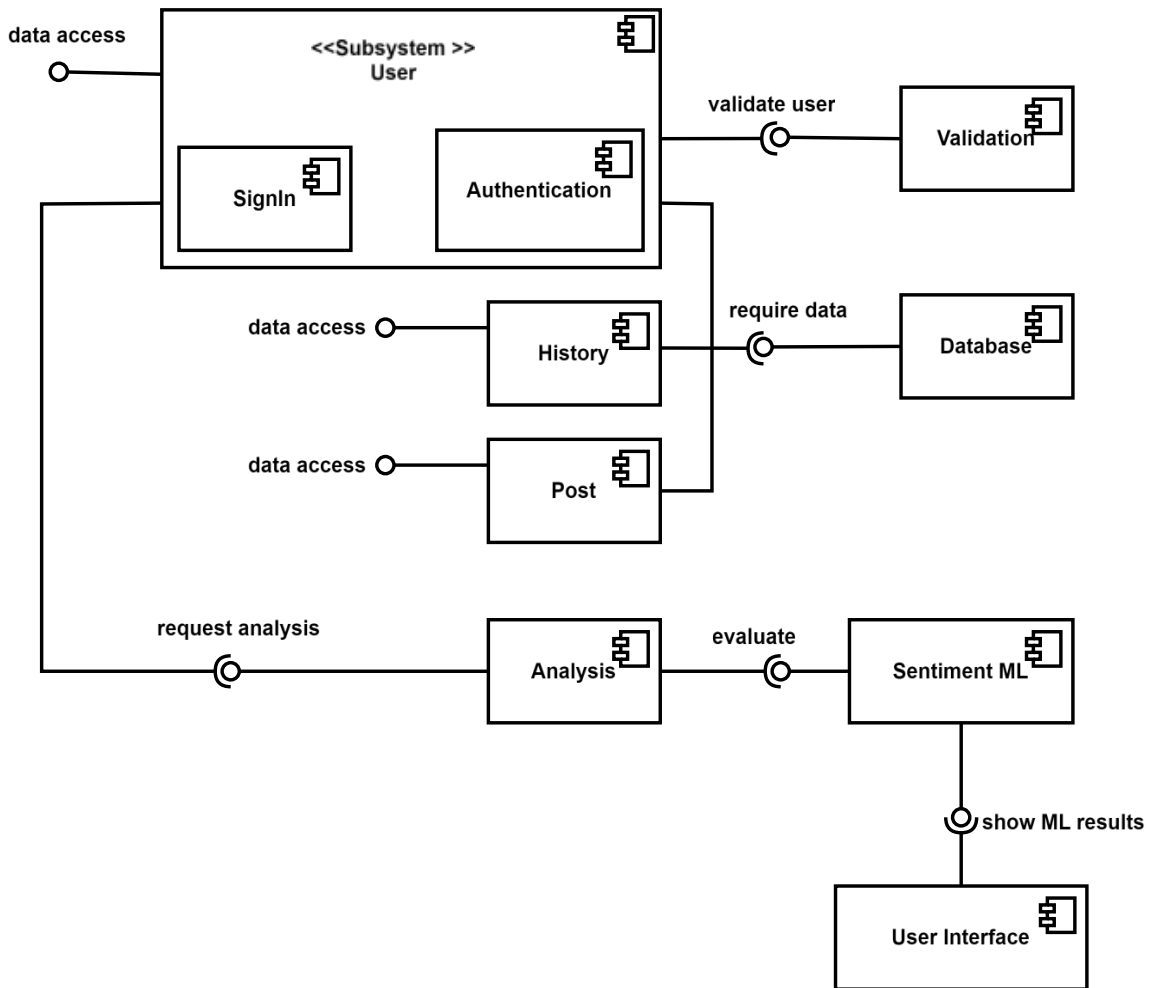


Figure 4.. Component Diagram

4.2 System Class Diagram

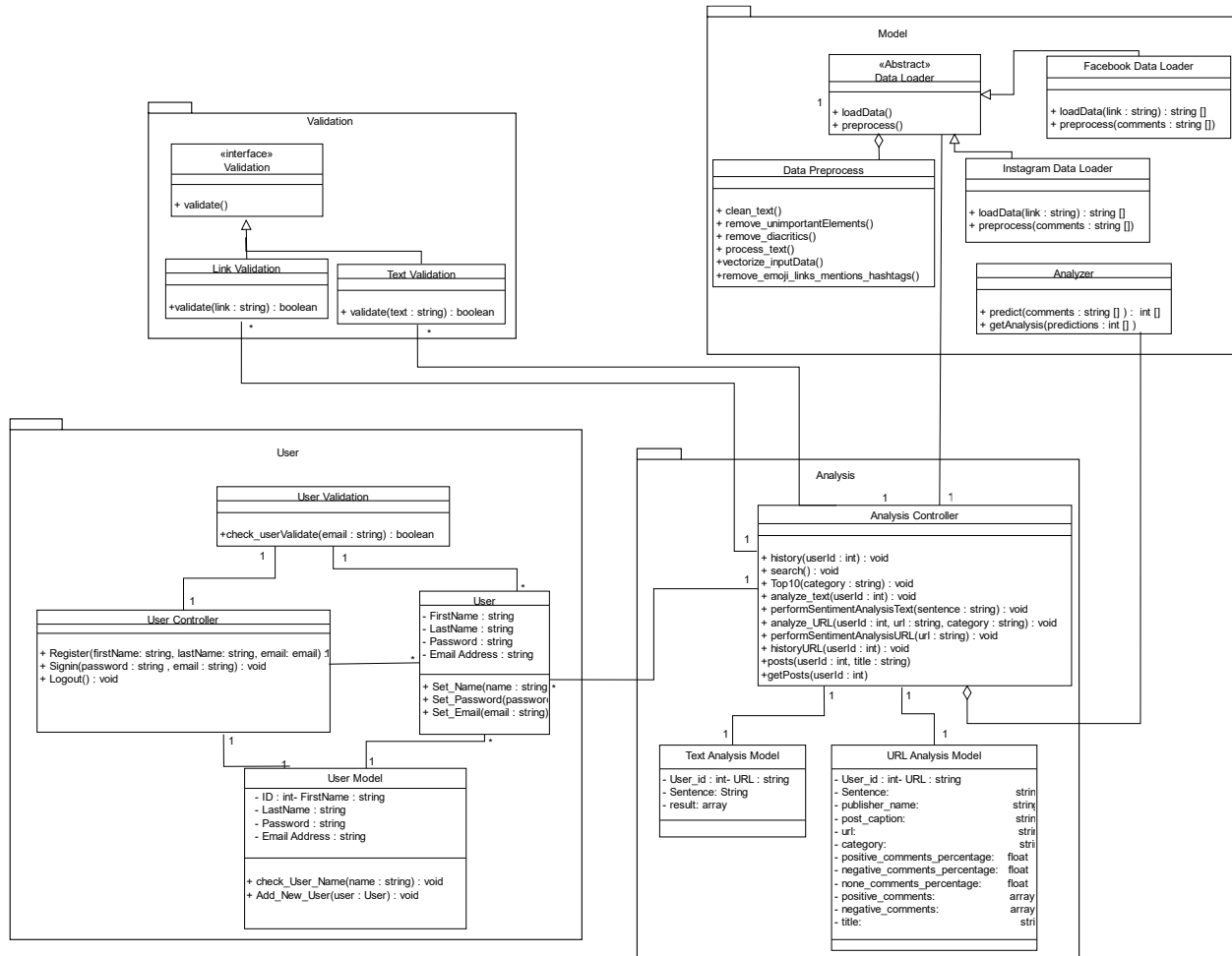


Figure 5.. Class Diagram

4.3 Sequence Diagrams

- Get Analysis for Post.

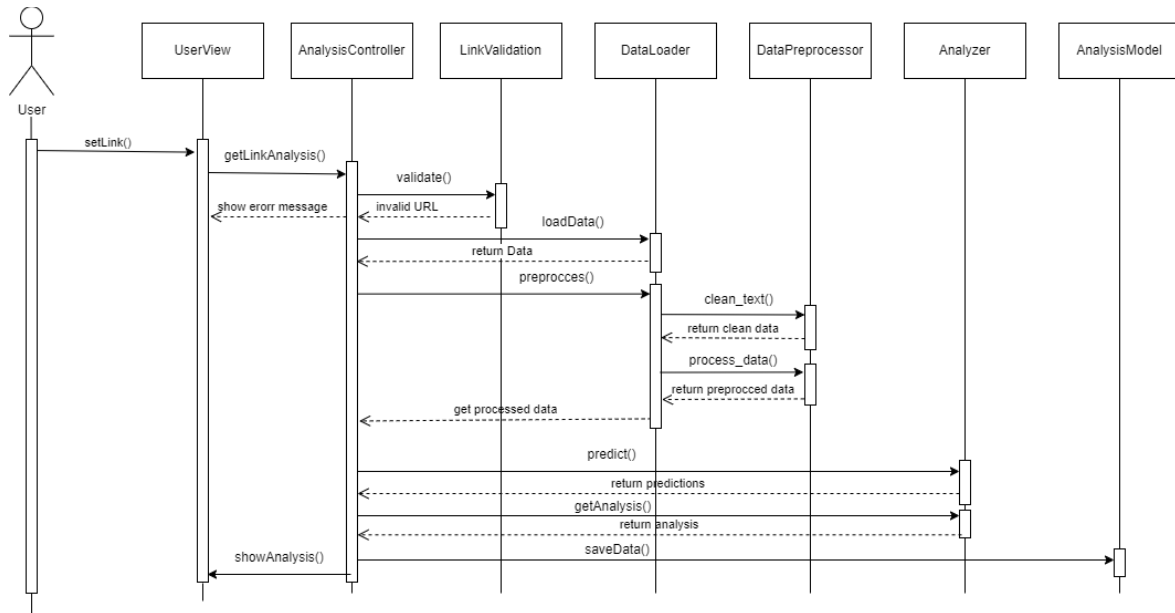


Figure 6.. Get Analysis for Post Sequence Diagram

- Show Top10 in specific category.

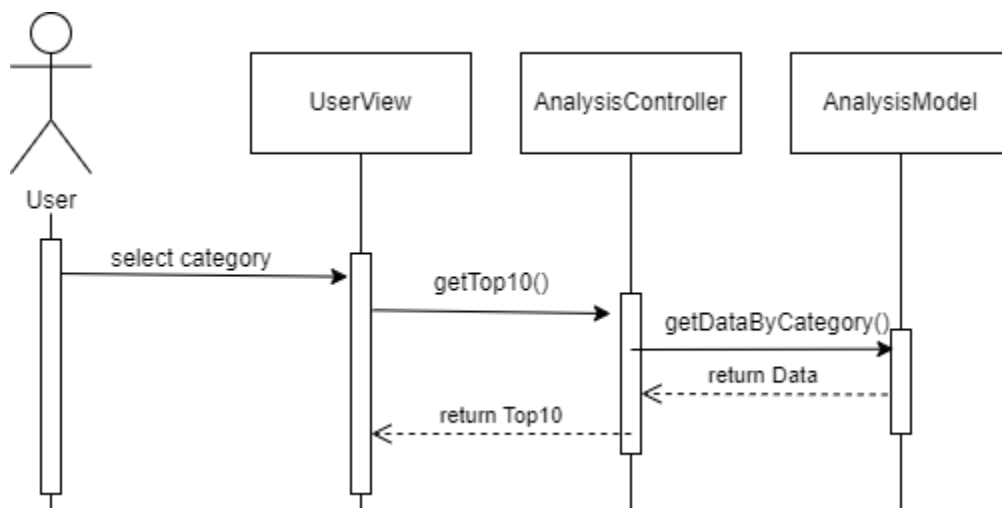


Figure 7..Show Top10 in specific category sequence diagram

- **Search for specific page name.**

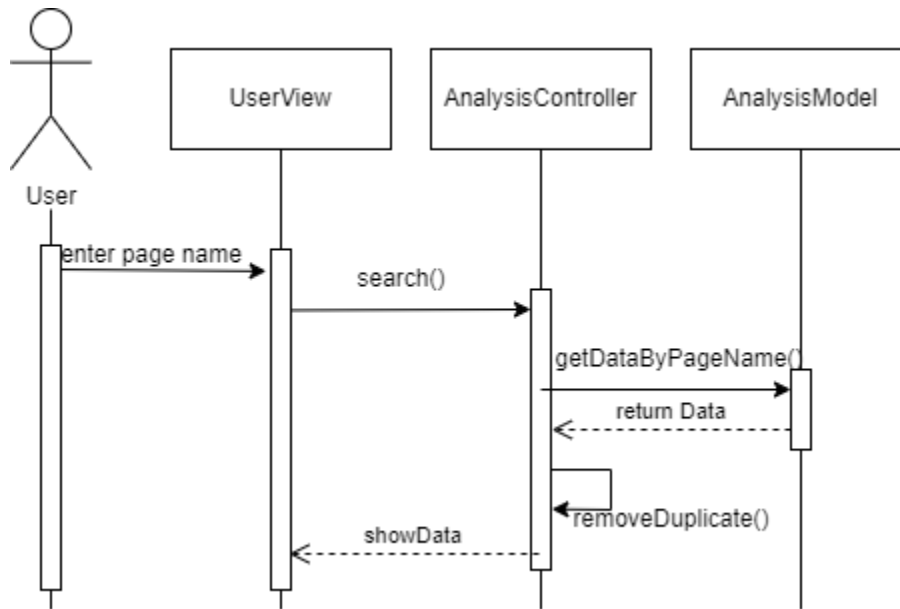


Figure 8..Search for specific page name sequence diagram

- **View User History.**

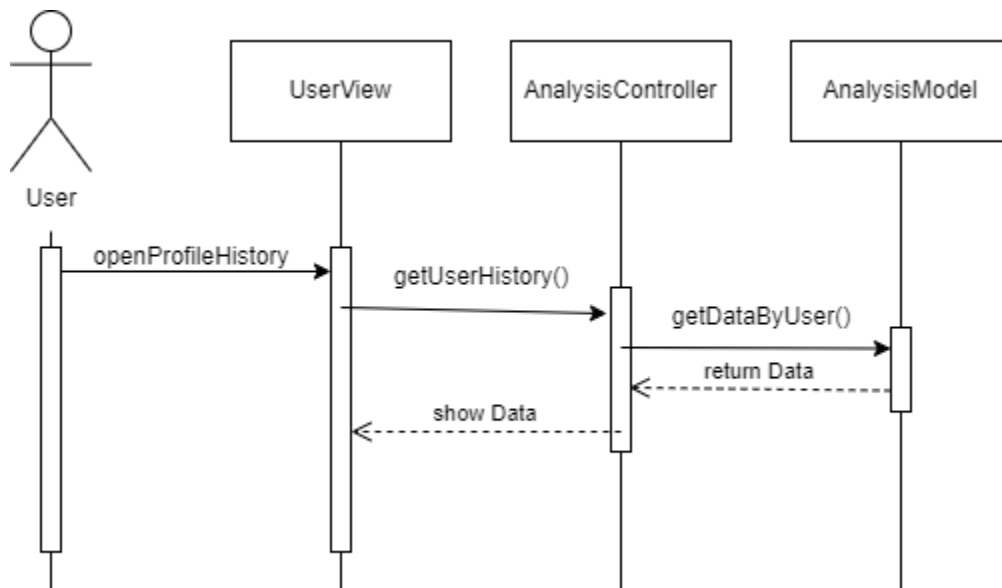


Figure 9.. View User History Sequence Diagram

4.4 Project ERD

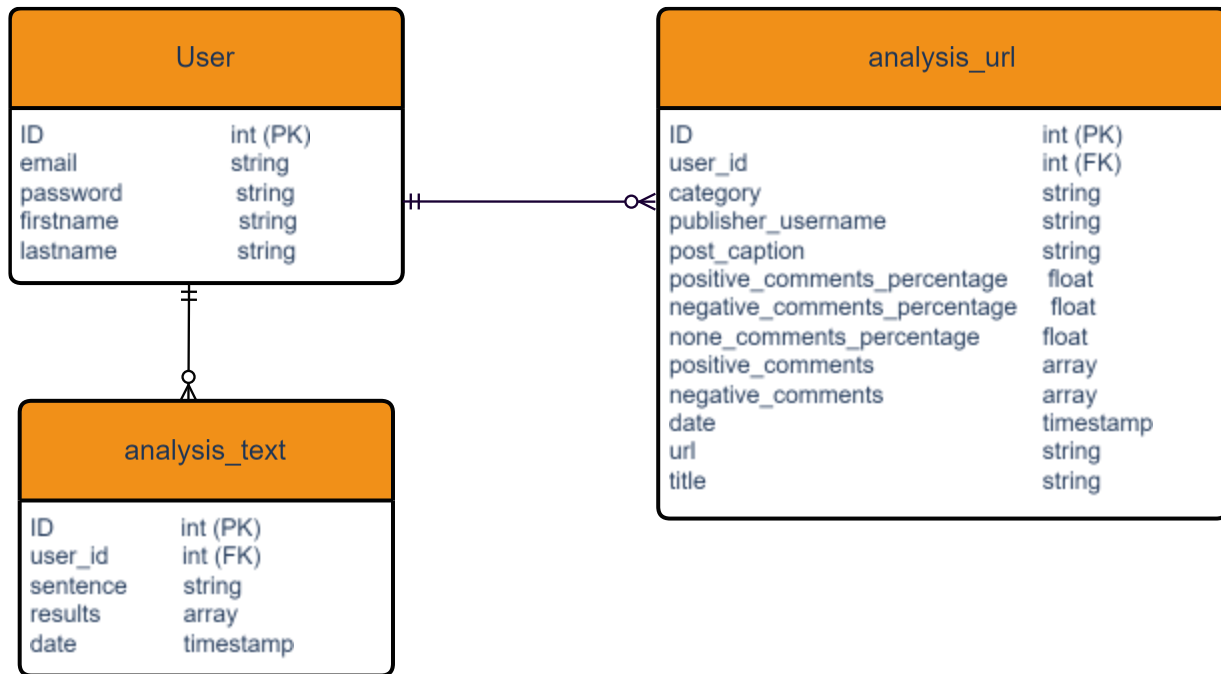


Figure 10.. ERD

4.5 Machine Learning Architecture

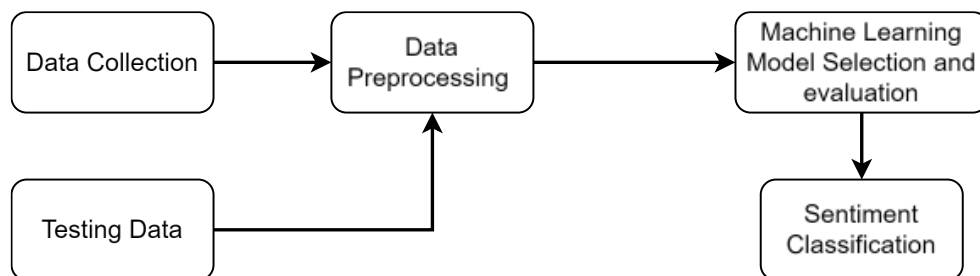


Figure 11.. Machine Learning Architecture

4.6 System GUI Design

- Home Page

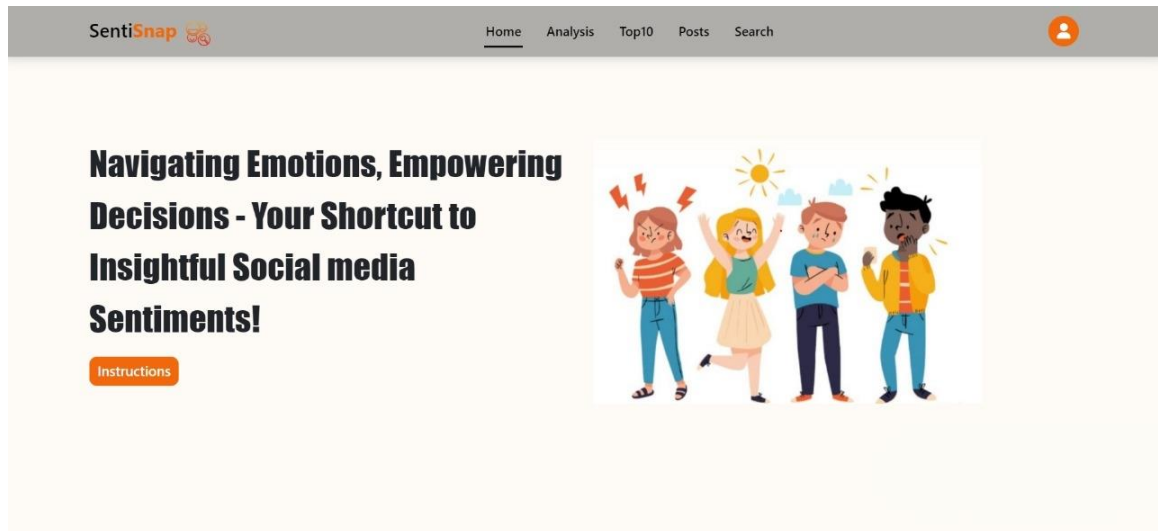


Figure 12.. Home Page GUI

- Analysis Page

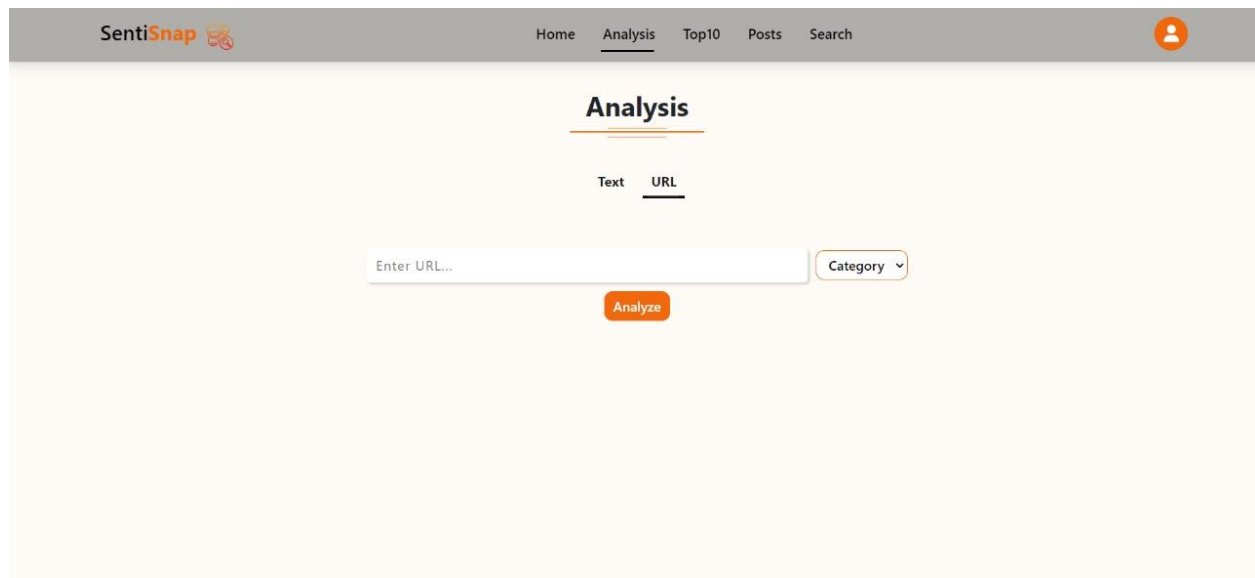


Figure 13.. Analysis Page GUI

- Top10 Page

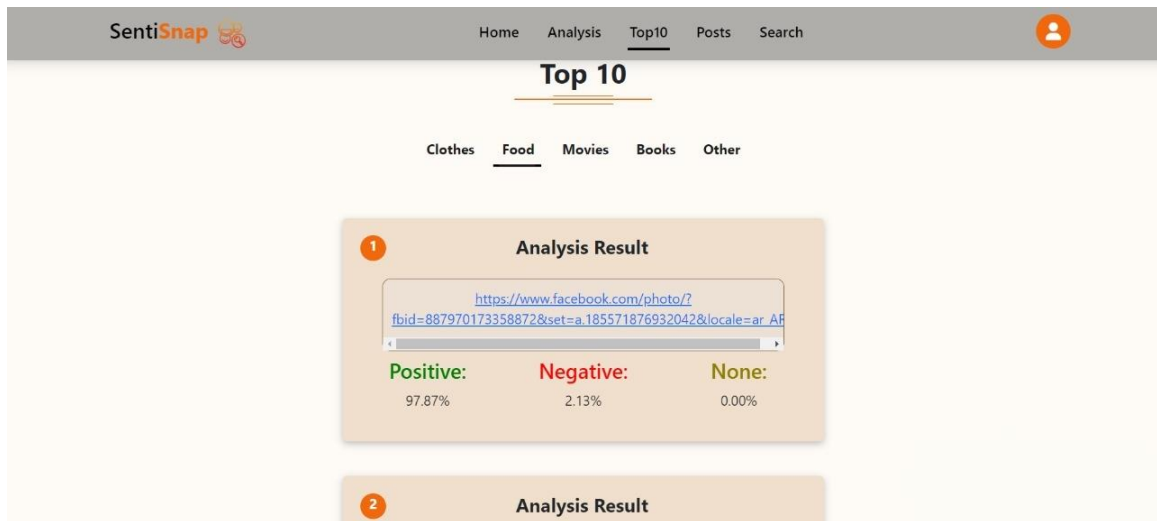


Figure 14.. Top10 Page GUI

- Posts Page

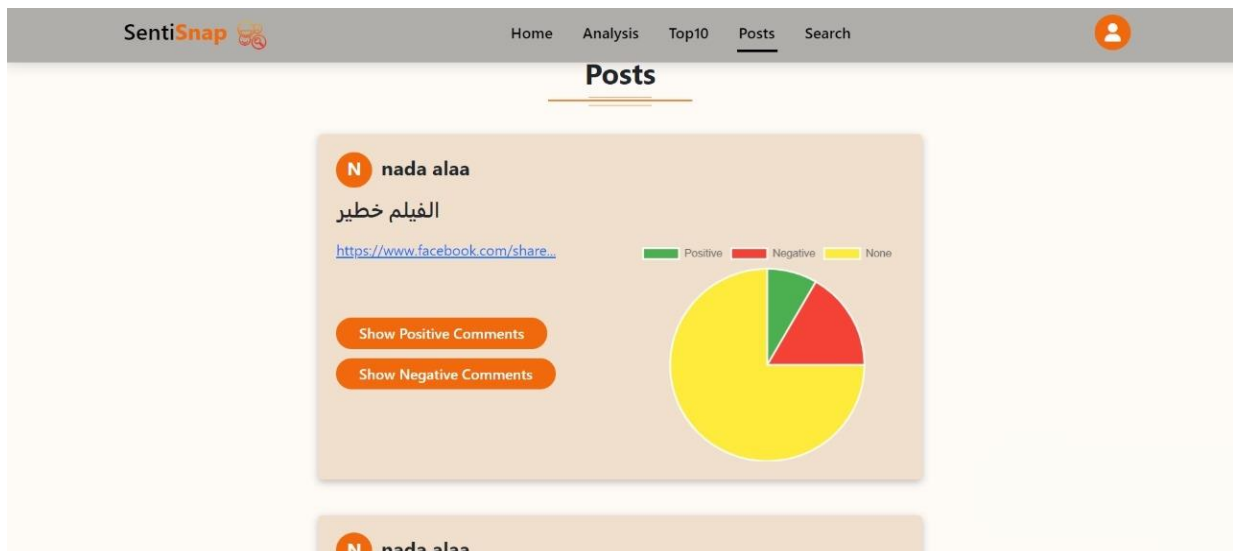


Figure 15.. Posts Page GUI

- History Page

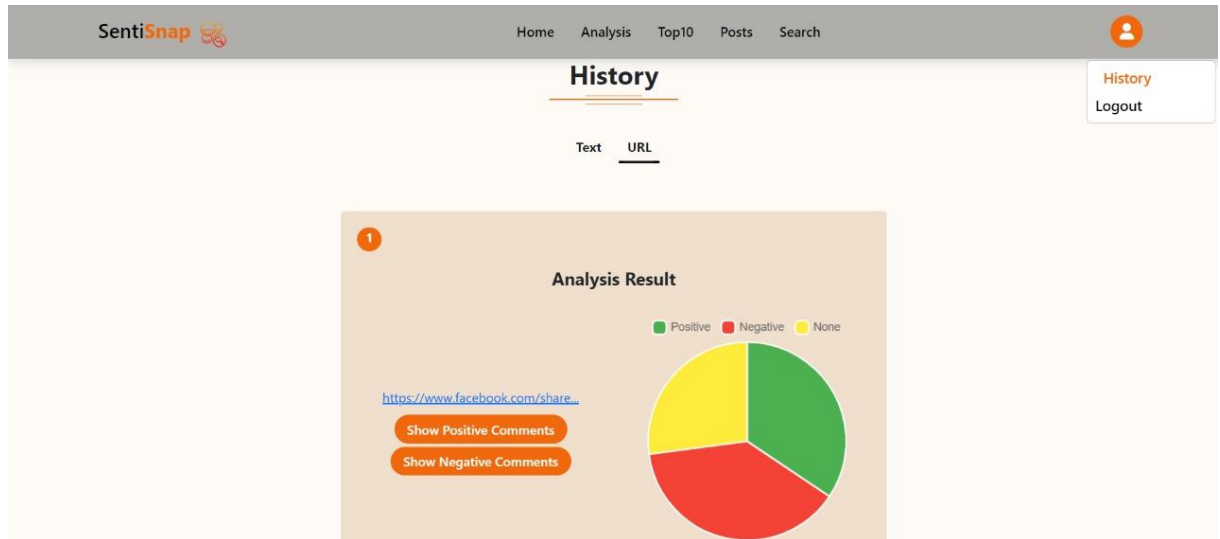


Figure 17.. History Page of URLs GUI

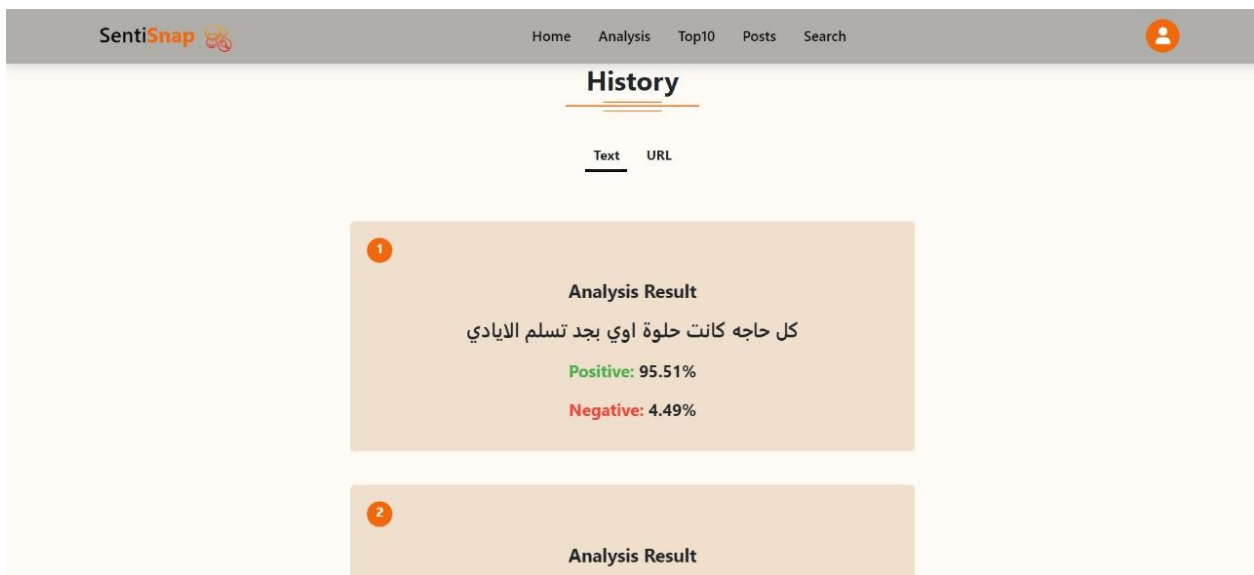
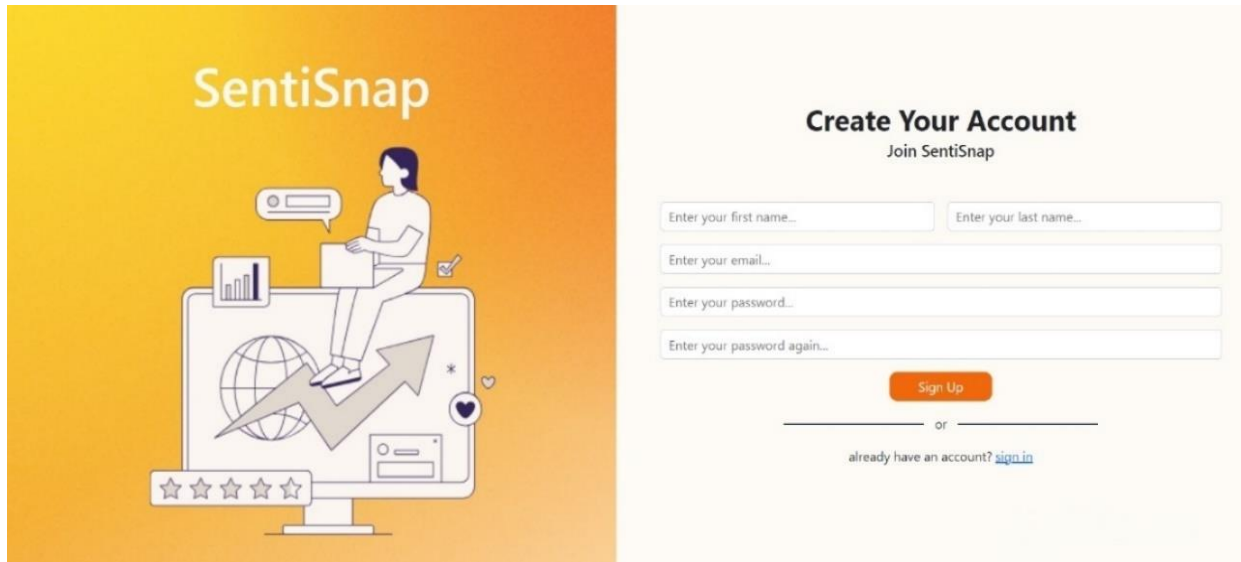


Figure 16.. History Page of Text GUI

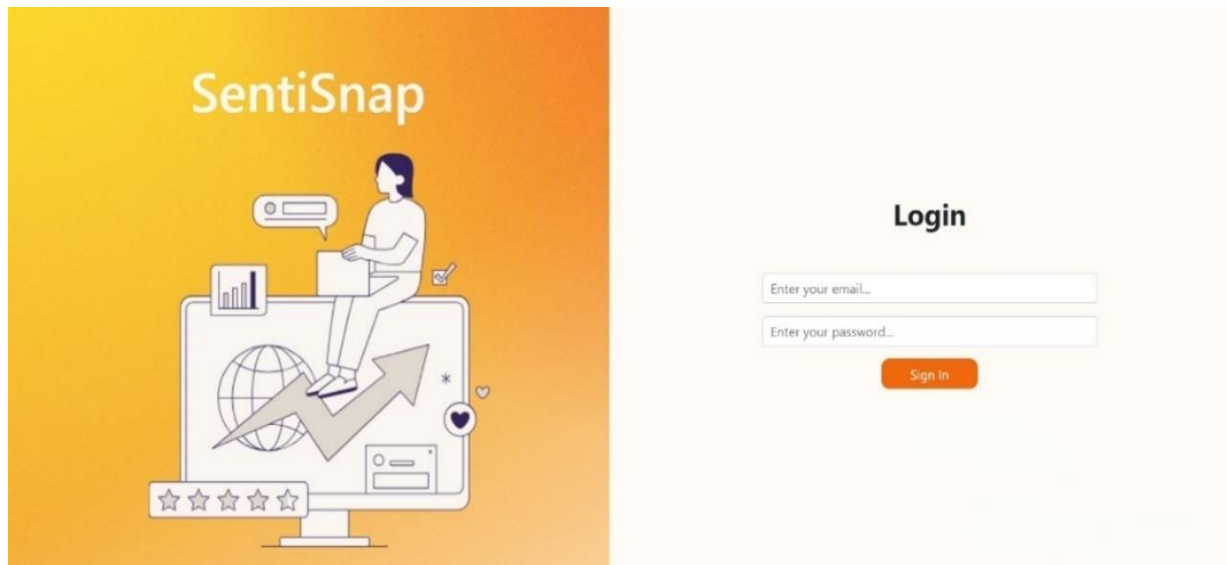
- Sign up Page



The image shows the SentiSnap sign-up page. On the left is a vertical orange banner with the SentiSnap logo at the top. Below the logo is an illustration of a person sitting on a large computer monitor. The monitor displays a bar chart, a globe, and a large upward-pointing arrow. There are also five stars at the bottom of the monitor. On the right side of the banner, there are icons for a speech bubble, a laptop, a checkmark, and a heart. The main content area on the right has a light beige background. It features the heading 'Create Your Account' and the subheading 'Join SentiSnap'. Below these are four input fields: 'Enter your first name...', 'Enter your last name...', 'Enter your email...', and 'Enter your password...'. Below the password field is another field labeled 'Enter your password again...'. A blue 'Sign Up' button is positioned below the fields. Below the button is a horizontal line with the word 'or' in the center. Below the line is the text 'already have an account? [sign in](#)'.

Figure 18.. Sign Up Page GUI

- Sign in Page



The image shows the SentiSnap sign-in page. On the left is a vertical orange banner with the SentiSnap logo at the top. Below the logo is an illustration of a person sitting on a large computer monitor. The monitor displays a bar chart, a globe, and a large upward-pointing arrow. There are also five stars at the bottom of the monitor. On the right side of the banner, there are icons for a speech bubble, a laptop, a checkmark, and a heart. The main content area on the right has a light beige background. It features the heading 'Login'. Below the heading are two input fields: 'Enter your email...' and 'Enter your password...'. A blue 'Sign In' button is positioned below the fields.

Figure 19.. Sign In Page GUI

Chapter 5: Implementation and Testing

5.1 Implementation

5.1.1 Frontend

In front end, we used HTML, CSS, JS, Bootstrap, and React js.

We used React js because it is a popular open-source JavaScript library used for building user interfaces, particularly for single-page applications where the interface needs to be dynamic and responsive.

The virtual DOM and efficient updating mechanism make React applications fast and responsive, even with complex UIs and frequent updates.

We used a lot of libraries from it, for example: formik and yup in validation.

5.1.2 Backend

Our main technology for the back end of our application was Laravel as we choose it because:

- 1. Elegant Syntax:** Laravel offers a clean and elegant syntax that simplifies the development process, making it simple and efficient for developers.
- 2. MVC Architecture:** Laravel applies Model-View-Controller (MVC) architectural pattern, which facilitates the separation of concerns, making the codebase more organized and maintainable.
- 3. Eloquent ORM:** Laravel's Eloquent ORM provides a beautiful, simple Active Record implementation for working with your database. Each database table has a corresponding "Model" which is used to interact with that table.
- 4. Built-in Authentication:** Laravel comes with an out-of-the-box authentication system, including user login and registration.
- 5. Robust Ecosystem:** Laravel has a rich ecosystem of tools and packages that can be easily integrated, from debugging tools to API documentation generators.
- 6. Security:** Laravel includes security features such as protection against SQL injection, cross-site request forgery (CSRF), and cross-site scripting (XSS), making your application more secure by default.

Laravel's comprehensive features and its emphasis on elegant, readable code make it an excellent choice for building our sentiment analysis application. It provides a robust framework that simplifies many of the common tasks associated with web development while maintaining a high level of security and performance.

We link our ML model with the backend using the ``escapeshellcmd`` and ``shell_exec`` functions. These functions allow us to execute the model seamlessly, providing several benefits:

- 1. Separation of Concerns:** The machine learning logic is encapsulated in Python scripts, allowing the backend to focus on application logic and data handling.
- 2. Flexibility:** The backend can easily call different scripts for different types of analyses (text or URL-based) without modifying the core application logic.
- 3. Security:** Using ``escapeshellcmd`` helps prevent command injection attacks, enhancing the security of the system.

5.1.3 Dataset

Arabic 100k Reviews Dataset: The dataset combines reviews from hotels, books, movies, products and a few airlines. It has three classes (Mixed, Negative and Positive). Most were mapped from reviewers' ratings with 3 being mixed, above 3 positive and below 3 negatives. Each row has a label and text. We dropped mixed reviews to make better analysis. Text (reviews) were cleaned by removing Arabic diacritics and non-Arabic characters. The dataset has no duplicate reviews.

	label	text	
0	Positive	...ممتاز نوعا ما . النظافة والموقع والتجهيز والشا	
1	Positive	...أحد أسباب نجاح الإمارات أن كل شخص في هذه الدول	
2	Positive	...هاتفه .. وقوية. تتفكك من صخب شوارع القاهرة الى	
3	Positive	...خلصنا .. مبدئيا التي مستلي ابهار زي الفيل الاز	
4	Positive	...ياسات جلوريا جزء لا يتجزأ من دبي . فندق متكامل	
...	
99994	Negative	...معرفش ليه كنت عاوزة أكملها وهي مش عاجباني من ا	
99995	Negative	...لا يستحق ان يكون في بوكنت لانه سيئ . لا شي. لا	
99996	Negative	...كتاب ضعيف جدا ولم استمتع به. في كل قصه سرد لحا	
99997	Negative	...مملة جدا. محمد حسن علوان فنان بالكلمات، والوصف	
99998	Negative	...لن ارجع إليه مرة أخرى . قريه من البحر. المكان	

66666 rows x 2 columns

Figure 20.. Sample of Arabic 100k Reviews Dataset

The dataset is balanced and mainly a compilation of several available datasets and a sampling of 100k rows and we get it from Kaggle. The hotels and book reviews are a subset of [HARD] and BRAD. The rest were selected from [hadyelsahar](#) with a little over 100 airlines reviews collected manually.

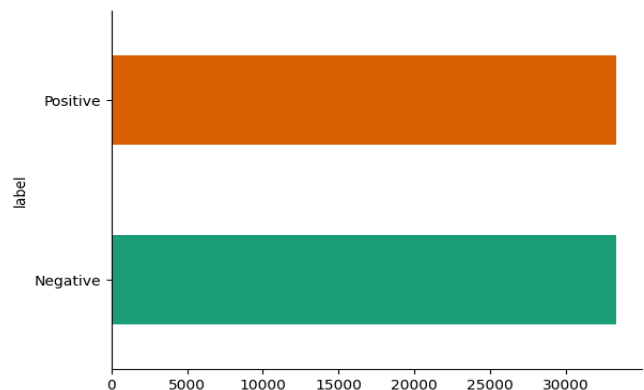


Figure 21.. Bar Chart of Dataset Polarity Count

5.1.4 Data preparation and Preprocessing

In any machine learning project, data preparation and preprocessing are critical steps to ensure the quality and relevance of the data fed into the model. For this project, which focuses on sentiment analysis of Arabic text, we implemented a comprehensive preprocessing pipeline to clean and standardize the text data. This pipeline involves several key steps: removing emojis, links, mentions, and hashtags; eliminating duplicate letters and

English words; and normalizing the text by removing diacritics and unnecessary punctuations. We utilized various libraries such as nltk for natural language processing tasks and pyarabic for handling Arabic-specific text operations. The following code snippet outlines the functions used in our preprocessing pipeline, ensuring that our dataset is refined and ready for feature extraction and model training.

```
1- import pyarabic.araby as araby
2- import nltk
3- nltk.download('stopwords')
4- nltk.download('punkt')
5- from nltk.corpus import stopwords
6- from nltk.tokenize import word_tokenize
7-
8-
9- def remove_emoji_links_mentions_hashtags(text):
10-     # Remove emojis
11-     emoticon_pattern = r"[\U0001F600-\U0001F64F" \
12-         r"\U0001F300-\U0001F5FF" \
13-         r"\U0001F680-\U0001F6FF" \
14-         r"\U0001F700-\U0001F77F" \
15-         r"\U0001F780-\U0001F7FF" \
16-         r"\U0001F800-\U0001F8FF" \
17-         r"\U0001F900-\U0001F9FF" \
18-         r"\U0001FA00-\U0001FA6F" \
19-         r"\U0001FA70-\U0001FAFF" \
20-         r"\U00002702-\U000027B0" \
21-         r"\U000024C2-\U0001F251]"+
22-
23-     text = re.sub(emoticon_pattern, '', text)
24-
25-     # Remove links
26-     text = re.sub(r'http\S+', '', text)
27-
28-     # Remove mentions
29-     text = re.sub(r'@[^\s@]+', '', text)
30-
31-     # Remove hashtags
32-     text = re.sub(r'#[^\s#]+', '', text)
33-
34-     # Remove duplicate letters
35-     text = re.sub(r'(\w)\1{2,}', r'\1', text)
36-
37-     # Remove English letters
38-     text = re.sub(r'[a-zA-Z]', '', text)
```

```

42- #Remove English words
43- text = re.sub(r'\b[a-zA-Z]+\b', '', text)
44-
45- #Remove consecutive duplicate words
46- text = re.sub(r'\b(\w+) (?:\s+\1\b)+', r'\1', text)
47-
48- #Remove longation
49- text = re.sub("[īīī!]", "!", text)
50- text = re.sub("ى", "ي", text)
51- text = re.sub("ؤ", "ء", text)
52- text = re.sub("ئ", "ء", text)
53- text = re.sub("ة", "ه", text)
54- text = re.sub("گ", "ك", text)
55-
56- return text.strip()
57-
58-
59- # Remove Tashkeel
60- def remove_diacritics(text):
61-     return araby.strip_diacritics(text)
62-
63- def clean_text(text):
64-     text = remove_emoji_links_mentions_hashtags(text)
65-     #text = remove_diacritics(text)
66-     text = "".join([word for word in text if word not in
67- punctuations_list])
68-     tokens = word_tokenize(text)
69-     text = ' '.join([word for word in tokens if word not in
70- arabicStopWords])
71-     return text
72-
73-

```

Figure 22.. Arabic Cleaning Function

5.1.4.1 Cleaning phase in Arabic

Here we do some steps to clean Arabic text.

- **Emoji, link, hashtag, and mentions removal**

Ensuring that the data doesn't have Emoji, link, hashtag, and mentions before making sentiment analysis.

- **English words and letters removal**

For sentiment analysis specifically in the Arabic language. The primary objective is to eliminate English letters and words that are irrelevant to the sentiment analysis task, ensuring that the focus remains on the Arabic content.

- **Consecutive duplicate words and letters removal**
It does not affect the validity of the sentiment analysis task.
- **Normalization**
Data Normalization: Ensuring the uniformity of Reviews is vital. Some words may have more than one acceptable spelling or discretization which, unless normalized, would result in redundant evaluation of the same word. Therefore, it is crucial to normalize reviews, so that all the words exist in a single form. This is done by removing all diacritics, normalizing Alef, Yaa, Taa Marboutah and Hamza characters, replacing multi-word sentiment terms.
- **Diacritics removal (Tashkeel)**
All diacritics and decoration are removed from the reviews.
- **Punctuations removal**
Remove special characters that are irrelevant to the sentiment analysis task.
- **Stop Words removal**
Stop words often appear frequently in the text and may not contribute significantly to the overall sentiment. By removing them, the noise level in the dataset decreases, enabling sentiment analysis algorithms to focus on more meaningful words and sentiment indicators.

5.1.4.2 Stemming

Stemming is a process used in Arabic sentiment analysis (and other natural language processing tasks) to reduce words to their root or stem form. The goal of stemming is to normalize words by removing prefixes, suffixes, and other morphological variations, so that words with similar meanings can be treated as the same base form. This process helps in reducing the vocabulary size, improving computational efficiency, and enhancing the accuracy of sentiment analysis algorithms.

In the context of Arabic sentiment analysis, stemming can be challenging due to the rich morphology and complex word patterns of the Arabic language. Arabic words can have different prefixes, suffixes, and infixes, indicating various grammatical aspects such as tense, gender, and plurality.

We used nltk.ISRIStemmer, it is a module within the Natural Language Toolkit (NLTK) specifically designed for stemming Arabic words, follows

an algorithm tailored to the Arabic language, addressing its unique morphological structure. It processes words by stripping common prefixes and suffixes, normalizing certain characters, and handling specific Arabic language patterns.

```
1- from nltk.tokenize import word_tokenize
2- import nltk
3-
4- def process_text(text):
5-     stemmer = nltk.ISRIStemmer()
6-     word_list = nltk.word_tokenize(text)
7-     #stemming
8-     word_list = [stemmer.stem(w) for w in word_list]
9-     return ' '.join(word_list)
10-
```

Figure 23.. Text Preprocessing

5.1.4.3 Word Embedding and Feature Extraction

In natural language processing, transforming textual data into numerical representations is essential for machine learning algorithms to process and analyze the data effectively. I initially experimented with Term Frequency-Inverse Document Frequency (TF-IDF) for feature extraction. However, due to the vast number of unique words in my dataset of 100,000 reviews, TF-IDF did not perform well. The high dimensionality and sparsity of the resulting vectors posed significant challenges for the model's performance.

To address these issues, I adopted Word2Vec, a more advanced word embedding technique. Word2Vec transforms words into dense vectors of fixed size, capturing semantic meanings and relationships between words. This method is particularly effective for large datasets and languages with rich morphology like Arabic.

Word2Vec uses a neural network-based approach to learn word representations. There are two primary architectures for Word2Vec:

- Continuous Bag of Words (CBOW): Predicts a target word based on the context of surrounding words.
- Skip-Gram: Predicts the context words given a target word.

For our project, we used the Skip-Gram model, which is more effective for capturing rare words and detailed semantic relationships.


```

1- from gensim.models import Word2Vec
2- from nltk.tokenize import word_tokenize
3-
4- df['tokens'] = df['cleanedtextnew'].apply(word_tokenize)
5- # train the model
6- word2vec_model = Word2Vec(sentences = df['tokens'], vector_size=100,
7- window=5, min_count=2 ,workers=4,sg=1)
8- # get word embeddings from word2vec
9- word_embeddings = word2vec_model.wv
10-
11-
12- document_embeddings = []
13- for tokens in df['tokens']:
14-     # Filter tokens that have embeddings
15-     tokens_with_embeddings = [token for token in tokens if token in
16- word_embeddings]
17-
18-
19-     # Calculate the mean of embeddings for tokens with embeddings
20-     if tokens_with_embeddings:
21-         doc_embedding = np.mean([word_embeddings[token] for token in
22- tokens_with_embeddings], axis=0)
23-     else:
24-         # If all tokens are missing embeddings, assign a zero vector
25-         doc_embedding = np.zeros_like(word_embeddings.vectors[0])
26-
27-
28-     # Append the document embedding to the list
29-     document_embeddings.append(doc_embedding)
30-

```

Figure 24.. Document and Word Embedding

5.1.5 Machine Learning Models and Algorithms

First, we made a function to evaluate the performance of a trained machine learning model on a test dataset and visualize the results using a confusion matrix.

Here is the function for calculating the accuracy of the models:

```

1- from sklearn.metrics import
2- confusion_matrix, accuracy_score, precision_score, recall_score,
3- f1_score
4- import seaborn as sns
5- import matplotlib.pyplot as plt
6-
7-
8- def get_accuracy(name, trained_model , x_test, y_test):
9-     tree_predict = trained_model.predict(x_test)
10-     print("Testing
11- accuracy      :", metrics.accuracy_score(y_test,
12- tree_predict)*100 , "%")
13-     print("precision : ", precision_score(y_test,
14- tree_predict, average='micro'))
15-     print("recall      : ", recall_score(y_test,
16- tree_predict, average='micro'))
17-     print("f1_score    : ", f1_score(y_test,
18- tree_predict, average='micro'))
19-
20-
21-
22-     cfl = confusion_matrix(y_test, tree_predict)
23-     sns.heatmap(cfl, annot=True, fmt = '.0f')
24-     plt.xlabel('prediction')
25-     plt.ylabel('Actual')
26-     plt.title(name+ ' Confusion Matrix')
27-     plt.show()
28-
29-

```

Logistic regression model:

We tried logistic regression algorithm to make sentiment analysis classifier that predicts the sentiment based on estimation is made by applying binary classification on the data allocated to training and test data. and convert review into numerical for using Word2Vec vectorizer.

```

1- from sklearn.linear_model import LogisticRegression
2-
3-
4- clf=LogisticRegression()
5-
6- trmodel=clf.fit(x_train,y_train)
7-
8- y_pred=clf.predict(x_test)
9-
10- get_accuracy("LogisticRegression",trmodel,x_test,y_test)

```

Figure 25.. Logistic Regression Model

Here the classification report of Logistic regression model:

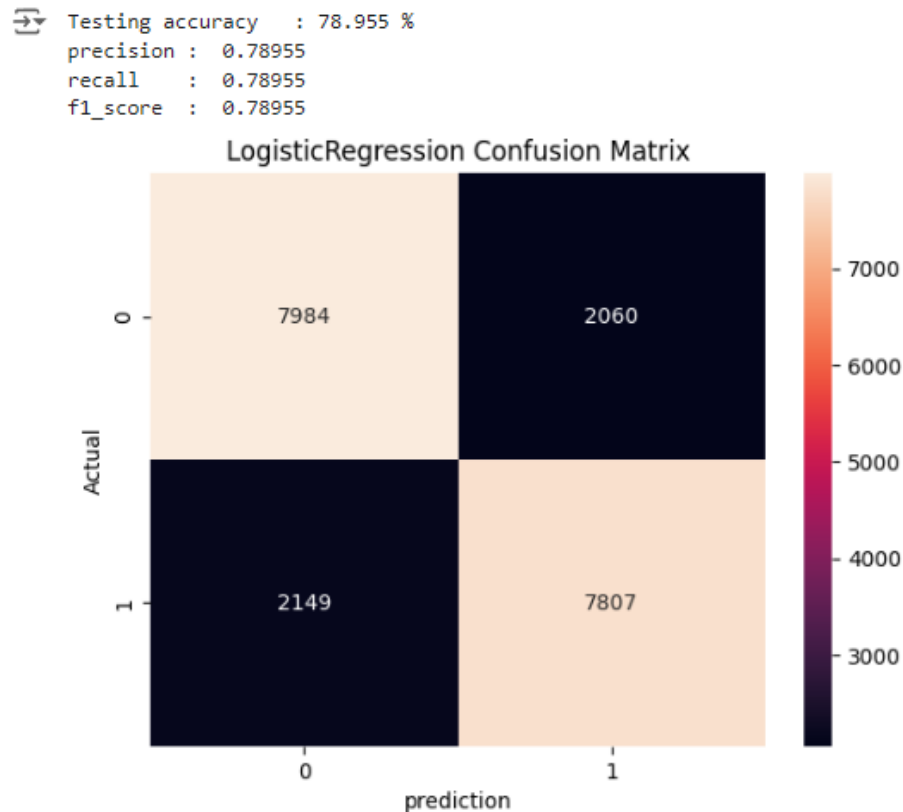


Figure 26.. classification report of Logistic regression model

Support Vector Machine (SVM) model:

Support Vector Machine is a popular machine algorithm tried for sentiment analysis in our dataset in Arabic. We use it and make feature extraction which converts the preprocessed text data into numerical features with Word2Vec.

```
1- from sklearn.svm import LinearSVC
2-
3- svm = LinearSVC()
4- trmodel=svm.fit(x_train,y_train)
5- get_accuracy("LogisticRegression",trmodel,x_test,y_test)
6-
```

Figure 27.. SVM Model

Here is the classification report of SVM model:

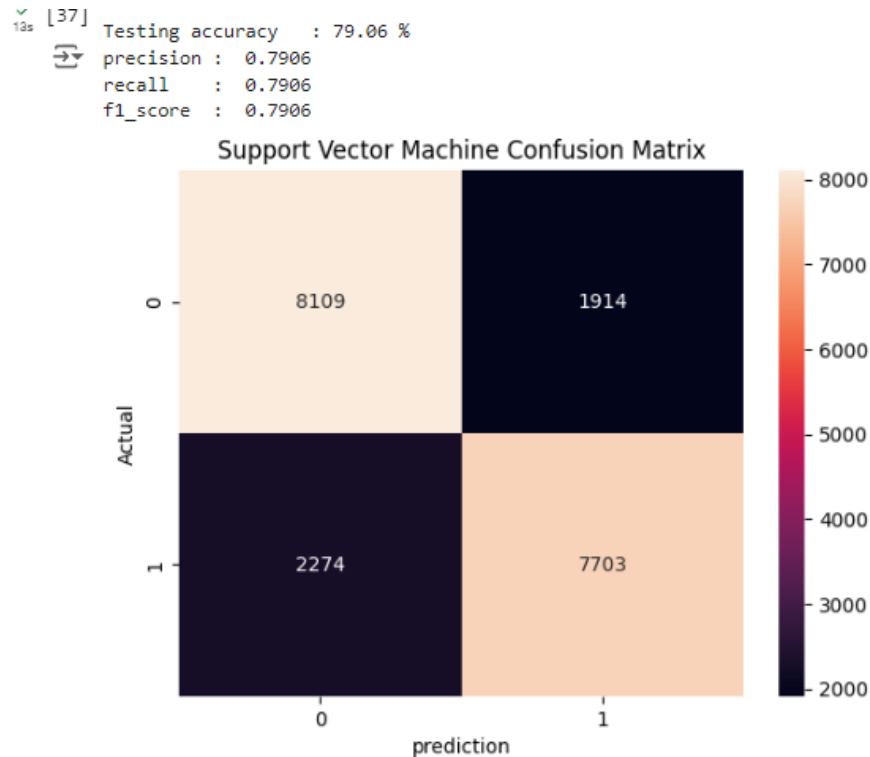


Figure 28.. Classification report of SVM model

Decision tree model:

Decision trees can be used for classification tasks. It can handle different data types and discrete or continuous values, and continuous values can be converted into categorical values using thresholds. It can also handle missing values.

```
1- from sklearn.tree import DecisionTreeClassifier
2-
3-
4- dt_classifier = DecisionTreeClassifier(criterion="entropy")
5- trmodel=dt_classifier.fit(x_train,y_train)
6- get_accuracy("DecisionTreeClassifier",trmodel,x_test,y_test)
```

Figure 29.. Decision Tree Model

Here is the classification report of Decision Tree Model:

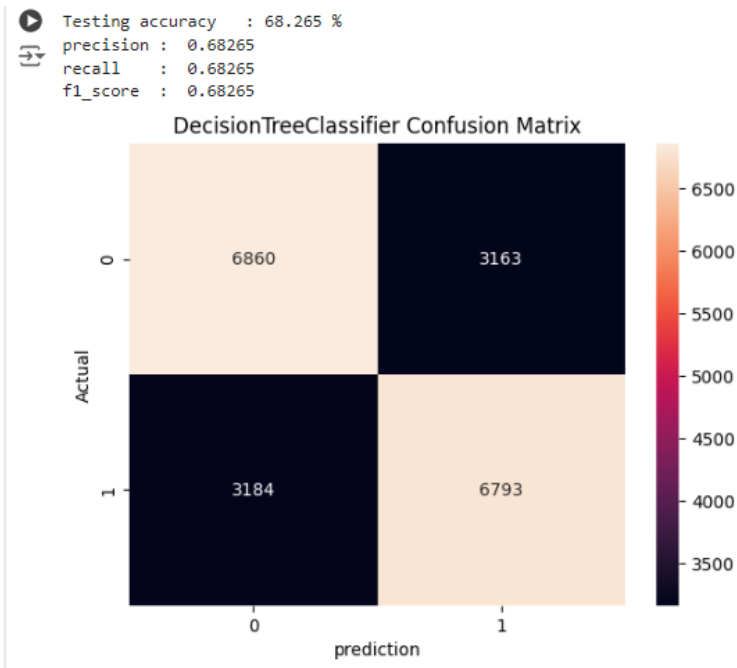


Figure 30.. Classification report of Decision Tree Model

KNN model:

We tried K-nearest neighbors to make sentiment analysis classifier that predicts the sentiment based on the similar reviews in data of input with odd number $k=7$ and even number $k=10$.

```
1- from sklearn.neighbors import KNeighborsClassifier
2-
3-
4- knn = KNeighborsClassifier(n_neighbors=7)
5- trmodel=knn.fit(x_train,y_train)
6- get_accuracy("KNeighborsClassifier With k=7",trmodel,x_test,y_test)
```

Figure 31.. KNN Model

Here is the classification report of KNN Model with $k = 7$:

```
Testing accuracy : 76.23 %  
precision : 0.7623  
recall : 0.7623  
f1_score : 0.7623
```

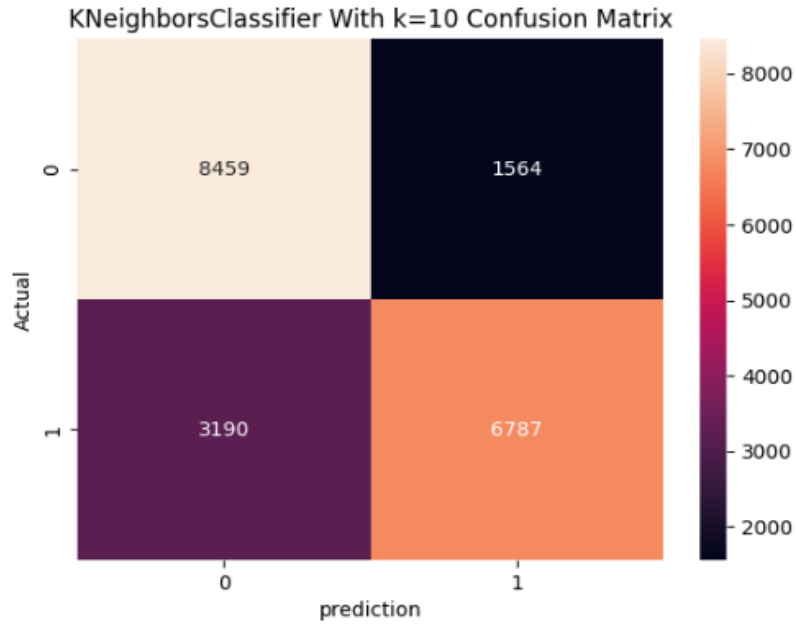


Figure 32.. Classification report of KNN Model with $k = 7$

And here is the classification report of KNN Model with $k = 10$:

```
Testing accuracy : 76.53 %  
precision : 0.7653  
recall : 0.7653  
f1_score : 0.7653
```

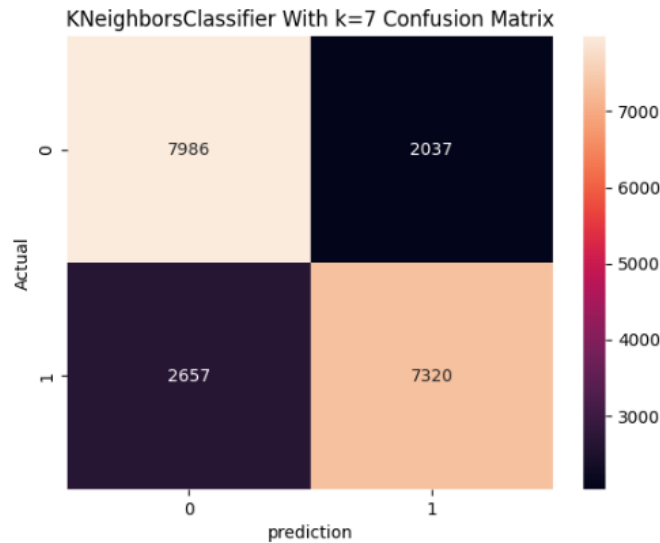


Figure 33.. Classification report of KNN Model with $k = 10$

Random Forest model:

Random Forest is another robust machine learning algorithm we employed for sentiment analysis on the dataset. This algorithm constructs multiple decision trees during training and outputs the mode of the classes (classification) of the individual trees for prediction.

```
1- from sklearn.ensemble import RandomForestClassifier
2-
3-
4- model = RandomForestClassifier()
5-
6- trmodel=model.fit(x_train,y_train)
get_accuracy("KNeighborsClassifier With k=10",trmodel,x_test,y_test)
```

Figure 34.. Random Forest Model

Here the classification report of Random Forest model:

```
Testing accuracy : 79.14999999999999 %
precision : 0.7915
recall : 0.7915
f1_score : 0.7915
```

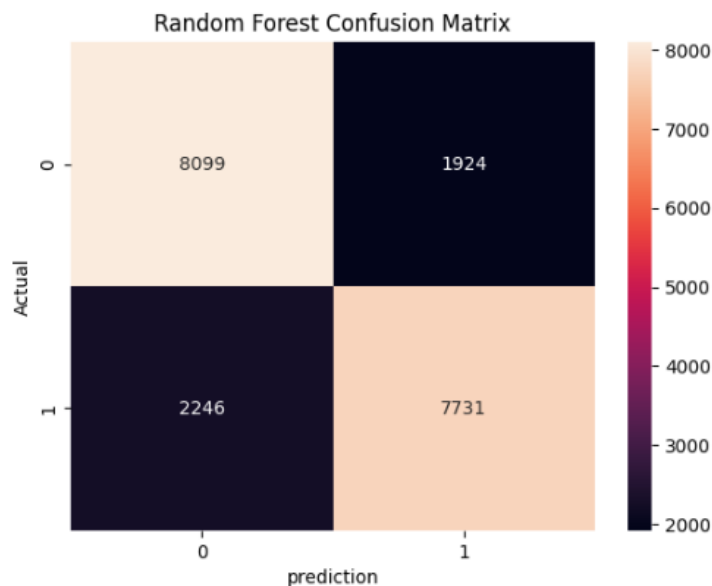


Figure 35.. Classification report of Random Forest model

Multi-Layer Perceptron model:

Multi-Layer Perceptron Classifier is a type of artificial neural network that we employed for sentiment analysis on the dataset. This algorithm is well-suited for complex datasets and can capture non-linear relationships.

```

1- from sklearn.neural_network import MLPClassifier
2-
3-
4- ann1 = MLPClassifier(
5-     hidden_layer_sizes=(50,),
6-     learning_rate_init=0.01,
7-     batch_size=128,
8-     max_iter=500,
9-     random_state=100
10- )
11-
12- trmodel=ann1.fit(x_train,y_train)
13-
14- get_accuracy("ANN",trmodel,x_test,y_test)

```

Figure 36.. Multi-Layer Perceptron Model

Here the classification report of MLP model:

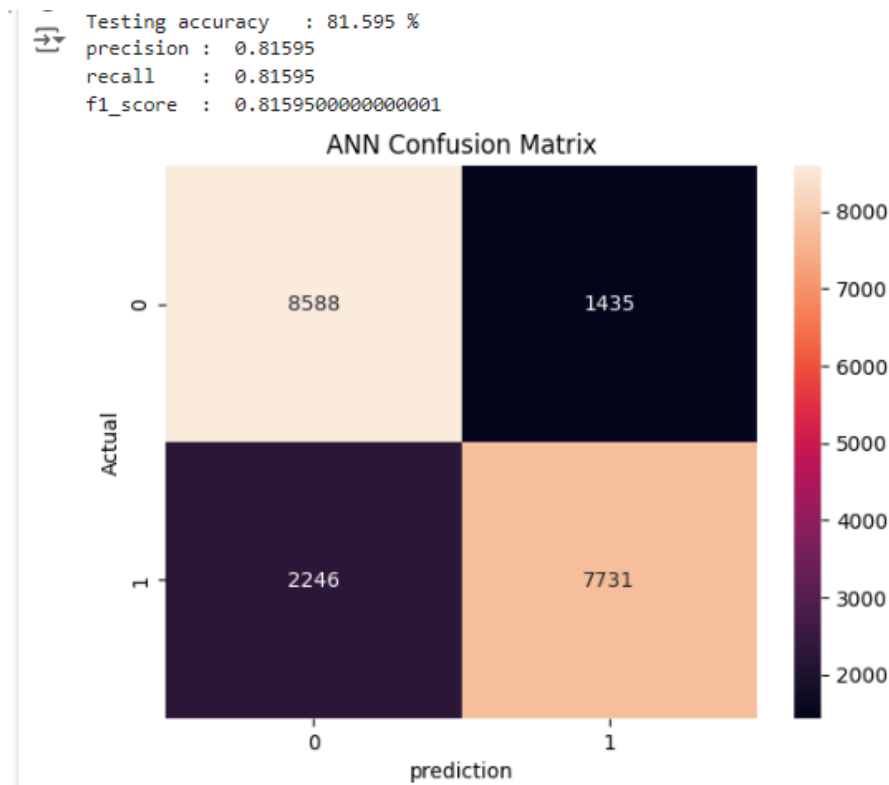


Figure 37.. Classification report of MLP Model

RNN Models:

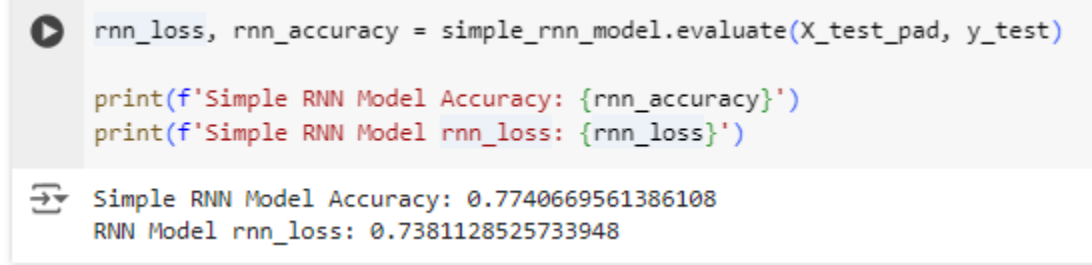
RNN models are commonly used for sequence- to-sequence modeling, where the input and output are both sequences of variable length. In the context of sentiment analysis, the sequence of variable length. In the context of sentiment analysis, the input sequence and the output are sentiment labels.

Here is an example of LSTM model we tried it.

```
1- from tensorflow.keras.layers import Embedding, SimpleRNN, LSTM,
2- Dense, Dropout
3-
4- def train_model(model_type, X_train, y_train, X_test, y_test):
5-     model = Sequential()
6-     model.add(Embedding(vocab_size, 128, input_length=maxlen))
7-
8-     if model_type == 'SimpleRNN':
9-         model.add(SimpleRNN(128, return_sequences=False))
10-    elif model_type == 'LSTM':
11-        model.add(LSTM(128, return_sequences=False))
12-
13-
14-    model.add(Dropout(0.5))
15-
16-    model.add(Dense(3, activation='softmax'))
17-
18-
19-    model.compile(optimizer='adam',
20- loss='sparse_categorical_crossentropy', metrics=['accuracy'])
21-
22-    model.fit(X_train, y_train, epochs=10, batch_size=1024,
23- validation_data=(X_test, y_test))
24-
25-    return model
26-
27- simple_rnn_model = train_model('SimpleRNN', X_train_pad, y_train)
28-
29- LSTM_rnn_model = train_model('LSTM', X_train_pad, y_train)
```

Figure 38.. LSTM Model

Here the classification report of RNN models:

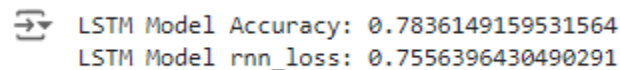


```
rnn_loss, rnn_accuracy = simple_rnn_model.evaluate(X_test_pad, y_test)

print(f'Simple RNN Model Accuracy: {rnn_accuracy}')
print(f'Simple RNN Model rnn_loss: {rnn_loss}')
```

Simple RNN Model Accuracy: 0.7740669561386108
RNN Model rnn_loss: 0.7381128525733948

Figure 39.. RNN Model Accuracy



```
LSTM Model Accuracy: 0.7836149159531564  
LSTM Model rnn_loss: 0.7556396430490291
```

Figure 40.. LSTM Model Accuracy

5.1.6 Facebook and Instagram Scrapers

Instaloader and facebook-scraper are two popular Python libraries designed for scraping Instagram and Facebook, respectively. They offer specific functionalities tailored to extracting data from these social media platforms while handling some of the complexities involved in web scraping.

- To perform sentiment analysis on user comments from Instagram and Facebook, we developed custom scrapers using Instaloader for Instagram and facebook_scraper for Facebook. These scrapers authenticate using user credentials or cookies and extract comments from specified posts. The extracted data is then processed and analyzed for sentiment using pre-trained models.
- The scrapers extract the short code or post ID from the URL and fetch the post comments and the post publisher username.
- The scrapers are designed to fetch all available comments, without any predefined limit.
- As the number of comments increases, the response time for the scraping process also increases. This is due to the additional time required to fetch and process a larger volume of data.

Here is an example for scraping an Instagram post:



Figure 42.. Instagram Post

And
here
is an



Figure 41.. Scraped Comments and Caption of the Instagram Post

example for scraping a Facebook post:



Figure 43.. Facebook Post



Figure 44.. Scraped Comments and Caption of the Facebook Post

5.2 Testing

5.2.1 System Testing

- **Test case 1:** Analyze Arabic Text

When the user inputs an Arabic Text, and click on Analyze, it shows the positive and negative score of the text.



Figure 45.. Testing when a user enters Arabic text

- **Test case 2:** Testing Non-Arabic Review

When the user inputs a non-Arabic text, the “Please Enter Arabic Text Only” will be displayed

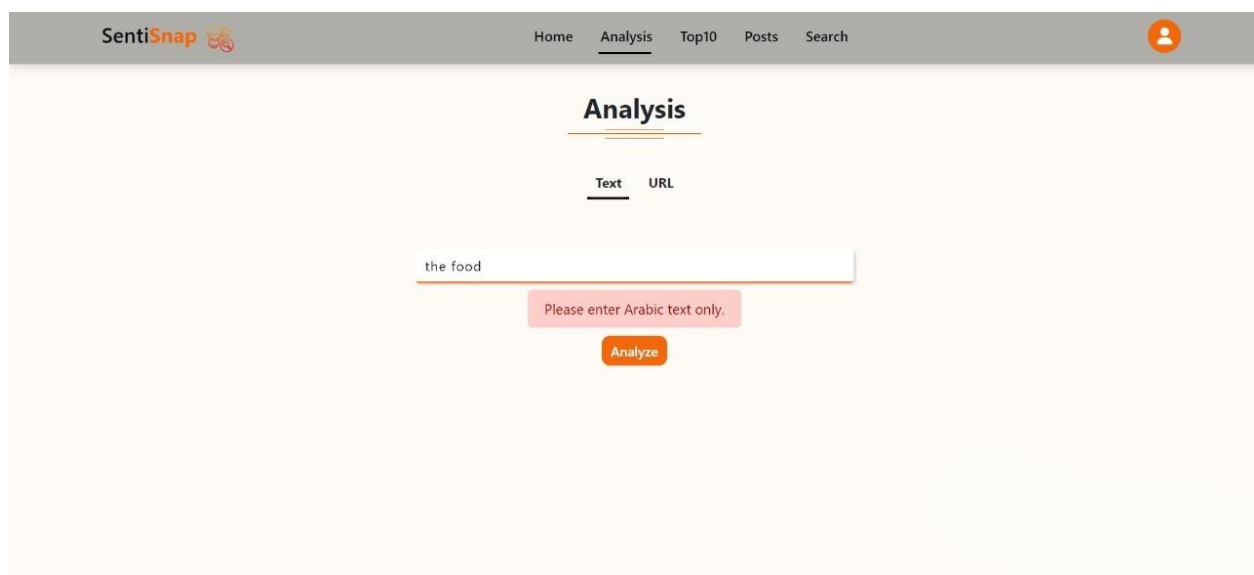


Figure 46.. Testing when user enters Non-Arabic Text

- **Test case 3: Analyzing Facebook or Instagram URL**

When the user inputs a facebook or instagram link to get its analysis, it displays the positive and negative score of all the comments on the post, and the user can share the analysis result.

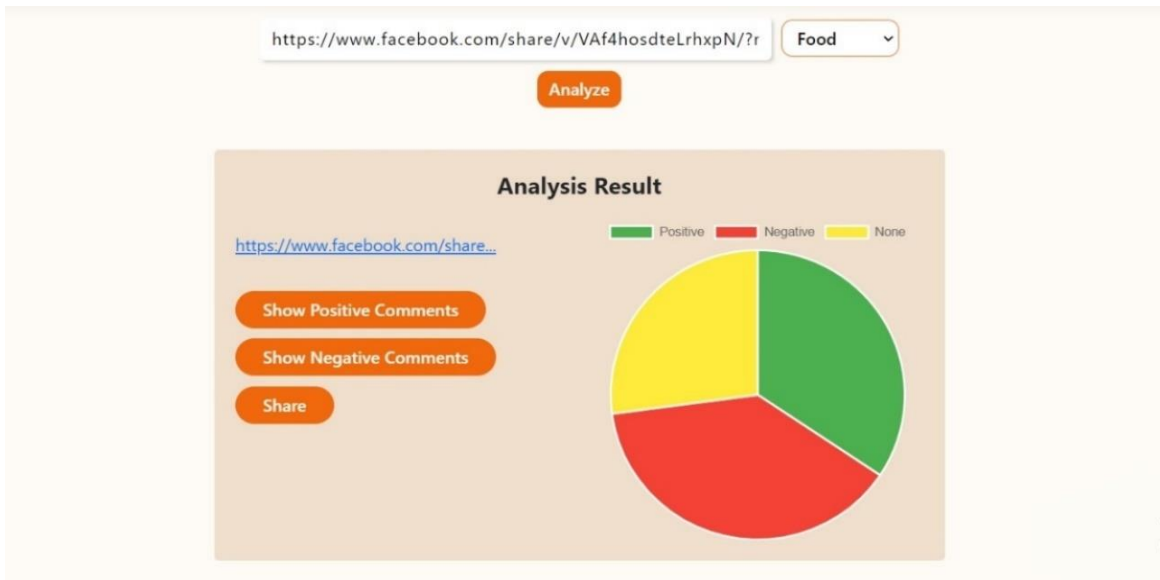


Figure 47.. Analyzing Facebook or Instagram URL

- **Test case 4: Show Positive Comments**



Figure 48.. Positive Comments Scraped

- **Test case 5: Show Negative Comments**



Figure 49.. Negative Comments Scraped

- **Test case 6: Testing Invalid URL (not facebook or instagram)**

When the user inputs an invalid link, the “**Please Enter a Valid Facebook or URL Link**” error will be displayed

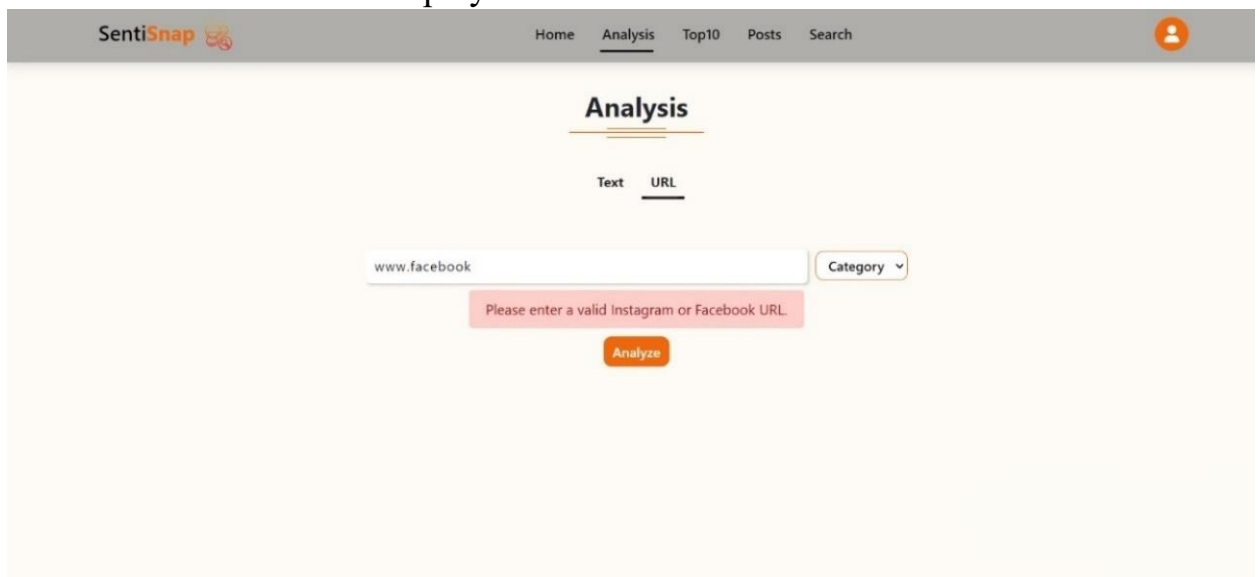


Figure 50.. Testing when user enters Invalid URL

- **Test case 7: Testing Post of Inaccessible Account (Private Account)**

When the user inputs a link of Instagram or Facebook post, if the post's account is private, or if it's a private group, the “**Unable to scrape the post. It may be private or inaccessible**” error will be displayed

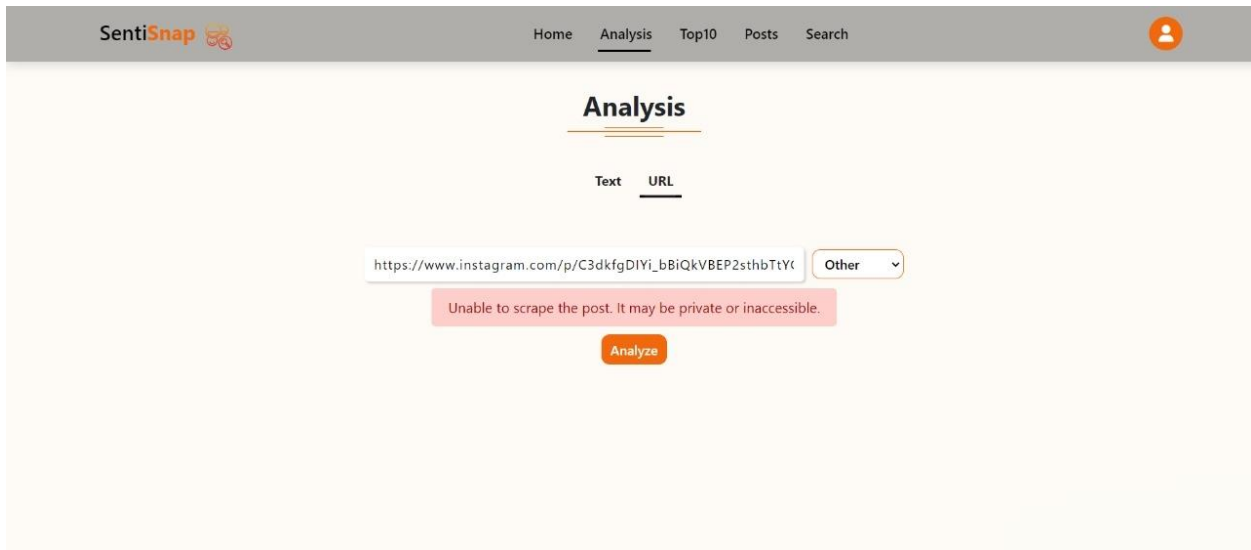


Figure 51.. Testing Link of a Post in a Private Account

- **Test case 8: Testing Search for a Word**

When the user searches for a word, it searches in the analyzed posts' captions, or in the page name

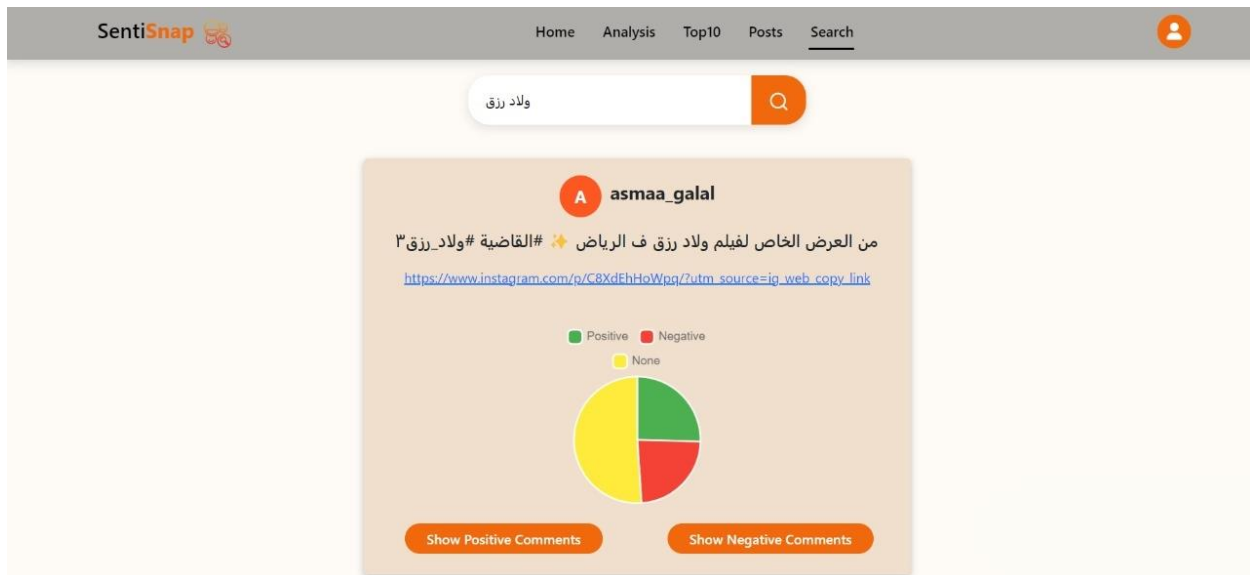


Figure 52.. Testing Search for a Word

- **Test Case 9: Testing Sharing a Post without Title (will not be shared)**

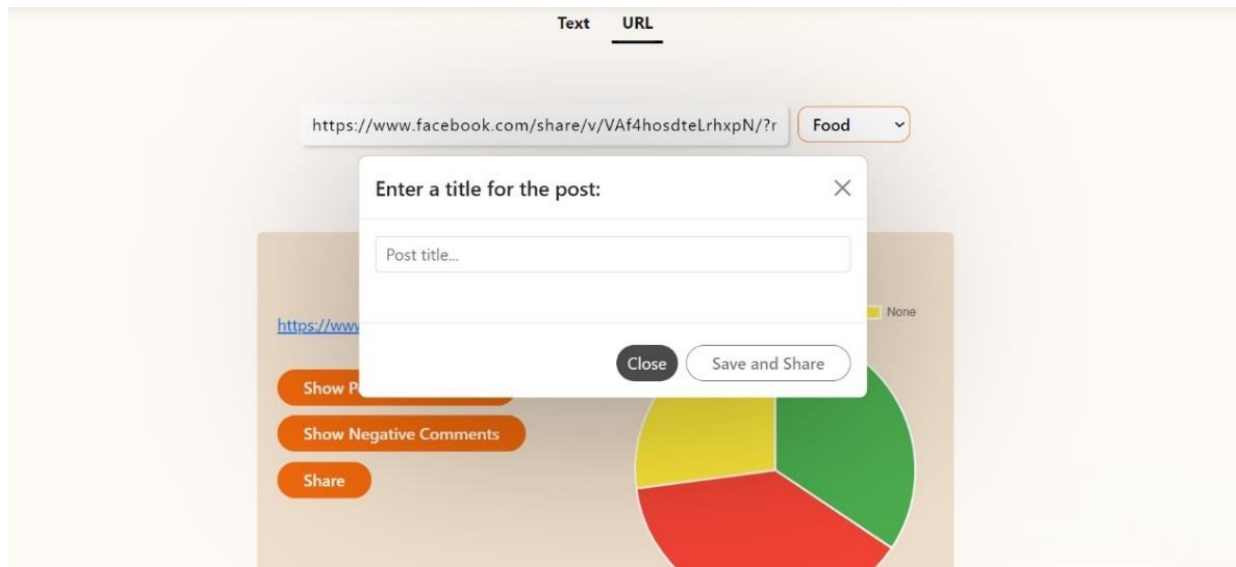


Figure 53.. Testing Sharing a Post without a Title

5.2.2 Integration Testing

In our SentiSnap project, we ran the integration test for each API as follows using Postman, here is a sample of our API results using Postman:

- Register API:

When registering a new account, if this account doesn't exist in the database, it'll register successfully.

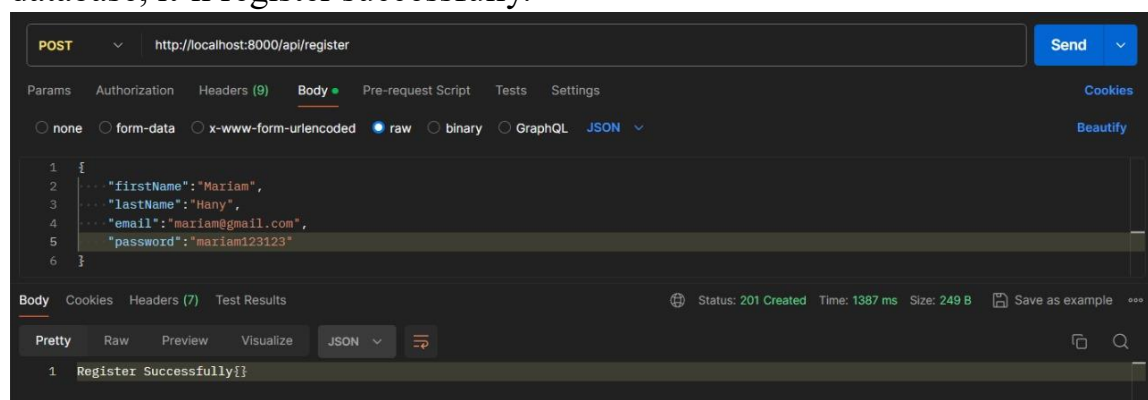


Figure 54.. Register API when register successfully

If the user's email is found in the database, it'll return "Email exists, enter another email"

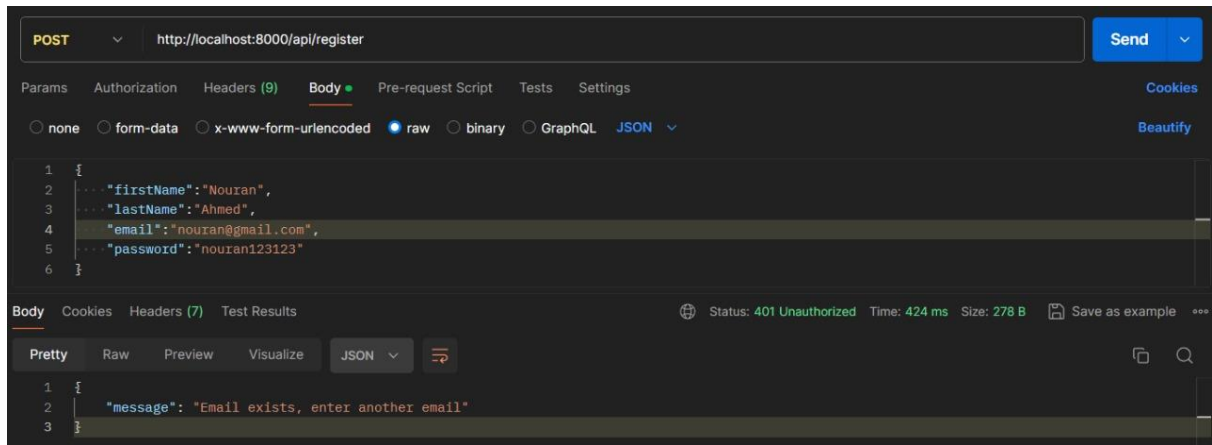


Figure 55.. Register API when email exists

- Signin API:

If the user enters the wrong email or password when signing in, the API returns it with status 401 Unauthorized.

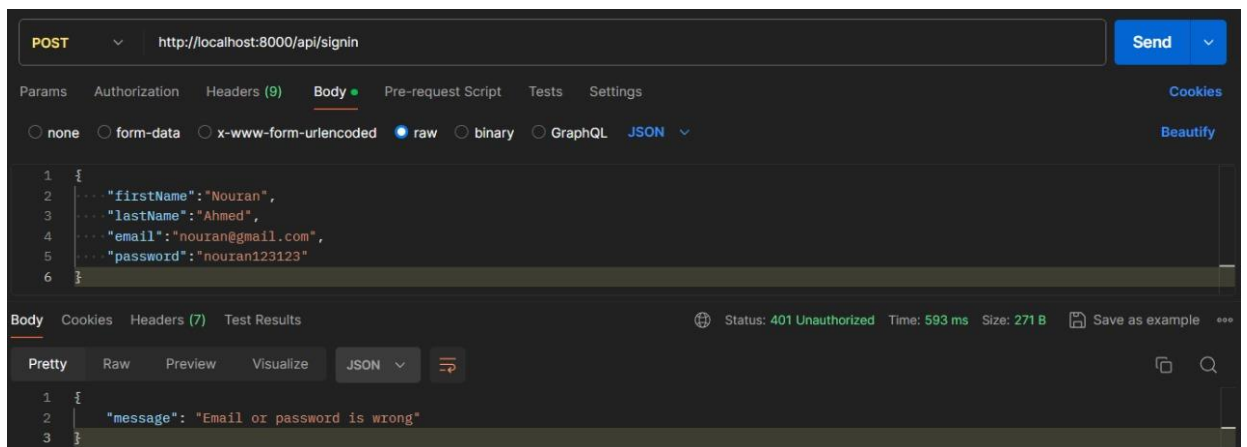


Figure 56.. Signin API when user enters wrong email or pass

If the user is signed up, when he sign in the API returns “User logged in Successfully”

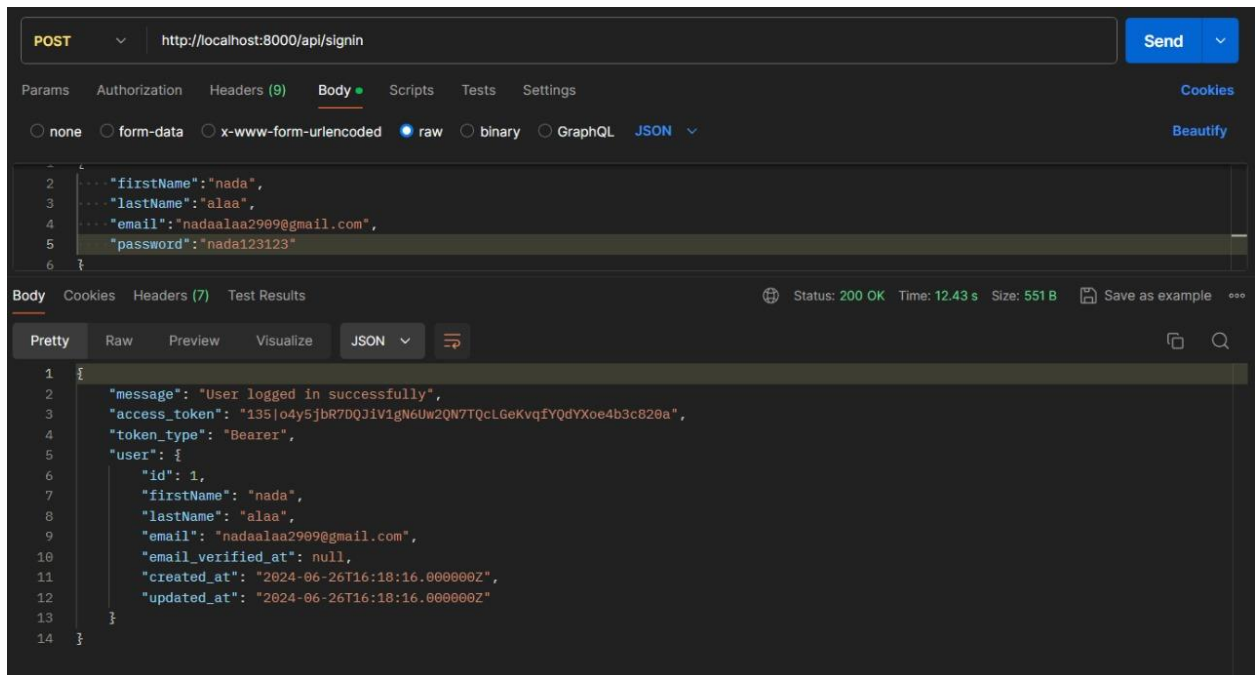


Figure 57.. Signin API when the user enters a correct email and pass

- Search API:

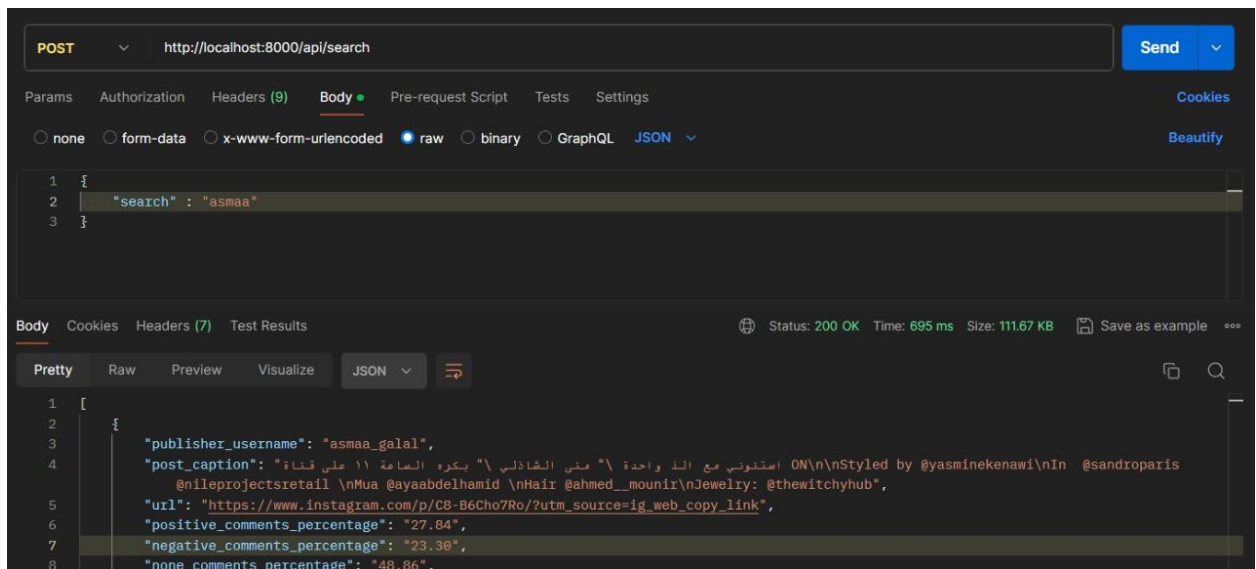


Figure 58.. Search API

When the user searches with anything, the API returns the analyzed posts that has what the user searched with.

References

- [1] [Sentiment Analysis - NLP using Machine Learning \(youtube.com\)](#) [تطبيق عملي: معالجة اللغات الطبيعية - تحليل المشاعر](#)
- [2] [Methods for Calculating Sentiment Score for Text | Analytics Vidhya](#)
- [3] [Arabic 100k Reviews \(kaggle.com\)](#)
- [4] [Design Patterns \(refactoring.guru\)](#)
- [5] [\(PDF\) Sentiment Analysis for Arabic Reviews in Social Networks Using Machine Learning \(researchgate.net\)](#)
- [6] [arabic companies reviews sentiment analysis | Kaggle](#)
- [7] N. Mahmoud, A. Mohamed and Amir Atiya, "Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing," in Association for Computational Linguistics, Lisbon, Portugal, 2015.
- [8] M. Hammad and M. Al-awadi, "Sentiment Analysis for Arabic Reviews in Social Networks Using Machine Learning," in Information Technolog: New Generations, 2016, pp. 131-139.
- [9] N. Abdulla, M. Al-Ayyoub and M. N. Al-Kabi, "An extended analytical study of Arabic sentiments," International Journal of Big Data Intelligence, pp. 103-113, 2014.
- [10] Inoue, G. a. Alhafni, B. a. Baimukan, N. a. Bouamor, H. a. Habash and Nizar, "The Interplay of Variant, Size, and Task Type in Arabic Pre-trained Language Models," in Proceedings of the Sixth Arabic Natural Language Processing Workshop, Association for Computational Linguistics, 2021.