OOAD Project

Byte Engine

Our Search Engine

Submitted for partial fulfilment of the requirements of the course
CS6359 - Object Oriented Analysis and Design

by

Sravya Mareedu	**sxm210171**
Hitesh Atluri	**hxa210029**
Monisha Ramu	**mxr210061**
Sai Tharun Reddy Garlapati	**sxg210161**
Tathya Thaker	**txt200018**
Dhruvi Sonani	**dxs210030**
Parth Sabhadiya	**pxs210002**
Kruti Marvania	**kxm200051**

Contents

List of Figures

Chapter 1

Introduction

1.1 Project Overview

This Project deals with the design and implementation of a powerful and feature-rich tool that optimizes our use case using a crawler-based search engine. Crawler-based means that the system utilizes information available in the web documents in a holistic manner to determine what might be interesting to the user. There are many general-purpose search engines. Hence, we took a particular domain. We focus on textual content that is written in a natural language as opposed to, say, images included in the documents. We call the presented system a search engine, as it contains components to retrieve and index web documents, and it provides a mechanism to return a ranked subset of the documents according to the user's requests. The system should be able to process millions of documents in a reasonable time and respond to queries with a low average latency.

1.2 Aim and Objective of the Project

The aim of this project work is to create a web-based search engine that yields quick and precise search results to any input related to the countries of the world that the user supplied in an environment that is very simple for user interaction. The method for retrieving information will be easy for non technical users. The objective of this research work is to :

1. Embed the system in an environment that the user is already familiar with.
2. To enable users to retrieve information with simple applications such as Web browsers.

1.3 Components of Crawler Based Search Engine

1. **Indexing :** Indexing is the process of looking at files, email messages, and other content on your PC and cataloging their information, such as the words and metadata in them. When you search your PC after indexing, it looks at an index of terms to find results faster.

2. **Crawling :** Crawling is the discovery process in which search engines send out a team of robots (known as crawlers or spiders) to find new and updated content. Content can vary — it could be a webpage, an image, a video, a PDF, etc.

3. **Ranking :** Ranking is the technique to assign the rank to the web pages based on how relevant the URLs are to the user's search query.

1.4 Functional Requirements

1. **Ranking or Ordering of Results :** The results must be ranked in the order of occurrence.

2. **Filter Stop Words :** The search engine should be able filter stop words like 'and' , 'or' to concentrate on key words.

3. **Auto-fill Support :** The search engine has to give suggestions using cache.

4. **Concurrent Operation :** The search engine has to be able to perform parallel processing.

5. **Number of Results per Page :** The search results should show certain number of results per page.

6. **Comparing Results :** The search engine compares results with Google and Microsoft.

1.5 Non-Functional Requirements

1. **User-Friendly** - The search engine should be user friendly.

2. **Responsive** with low latency - The search engine should give the results within a few milliseconds.

3. **Scalable**

4. **Reusable** with good performance.

5. **Adaptive** - Search Engine should be able to adapt any new changes in the system without affecting any other functionalities.

6. **Portable** - Search Engine should be able to run on any system.

7. **Concurrency** - More than one user should be able to access the search engine simultaneously.

1.6 Methods, tools, and techniques

The system is implemented in Python language. Our project team is using Microsoft Teams for easy team member communications and collaborations. Tools: We have made use of the following technology stack:

1. Python

2. Django
3. NLTK
4. HTML
5. CSS
6. Bootstrap
7. SQL
8. Apache Nutch
9. Apache Solr

1.7 Team Organization

All the team members have equally involved in the completion of project from the beginning. The total team is divided into three teams, i.e. Front End Team, Back End Team and Scrapping Team and every team member has taken leadership responsibility at least once in any phase of the project.

Phase II Leader : Dhruvi Sonani
Team Organization

Responsibility	**Team Members**	**Team Leader**
Front End	Dhruvi Sonani Kruti Marvania Monisha Ramu	Dhruvi Sonani
Back End	Parth Sabadhiya Tathya Thaker Sravya Mareedu	Parth Sabadhiya
Scrapping	Hitesh Atluri Sai Tharun Reddy	Hitesh Atluri

Apart from this, we have also had many team meetings to discuss the progress of the project and to suggest and implement changes. We have considered every suggestion and everyone's idea.

Date	Location	Summary	Participants	Meeting Leader
05/27/2022	Student Union	Introduced each other and decided on the languages to work on.	Whole Team	Hitesh
06/10/2022	MS Teams	Created Team Website and formed sub-groups(Front end, back end, Scraping team)	Whole Team	Tathya
06/15/2022	MS Teams	Decided on the contents of deliverables of phase 1 and divided work amongst us	Whole Team	Monisha
06/18/2022	Student Union	Discussed the suggestions given by the professor and made changes accordingly.	Whole Team	Sravya
06/28/2022	MS Teams	Finalized how the front end should be	Front End Team	Dhruvi
07/01/2022	MS Teams	Completed Connectivity to database	Scraping Team	Tharun
07/08/2022	MS Teams	Decided and collected data	Back End Team	Parth
07/12/2022	Student Union	Discussed about the improvements to be made to the UI	Whole Team	Kruti

1.8 Team Website

Link : https://iridescent-baklava-c51ff5.netlify.app/

Chapter 2

UML Diagrams

We have added some additional functional and non-functional needs based on the conversations and feedback we have received. The UML diagrams from our phase 1 submissions have also undergone revisions, and we have contributed some new diagrams.

2.1 The Big Picture

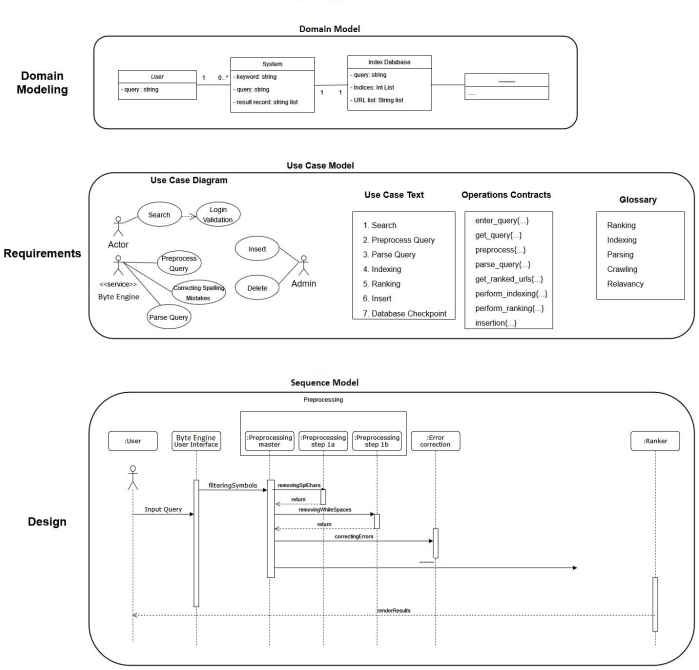

Figure 2.1: The Big Picture

Essentially, UML is visualizing software through diagrams. Figure 1 given above shows the big picture inspired by Craig Larman's big picture architecture. This picture has 3 phases – namely Domain Modeling, Requirements and Design.

2.2 The domain model

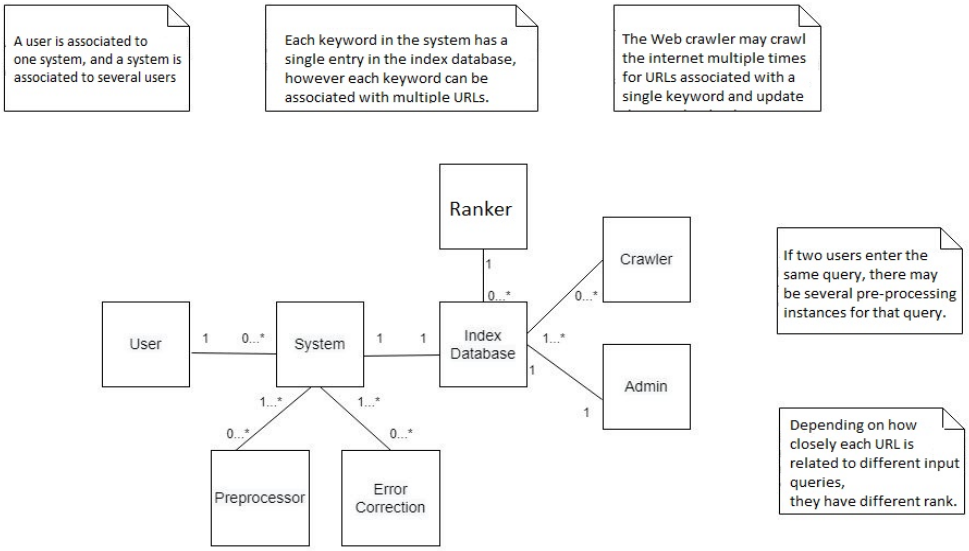

Figure 2.2: Domain Model

It is clear from the diagram that a user may be connected to several different systems. Many preprocessing and error-correction units can be found in a single system. Although each keyword in the system has only one item in the database, it is possible for a keyword to be linked to many URLs. At least one system must include a preprocessing unit and an error correcting unit. There is only one Index database per system. A ranker can be used with several index databases. The number of admins in an Index database is one. Links from numerous crawlers may be present in an Index database.

2.3 Use Case Diagram

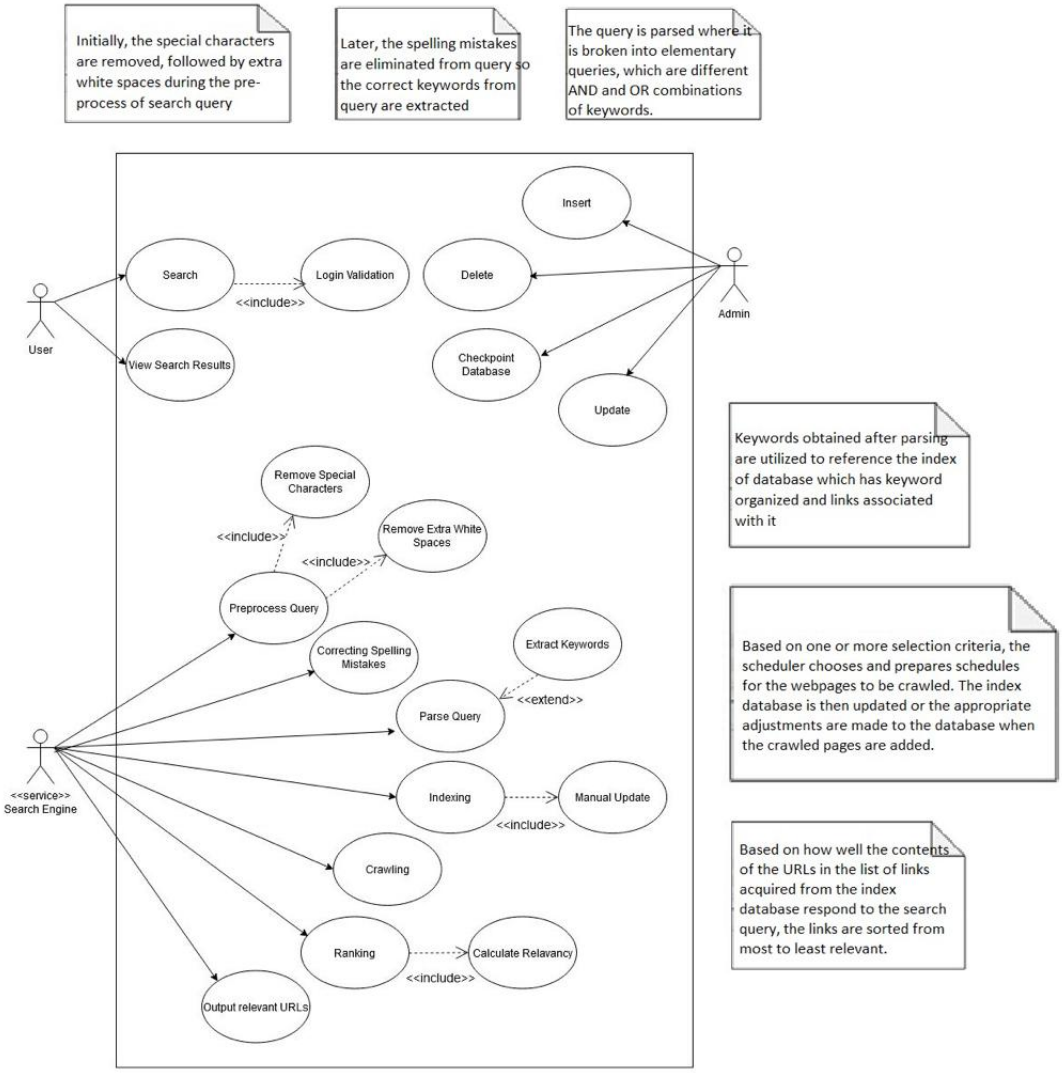

Figure 2.3: Use Case Diagram

The scope and high-level functions of a system are described in use-case diagrams. The interactions between the system and its actors are also depicted in these diagrams. Utilize-case diagrams show what the system does and how the actors use it, but they do not show how the system works within. The use case diagram has been revised to include admin actor functionalities as opposed to the use diagram presented in phase 1.

These features include database check pointing and the addition, deletion, and updating of URLs. The user can log into his account as well. The search feature consists of this login functionality. In order to distinguish its functions from those of the admin and other services, the search engine has been added as a separate service.

2.4 Use Case Template

Name	Search
ID	UC1
Description	Users enter input query for which result URLs will be displayed
Countries	User
Includes	Login Validation

Name	Insert URL
ID	UC4
Description	Admin inserts URLs manually into the index database
Countries	Admin

Name	Indexing
ID	UC6
Description	Index database indexes the result URL for easy access
Countries	Search Engine <<service>>
Excludes	Manual Update

Name	Ranking
ID	UC9
Description	Rank the indexed URLs for better search results
Countries	Search Engine <<service>>
Includes	Calculate Relevancy

Figure 2.4: Use Case Template

The major use cases that were put into practice during the project are described in detail in the use case template. The use case template above gives information on the four most significant use cases, which are

Indexing, ranking, search, and adding a URL.

2.5 Sequence Diagram

A sequence diagram is an interaction diagram that demonstrates the manner and timing of an object's interactions with other items. These exchanges are organized chronologically. The series.

In-depth interactions between the user and the system are also depicted in the diagram. The interactions may involve communications between the system and other systems or communications between the sub-systems rather than just those between the system and the user.

Three sequence diagrams, each pertaining to a distinct group of operations, have been created. The initial one is for pre-processing data. The second is for the central functions of the system. The final one is for administrative functions.

2.5.1 Sequence Diagram for Pre-processing

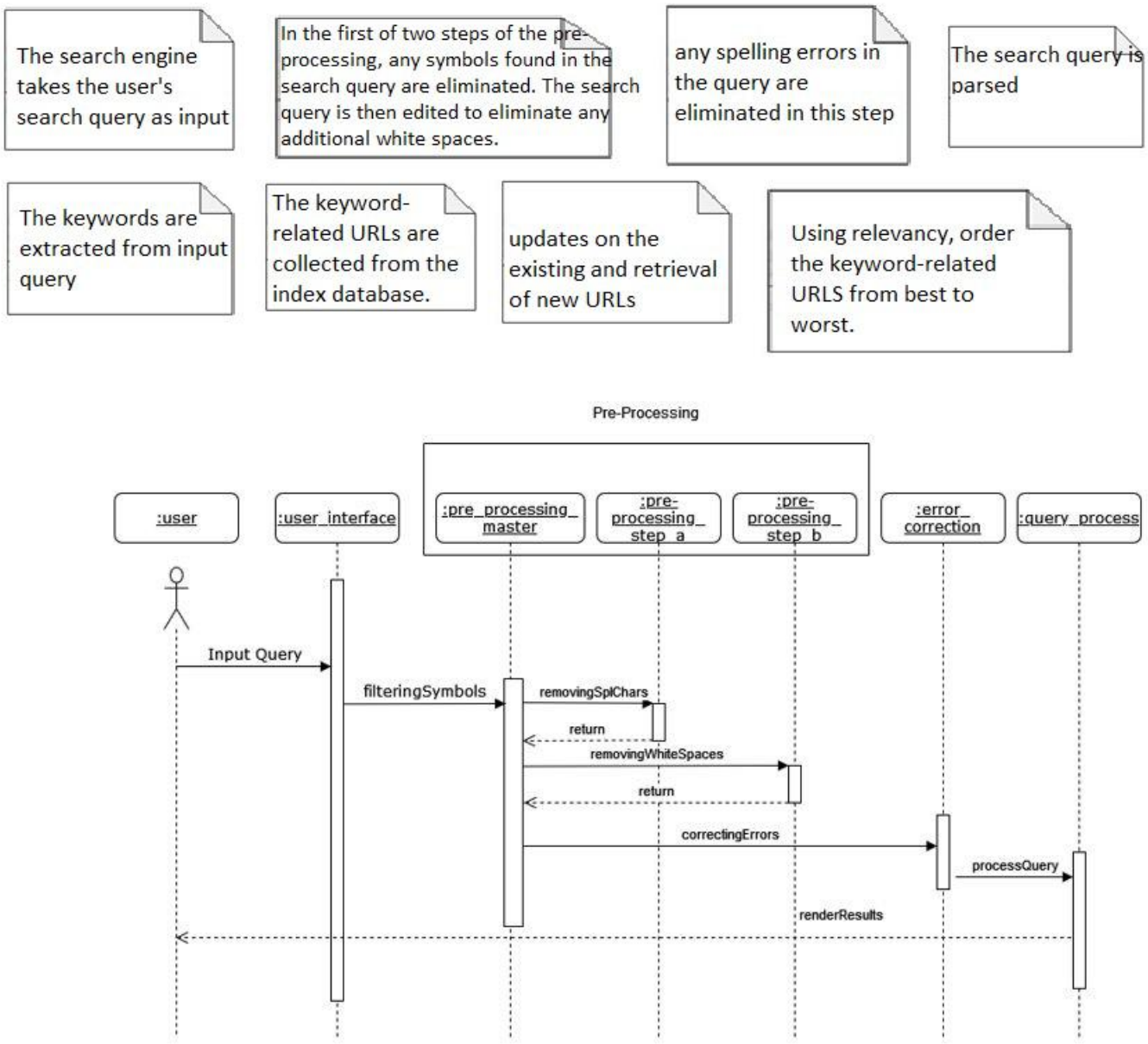

Figure 2.5: Pre-Processing Sequence Diagram

The input question serves as the trigger in the sequence diagram above, beginning a series of interactions. The first stage in pre-processing the query entails three phases. After the input query is entered, the initial action is to eliminate symbols from the query. The query's special characters are then deleted. If there are any additional white spaces in the processed query from the previous stage, they are also eliminated. Finally, any typographical errors are fixed if they exist.

2.5.2 Sequence Diagram for Search Engine Core

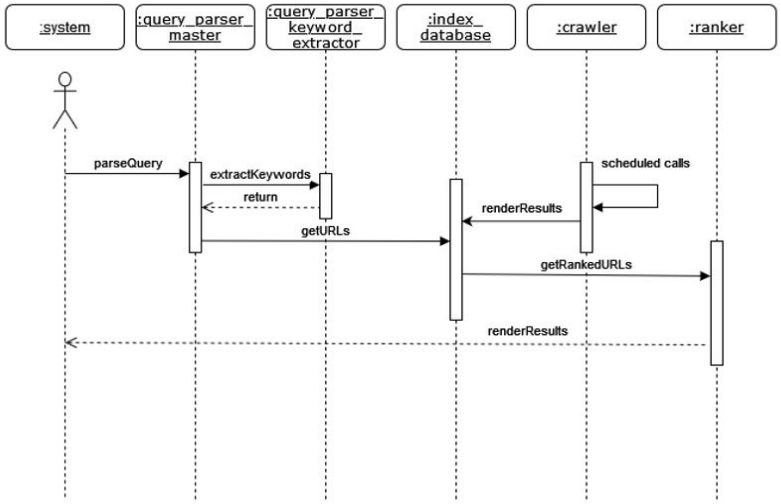

Figure 2.6: Search Engine Core Sequence Diagram

The query is now parsed in the system's core, where the keywords are then gleaned. The retrieved keywords are utilized to search the index database and find the pertinent URLs. The web crawler searches the internet for both fresh and abandoned web pages. The outdated pages are taken out of the index database and the new pages are added. The user is then presented with a ranking of the URLs that were acquired from the index database according to how relevant they are to the input query.

2.5.3 Sequence Diagram for Admin

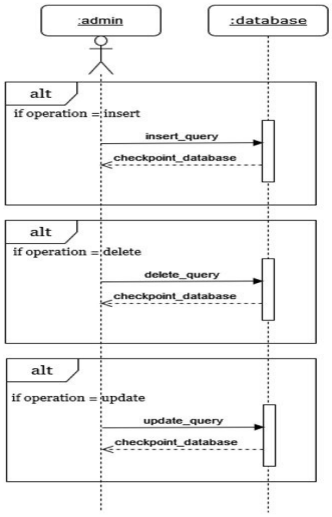

Figure 2.7: Pre-Processing Sequence Admin

A manual database update can be made by the administrator. At any time, he has the ability to update, delete, or enter data into the database. The frame with the 'ALT' tag has been used

to represent the same in the sequence diagram. A new URL can be manually entered into the database by the admin. An existing URL in the database can also be updated or deleted by the admin. The administrator always runs a database checkpointing procedure on the database after carrying out any of these operations.

2.6 Class Diagram

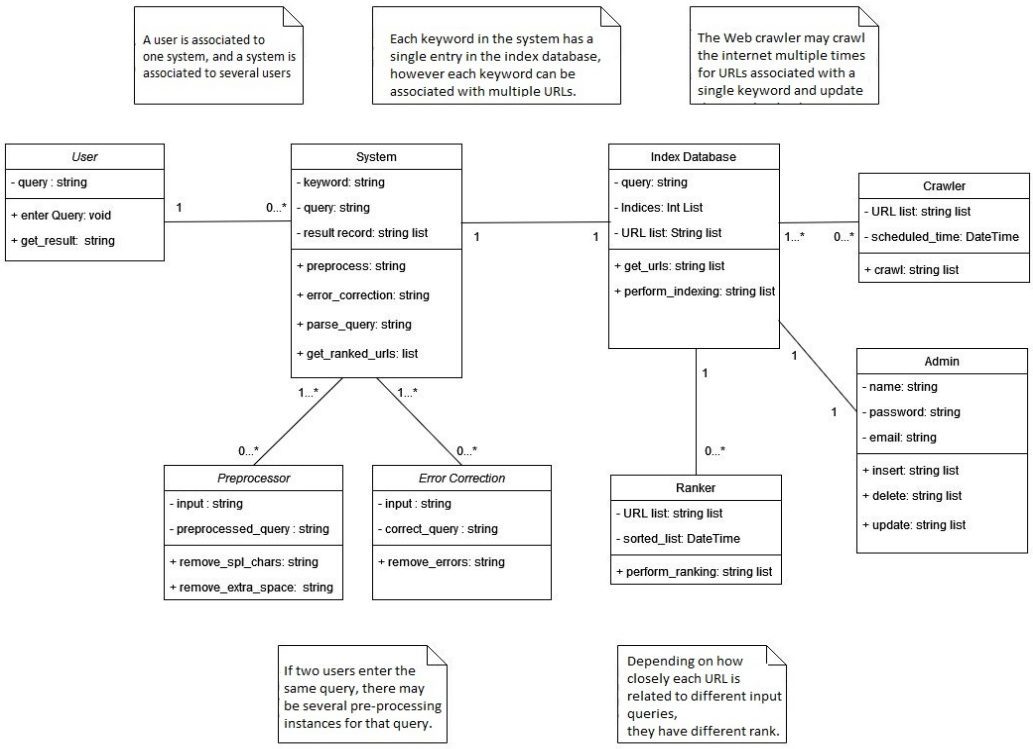

Figure 2.8: Class Diagram

The core component of object-oriented modeling is the class diagram. It is used for detailed modeling, which converts the models into computer code, as well as for general conceptual modeling of the application's structure. The administrative features that were lacking from the previously provided class diagram are now included in this upgraded version. The administrator has the option of adding a new URL or removing an existing URL from the database. An already-existing URL in the database may also be changed by the admin.

2.7 Object Diagram

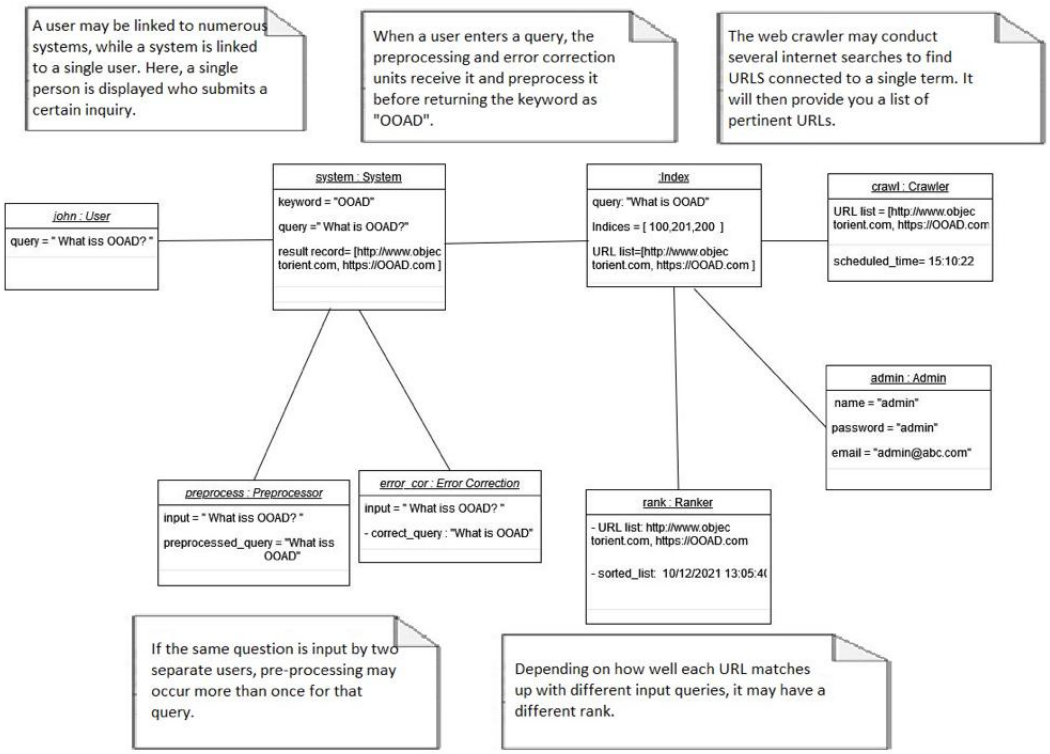

Figure 2.9: Object Diagram

The instances for the class diagram are included in the object diagram.Here, John is the user who submits this inquiry. 'Where is Denmark?', that is the question. There are grammatical errors and excessive white spaces in this query. Thus, preprocessing units are employed to extract keywords. Special characters and unnecessary white space will be eliminated by the preprocessing unit. Typographical errors will be removed by the error correction unit, and when we combine them, we get the keywords. The URLs that correspond to the provided keyword are then checked in the index database. Database functionalities including insertion, deletion, updating, and checkpointing are handled by the administrator. All URLs on the internet will be listed by the crawler. All URLs will be ranked by Ranker based on their relevance.

2.8 State Transition Diagram

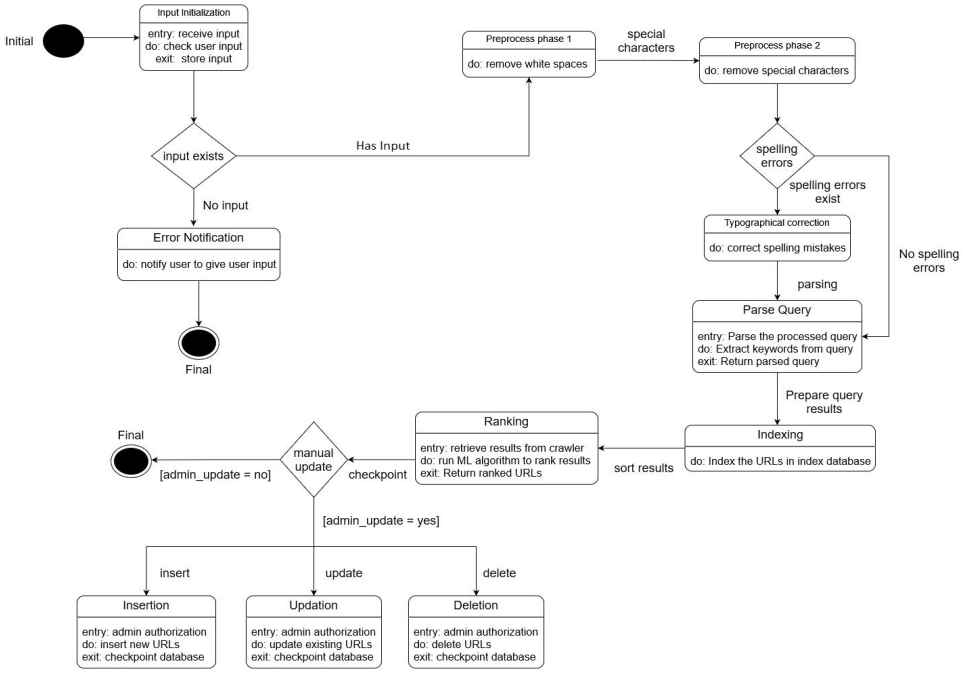

Figure 2.10: State Transition Diagram

The states that an object can be in, the circumstances under which it can change states, the guards that must be in place before the transition can take place, and the activities carried out during an object's existence are all described in state-transition diagrams (actions). State-transition diagrams are highly helpful for describing how different objects behave over the entire range of use cases that have an impact on those items. For illustrating the cooperation between items that results in the transitions, state-transition diagrams are ineffective.

Our search engine's state diagram comprises two end states and one initial state. Wherever decisions need to be made based on how various alternative flows would play out, decision boxes are employed. The state transitions are also determined by guard conditions.In the preprocessing stage, transitions will happen automatically.

2.9 Activity Diagram

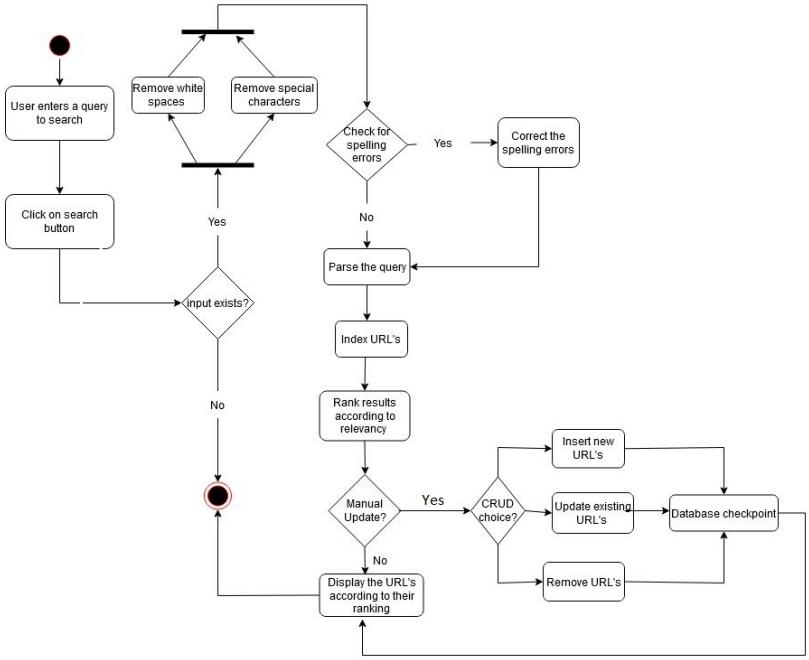

Figure 2.11: Activity Diagram

A flowchart used to show how one activity leads to another is called an activity diagram. The action might be referred to as a system operation. The dynamic features of the system are described using activity diagrams.

The user enters a search query in the box shown in the above diagram, after which the user clicks the search button to begin the process of doing the search. If the user doesn't supply any input, the system will proceed to the final state; otherwise, the system will process the inquiry by eliminating spaces, special characters, and correcting any spelling issues before sending a query for parsing. After parsing, the search results are indexed, and the user is presented with the most pertinent results at the top of the list.

2.10 Component Diagram

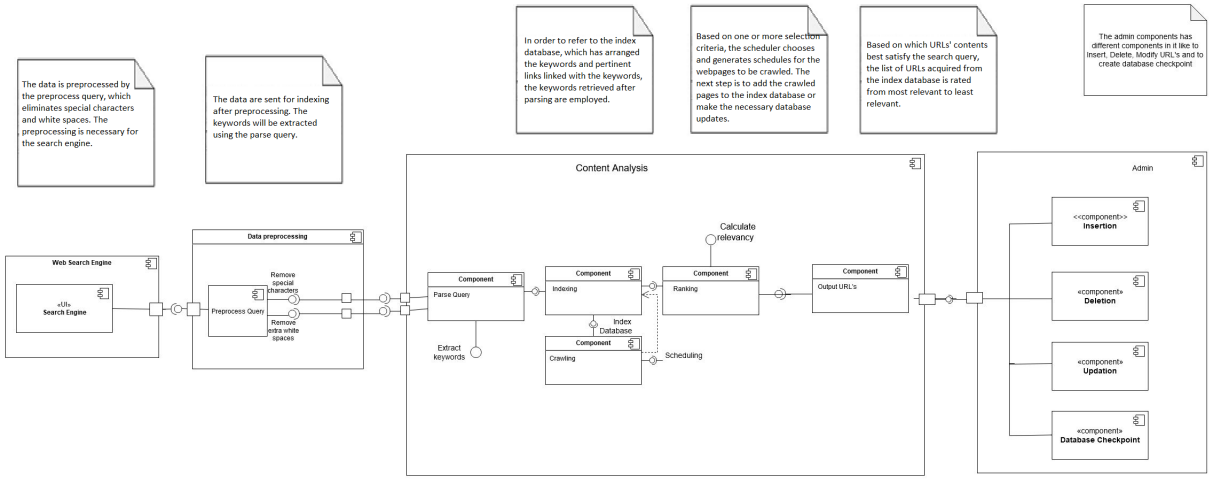

Figure 2.12: Component Diagram

Crawler, ranker, and index databases are all used in content analysis. Typographical errors, excess spaces, and special characters are removed using the query preprocessing unit. An administrator has access to index databases and can create, delete, and update database operations.

There are four parts to the component diagram shown above: the web search engine, data preparation, content analysis, and administration. The user inputs the specifics of his or her search in the web search engine component, which is a UI component of the system.

The data is transferred from the web search engine to Data preparation. Through a port, the two components are connected. The web search engine uses the necessary interface because it needs the assistance of the data preparation component. The superfluous white spaces and special characters are eliminated during data preprocessing before the data is transmitted to the content analysis port.

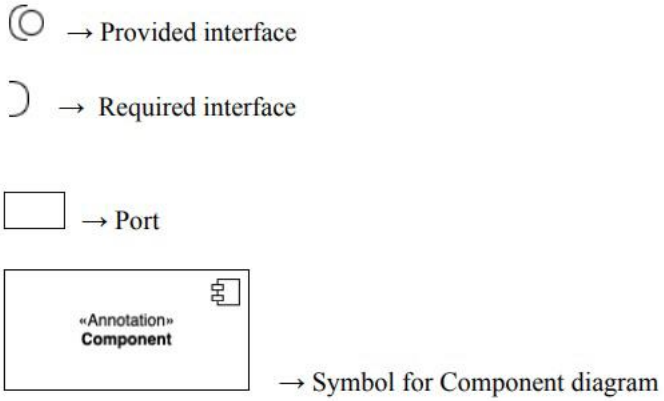

Figure 2.13: Notations

There are several parts to content analysis, including parse queries, indexing, crawling, ranking, and output URLs. We take the keyword from the Parse query component and send it to Indexing to acquire the Index Database (which organizes the keywords and associated links with the keyboard).

The crawling component receives the acquired keywords from indexing and schedules them (organizing the data). The data is then sent back to indexing after scheduling so that the database modifications can be made. The relevancy will then be determined by the ranking component, and the URLs are ranked from the most to the least relevant search criteria. The URls derived from ranking are passed to the component of the output url that displays the output to the user. The Admin component carries out various tasks such database checkpoints, URL updates, deletions, and insertions.

2.11 Deployment Diagram

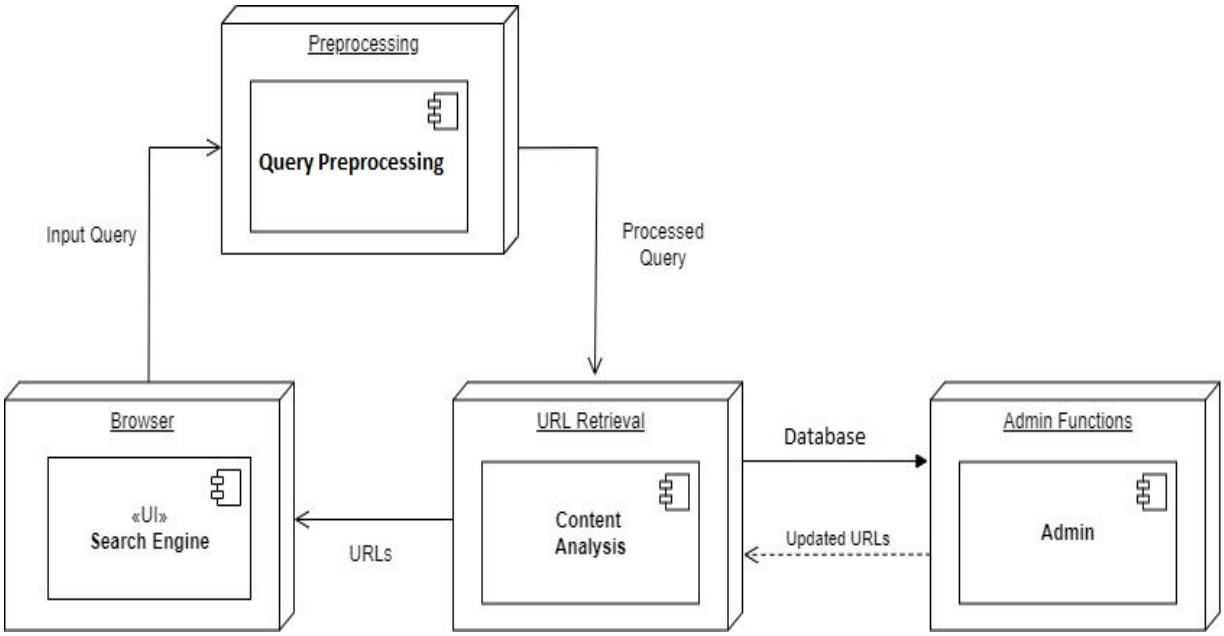

Figure 2.14: Deployment Diagram

The system's execution units are shown in the deployment diagram. It stands for the physical components that contain memory, software, and middleware. Query processing, search engine, content analysis, and administration are referred to as nodes in the diagram above since they are runtime components. The user interface where a query can be entered is the browser.

Crawler, ranker, and index databases are all used in content analysis. Typographical errors, excess spaces, and special characters are removed using the query preprocessing unit. An administrator has access to index databases and can create, delete, and update database operations.

Chapter 3

Test Cases

3.1 Test Cases

1. **TC01:** Entering valid input and checking returned result URLs.
2. **TC02:** Entering no input and trying to search.
3. **TC03:** Testing Admin CRUD functions
4. **TC04:** Input having typographical mistakes
5. **TC05:** Validating ranked URLs
6. **TC06:** Checking indexing methodology in database
7. **TC07:** Having special characters and white spaces in the input

Chapter 4

Implementation

4.1 Deployment Diagram

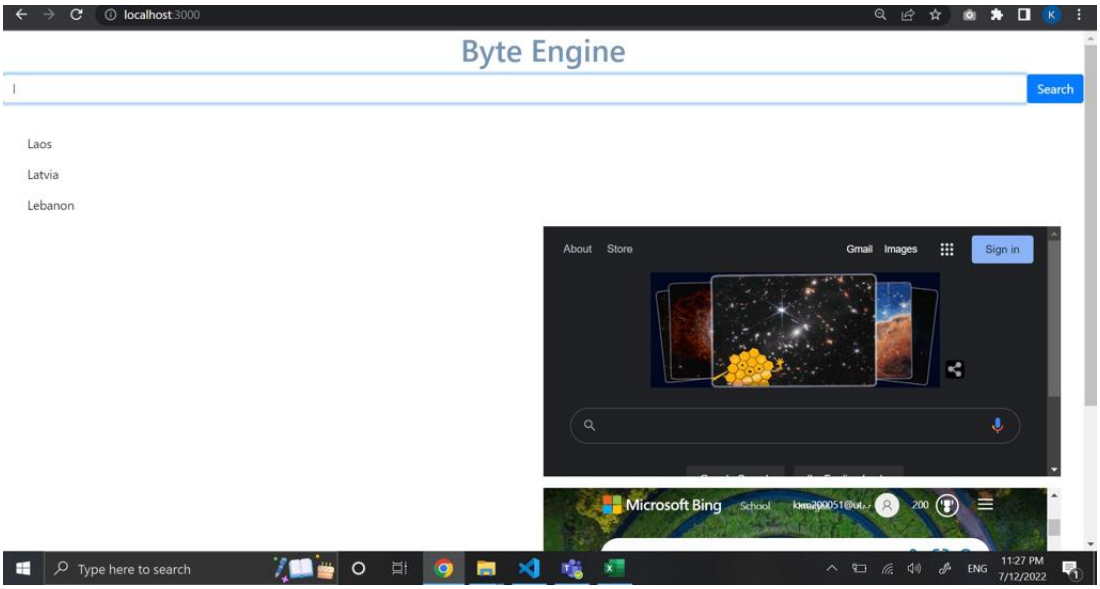

Figure 4.1: Front End

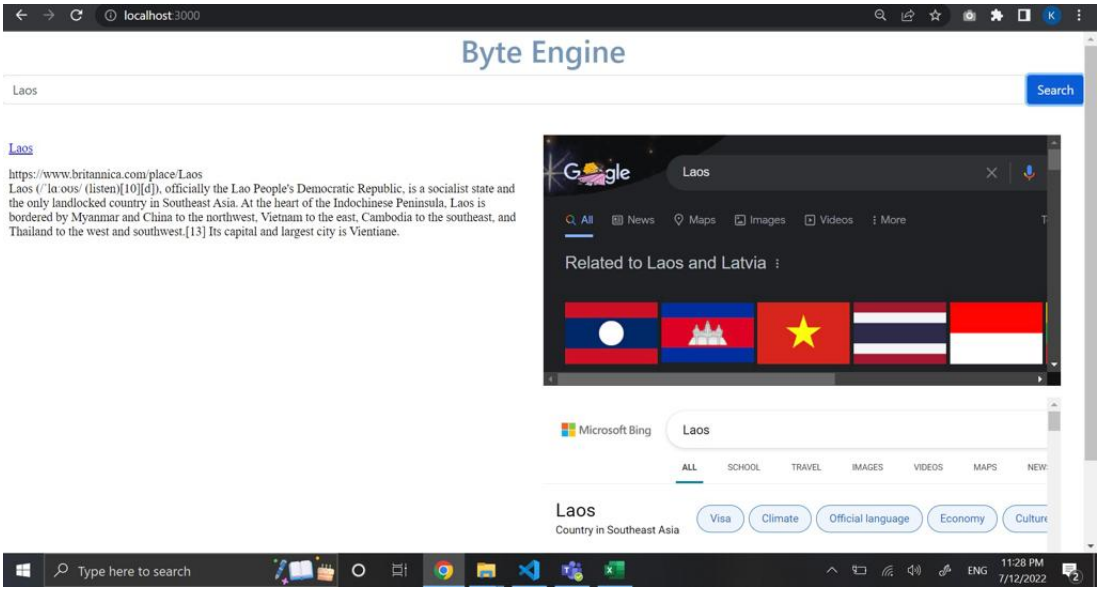

Figure 4.2: Searching for a country Name

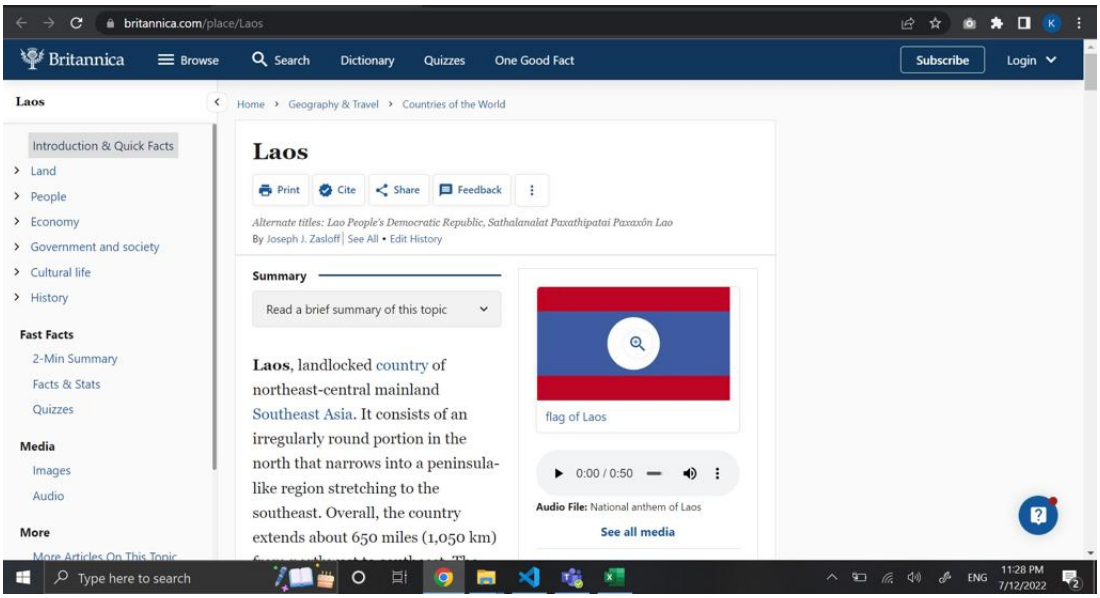

Figure 4.3: Results

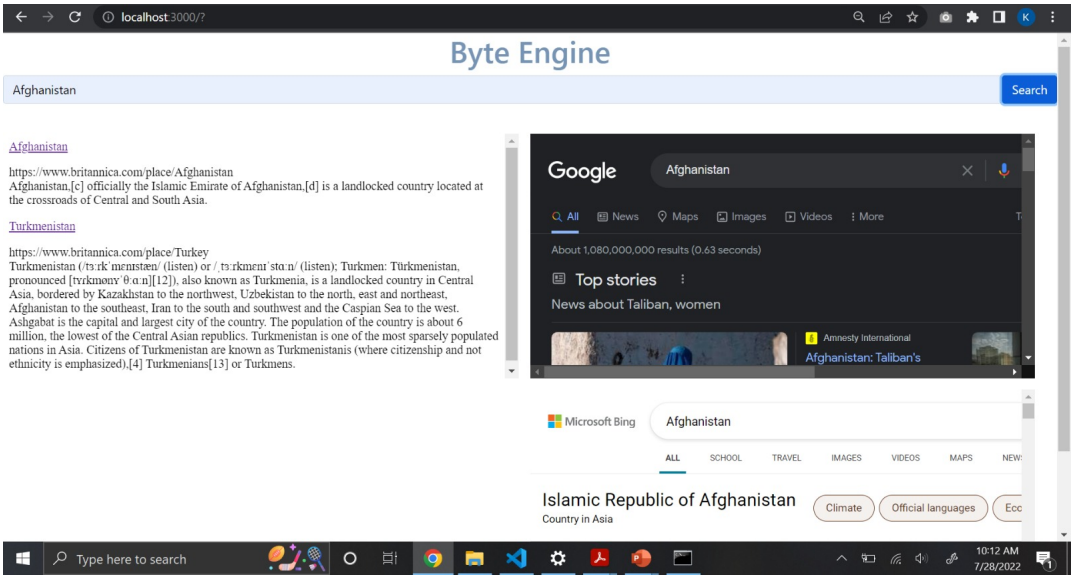

Figure 4.4: Searching for Results of Afganisthan

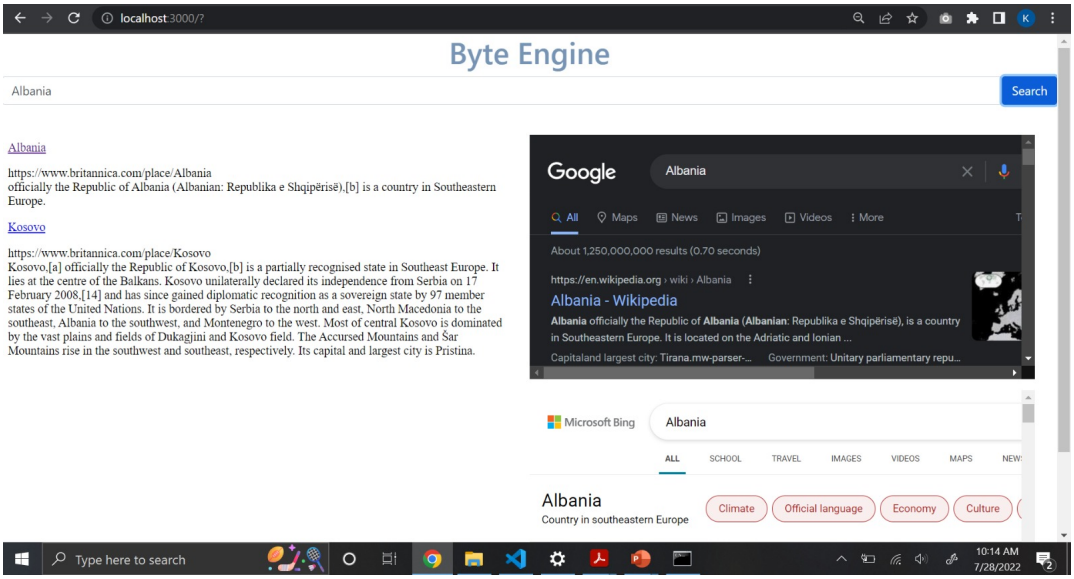

Figure 4.5: Results 2

Chapter 5

Conclusion

---

1. A search engine gives us knowledge in a split second in this age of expanding information, something we couldn't have believed was feasible just a few decades ago. Every day, it gains more powers.
2. In addition to offering information, search engines assist businesses in promoting their websites.
3. Indexing and ranking work together to improve the search engine's accuracy and optimization. Modern machine learning techniques could be incorporated into the architecture of search engines to increase their effectiveness.
4. Use case, Class Diagram, Object, Sequence Diagram, State Transition, Activity, Component, and Deployment Diagram are among the eight UML diagrams that we created.
5. We used UML diagrams to represent the project and choose the next actions to take.
6. We also compared the search results with Google and Microsoft search engines.
7. Use case, Sequence, State transition, and Activity diagrams helped us grasp the system's dynamic features, while Class, Object, Component, and Deployment diagrams helped us understand the system's static aspects.