Summify - Functional Specification

Student 1 Name: Benjamin Olojo **ID Number:** 19500599

Student 2 Name: Przemyslaw Majda **ID Number:** 20505049

Staff Member Consulted for supervision: Dr. Jennifer Foster

Table of Contents

1. Introduction

1.1 Overview

Summify is a web application that allows users to generate text summaries of videos through Natural Language Processing. The application primarily aims to automate the process of summarising lengthy educational videos, such as video lectures, through creating abstractive summaries of the video transcripts. Within the application, video transcripts will be fetched through user-specified URLs using the python YouTube Transcript API. These video transcripts will then be used as input material for generating abstractive summaries containing key information that will be provided to the user.

The video transcripts will be summarised using the GPT-3[1] language model fine-tuned for summarising video transcripts. The pre-trained model will be fine-tuned using the VTSSum [2] dataset, a benchmark dataset for video transcript segmentation and summarisation which includes 125K transcript-summary exemplar pairs from 9,616 video lectures. Once fine-tuned for video transcript summarisation, the model will be deployed on the OpenAI servers and accessed within the application through the OpenAI Developers API.

The application will additionally feature an Information Retrieval functionality that will leverage the spaCy NLP[3] library and TextRank[4] Keyword Extraction library to identify key terms within the video transcripts. Once keywords are identified, the user will be provided with links to relevant web-based resources, such as websites and videos, through the python Google Search API.

1.2 Business Context

The application could serve as an aid to a wide range of users for summarising video lectures or seminars into an easily readable form, revising video lecture material or creating notes of educational videos for personal study. Students of academic institutions could also use the Summify application to generate text summaries of publicly available lecture videos, or alternatively use the application to create summaries of long-form videos provided as supplementary course content.

However, the project is not intended for specific use within any business organisation and would primarily serve as an application for personal use.

[1] https://openai.com/api/
[2] https://github.com/Dod-o/VT-SSum
[3] https://spacy.io/
[4] https://web.eecs.umich.edu/~mihalcea/papers/mihalcea.emnlp04.pdf

1.3 Glossary

NLP → Natural Language Processing refers to the application of computational techniques to the analysis and synthesis of natural language and speech.

IR → Information Retrieval refers to the techniques of storing, recovering and disseminating recorded data through the use of a computerised system.

Fact Extraction → The task of automatically extracting structured information from unstructured and/or semi-structured machine-readable documents.

GPT-3 → Generative Pre-trained Transformer 3 is an autoregressive language model that uses deep learning to produce human-like text.

Extractive summarisation → Aims to identify the salient information within text that is then extracted and grouped to form a concise summary.

Abstractive summarisation → An alternative to extractive summarisation, abstractive summarisation is a method for generating novel sentences to represent the salient information in a body of text.

Pre-training → Pre-training refers to training a model with a series of tasks to help it produce useful data representations that can be used in other tasks.

Fine-tuning → Re-training a pre-trained model using custom data so that the weights of the original model are updated to account for the characteristics of the domain data.

spaCy → Open-source python library for advanced natural language processing.

PageRank → An algorithm used by Google Search to rank websites in their search engine results.

ROUGE →Recall Oriented Understudy for Gisting Evaluation, or ROUGE, is a set of metrics for evaluating automatic summarisation in NLP (Lin 2004) The metrics compare an automatically produced summary against a reference or goal (human-produced) summary.

2. General Description

2.1 Product/System Functions

The project will consist of a frontend created using Node.js and Vue.js as well as a backend created using Flask. The frontend will consist of a Vue.js interface used to obtain user input and display results generated by the backend containing the NLP components for text summarisation, keyword extraction and information retrieval. Both the frontend and backend of the application will consist of API endpoints to facilitate HTTP requests between the servers.

Using the link submission feature, users will be able to submit a URL to a YouTube video they want to summarise within the UI. Using the provided video URL, the captions for the specified video will then be obtained using the python YouTube Transcript API and subsequently passed via the OpenAI API to the fine-tuned GPT-3 model hosted on the OpenAI servers. Once the summary has been generated by the GPT-3 model and returned via the OpenAI API, it will be displayed to the user who will have the additional option to copy the generated summary to their clipboard for personal use. Additionally, users will be able to play the video within the UI through an embedded video player.

The Information retrieval functionality will utilise the TextRank algorithm to perform Fact Extraction on the original transcript of the video. This will be done through the PageRank based python TextRank Fact Extraction library used in conjunction with the spaCy library to rank the most important terms mentioned within the video transcript. The top ranked terms will then be passed to the python Google Search API which will return links to the top related web resources that will be provided to the user.

2.2 User Characteristics and Objectives

Within the frontend of the application, we intend to provide an interface and experience that doesn't require the user to have any knowledge of natural language processing or a skill set that spans beyond basic computer literacy. This is primarily due to the varied composition of our target audience, with the intended users having a wide range of computer skills and varied levels of knowledge in regards to natural language processing. All users are simply expected to understand basic mouse/keyboard functionality, particularly for copy and pasting text such as URLs.

All users of the application would be categorised as standard users and within the UI should expect to be able to paste URLs to YouTube videos for summarisation, play the embedded YouTube video, view and copy the generated summary and access hyperlinks to the top Google search results of extracted key terms.

Aside from the core functionality of the systems, users could also benefit from additional functionality that includes viewing and editing video transcripts before summarisation, generating transcripts for YouTube videos without transcripts available and MP4 files and finding all occurrences of specific terms within the video transcripts using the Fact Extraction component.

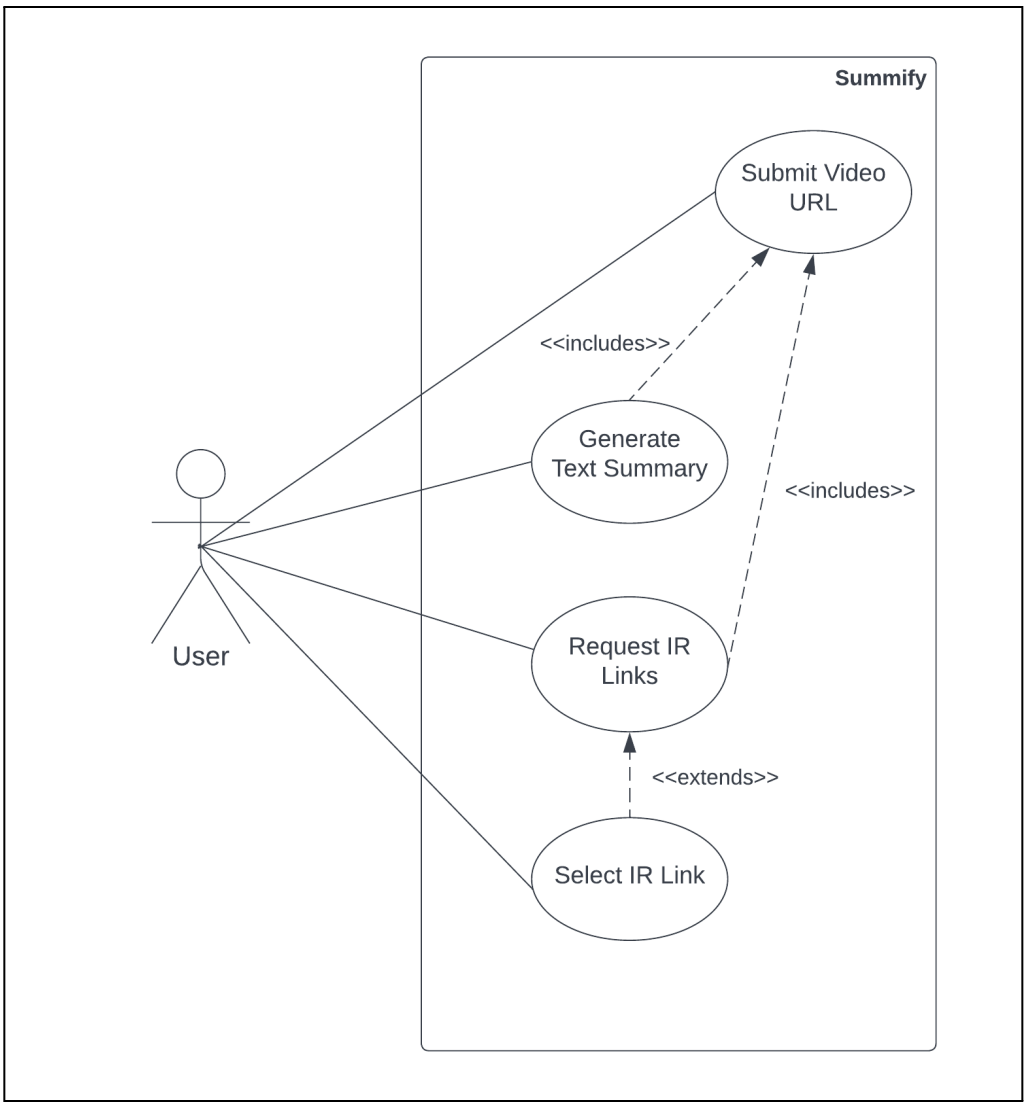
2.3 Operational Scenarios

For standard users, the main use cases of the Summify application will be obtaining text summaries of YouTube videos and accessing links to web-based resources based on key terms extracted from videos as depicted in Figure 1.

For obtaining text summaries of YouTube videos, the user will simply paste and submit the URL of the intended video within the UI. The transcript of the video will then be obtained and summarised before being displayed to the user.

For accessing links based on extracted key terms, the user will have to submit their request through a button within the UI and simply click on the links to the web resources subsequently displayed. These web resources, such as educational websites and related videos, will be based on the same video intended for summarisation by the user.

Fig. 1 - Use Case Model for Standard User



2.4 Constraints

2.4.1 Accuracy

It is necessary that the application produces high quality, human-like summaries to be of value to the user. In order to ensure the quality of the video transcript summarisation process, the development team must test summaries produced by the GPT-3 model using evaluation metrics such as ROUGE, a method used to compare text summarisation results to “gold-standard” summaries produced by experts.
Additionally, the Fact Extraction component using the TextRank library must also accurately highlight key terms within the video transcripts for relevant web-based to be provided. The development team should ensure that key terms extracted from the video transcripts accurately represent the main topics of the videos.

2.4.2 Response Time

The application should attempt to provide minimal latency between the submission of the video URL and the generation of the transcript summary. In addition, there should be minimal latency between the user’s IR request and the response from the Google Search API outlining the relevant web resources.

Since text summarisation is a computational expensive task, additional steps may be necessary in order to facilitate low latency responses from API calls to GPT-3. The development team should consider parallelising multiple API requests based on segments of the video transcript and creating an overall summary of the segmented transcript summaries. This could be coupled with fine-tuning the GPT-3 model to produce summaries for video transcripts prior to querying the model within the deployed application.

2.4.3 Project Deadlines

The design team will be constrained by the project deadlines outlined by the DCU School of Computing. The project must be completed by Friday the 24th of February 2023 at 5pm. The development team must effectively plan and carry out the implementation, testing and documentation of the project prior to this deadline.

2.4.4 GPT-3 Usage

Usage of the GPT-3 model must align with the OpenAI use-case and content policies. The development team must adhere to the OpenAI use-case policies which prohibit illegal or harmful industries, misuse of personal data, promoting dishonesty, deceiving or manipulating users and trying to influence politics.

2.4.5 Pre-Training Hardware Platforms

To take advantage of the parallel architecture of GPUs, the development team should consider offloading the intensive compute tasks involved in fine-tuning the GPT-3 model to one or more GPUs made available through Google Colab.

3. Functional Requirements

3.1 Video Link Submission

Description	The system will take a video URL and pass it to the Youtube Transcript API
Criticality	High
Technical Issues	The input must be a valid Youtube video URL
Dependencies with other requirements	None

3.2 Obtain Video Transcript

Description	The system will obtain the full video transcript using the Youtube Transcript API
Criticality	High
Technical Issues	The Youtube video must have a transcript or captions available. Some videos may have low quality auto-generated captions that make summarisation difficult
Dependencies with other requirements	Dependency on requirement 3.1 to obtain video link

3.3 Generate Transcript Summary

Description	The system will pass the transcript to the GPT-3 model through an API call, which will return the summarised transcript
Criticality	High
Technical Issues	The GPT-3 model must be available and running on the OpenAI servers and accessible through the OpenAI API
Dependencies with other requirements	Dependency on requirement 3.2 to provide the video transcript

3.4 Keyword Extraction

Description	The system will identify and extract keywords from the transcript
Criticality	Medium
Technical Issues	Certain video transcripts are difficult to extract keywords from due to sentence structure or typos
Dependencies with other requirements	Dependency on requirement 3.2 to provide the video transcript

3.5 Generate IR Links

Description	The system will perform a search using the Google Search API and will display links to results based on extracted key terms
Criticality	Medium
Technical Issues	Search results could be an inaccurate representation of the key video topics if the most mentioned keywords are broad terms
Dependencies with other requirements	Dependency on requirement 3.4 to extract keywords from the transcript

3.6 Embed Video

Description	The video associated with the URL provided by the user will be embedded within the webpage UI
Criticality	Low
Technical Issues	Videos may have embedding disabled in their Youtube settings
Dependencies with other requirements	Dependency on requirement 3.1 to provide a working Youtube video link

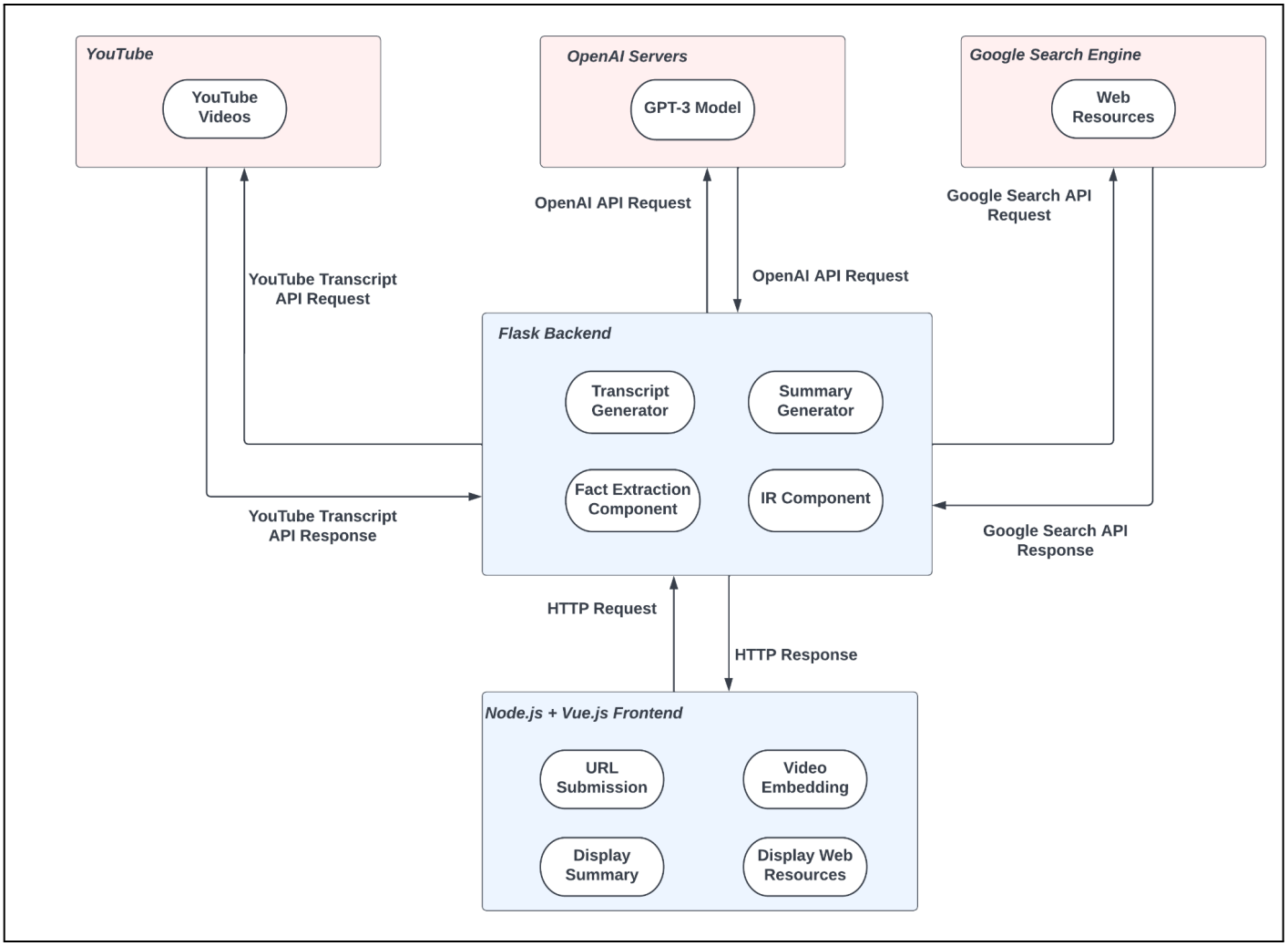
4. System Architecture

Designed to adhere to the client-server model, the Summify web application will feature a Flask application running on a backend server with a Node.js and Vue.js application running on a frontend server. The Summify application will also operate in conjunction with third-party systems that include the GPT-3 Model hosted on the OpenAI servers, the Google Search Engine service and the YouTube application.

The Flask backend will contain the components for making API calls to these third-party systems and utilise the results for transcript and summary generation, fact extraction of key terms and the associated top Google search results. The Node.js and Vue.js frontend will serve as the main mechanism for obtaining user input and displaying results within a UI available to the user through a web browser.

Communication between the backend and frontend applications will be facilitated through HTTP exchanges between the Flask and Node.js application API endpoints.

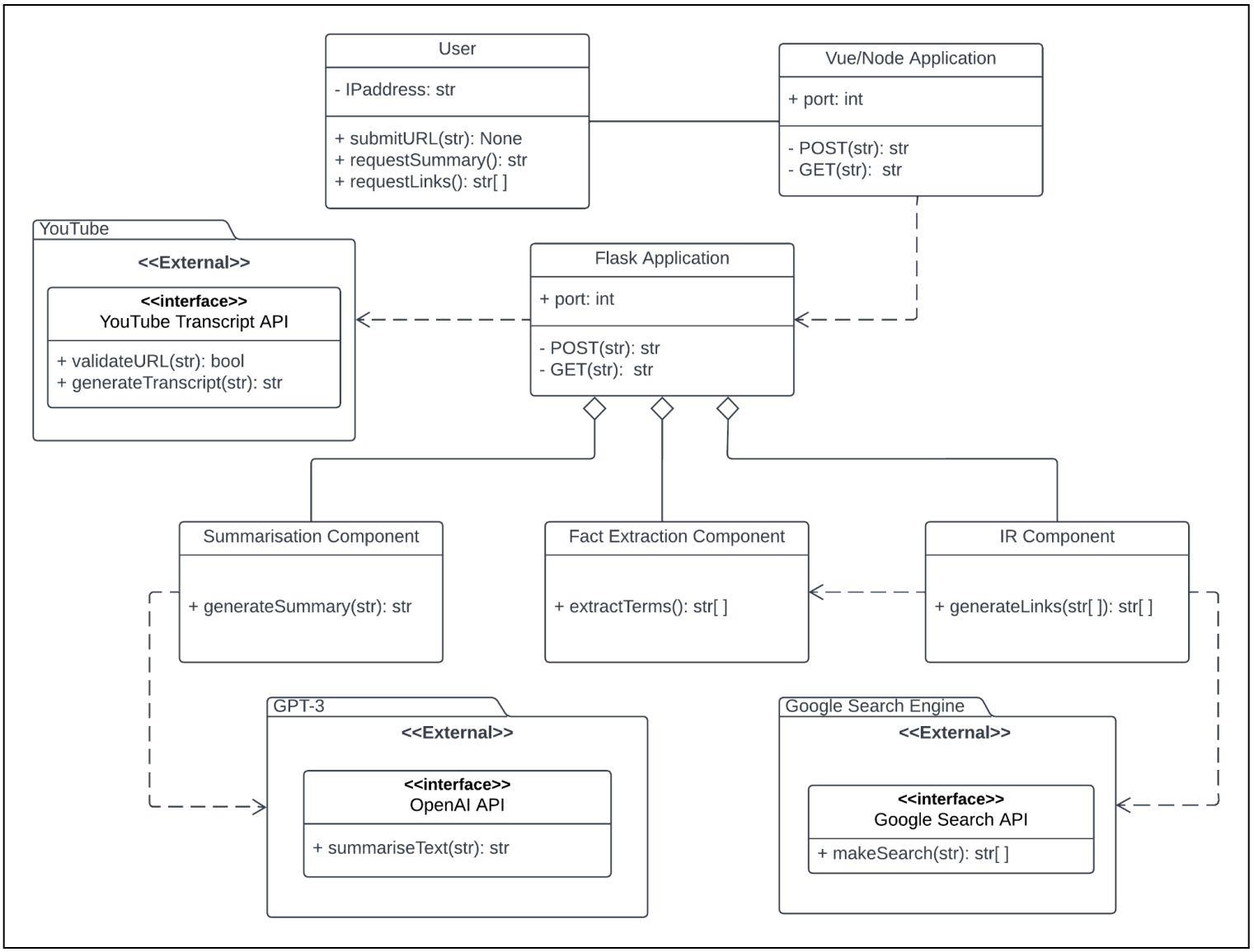
Fig. 2 - Summify System Architecture



5. High-level Design

5.1 Class Diagram

Fig. 3 - Summify Class Diagram



5.2 Sequence Diagrams

Fig. 4 - Video Summarisation Sequence Diagram

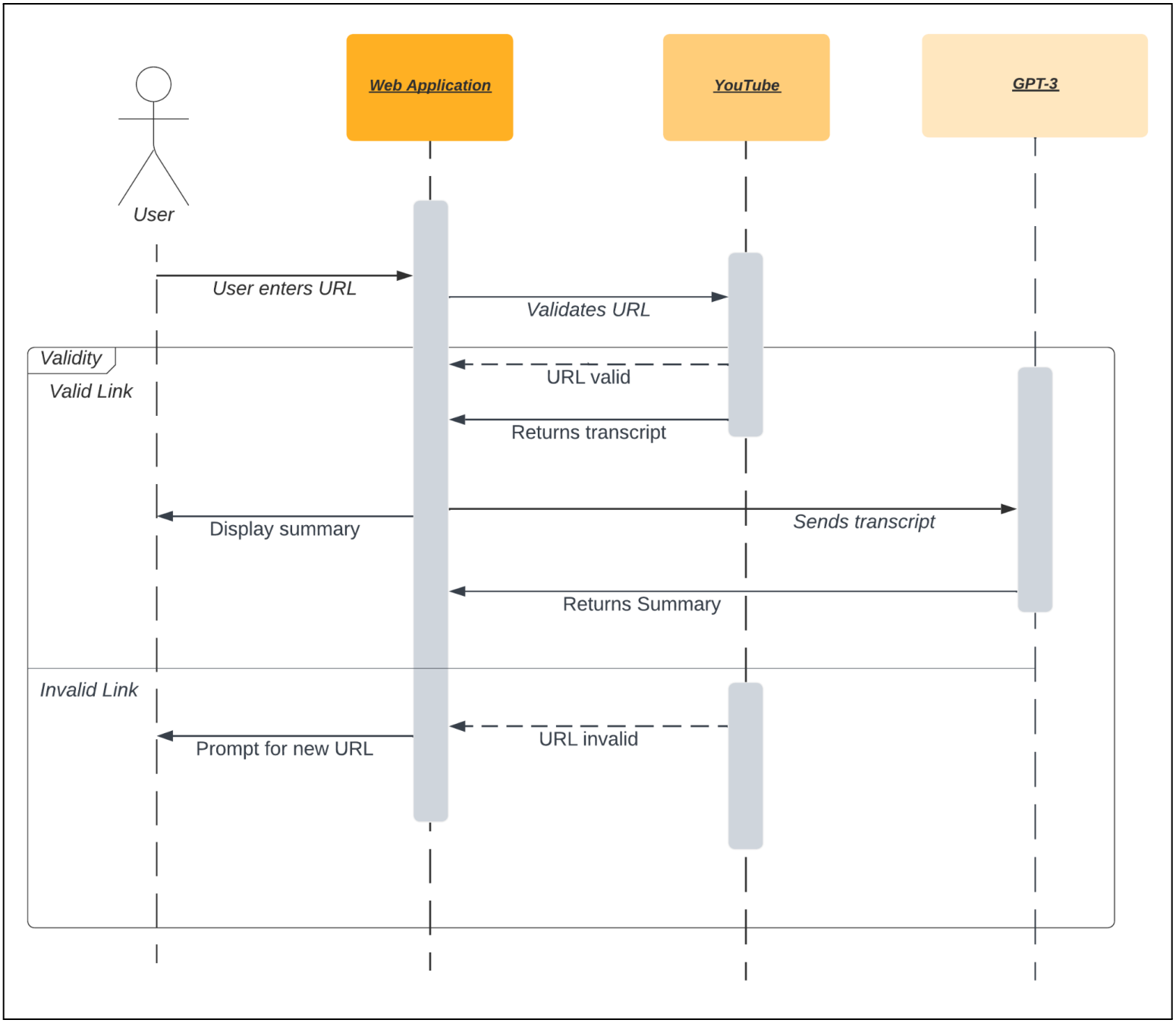
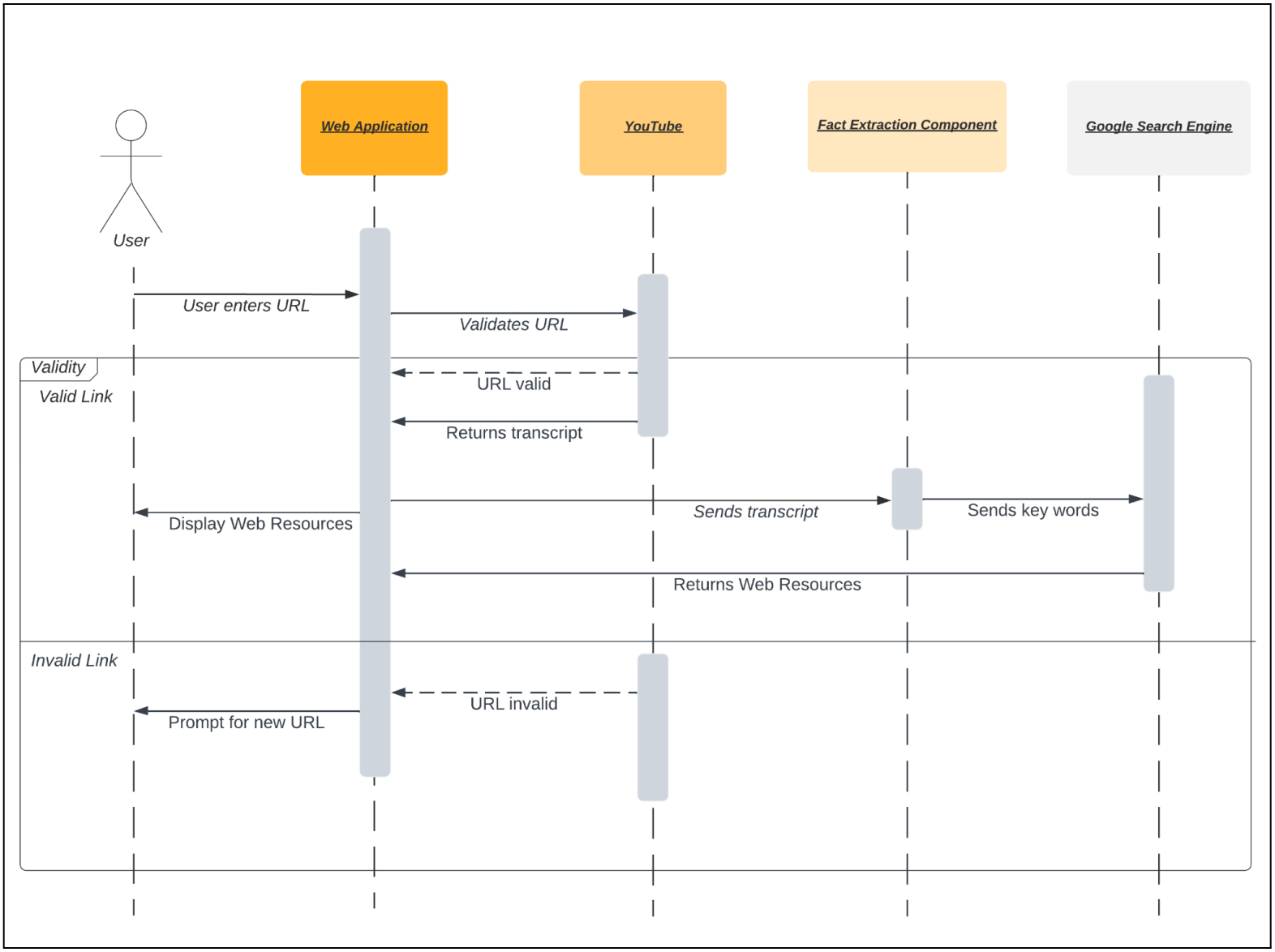
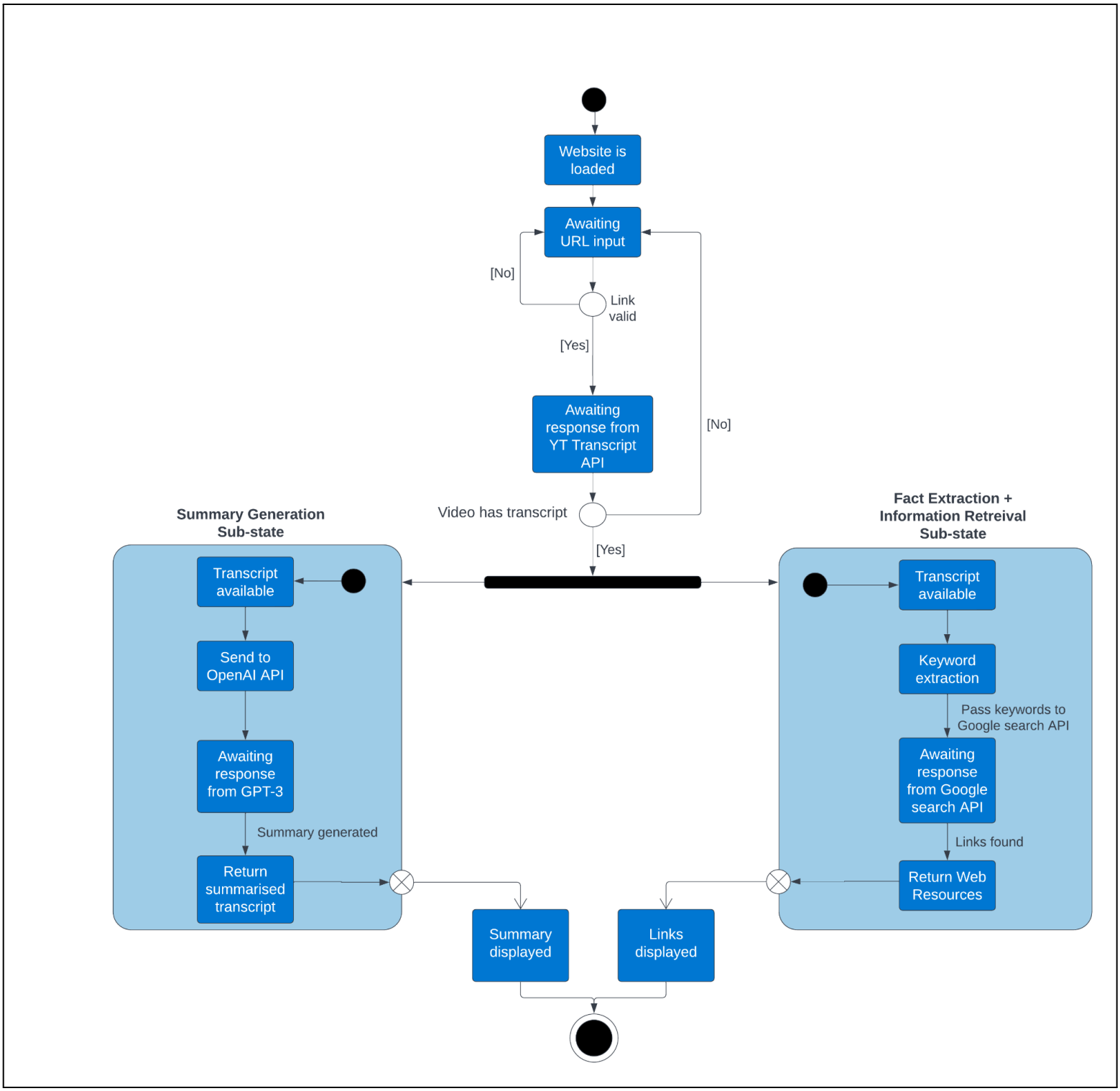


Fig. 5 - Information Retrieval Sequence Diagram



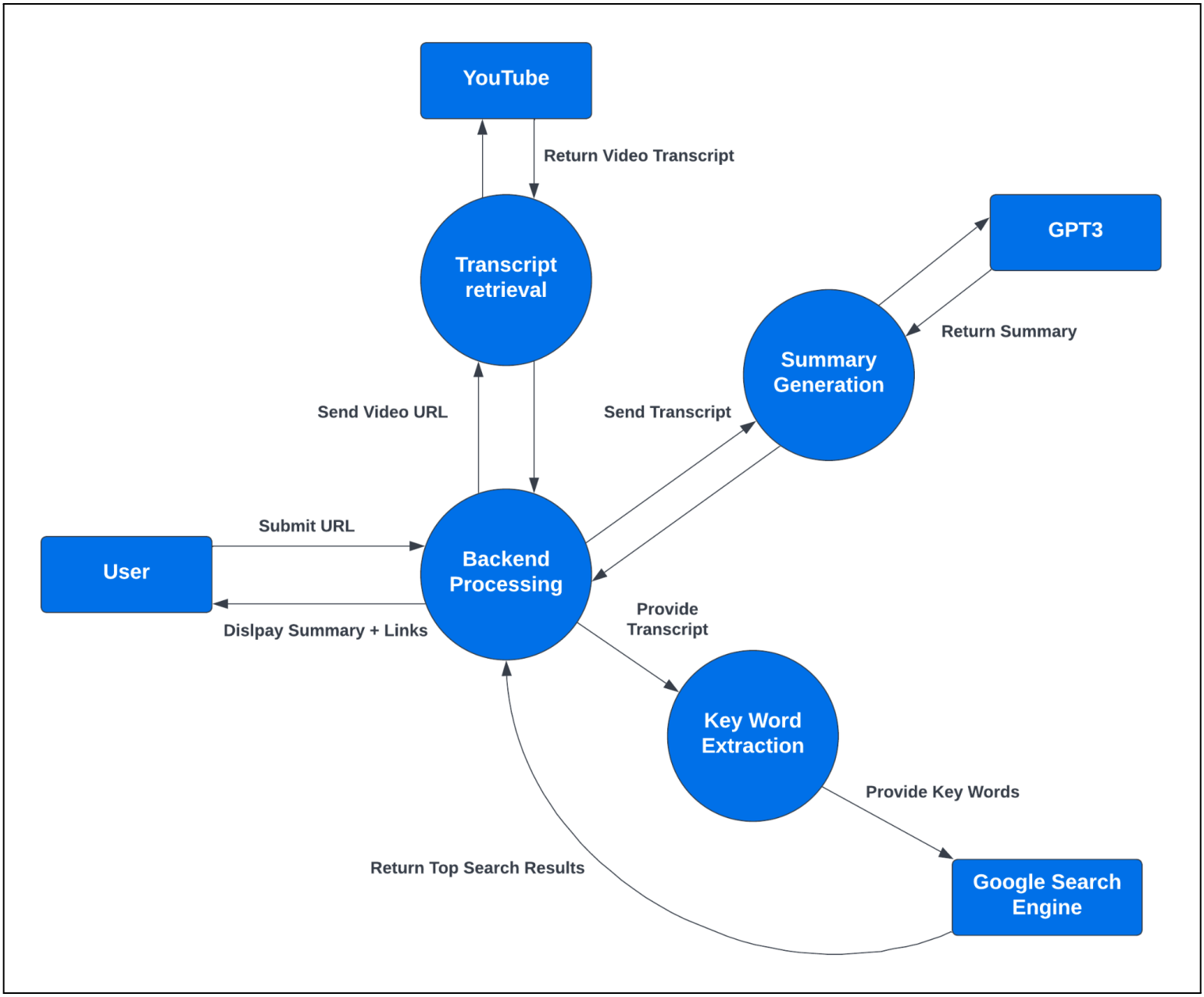
5.3 State Machine Diagram

Fig. 6 - Summify State Machine Diagram



5.4 Data Flow Diagram

Fig. 7 - Summify Data Flow Diagram

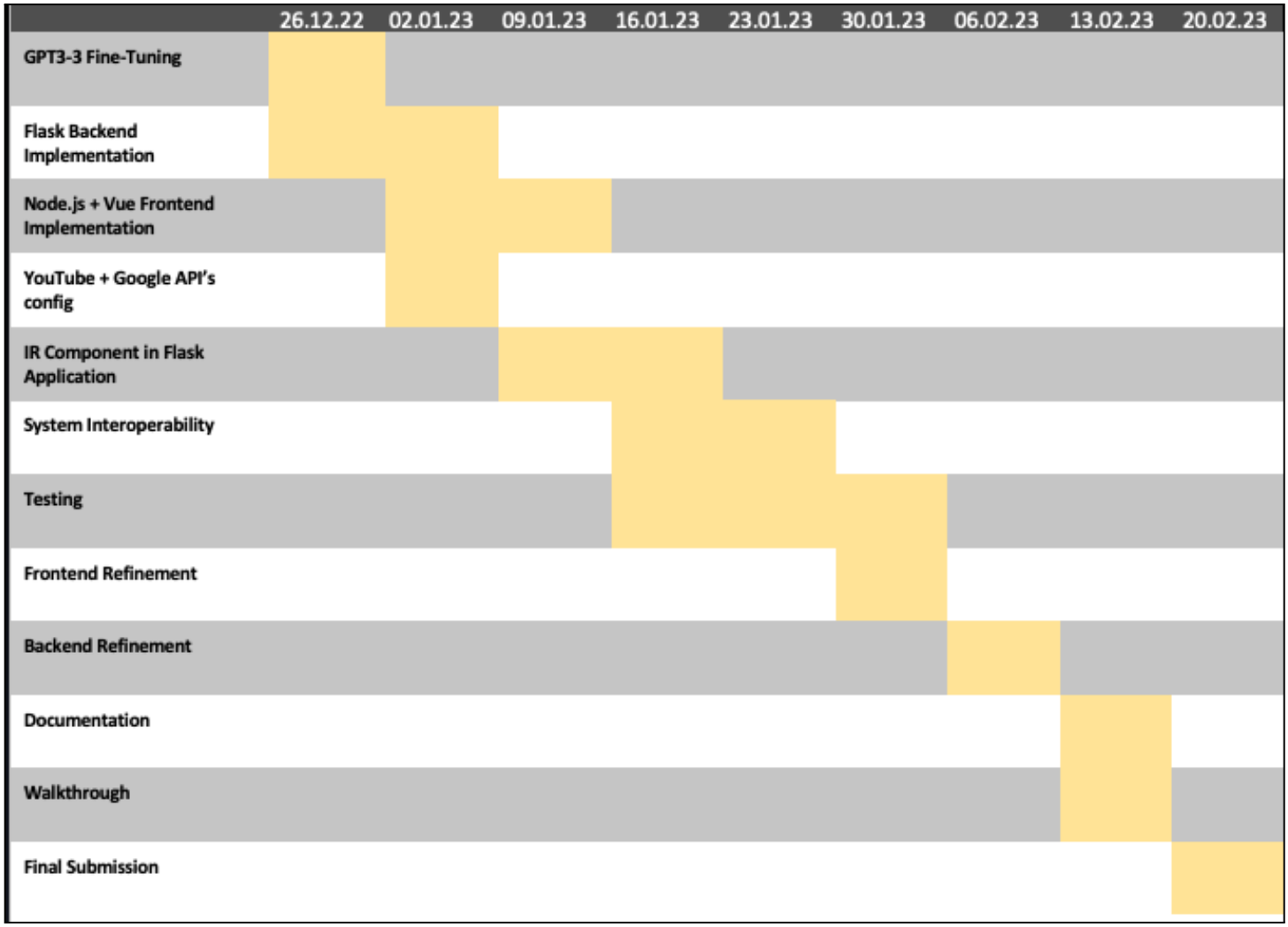


6. Preliminary Schedule

The GANTT Chart displayed in Figure 8 below indicates the plan for completing the project on a weekly basis. The development team also already has access to all the software/hardware tools necessary to complete the project.

Aspects of the project plan such as testing and documentation will also be carried outside of the outlined timeframes when appropriate to ensure their thorough completion. However, these aspects of the project will be the core focus of the development team during the periods outlined.

Fig.8 - Project Plan GANTT Chart



7. Appendix

- ROUGE Automatic Summary Evaluation Formula
  - Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
  - https://medium.com/the-ai-herald/a-qualitative-introduction-to-automatic-text-summarization-30f025c853c0

Fig. 9 - ROUGE-N Formula

$$ROUGE-N = \frac{\sum_{S \in S_H} \sum_{g_n \in S} C_m(g_n)}{\sum_{S \in S_H} \sum_{g_n \in S} C(g_n)}$$

where

S_H is the set of manual summaries

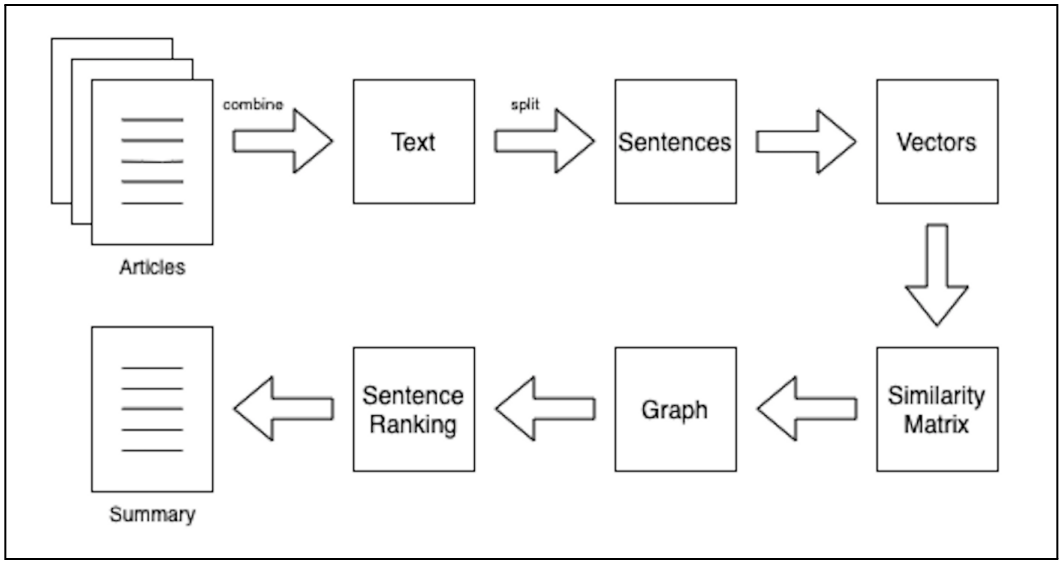
S is an individual manual summary

g_n is an N-gram

$C(g_n)$ is the number of co-occurrences of g_n in the manual summary and automatic summary

- TextRank Algorithm
  - https://ardaozmen.medium.com/extractive-text-summarization-using-textrank-algorithm-basic-logic-a6e73e9d60f0

Fig. 10 - TextRank Process Flowchart



- GPT-3 Usage Policies
    - https://beta.openai.com/docs/usage-policies

- GPT-3 Attention Based Architecture
    - https://dzlab.github.io/ml/2020/07/25/gpt3-overview/

Fig. 11 - GPT-3 Architecture Depiction