VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JNANA SANGAMA” ,BELAGAVI-590018

A Project synapses on

“ YOUTUBE TRANSCRIPT SUMMARIZER”

Submitted in the partial fulfillment of the requirement for the award of degree of

BACHELOR OF ENGINEERING
IN
ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Submitted by

MADHURI H S	1VJ22AI015
ZEBA AKTHER	1VJ21AI035
CHOWDRI FURKHAN	1VJ21AI007
MALLAREDDYGARI GAYATHRI	1VJ21AI016

Under the Guidance of

Dr. J. Venkatagiri
Professor, Dept. of AI&ML

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

VIJAYA VITTALA INSTITUTE OF TECHNOLOGY

Affiliated to Visvesvaraya Technological University, Belagavi | Recognized by Govt. of Karnataka |Approved by AICTE, New Delhi

KOTHANUR, BENGALURU – 560 077

TABLE OF CONTENTS

FIGURES

ABSTRACT

This project focuses on creating a comprehensive summary of YouTube video transcripts, aimed at enhancing content accessibility and user engagement. By employing natural language processing techniques, we analyze video transcripts to extract key themes, main ideas, and essential details. The summaries are designed to be concise yet informative, allowing viewers to quickly grasp the video's core message.

The project also explores the potential of summarization for educational purposes, aiding learners in efficiently digesting video content. Ultimately, our goal is to bridge the gap between extensive video content and user comprehension. Through this initiative, we aspire to empower viewers with the ability to easily navigate and absorb valuable information from diverse YouTube channels.

CHAPTER 1

INTRODUCTION

1.1. Overview

The user interface of the application is straightforward and user-friendly. Upon launching the app, users are greeted with a title and a text input field where they can enter the YouTube video link. Once a valid link is provided, the application extracts the video ID, retrieves the corresponding thumbnail image, and displays it alongside the input field. Users can then click the "Get Detailed Notes" button, which triggers the transcript extraction and summary generation process. If successful, the application presents the generated summary in a well-structured format, allowing users to easily digest the key points of the video content.

This YouTube transcript summarizer application combines effective use of APIs and generative AI to enhance the user experience by providing quick and meaningful summaries of video content. Its modular design and clear functionality make it a valuable tool for anyone looking to save time while still accessing important information from YouTube videos. Potential enhancements could include improved error handling, input validation, and additional customization options for users to tailor the summary output to their specific needs. Overall, this application exemplifies the power of modern technology in streamlining content consumption in the digital age.

1.2 Problem definition and scenarios

In an era where video content is proliferating, platforms like YouTube host vast amounts of information that can be time-consuming to consume. Users often seek quick access to the key points of videos without watching them in their entirety. Traditional methods of summarization, such as manually taking notes or relying on lengthy descriptions, can be inefficient and may lead to the omission of crucial details. There is a need for a tool that can automate the process of summarizing YouTube video transcripts, providing users with concise, relevant, and easily digestible information.

Scenarios

➢ **Student Research**

- **Context**: A student is conducting research on a specific topic and finds several relevant YouTube videos.
- **Action**: The student uses the YouTube Transcript Summarizer to enter the links of these videos.
- **Outcome**: The student receives concise summaries of each video transcript, enabling them to quickly identify which videos contain the most relevant information for their research, thus streamlining their study process.

➢ **Professional Development**

- **Context**: A professional is looking to improve their skills and comes across numerous tutorial videos on YouTube.
- **Action**: They input the video links into the summarizer to obtain quick overviews of the content.
- **Outcome**: The professional can efficiently determine which tutorials are worth watching in full, focusing only on those that align with their learning objectives.

1.3 Overview of documentation

The documentation for the YouTube Transcript Summarizer provides a comprehensive guide for users and developers, detailing the application's purpose, features, installation process, and usage instructions. It begins with an introduction that outlines the tool's goal of efficiently summarizing YouTube video transcripts, catering to a diverse audience including students, professionals, and content creators. The documentation includes step-by-step installation instructions, an overview of the code structure, and detailed explanations of key functions such as transcript extraction and summary generation using the Google Gemini Pro AI model. Additionally, it addresses common errors and troubleshooting tips, while also suggesting potential enhancements for future development. By offering clear guidance and insights, the documentation ensures that users can effectively utilize the application and that developers can understand its architecture and contribute to its improvement.

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction

The explosion of content on platforms like YouTube has made it increasingly challenging for users to extract meaningful information from lengthy videos. This paper presents a comprehensive survey of existing literature on YouTube transcript summarization, focusing on the methodologies employed, particularly those leveraging Natural Language Processing (NLP) models such as Hugging Face Transformers. We explore the effectiveness of these techniques, the challenges faced, and future directions for enhancing transcript summarization tools.

2.2 Review paper on Youtube Transcript Summarizer

YouTube has become a vital source of information and education, with millions of users relying on it for diverse content. However, the abundance of videos often leads to information overload, necessitating efficient summarization tools. YouTube transcript summarizers aim to provide quick, concise, and informative summaries, enabling users to save time and enhance their learning experience. This paper synthesizes insights from two key projects: the work of Sourav Biswas et al. and the project by Vijay Mane et al., which utilize advanced NLP techniques to summarize video content.

2.3 A Comprehensive Survey: Youtube Transcript Summarizer

This survey examines advancements in YouTube transcript summarization, focusing on methodologies and technologies that enhance user experience. Key areas include **Model Development**, utilizing transformer models like BERT for accurate summaries, and **User - Centric Design**, prioritizing intuitive interfaces based on user feedback. Additionally, **Cross-Linguistic Capabilities** are explored to ensure accessibility for diverse audiences. Recent projects, such as those by Sourav Biswas and Vijay Mane, highlight the effective use of Hugging Face transformers and Python APIs, achieving high accuracy rates in summarization. Future research should emphasize multilingual support and improved accuracy to further streamline content consumption on YouTube.

2.4 Literature study

The literature reveals a growing interest in the application of NLP for video summarization, with several studies highlighting the effectiveness of different methodologies. For instance, extractive summarization techniques, which select key sentences from transcripts, have been widely adopted due to their simplicity and effectiveness. However, recent advancements in deep learning have paved the way for more sophisticated abstractive summarization methods that generate coherent summaries by rephrasing the original content. The studies reviewed indicate that while extractive methods are useful, abstractive techniques offer a more nuanced understanding of the video's context and can produce summaries that are more aligned with human comprehension.

2.5 Overview of Literature Review

The literature on YouTube transcript summarization can be divided into three main areas. **Model Development** focuses on creating and fine-tuning transformer models to enhance the accuracy and coherence of summaries. **User -Centric Design** prioritizes user experience, ensuring that summarization tools meet diverse needs through intuitive interfaces and relevant outputs. Lastly, **Cross-Linguistic Capabilities** explores adapting summarization tools for multiple languages, making them accessible to non-English speakers and expanding the user base. Together, these areas aim to develop effective and inclusive tools that facilitate efficient information retrieval from the extensive content available on YouTube

2.6 Conclusion

YouTube transcript summarization is a critical area of research that addresses the challenges posed by the vast amount of content available online. The application of advanced NLP models, particularly transformer architectures, has shown significant promise in improving the quality of summaries. However, challenges such as language diversity and varying user expectations remain. Future research should focus on enhancing cross-linguistic capabilities and refining models to cater to diverse user needs. As the demand for efficient information retrieval continues to grow, YouTube transcript summarization tools will play an increasingly important role in facilitating knowledge acquisition

CHAPTER 3
PROJECT DESCRIPTION

3.1 Objective

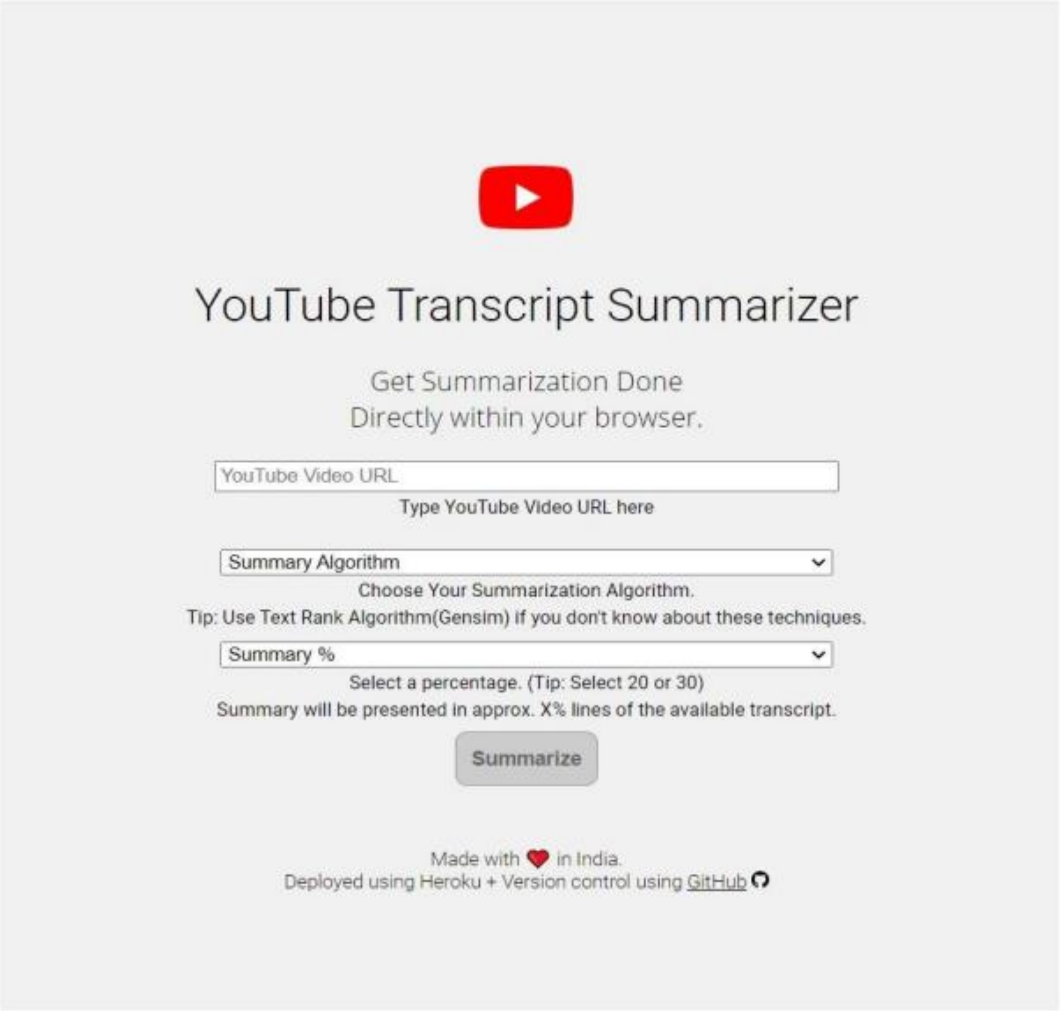
The objective of this project is to develop an automated system that retrieves and summarizes YouTube video transcripts using advanced natural language processing techniques. By leveraging the capabilities of the ‘YouTubeTranscriptApi’ to extract textual content from videos and employing a generative AI model (specifically, the "gemini-pro" model), the system aims to provide users with concise and meaningful summaries of video content.

3.2 Existing Systems

Even though many systems and architectures exist to perform text summarization, this project demonstrates a different approach of accessing the backend python API, that is, in the form a chrome extension, which upon clicking the summarize button provided makes a http request to the backend to acquire the summarized text for the obtained transcripts of a video. However, when it comes to improving or enhancing the object detection model, there are very few solutions available. One such method is where a backend is hosted on a server and a request for the API is made by pasting the URL of the video in the placeholder of the label provided in the frontend. The following sub chapter details the method

3.2.1 YouTube video summarization over Flask

The backend here uses Flask framework to receive API calls and then respond to the calls with a summarized text. The API here can only work on the YouTube videos which have well formatted transcripts available. It hosts a web version of the Summarizer to make the API calls in an easier way and show the output within a webpage. The figures 3.1 and 3.2 below show the working and GUI interface of the system respectively. In this existing system, the user should manually paste the URL of the video to acquire the summarized transcripts.



YouTube Transcript Summarizer

Get Summarization Done
Directly within your browser.

Type YouTube Video URL here

Choose Your Summarization Algorithm.
Tip: Use Text Rank Algorithm(Gensim) if you don't know about these techniques.

Select a percentage. (Tip: Select 20 or 30)
Summary will be presented in approx. X% lines of the available transcript.

Made with ❤ in India.
Deployed using Heroku + Version control using GitHub

Figure 3.2 User Interface.

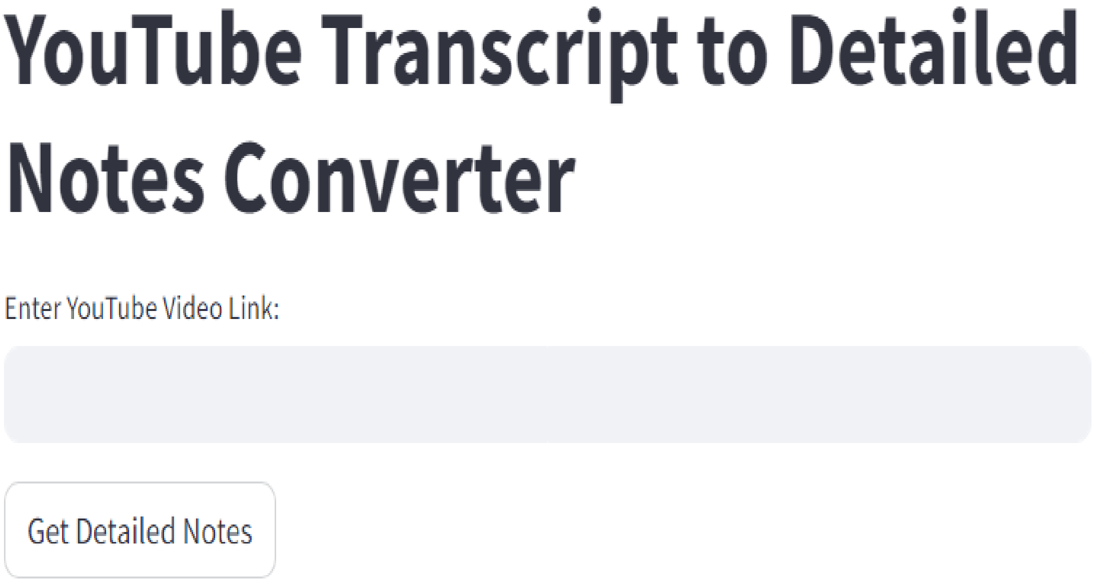
3.3 Proposed system

To tackle the problem of the mentioned existing systems in the above sub chapter, a solution for acquiring transcripts of a video, a high-level approach is given in the following steps:

- **User Interface (UI):** A web application built using Streamlit, providing an intuitive interface for users to input YouTube video links.
- **Transcript Extraction Module:** Utilizes the YouTubeTranscriptApi to fetch the transcript of the video based on the provided URL.
- **Summarization Engine:** Integrates with the Google Gemini Pro generative AI model to process the extracted transcript.
- **Output Display:** Presents the generated summary in a clear and organized format, allowing users to easily read and understand the key takeaways from the video.

3.3.1. Youtube Video Summarization over Streamlit

The YouTube Transcript Summarizer is a Streamlit-based web application that enables users to input a YouTube video link and receive a concise summary of the video's content. By utilizing the YouTube Transcript API, the application extracts the transcript of the specified video and then employs the Google Gemini Pro generative AI model to summarize the transcript into key points, limited to 250 words. Users can easily enter a video URL, view the corresponding thumbnail, and click a button to generate detailed notes. The application is designed to provide an interactive and user-friendly experience, making it simple for users to quickly access essential information from YouTube videos without having to watch the entire content. Overall, this tool enhances the accessibility of video information, catering to learners and researchers seeking efficient ways to digest video content.

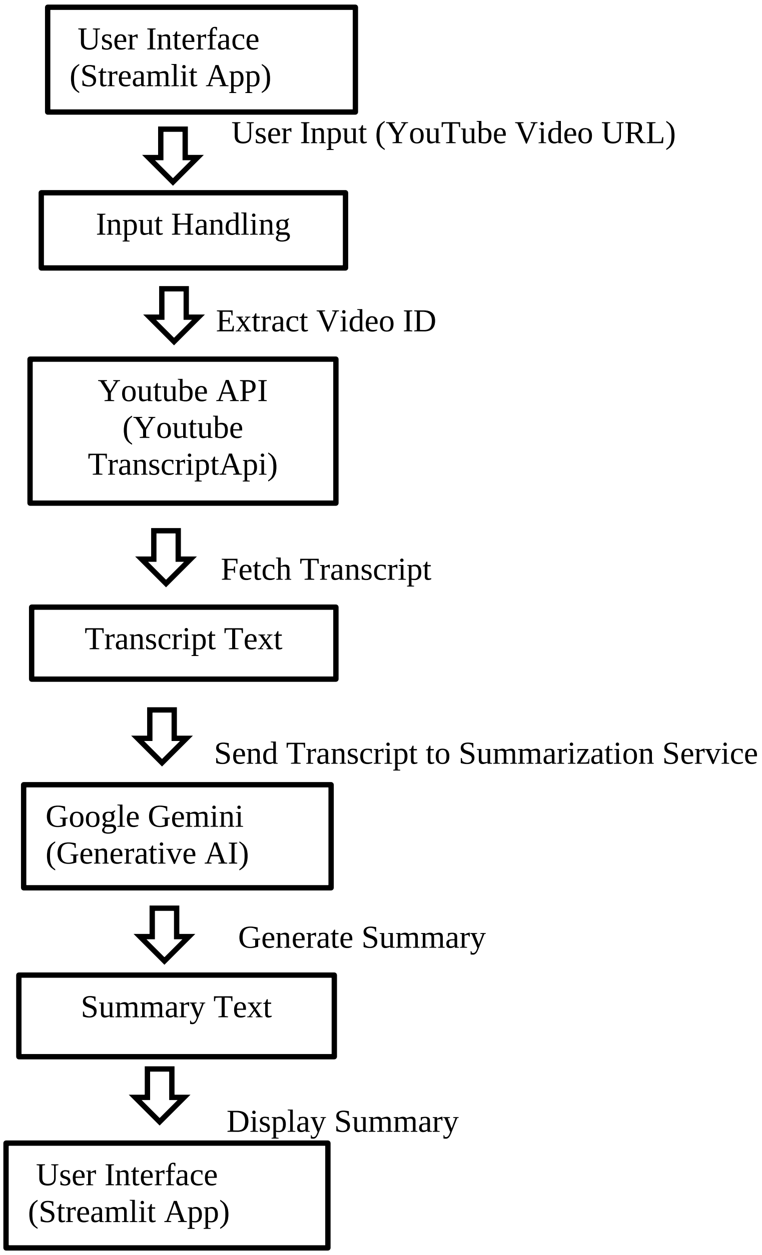


YouTube Transcript to Detailed Notes Converter

Figure:3.3.1

Figure 3.3.1.a

System Architecture Diagram



User Interface (Streamlit App)
User Input (YouTube Video URL)
Input Handling
Extract Video ID
Youtube API (Youtube TranscriptApi)
Fetch Transcript
Transcript Text
Send Transcript to Summarization Service
Google Gemini (Generative AI)
Generate Summary
Summary Text
Display Summary
User Interface (Streamlit App)

3.4. Overview of Project Description

The project is a Streamlit application designed to convert YouTube video transcripts into concise and informative summaries. By utilizing the YouTube Transcript API, the application extracts the spoken content from a given video link, enabling users to easily access the main points of the video without watching it in full. The extracted transcript is then processed using Google Gemini Pro's generative AI capabilities, which summarizes the content into a structured format, highlighting the key takeaways within a limit of 250 words. The user-friendly interface allows users to input a YouTube video link, view a thumbnail of the video, and obtain detailed notes with just a click. This tool is particularly beneficial for students, researchers, and professionals seeking to efficiently digest video content for study or reference purposes.

CHAPTER 4

PROJECT REQUIREMENTS

4.1 Software used

The code utilizes several key software components and libraries to function effectively. Streamlit is employed as the primary framework for building the web application, providing an interactive user interface that allows users to input YouTube video links and display results seamlessly. The dotenv library is used to manage environment variables, enabling secure handling of sensitive information such as API keys. The google.generativeai library is integrated to leverage the generative capabilities of Google's Gemini Pro model, which is responsible for summarizing the video transcripts. Additionally, the youtube_transcript_api library is utilized to extract the transcript of the YouTube videos, allowing the application to retrieve and process spoken content efficiently. Together, these software components create a cohesive and functional application that enhances the user experience by simplifying the process of obtaining video summaries.

4.2 Libraries and frameworks used

The list of Python libraries and frameworks used in the development of this project is mentioned in this section.

- **Streamlit:** A popular open-source framework for building interactive web
- applications in Python. It allows for the creation of user-friendly interfaces with minimal effort, making it ideal for data-driven applications.
- **dotenv**: This library is used for loading environment variables from a .env file, which helps manage sensitive information such as API keys securely without hardcoding them into the source code.
- **google.generativeai**: This library provides access to Google's Gemini Pro generative AI model, which is used in the application to generate summaries of the video transcripts.
- **youtube_transcript_api**: A Python library that allows users to retrieve transcripts for YouTube videos easily. It handles the extraction of spoken content from videos, enabling the application to summarize the text.

These libraries and frameworks work together to create a streamlined application that extracts, processes, and summarizes content from YouTube videos.

4.3 Overview of Project Requirements

This chapter briefs out on the requirements of the proposed project. The project is entirely a software product; therefore, the requirements are only software- based, and no hardware components are used in this project. The chapter briefs on the programming languages used and the libraries/packages used in the development of the system along with their use case.

CHAPTER 5

IMPLEMENTATION

5.1 Introduction

The YouTube Transcript Summarizer is a web application built using Streamlit that allows users to input a YouTube video link and automatically generates a concise summary of the video's transcript. By leveraging the YouTube Transcript API and Google's generative AI, the application enhances content accessibility and provides users with key insights in an easily digestible format.

5.2 Setting Up the Environment

Before running the code, ensure you have the necessary libraries installed. You can do this by running:

```
pip install streamlit python-dotenv google-generativeai youtube-transcript-api
```

You will also need to set up a .env file in your project directory to store your Google API key. The .env file should look like this:

```
GOOGLE_API_KEY=your_google_api_key_here
```

5.3 Coding:

```
import streamlit as st
from dotenv import load_dotenv
import os
import google.generativeai as genai
from youtube_transcript_api import YouTubeTranscriptApi

# Load environment variables
load_dotenv()
genai.configure(api_key=os.getenv("GOOGLE_API_KEY"))
```

```
# Prompt for summarization
prompt = """You are a YouTube video summarizer. You will be taking the transcript
text
and summarizing the entire video and providing the important summary in points
within 250 words. Please provide the summary of the text given here:  """
# Function to extract transcript details
def extract_transcript_details(youtube_video_url):
    try:
        # Extract video ID from the URL
        video_id = youtube_video_url.split("=")[1]
        # Fetch the transcript using YouTubeTranscriptApi
        transcript_text = YouTubeTranscriptApi.get_transcript(video_id)
        # Join the transcript text into a single string
        transcript = " ".join([i["text"] for i in transcript_text])
        return transcript
    except Exception as e:
        st.error(f"Error fetching transcript: {e}")
        return None
# Function to generate summary using Google Gemini Pro
def generate_gemini_content(transcript_text, prompt):
    model = genai.GenerativeModel("gemini-pro")
    # Generate content using the model
    response = model.generate_content(prompt + transcript_text)
    return response.text
# Streamlit application layout
st.title("YouTube Transcript to Detailed Notes Converter")
youtube_link = st.text_input("Enter YouTube Video Link:")

# Display video thumbnail if a link is provided
if youtube_link:
    video_id = youtube_link.split("=")[1]
    st.image(f"http://img.youtube.com/vi/{video_id}/0.jpg",
use_container_width=True)
```

```
# Button to get detailed notes
if st.button("Get Detailed Notes"):
    transcript_text = extract_transcript_details(youtube_link)
    if transcript_text:
        summary = generate_gemini_content(transcript_text, prompt)
        st.markdown("## Detailed Notes:")
        st.write(summary)
```

5.4 Explanation of Each Component

➢ **Imports:**

- **streamlit**: A library for creating web applications in Python.
- **dotenv**: A library to load environment variables from a **.env** file.
- **os**: A module to interact with the operating system.
- **google.generativeai**: A library to interact with Google’s generative AI models.
- **youtube_transcript_api**: A library to fetch transcripts from YouTube videos.

➢ **Loading Environment Variables:**

The **load_dotenv()** function loads the environment variables from the **.env** file, allowing access to the Google API key.

➢ **Prompt for Summarization:**

A prompt is defined to instruct the AI model on how to summarize the transcript.

➢ **Function to Extract Transcript:**

- **extract_transcript_details(youtube_video_url)**:

  This function takes a YouTube video URL, extracts the video ID, and retrieves the transcript using the **YouTubeTranscriptApi**. It concatenates the transcript text into a single string and returns it. If an error occurs, it displays an error message using **st.error()**.

➢ **Function to Generate Summary:**

- **generate_gemini_content(transcript_text, prompt)**: This function takes the transcript text and the prompt, initializes the Google Gemini Pro model, and generates a summary.

It returns the generated summary text.

- **Streamlit Application Layout**:

  The application title is set using **st.title()**.

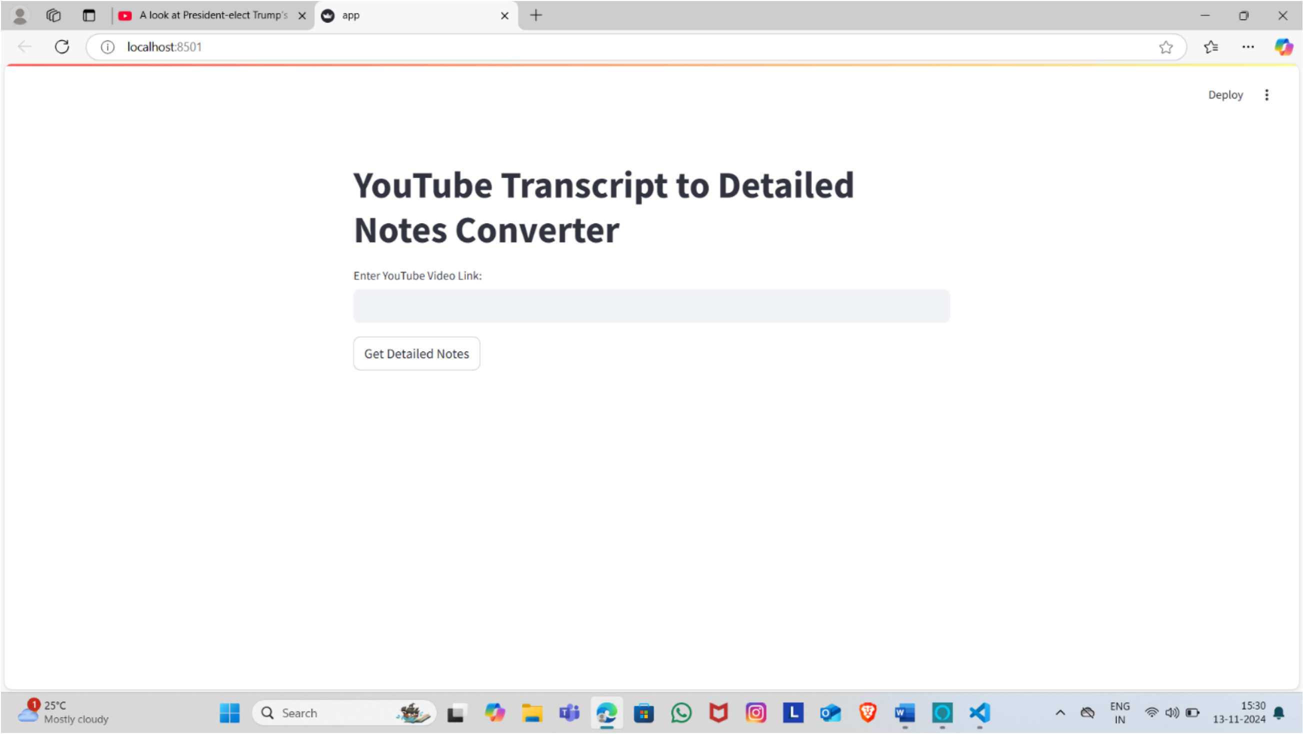
  A text input field is created for users to enter the YouTube video link.

  If a link is provided, the video thumbnail is displayed using the video ID extracted from the URL.

  A button labeled "Get Detailed Notes" triggers the summarization process. When clicked, it calls the **extract_transcript_details()** function to fetch the transcript and then generates a summary using the **generate_gemini_content()** function. The summary is displayed in a markdown format.

CHAPTER 6

SNAPSHOTS

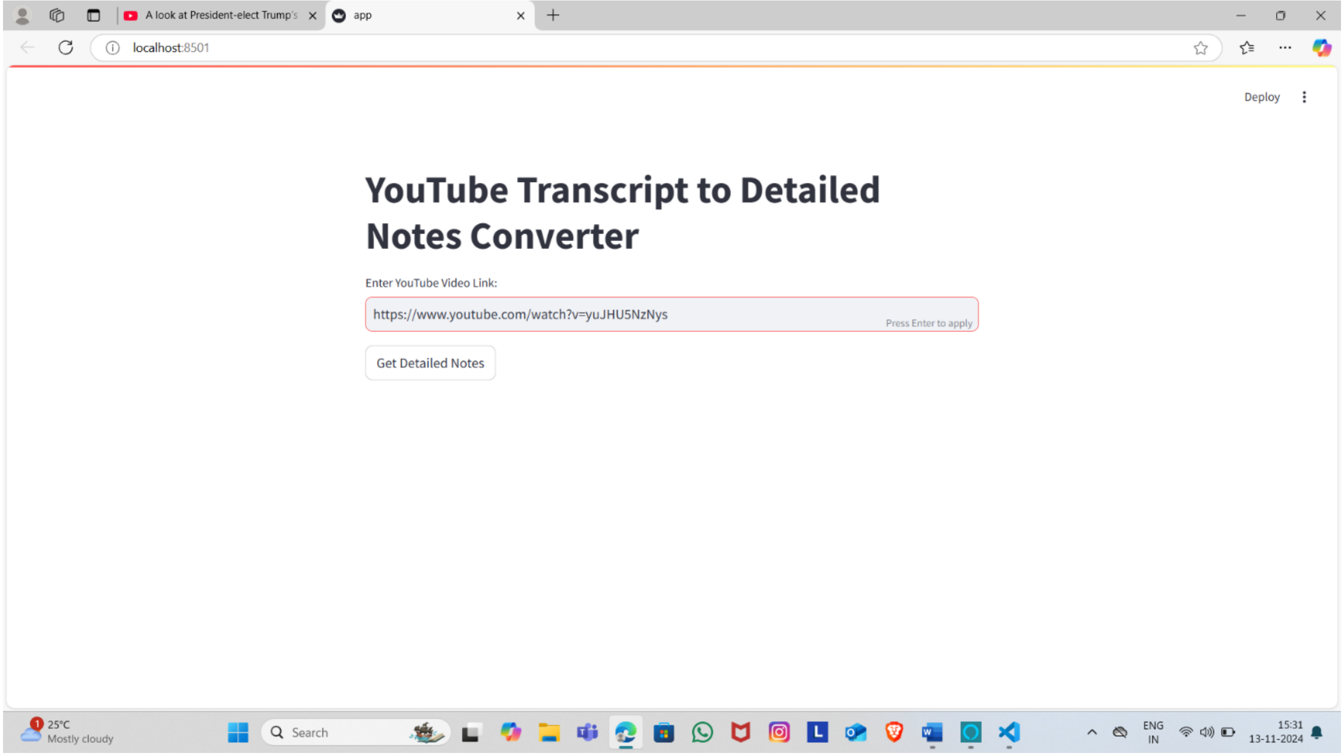


YouTube Transcript to Detailed Notes Converter

Enter YouTube Video Link:

Get Detailed Notes

Fig 6.1: User Interface



YouTube Transcript to Detailed Notes Converter

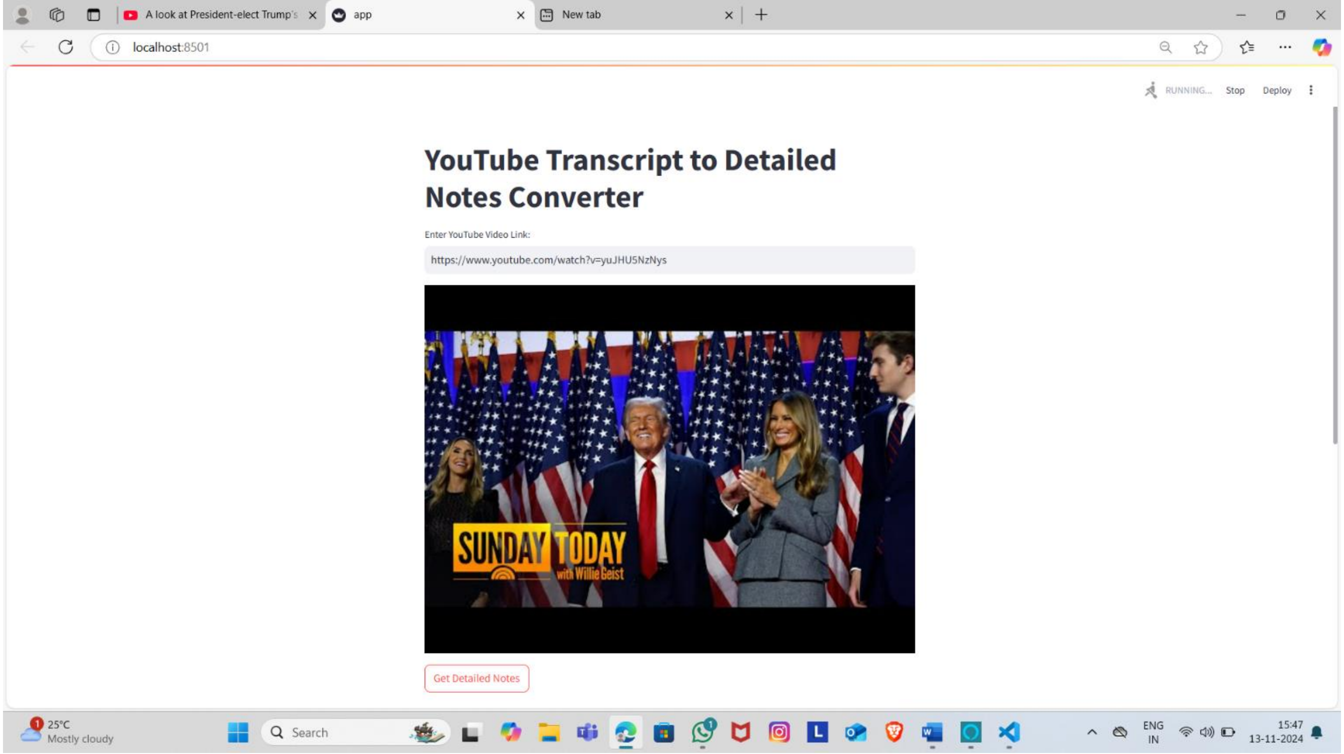
Enter YouTube Video Link:

https://www.youtube.com/watch?v=yuJHU5NzNys

Press Enter to apply

Get Detailed Notes

Fig 6.2: User Interface (Entering URL)

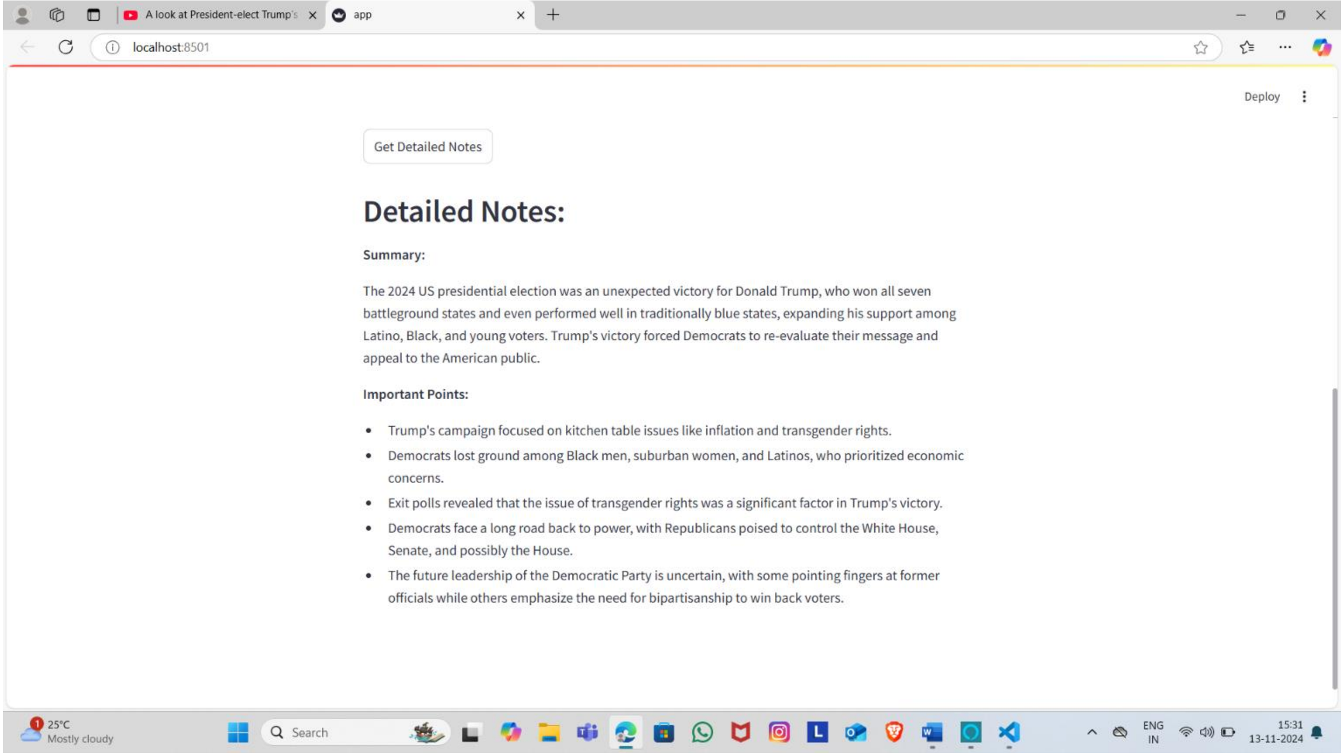


YouTube Transcript to Detailed Notes Converter

Enter YouTube Video Link:

https://www.youtube.com/watch?v=yuJHU5NzNys

Fig 6.3: Displaying Thumbnail of URL



Detailed Notes:

Summary:

The 2024 US presidential election was an unexpected victory for Donald Trump, who won all seven battleground states and even performed well in traditionally blue states, expanding his support among Latino, Black, and young voters. Trump's victory forced Democrats to re-evaluate their message and appeal to the American public.

Important Points:

- Trump's campaign focused on kitchen table issues like inflation and transgender rights.
- Democrats lost ground among Black men, suburban women, and Latinos, who prioritized economic concerns.
- Exit polls revealed that the issue of transgender rights was a significant factor in Trump's victory.
- Democrats face a long road back to power, with Republicans poised to control the White House, Senate, and possibly the House.
- The future leadership of the Democratic Party is uncertain, with some pointing fingers at former officials while others emphasize the need for bipartisanship to win back voters.

Fig 6.4: Summarized Output Of The Entered URL

CHAPTER 7

CONCLUSION

Started with the motive of contributing to the applications of text summarization, the system from the start thrived to stick to its initial goal and in the end, was successfully implemented. Even though the project has been completed, the journey doesn’t end here, the project has a huge scope of expansion which can augment the growth of the delivered system. The scopes and enhancements of the project are proposed in the following subchapters.

7.1 Conclusion

Set out to look into the field of text summarizing and choose one or more methods for a prototype. The prototype should be able to generate legible summaries of video transcripts automatically. Furthermore, the quality of the generated summaries should be assessed in a uniform manner. Therefore, we can come to the conclusion that the set-out project, “Youtube Transcript Summarizer” has been successfully planned, proposed and implemented.

The project's goal of implementing a prototype summarizer has been met, although the summarizer's usefulness could be improved with future code enhancements. Furthermore, some of the more complex approaches, some of which rely on external lexical sources, offer an intriguing approach to improving the quality of the implemented summarizer.

7.2 Future Enhancement

There are several potential enhancements that could be implemented to improve the YouTube transcript summarizer application:

- **Multi-Language Support**: Expand the application's capabilities to support transcripts in multiple languages. This could involve integrating translation services to provide summaries in the user's preferred language.

- **Custom Summary Length**: Allow users to specify the desired length of the summary (e.g., short, medium, long) to cater to different preferences for detail and conciseness.

- **Keyword Extraction**: Implement a feature to extract and highlight key phrases or keywords from the transcript, providing users with quick insights into the main topics discussed in the video.

- **Sentiment Analysis**: Incorporate sentiment analysis to give users an understanding of the emotional tone of the video content, which could be useful for reviews or opinion pieces.

- **Feedback Mechanism**: Implement a feedback system where users can rate the quality of the summaries, allowing for continuous improvement of the summarization algorithm.

7.3 Summary

This chapter, in detail, draws out the conclusion of the project upon the implementation of the proposed system in the beginning. It briefs out the completion of the project in regard to its vision and abstract and in the later part talks about the future of the same system, on how it can be enhanced for a higher level of its scope.

CHAPTER 8

REFERENCES

[1] I. Awasthi, K. Gupta, P. S. Bhogal, S. S. Anand and P. K. Soni, "Natural Language Processing (NLP) based Text Summarization - A Survey," 2021 6th International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 2021.

[2] Adhika Pramita Widyassari, Supriadi Rustad, Guruh Fajar Shidik, Edi Noersasongko, Abdul Syukur, Affandy, De Rosal Ignatius Moses Setiadi, Review of automatic text summarization techniques &methods, Journal of King Saud University - Computer and Information Sciences,2020, ISSN 1319-1578.

[3] P. R. Dedhia, H. P. Pachgade, A. P. Malani, N. Raul, and M. Naik, "Study on Abstractive Text Summarization Techniques," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020.

[4] A. Dilawari and M. U. G. Khan, "ASoVS: Abstractive Summarization of Video Sequences," in IEEE Access, vol. 7, 2019.

[5] Denis Jouvet, David Langlois, Mohamed Amine Menacer, Dominique Fohr, Odile Mella, et al.. Adaptation of speech recognition vocabularies for improved transcription of YouTube videos. Journal of International Science and General Applications, 2018.

[6] Dictionary.com. (n.d.). Summary definition & meaning. Dictionary.com. Retrieved March 15, 2022, from https://www.dictionary.com/browse/summary

[7] Mehta, P., & Majumder, P. (2019). Introduction. From Extractive to Abstractive Summarization: A Journey, 1–9. https://doi.org/10.1007/978-981-13-8934-4_1

[8] https://www.sciencedirect.com/topics/computer-science/text-summarization

[9] Sourav Biswas1, Atul Kumar Patel2, YouTube Transcript Summarizer To Summarize the content of YouTube, Volume: 09 Issue: 04 | Apr 2022

[10] Vijay Mane, Kshitija Dupare, Prashik Dupare, Tushar Ganvir, Rutvik Ghode, YouTube Transcript Summarizer using HuggingFace Transformer, Vishwakarma Institute of Technology, Pune, Maharashtra, India-411037