YouTube Transcript Summarizer

A

MAJOR PROJECT REPORT

Submitted in partial fulfillment of the requirements
For the award of the degree of

BACHELOR OF TECHNOLOGY
In
Computer Science and Engineering

Submitted to

RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA,

BHOPAL (M.P.)

Submitted by

Vijay Pandey (0115CS201116)

Rohit Gupta (0115CS201089)

Prashant Anand (0115CS201077)

Under the Guidance of

PROF. Mridula

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NRI INSTITUTE OF INFORMATION SCIENCE & TECHNOLOGY

BHOPAL

NRI INSTITUTE OF INFORMATION SCIENCE & TECHNOLOGY - BHOPAL (M.P.)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We hereby declare that the Project entitled **“YouTube Transcript Summarizer"** is our own work conducted under the supervision of **Guide, PROF. Mridula**, Department Of Computer Science And Engineering at **NRI Institute of Information Science & Technology, Bhopal.**

We further declare that to the best of our knowledge this report does not contain any part of work that has been submitted for the award of any degree either in this institute or in other institute without proper citation.

Vijay Pandey (0115CS201116)
Rohit Gupta (0115CS201089)
Prashant Anand (0115CS201077)

NRI INSTITUTE OF INFORMATION SCIENCE & TECHNOLOGY - BHOPAL (M.P.)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the work done in this project entitled **“YouTube Transcript Summarizer"** being submitted by Vijay Pandey (0115CS201116), Rohit Gupta (0115CS201089), Prashant Anand (0115CS201077) in partial fulfillment of the requirement for the award of the degree of the **Bachelor of Technology (Computer Science and Engineering)** to Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal (M.P.) is a record of bonafide piece of work, carried out by them under our supervision and guidance in the **Department of Computer Science and Engineering**, **NRI Institute of Information Science and Technology, Bhopal (M.P.).**

Guided By	**Approved By**
Prof. Mridula	**Prof. Anurag Shrivastava**
Assistant Professor Computer Science and Engineering NIIST, Bhopal	Associate Professor and Head Computer Science and Engineering NIIST, Bhopal

Prof. Puran Gour

Principal
NIIST, Bhopal

NRI INSTITUTE OF INFORMATION SCIENCE & TECHNOLOGY - BHOPAL (M.P.)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ACKNOWLEDGMENT

The satisfaction that accompanies the successful completion of the task would be incomplete without the mention of the people whose ceaseless cooperation made it possible whose constant guidance and encouragement crown all efforts with success. Firstly, I would like to thanks **Dr. Puran Gour**, Principal under whose supervision I choose this project under his precious guidance and suggestions made to complete the project. I am heartily thankful to him for providing me his valuable suggestion. A am very thankful **to Prof. Anurag Shrivastava** HOD, CSE and all the staff for their valuable guidance and support during the entire period.

Place & Date **TEAM MEMBERS**

Vijay Pandey (0115CS201116)
Rohit Gupta (0115CS201089)
Prashant Anand (0115CS201077)

NRI INSTITUTE OF INFORMATION SCIENCE & TECHNOLOGY - BHOPAL (M.P.)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ABSTRACT

An automatic YouTube transcript summarizer is a tool that generates a summary of the content in a YouTube video by analyzing the transcript of the video's.
This is a useful tool for users who want to quickly understand the main points of a video without having to watch the entire video.

The system processes transcripts, identifies important topics, extracts meaningful sentences, and compiles them into a meaningful summary, making it easier for users to grasp the video's content and decide whether to watch it.
This is able to accurately and efficiently extract the main points and key information form the transcript.

TABLE OF CONTENTS

S. No.	Topics
1	Declaration
2	Certificate
3	Acknowledgement
4	Abstract

CONTENTS

Chapter 1- INTRODUCTION

1.1 Objective:-

The objective of the YouTube Transcript Summarizer Project is to develop an automated system that accurately extracts key information from YouTube video transcripts and generates informative summaries to accurately and efficiently extract the main points and key information from the transcript.   .

1.2 Scope:-

The goal of this project is to develope an automatic YouTube transcript summarizer that generates a summary of the content in a YouTube video by analyzing the transcript of the video's.

YouTube videos are a huge source of information in today's time but learning anything from them is a task as it takes a lot of time in grabbing information from a video. The videos contain a lot of unwanted and wasteful information which can be skipped by summarizing the transcript of the YouTube video.

1.3 Problem definition:-

In today's digital age, the vast amount of content available on platforms like YouTube, presents a significant challenge for users to efficiently access and consume relevant information. YouTube videos, in particular, often contain valuable insights, tutorials, or entertainment, but their length can be daunting. Users often spend a considerable amount of time scrolling through videos to find the content they need, and even after identifying a relevant video, they may not have the time or patience to watch it in its entirety.

To address the problem of summarizing YouTube video content effectively, we propose the development of a YouTube Video Transcript Summarizer (YTTS). YTTS will generate meaningful summaries of YouTube videos.

1.4 Technologies Used:- Technologies that are often used in projects involving YouTube transcript summarization

1.4.1 Software Requirements:-

- **Python:** A widely used programming language for web scraping, data processing, and machine learning tasks.
- **Django:** A high-level Python web framework often used for web application development, including handling web requests and responses.
- **YouTube API:** An API provided by YouTube that allows developers to interact with YouTube's features, including accessing video information and transcripts.
- **HTML/CSS:** Front-end technologies used for creating the user interface and handling client-side interactions on the web application.

1.4.2 Hardware Requirements:-

Operating System: Win 7 or more(32 bit)

RAM : 8 GB (recommended)

Must have any **Web Browser**

Chapter 2- Software Requirement Specifications

2.1 Introduction:-

The “**YouTube Transcript Summarizer**” generate a summary of the YouTube video, by analyzing the transcript (subtitle) of the video content.

The "YouTube Transcript Summarizer" app automates video content comprehension by extracting and processing YouTube video transcripts. The application offers a user-friendly interface, making it easy to input video URLs and access informative summaries, revolutionizing how users engage with online video content.

2.2 Overall Description

2.2.1 Product Architecture:-

The architecture comprises various modules that work together to achieve the project's objective of summarizing YouTube video transcripts.

➢ **User Interface (UI) Module**

- Description: The UI module allows users to interact with the system by entering a YouTube video URL and initiating the summarization process.
- Technologies: HTML, CSS, JavaScript, front-end frameworks.

➢ **YouTube API Integration Module**

- Description: This module interfaces with the YouTube API to fetch the transcript for the specified YouTube video URL.
- Technologies: YouTube API, Python.

- **Text Processing Module**
- Description: Responsible for cleaning, preprocessing, and organizing the raw transcript data.
- Technologies: Python libraries, Django

- **Summarization Engine**
- Description: Utilizes summarization algorithms to generate a concise summary from the processed transcript data.
- Technologies: Summarization algorithms (e.g., Extractive Summarization, Abstractive Summarization) Python.

This architecture ensures that the system follows a modular and efficient design, allowing for scalability, maintainability, and effective summarization of YouTube video transcripts.

2.2.2 Product Functional Requirements:-

Product Functional Requirements define the specific functions and capabilities that the YouTube Transcript Summarizer application must have to meet its objectives and user needs. Here's a list of functional requirements for the YouTube Transcript Summarizer.

- **1. Transcript Extraction:**
- Requirement 1.1: The system must allow users to input a valid YouTube video URL.
- Requirement 1.2: The system must extract the transcript of the specified YouTube video using the YouTube API.
- **2. Text Processing:**
- Requirement 2.1: The system must clean and preprocess the raw transcript data to remove noise and irrelevant information.
- Requirement 2.2: The system must organize the cleaned transcript data for further processing.

- **3. Summarization:**
- Requirement 3.1: The system must generate a concise summary of the video transcript.
- Requirement 3.2: The summary should accurately represent the key points and context of the video content.
- **5. User Interface:**
- Requirement 5.1: The system must have an intuitive and user-friendly interface.
- Requirement 5.2: Users should be able to initiate the summarization process easily by providing a YouTube video URL.
- **6. Display of Results:**
- Requirement 6.1: The system must display the summary of the video transcript.
- Requirement 6.2: The system should also display the extracted keywords for reference.
- **8. Performance:**
- Requirement 8.1: The system should provide a summary within a reasonable time frame, even for longer video transcripts.
- Requirement 8.2: The system should be able to handle a moderate number of concurrent users without a significant degradation in performance.

These functional requirements ensure that the YouTube Transcript Summarizer application can effectively extract transcripts, process the data, generate summaries, and display the results in a user-friendly manner, meeting the needs of its intended users.

2.3 System Features:-

YouTube Transcript Summarizer System Features

- **1. Transcript Extraction:**
- Feature 1.1: Automatic Transcript Retrieval
  Description: The system should allow users to input a YouTube video URL and automatically fetch the video's transcript using the YouTube API.

- **2. Text Processing:**
- Feature 2.1: Text Cleaning
  Description: The system should clean the transcript data by removing special characters, timestamps, and any irrelevant information.
- Feature 2.2: Preprocessing
  Description: The system should preprocess the cleaned text, including sentence and word tokenization, and removing stop words.
- **3. Summarization:**
- Feature 3.1: Extractive Summarization
  Description: The system should use extractive summarization techniques to generate a summary by selecting the most significant sentences from the preprocessed text.
- Feature 3.2: Abstractive Summarization (Optional)
  Description: Optionally, the system could use abstractive summarization techniques to generate a more concise and coherent summary.
- **4. User Interface:**
- Feature 4.1: User-Friendly Input
  Description: The system should provide an intuitive and user-friendly interface for users to input the YouTube video URL.
- Feature 4.2: Summary Display
  Description: The system should present the summarized content in an easy-to-read format for users.

- **5. Display of Results:**
- Feature 5.1: Concise Summary Display
  Description: The system should display a concise summary of the video transcript, capturing the main ideas and key points.
- Feature 5.2: Keyword Highlighting
  Description: The system should highlight the extracted keywords within the summary for quick reference.

- **6. Error Handling:**

- Feature 6.1: Input Validation
  Description: The system should validate the YouTube video URL input and provide appropriate error messages for invalid or incorrect inputs.

These system features provide a comprehensive overview of the essential functionalities required to build an effective YouTube Transcript Summarizer.

2.4 System models:-

In the context of a YouTube Transcript Summarizer, scenarios represent real-life use cases or situations where the system is utilized to fulfill a specific objective. Here are four scenarios that demonstrate how users might interact with the YouTube Transcript Summarizer:

- **2.4.1 Scenarios:-**

The user copies the URL of the YouTube video and pastes it into the summarizer's input field. The system retrieves the transcript of the video using the YouTube API.

The system processes the transcript, generates a summary, and displays it to the user.

Outcome: The user obtains a concise summary of the video's content, allowing for a quick understanding of the main points discussed.

- **2.4.2 Software Development Model:-**

Choosing an appropriate software development model is crucial for the success of a project like the YouTube Transcript Summarizer. Let's outline the Agile development model as it is often a suitable choice for projects that require flexibility, quick iterations, and the ability to adapt to changing requirements.

- **Overview:**

Agile is an iterative and incremental approach to software development that emphasizes flexibility, collaboration, and customer satisfaction.

The ability to respond to change and reprioritize based on customer feedback and evolving requirements.

Phases:

1. **Requirements Gathering:**

Gather initial requirements and features from stakeholders, defining the scope of the project.

Create a product backlog containing all desired features and functionalities.

2. **Sprint Planning:**

Select a subset of features from the product backlog for the upcoming iteration (sprint).Break down features into tasks and estimate the effort required for each task.

3. **Sprint Development:**

Develop the selected features within the defined sprint duration (e.g., 2-4 weeks).

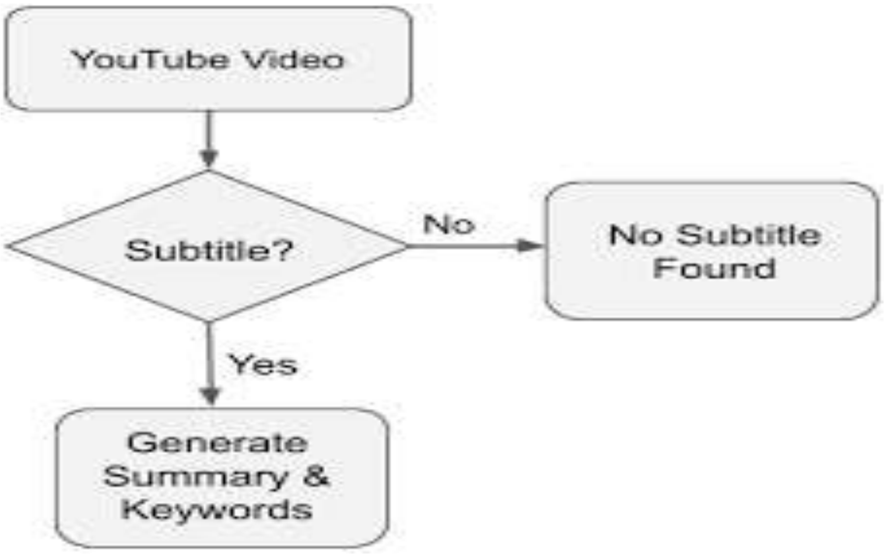
Conduct daily stand-up meetings to discuss progress, challenges, and plan the day's work.

4**. Sprint Review:**

Present the completed features to stakeholders for feedback.

Collect feedback and make necessary adjustments to the product backlog.

➢ **2.4.3 Use Case Model Description:-**



➢ **2.4.4 Object model:-**

The Object Model for the YouTube Transcript Summarizer provides an overview of the key objects and their relationships within the system. In this context, we'll focus on the main objects and their associations relevant to the functionality of the YouTube Transcript Summarizer:

1. YouTube Video

Attributes:

Video Id: String (unique identifier for a YouTube video)

Url: String (URL of the YouTube video)

Transcript: String (transcript extracted from the video)

2. Summarization Engine

Attributes:

Summary: String (the generated summary)

Methods:

Generate Summary(transcript: String): Generates a summary from the preprocessed transcript.

- **2.4.5 Dynamic models**

Sequence Diagram for YouTube Transcript Summarizer:-

Interaction 1: Extracting Transcript

User Interface -> Transcript Processor: User provides a YouTube video URL

Transcript Processor -> YouTube Video: Extracts transcript using YouTube API

Transcript Processor --> User Interface: Returns the extracted transcript

Interaction 2: Preprocessing Transcript

Transcript Processor -> Transcript Processor: Cleans the transcript (remove noise)

Transcript Processor -> Transcript Processor: Preprocesses the transcript (tokenization, stop words removal)

Interaction 3: Generating Summary

Transcript Processor -> Summarization Engine: Sends preprocessed transcript

Summarization Engine --> Transcript Processor: Generates a summary

Interaction 4: Displaying Summary and Keywords

Summarization Engine --> User Interface: Sends the summary

Keyword Extractor -> Summarization Engine: Extracts keywords

Keyword Extractor --> User Interface: Sends the keywords

User Interface: Displays the summary and keywords to the user.

These interactions showcase how the components interact during the process of summarizing a YouTube transcript. The User Interface takes input from the user, which triggers a series of actions involving the Transcript Processor, Summarization Engine, and Keyword Extractor to generate a summary and display it along with keywords to the user.

2.5 External Interface Requirements

- ### 2.5.1 User Interfaces:-

User Interfaces for YouTube Transcript Summarizer

1. **Input Screen**

   Description: This screen allows the user to input the YouTube video URL.

   Features: Input field to enter the YouTube video URL.

2. **Transcript Display Screen**

   Description: This screen displays the extracted transcript.

   Features: Display area to show the extracted transcript.

   Interaction: Shown after successful transcript extraction.

3. **Preprocessing Screen**

   Description: This screen displays the preprocessing progress.

   Features: Progress bar or loader indicating the preprocessing status.

   Interaction: Shown during transcript preprocessing.

4. **Summary Display Screen**

   Description: This screen displays the generated summary.

   Features: Display area to show the generated summary.

   Interaction: Shown after summary generation.

- ### 2.5.2 Hardware Interfaces

The YouTube Transcript Summarizer primarily operates as a software application and doesn't require specific hardware interfaces. However, it does interact with the user's device, which could include various hardware components. Here's a description of the relevant hardware interfaces:

1. Computer or Device

Description: The user interacts with the YouTube Transcript Summarizer using a computer, smartphone, tablet, or any device capable of running the application.

Requirements: An operating system that supports the application (e.g., Windows, macOS, Linux, Android, iOS).
Sufficient processing power and memory to run the application smoothly.

2. Internet Connection

Description: An active and stable internet connection is required for accessing YouTube and fetching the video transcript.
Requirements: Broadband or a reliable internet connection to stream YouTube content and access the YouTube API.

3. Storage

Description: Local storage on the user's device may be used to store temporary data, such as the fetched transcript, generated summary, or keywords.
Requirements: Adequate storage space to temporarily store the processed data during the summarization process.

4. User Input Devices

Description: Various input devices allow the user to interact with the application, such as a keyboard for typing the YouTube video URL and a mouse or touchpad for navigation.
Requirements: Functional and responsive input devices for ease of use and navigation within the application.

5. Display

Description: The application presents information to the user via a display screen on their device.
Requirements: A clear and functional display screen to present the extracted transcript, summary, keywords, and other information to the user.

➢ 2.5.3 Software Interfaces

Software interfaces enable the YouTube Transcript Summarizer to interact with other software components or services. Below are the key software interfaces for the YouTube Transcript Summarizer:

1. **YouTube Data API**

Description: The application interacts with the YouTube Data API to fetch video details, including the transcript.

Usage: Retrieving video details based on the provided YouTube video URL.

2. **Summarization Algorithms**

Description: The application employs summarization algorithms (e.g., extractive or abstractive) to generate a concise summary of the transcript.

Usage: Extracting key sentences or generating an abridged version of the transcript.

3. **Frontend Framework or Library**

Description: The application utilizes frontend frameworks or libraries to develop the user interface.

Usage: Building the graphical elements and enabling user interaction for a seamless user experience.

These software interfaces enable the YouTube Transcript Summarizer to interact with various components, services, and algorithms necessary for extracting, processing, summarizing, and presenting YouTube video transcripts in a condensed and understandable format. The application relies on these interfaces to deliver a functional and efficient summarization service to the users.

➢ 2.5.4 Communications Interfaces

Communications Interfaces for YouTube Transcript Summarizer:-

1. HTTP/HTTPS

Description: The application uses HTTP/HTTPS protocols to communicate with external servers, APIs, and fetch data.

Usage: Fetching YouTube video details and transcript data from the YouTube Data API.

2. YouTube Data API Endpoint

Description: The application interacts with the specific endpoint of the YouTube Data API to request video details.

Usage: Sending HTTP requests to the YouTube Data API endpoint to fetch video details using the provided YouTube video URL.

3. API Request and Response Formats

Description: The application uses predefined request and response formats when making API requests to the YouTube Data API.

Usage: Structuring requests (e.g., including the API key, video ID) and parsing responses to extract the necessary video details.

4. Internal Application APIs

Description: The application might have internal APIs for communication between frontend and backend components.

Usage: Enabling communication between frontend (user interface) and backend (processing, summarization) components within the application.

These communications interfaces facilitate the interaction of the YouTube Transcript Summarizer application with external services, APIs, and components. They ensure efficient data retrieval, real-time updates, error handling, and user feedback, contributing to a seamless user experience.

2.6 Nonfunctional Requirements

Nonfunctional requirements define the quality attributes of a system, specifying how it should behave, rather than what it should do. Below are the nonfunctional requirements for the YouTube Transcript Summarizer:

➢ 2.6.1 Performance Requirements

1. Performance/Response Time:
The system shall provide a summary within 10 seconds of initiating the summarization process.
2. Scalability: The system shall support concurrent requests from multiple users without significant performance degradation.

➢ 2.6.2 Security / Safety Requirements

Data Privacy:
The system shall ensure that user data and video URLs are handled securely and are not shared or exposed to unauthorized users.
Authentication and Authorization:
The system shall implement authentication and authorization mechanisms to ensure that only authorized users can access the application.

2.7 Constraints / Assumptions and Dependencies

➢ 2.7.1 Constraints

Internet Connectivity:
The system is dependent on a stable internet connection to access YouTube video data and its transcript through the YouTube Data API.

Third-Party Service Dependency:
The system is reliant on the YouTube Data API for fetching video details and transcripts. Any changes or restrictions to this API could impact the functionality of the system.

API Rate Limits:

The system must adhere to the rate limits imposed by the YouTube Data API. Exceeding these limits could temporarily disable certain functionalities.

Processing Capacity:

The summarization process heavily relies on the processing capacity of the system. Inadequate processing power may result in slower summarization or timeouts.

➢ 2.7.2 Assumptions

1. **Transcript Availability:**

It is assumed that the YouTube videos selected for summarization have machine-generated or human-generated transcripts available through the YouTube Data API.

2. **Transcript Quality:**

It is assumed that the quality and accuracy of the transcripts fetched from the YouTube Data API are sufficient for summarization.

3. **User-provided Video URLs:**

Users will provide valid and accessible YouTube video URLs for summarization.

4. **Sufficient System Resources:**

The system assumes that it has adequate resources (processing power, memory, storage) to handle multiple users and summarization requests concurrently.

➢ 2.7.3 Dependencies

1. **YouTube Data API:**

The system is dependent on the YouTube Data API to retrieve video details, including the transcript, based on the provided YouTube video URL.

2. **Natural Language Processing (NLP) Libraries:**

The system depends on NLP libraries and tools for preprocessing the transcript and generating the summary.

3. **Summarization Algorithms:**

The system relies on summarization algorithms (extractive or abstractive) to generate a concise summary of the transcript.

4. **Keyword Extraction Tools:**

The system may use external keyword extraction tools or libraries to extract important keywords from the transcript.

5. **Frontend and Backend Frameworks:**

The system depends on frontend and backend frameworks or languages for user interface development and server-side logic implementation.

6. **Web Browser:**

The system relies on web browsers to display the user interface and enable user interactions.

Chapter 3- System / Application Design

3.1 Design Methodology:-

The design methodology for developing the YouTube Transcript Summarizer involves a systematic approach to ensure the system is well-structured, efficient, and meets the specified requirements. We'll use an iterative and incremental design methodology, incorporating principles from Agile and iterative development.

- **Requirements Understanding and Analysis:**

  Initial step to understand and analyze requirements, constraints, assumptions, and dependencies.
- **High-Level Design (HLD):**

  Create a high-level architectural design.

  Identify key components and their interactions.
- **Detailed Design (LLD):**

  Refine each component identified in the high-level design.

  Develop UML diagrams (class, sequence, state) for detailed representation.
- **Prototype Development:**

Develop a prototype or MVP with essential features.

- **Iterative Development and Testing:**

  Implement and enhance specific functionalities in iterative cycles.

  Rigorously test each iteration to identify and fix bugs.
- **Continuous Integration and Deployment:**

  Implement CI/CD processes for smooth integration and deployment.
- **User Feedback and Iteration:**

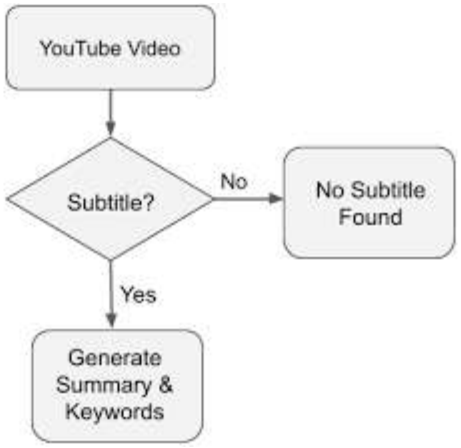
  Collect feedback and iterate based on user and stakeholder input.
- **Optimization and Performance Tuning:**

  Optimize the system for performance and scalability.
- **Documentation:**

  Maintain comprehensive documentation throughout the development process.
- **Final Testing and Quality Assurance:**

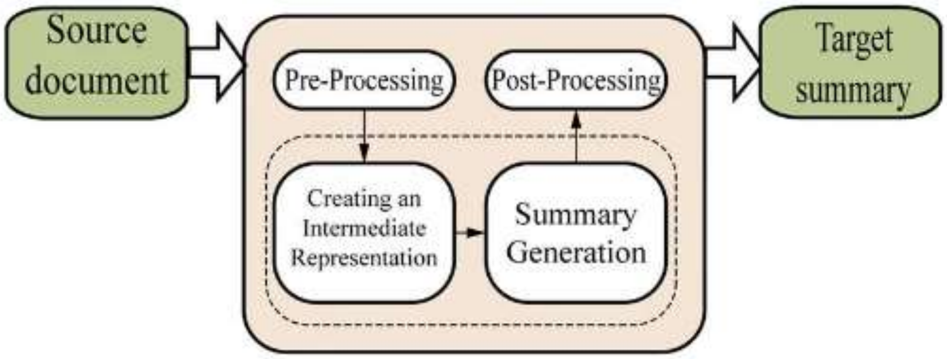
  Conduct final testing and quality assurance

3.2 Flowcharts:-



3.3 Database Design:-

- **3.3.1 Layout**



- **3.4 DFD (show different levels of DFDs)**

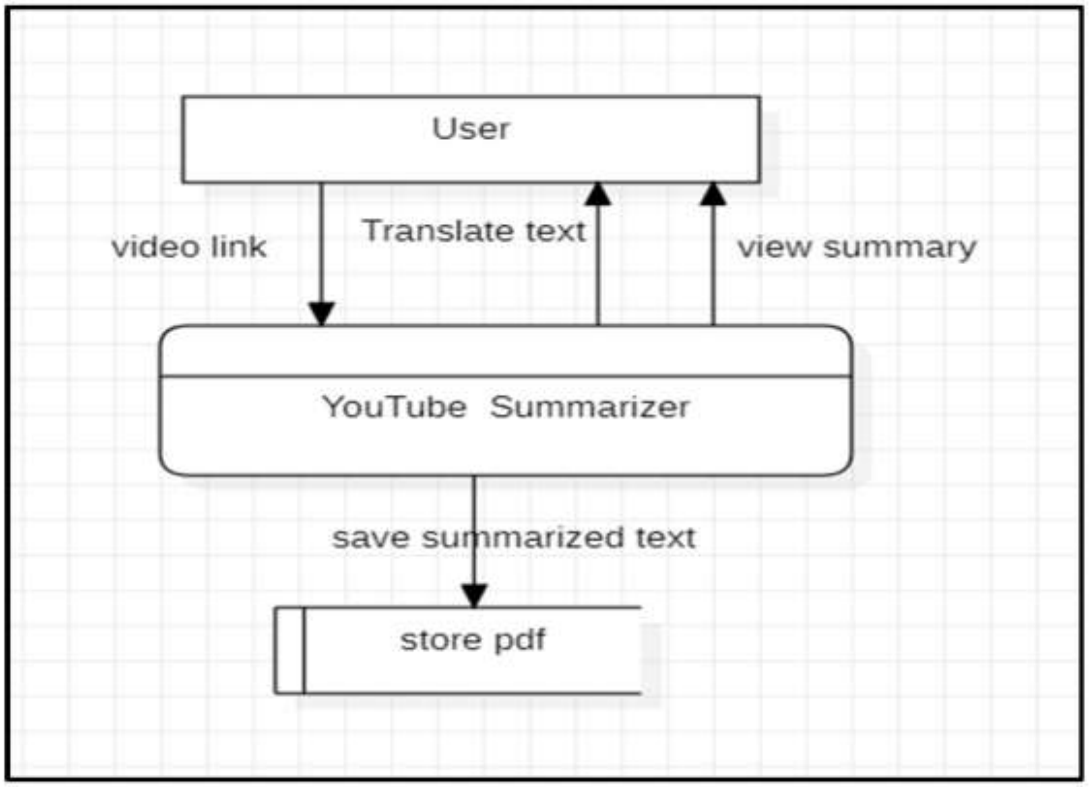


Fig - : LEVEL 0 DFD

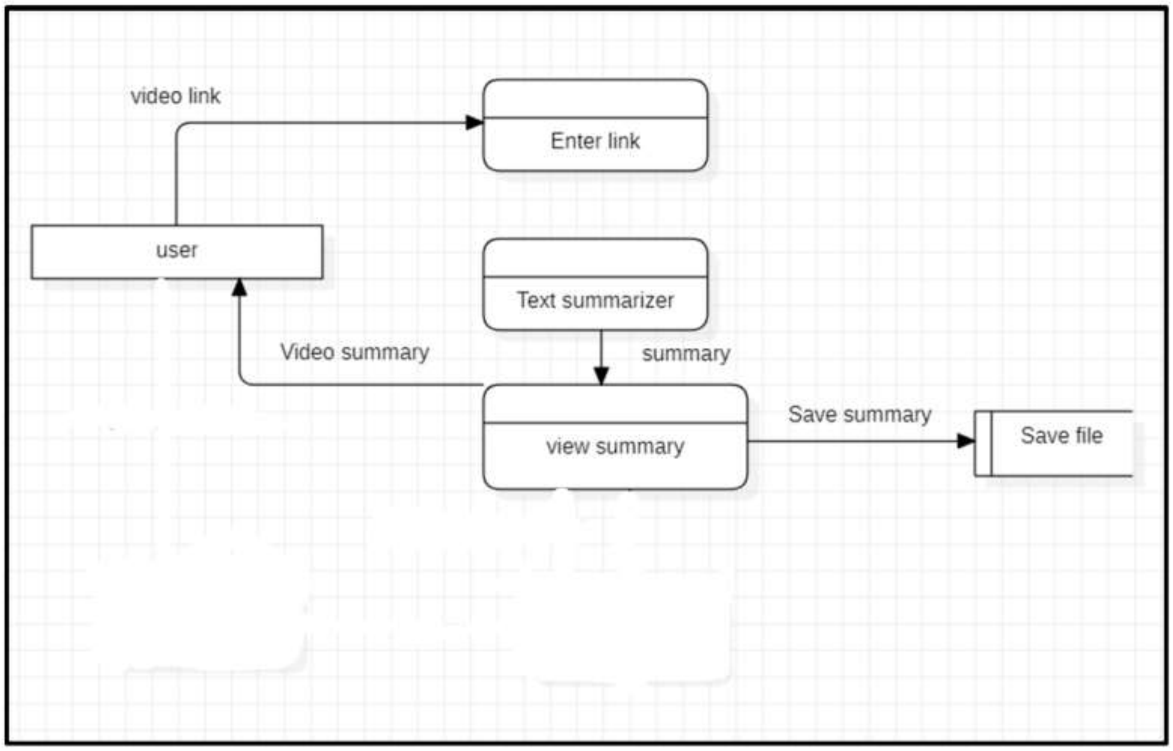


Fig - : LEVEL 1 DFD

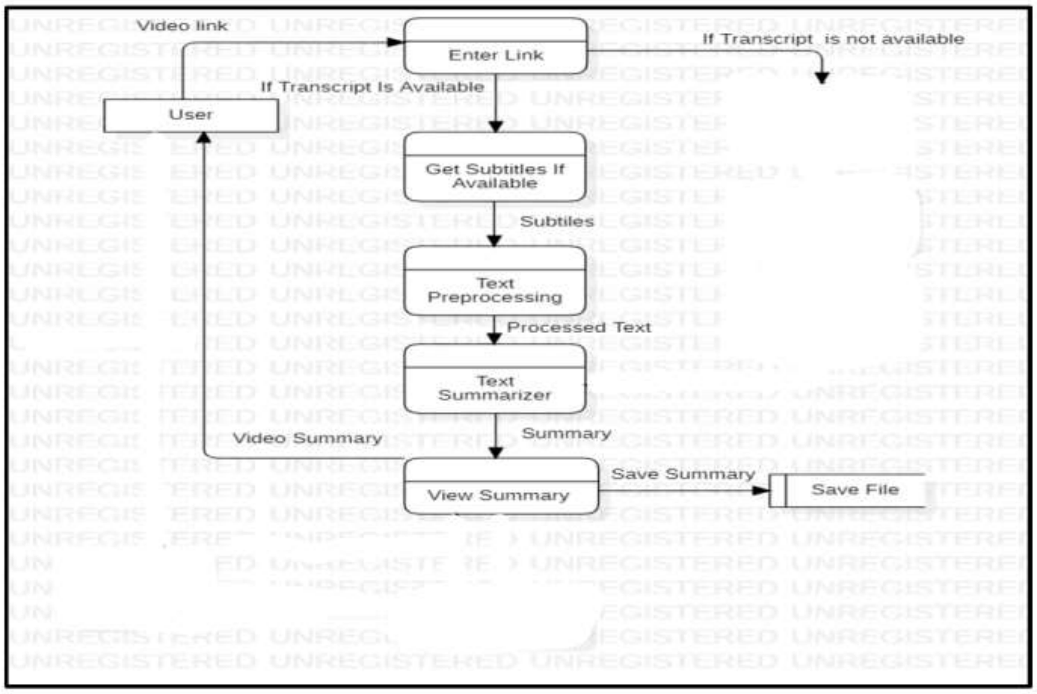


Fig - : LEVEL 2 DFD

Chapter 4- System Implementation

4.1 Coding:- The coding portion of this project is organized into files, each of which is written in segments.

4.1.1 Logical Implementation code :-

```
import youtube_transcript_api
from summarizer import Summarizer

def fetch_transcript(video_id):
    try:
        transcript =
youtube_transcript_api.YouTubeTranscriptApi.get_transcript(video_id)
        text = ' '.join([entry['text'] for entry in transcript])
        return text
    except youtube_transcript_api.transcripts.CouldNotRetrieveTranscript:
        print("Could not retrieve transcript.")
        return None

def generate_summary(text):
    model = Summarizer()
    summary = model(text)
    return summary

if __name__ == "__main__":
    # Replace with the actual YouTube video ID
    video_id = "your_youtube_video_id_here"

    # Fetch transcript
    transcript_text = fetch_transcript(video_id)

    if transcript_text:
        # Generate summary
        summary = generate_summary(transcript_text)

        # Print the summary
        print("Summary:")
        print(summary)
    else:
        print("Transcript fetch failed.")
```

File :- summarize.py

```
from pytube import YouTube
from summarizer.models import *
from youtube_transcript_api import YouTubeTranscriptApi
import re

############Summary Conversion
def summarize_video(youtube_url,video_id):
    try:
        yt = YouTube(youtube_url)
        video_title = yt.title
        video_transcripts = YouTubeTranscriptApi.get_transcript(video_id)
        transcript=[]
        for index,item in enumerate(video_transcripts):
            transcript.append(item['text'])

        video_stream = yt.streams.get_highest_resolution()
        video_path = video_stream.download()
        cleaned_transcript = ' '.join(transcript)
        cleaned_transcript = re.sub(r"[^\w\s]", "", cleaned_transcript)
        desired_length = 3
        sentences = cleaned_transcript.split('. ')
        summary = '. '.join(sentences[:desired_length])

        return summary

    except Exception as e:
        print(f"Error: {str(e)}")
        return None
```

File :- models.py

```
from django.db import models

class Video(models.Model):
    title = models.CharField( max_length=200)
    youtube_url = models.URLField()
    summary = models.TextField(blank=True,null = True)

    def __str__(self):
      return self.title
```

File:- Home.html

```
{% extends 'base.html' %}

{% block content %}
<nav class="navbar bg-primary">
    <div class="container-fluid">
      <a class="navbar-brand text-white">Video Summarizer</a>
      <form class="d-flex" role="search">
        <input class="form-control me-2" type="search" placeholder="Search" aria-
label="Search">
        <button class="btn  text-white bg-dark" type="submit">Search</button>
      </form>
    </div>
  </nav>


    <div class="container bg-secondary mt-5 py-3">
        <h1 class="mt-4 text-white">YouTube Video Summarizer</h1>
        <form method="post" action="{% url 'home' %}">
            {% csrf_token %}
            <div class="form-group">
                <label for="youtube_url " class="text-white">YouTube URL:</label>
                <input type="text" class="form-control" id="youtube_url"
name="youtube_url" placeholder="Enter YouTube URL" required>
            </div>
            <button type="submit" class="btn btn-primary my-2">Summarize</button>
        </form>
    </div>

    <footer class="footer">
        <div class="container mt-5">
            <p class="text-center">&copy; 2023 YouTube Video Summarizer. All rights
reserved.</p>
        </div>
    </footer>
{% endblock %}
```

File:- Summary.html

```
{% extends 'base.html' %}

{% block content %}
<nav class="navbar bg-primary">
    <div class="container-fluid">
      <a class="navbar-brand text-white " href="#">Video Summarizer</a>
      <form class="d-flex" role="search">
        <input class="form-control me-2" type="search" placeholder="Search" aria-
label="Search">
        <button class="btn  text-white bg-dark" type="submit">Search</button>
      </form>
    </div>
  </nav>

<div class="container rounded py-3 mt-5 bg-info">

    <h1 class="mt-4 "> "{{ video.title }}"</h1>
    <div class="card mt-4  ">
        <div class="card-body">
            <h5 class="card-title">Video Title: {{ video.title }}</h5>
            <h6 class="card-subtitle mb-2 text-muted">Video URL: <a href="{{ video.youtube_url
}}">{{ video.youtube_url }}</a></h6>
            <p class="card-text fw-bold">Transcript: {{ video.summary }}</p>
        </div>
    </div>
</div>
<footer class="footer">
    <div class="container">
        <p class="text-center">&copy; 2023 YouTube Video Summarizer. All rights reserved.</p>
        <p class="text-center">Developed by PRASHANT, VIJAY , ROHIT</p>
    </div>
</footer>

{% endblock %}
```

4.1.2 Database Implementation code

File:- Setting.py

```
Django settings for video_summarizer project.

Generated by 'django-admin startproject' using Django 2.2.7.

For more information on this file, see
https://docs.djangoproject.com/en/2.2/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/2.2/ref/settings/
"""
import os
# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/2.2/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = '__!v$gl1!r3th85(p3b@v!oc!&6*nv2+jsx@so@$yvl^sc$w2='

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []

# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'summarizer',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]
ROOT_URLCONF = 'video_summarizer.urls'
ROOT_URLCONF = 'video_summarizer.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },},]

WSGI_APPLICATION = 'video_summarizer.wsgi.application'

# Database
# https://docs.djangoproject.com/en/2.2/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
# Password validation
# https://docs.djangoproject.com/en/2.2/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]
# Internationalization
# https://docs.djangoproject.com/en/2.2/topics/i18n/

LANGUAGE_CODE = 'en-us'
TIME_ZONE = 'UTC'
USE_I18N = True
USE_L10N = True
USE_TZ = True
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/2.2/howto/static-files/
STATIC_URL = '/static/'
```

File:-Manage.py

```
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys


def main():
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'video_summarizer.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)


if __name__ == '__main__':
    main()
```

4.2 How to run code (Setup):-

~~~~
# Django_Youtubevideo_Summary
Implemented YouTube Data API integration to fetch video details, including metadata, captions,
and transcripts, enabling efficient data retrieval for summarization using Django.

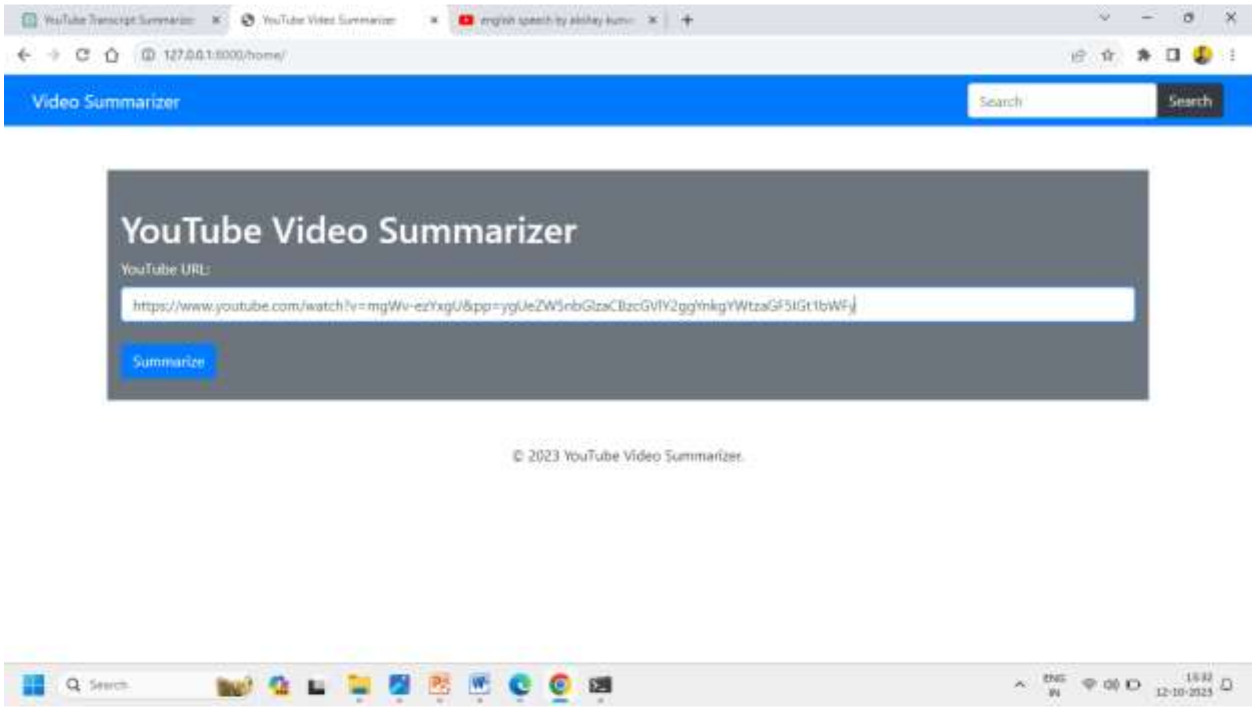
## Installing
Step by step commands on how to run this project on your computer.

1)- Install Virtualenv
```
pip install virtualenv
```
2)- Create Virtualenv
```
virtualenv venv
```
3)- Activate virtual env
```
source env/bin/activate
```
4)- Install requirements
```
pip install -r requirements.txt
```
Note: Above lines are required for first time installation.

5)- Execute below commands
```
python manage.py makemigrations
python manage.py migrate
```
Note: Above commands should be executed if there is any db level changes

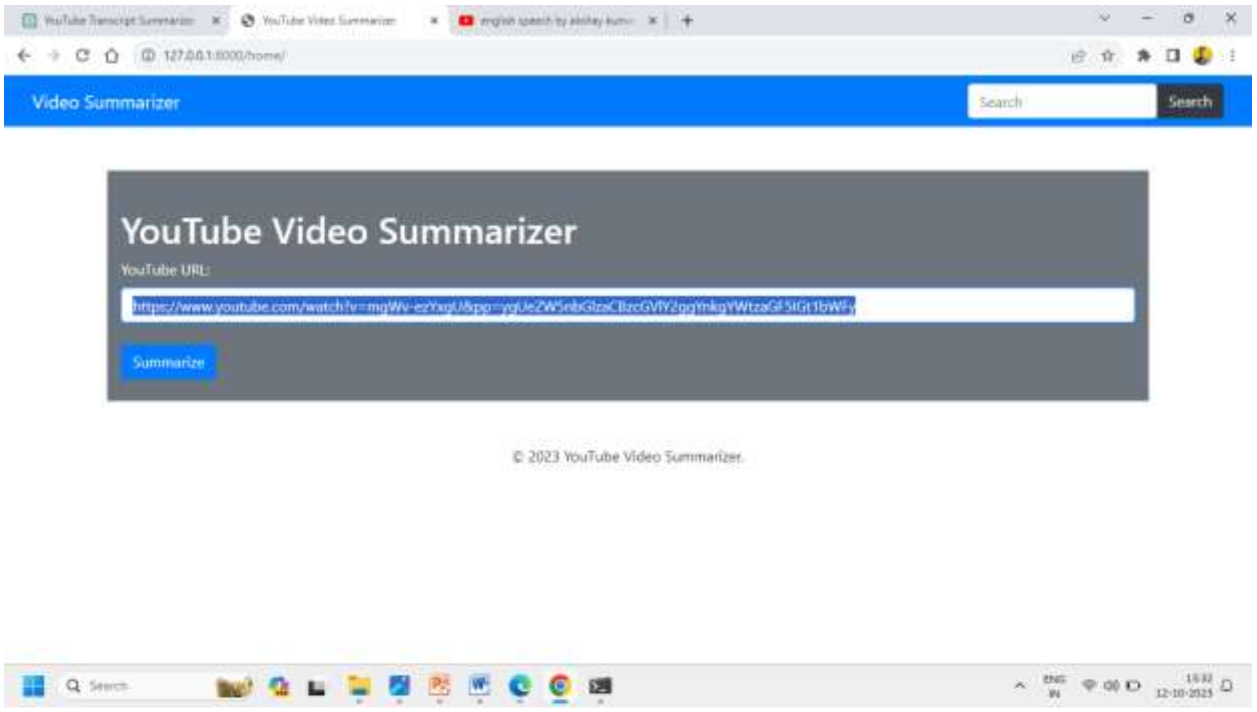
6)- Create superuser for admin access and follow instruction, if not created one
```
python manage.py createsuperuser
```
## Running the server
```
python manage.py runserver
```
~~~~

4.3 Snapshots:-



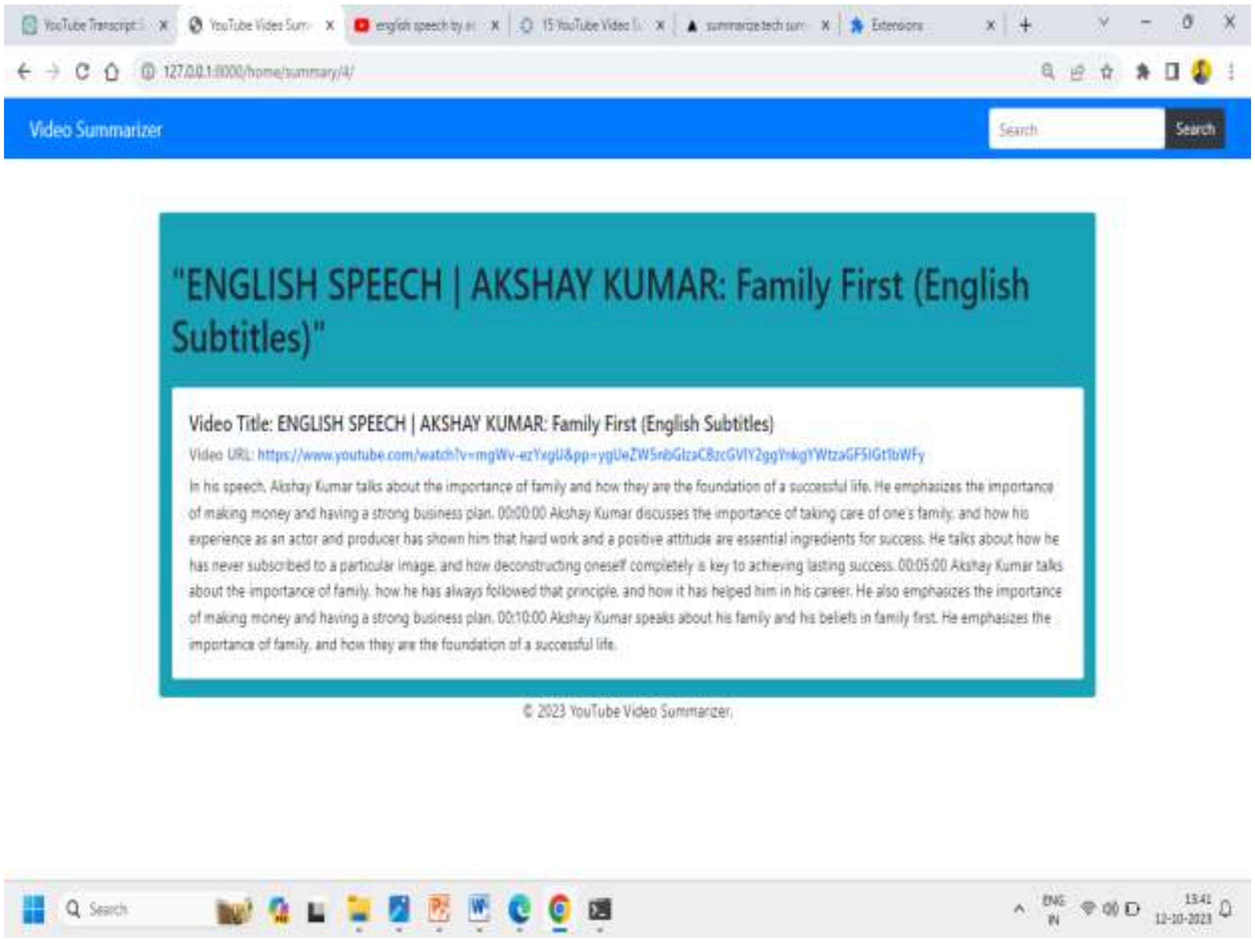
Video Summarizer
Search
YouTube Video Summarizer
YouTube URL:
https://www.youtube.com/watch?v=mgWv-ezYxgU&pp=ygUeZW5nbGlzaCBzcGVlY2ggYnkgYWtzaGF5IGt1bWFy
Summarize
© 2023 YouTube Video Summarizer.

Screenshot 1.(Home screen)



Video Summarizer
Search
YouTube Video Summarizer
YouTube URL:
https://www.youtube.com/watch?v=mgWv-ezYxgU&pp=ygUeZW5nbGlzaCBzcGVlY2ggYnkgYWtzaGF5IGt1bWFy
Summarize
© 2023 YouTube Video Summarizer.

Screenshot 2.(link pasted)



"ENGLISH SPEECH | AKSHAY KUMAR: Family First (English Subtitles)"

Video Title: ENGLISH SPEECH | AKSHAY KUMAR: Family First (English Subtitles)

Video URL: https://www.youtube.com/watch?v=mgWv-ezYxgU&pp=ygUeZW5nbGlzaCBzcGVlY2ggYnkgYWtzaGF5IGt1bWFy

In his speech, Akshay Kumar talks about the importance of family and how they are the foundation of a successful life. He emphasizes the importance of making money and having a strong business plan. 00:00:00 Akshay Kumar discusses the importance of taking care of one's family, and how his experience as an actor and producer has shown him that hard work and a positive attitude are essential ingredients for success. He talks about how he has never subscribed to a particular image, and how deconstructing oneself completely is key to achieving lasting success. 00:05:00 Akshay Kumar talks about the importance of family, how he has always followed that principle, and how it has helped him in his career. He also emphasizes the importance of making money and having a strong business plan. 00:10:00 Akshay Kumar speaks about his family and his beliefs in family first. He emphasizes the importance of family, and how they are the foundation of a successful life.

© 2023 YouTube Video Summarizer.

Screenshot 3.(Summary)

Chapter 5- Conclusions and future scope

5.1 Conclusion:-

In conclusion, the "YouTube Video Transcript Summarizer" project represents a valuable tool for users seeking to extract key insights and information from YouTube videos efficiently. With its ability to process video transcripts and generate concise summaries, the project addresses a significant need in content consumption and knowledge acquisition.

Ultimately, the "YouTube Video Transcript Summarizer" project is an innovative solution that simplifies the process of extracting valuable information from YouTube videos, making it a valuable tool for a wide range of users. With a commitment to ongoing development and improvement, we look forward to further refining this project to meet the evolving needs of our users and providing an even more robust summarization experience.

5.2 Future Scope

Multilingual Support: Enhance the project to support multiple languages, allowing users to summarize videos in various languages.

Audio and Visual Elements: Currently this summarize only the video that have transcript(summarizer) in Future we Explore ways to incorporate audio and visual elements, such as speech recognition for audio content and image recognition for visual content, into the summarization process.

Overall, there are many opportunities for improving and expanding the capabilities of YouTube transcript summarizers, and the future scope of this technology is likely to be broad and varied.