A PROJECT REPORT

on

"YouTube Summarizer"

Submitted by

Mr. Suraj Maruti Tavare

Seat No:

in partial fulfillment for the award of the degree

of

BACHELOR OF SCIENCE

In

COMPUTER SCIENCE

under the guidance of

Prof. Gyaneshwari Pawar

Department of Computer Science

VIDYAVARDHINI'S

A. V. COLLEGE OF ARTS, K. M. COLLEGE OF COMMERCE,

E. S. A. COLLEGE OF SCIENCE,
VASAI(WEST), PALGHAR-401208,
MAHARASHTRA

(SEM-V)

(2024-2025)

DECLARATION

I , **Mr. Suraj Maruti Tavare**, hereby declare that the project entitled "**YouTube Summarizer** " submitted in the partial fulfillment for the award of **Bachelor of Science** in Computer Science during the academic year **2024 – 2025** is my original work and the project has not formed the basis for the award of any degree, associate ship, fellowship or any other similar titles.

Signature of the Student:

Place:

Date:

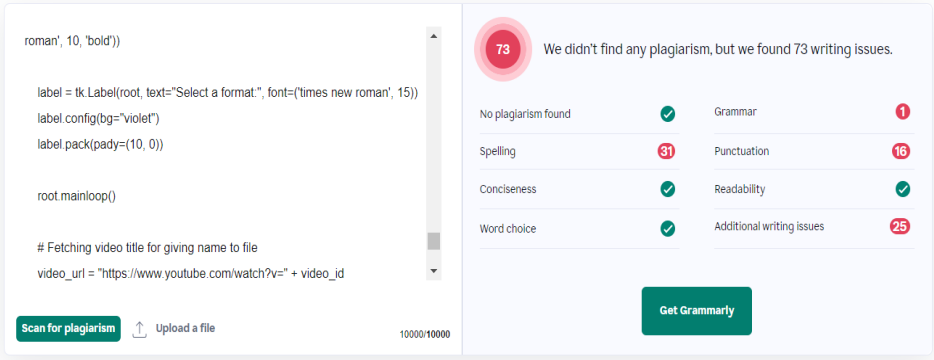
ACKNOWLEDGEMENT

I would like to acknowledge my sincere thanks towards our project guide Head of Computer Science Department **Mrs. Srimathi Narayanan** for their valuable guidance and suggestions and providing me an opportunity to do the project work in the college lab and which made me complete the project successfully.

I am very thankful to **Prof. Gyaneshwari pawar** for providing such nice guidance in form of comments and corrections. I am thankful to and fortunate enough to get constant encouragement, support and guidance from all teaching staff of Computer Science which helped us in Successfully completing our project work. Also, I would like to extend our sincere esteems to all staffs in laboratory for their timely support.

- **Mr. Suraj Maruti Tavare**
T.Y.BSC (Computer Science)

PLAGIARISM REPORT



GANTT CHART

INDEX PAGE

Sr. No	Contents	Page No.	SIGN
1.	**Introduction and Objective**	7-8	
2.	**Scope of Project and Technology stack**	9-10	
3.	**System Features and modules**	11	
4.	**Use case and Advantages**	12-13	
5.	**Requirement Specification** (Software and Hardware Requirement)	14-15	
6.	**System Design Details** A. Event Table B. Entity Relationship Diagram C. Class Diagram D. Activity Diagram E. Use Case Diagram F. Sequence Diagram G. Component Diagram H. Deployment Diagram	 16 17 18 19 20 21 22 23	
7.	**System Implementation (CODE)**	24-43	
8.	**User Interface(Screenshot)**	44-49	
9.	**Reference**	50	

INTRODUCTION

TITLE OF THE PROJECT : YouTube Summarizer

Introduction

1. The rise of YouTube as a dominant force in the digital media landscape has reshaped the way we consume and share information. With over a billion hours of video content watched on the platform every day, YouTube has become a primary source of entertainment, education, and inspiration for users around the globe. However, the format of video content poses significant challenges for certain segments of the population.
2. For individuals with hearing impairments, relying solely on audio content can be impractical or impossible. While YouTube does offer automatic captioning for some videos, the accuracy of these captions can vary widely, and they often fail to capture the nuances of spoken language accurately. Moreover, manually transcribing and summarizing videos is a time-consuming and labor-intensive process, making it impractical to generate detailed textual notes for every video.
3. Given these challenges, there is a clear need for an automated system that can accurately transcribe YouTube videos and generate detailed textual notes in a timely and efficient manner. Such a system would not only improve accessibility for individuals with hearing impairments but also enhance the overall usability and utility of YouTube content for all users.

Objectives

The main objectives of the YouTube Summarizer are to**:**

- Enhance Learning Efficiency:
    - Provide concise summaries to help users quickly grasp essential points from videos, facilitating faster learning and retention.

- Increase Accessibility:
    - Offer translation options for summaries, making content accessible to non-native speakers and a wider audience.
- Support Multi-Modal Learning:
    - Enable users to listen to summaries, catering to different learning preferences and enhancing engagement.
- Facilitate Easy Content Management:
    - Allow users to save summaries in various formats (TXT, DOC, PDF) for easy reference and organization.
- Enable Offline Access:
    - Provide the ability to download videos, allowing users to view content offline at their convenience.
- Encourage Deeper Exploration:
    - Offer a feature to search for more information on video topics, promoting further research and understanding.
- Improve Note-Taking:
    - Introduce a notes feature to help users document personal insights or highlights, aiding in review and study.
- Boost User Engagement:
    - Create an interactive experience that encourages users to engage more deeply with video content and related topics.
- Streamline Content Consumption:
    - Help users save time by summarizing lengthy videos, enabling them to make informed decisions about which videos to watch fully.
- Enhance User Experience:
    - Provide a user-friendly interface that simplifies access to all features, ensuring a seamless experience.

SCOPE OF PROJECT AND TECHNOLOGY STACK

Scope

1. **Core Functionality:**
   - **Transcript Summarization:** Automatically generate concise summaries from YouTube video transcripts.
   - **Translation:** Provide translation options for summaries into multiple languages.
   - **Audio Playback:** Allow users to listen to the summarized content.

2. **Content Management:**
   - **Save Options:** Enable users to save summaries in various formats, including TXT, DOC, and PDF.
   - **Download Video:** Implement functionality for users to download the YouTube video for offline viewing.

3. **Research and Exploration:**
   - **Related Search:** Include a feature that allows users to search for additional information related to the video topic.
   - **Notes Feature:** Provide a button to create and manage personal notes based on the summary.

4. **User Interface:**
   - **Intuitive Design:** Develop a user-friendly interface that allows easy navigation and access to features.
   - **Customization Options:** Allow users to customize settings (e.g., preferred language for translation).

5. **Performance and Compatibility:**
   - **Chrome Compatibility:** Ensure the extension functions smoothly across different versions of Google Chrome.

Technology Stack:

The app is built using the following technologies:

1. Frontend Development:

- **HTML**: For structuring the user interface of the extension.
- **CSS**: For styling and layout, ensuring a visually appealing design.
- **JavaScript**: For interactivity and handling user events.

2. Backend Development:

- **Flask (Python):** A lightweight web framework for building the server-side logic. This will handle requests for summarization, translation, and other backend services.
- **Flask-RESTful**: For creating RESTful APIs to facilitate communication between the frontend and backend.

3. APIs and Services:

- **YouTube API:** For retrieving video details, including transcripts, descriptions, and metadata.
- **Translation API**: Google Cloud Translation API for translating summaries into different languages.
- **Generative AI for Notes:** GenAI model to generate notes based on user input or summaries.
- **Google Custom Search API:** For searching more information related to the video topic using a Custom Search Engine ID.

4. Text-to-Speech Functionality:

- **SpeechSynthesis API:** Utilizing SpeechSynthesisUtterance for converting text summaries into speech, allowing users to listen to the content.

5. File Handling:

- File System API: For downloading summaries in formats like TXT, DOC, and PDF.
- PyPDF2 or ReportLab (Python): For generating PDF files from the server.

SYSTEM FEATURES AND MODULES

System Features

1. Transcript Summarization
- Automatically generate concise summaries from YouTube video transcripts.

2. Text-to-Speech Functionality
- Allow users to listen to the summarized content using the SpeechSynthesisUtterance.

3. Translation Options
- Translate summaries into multiple languages using Google Cloud Translation API.

4. Download Options
- Save summaries in various formats, including TXT, DOC, and PDF.
- Option to download the YouTube video for offline viewing.

5. Video Details Retrieval
- Fetch and display metadata such as title, description, and thumbnails using the YouTube API.

6. Related Search Feature
- Use Google Custom Search API to search for additional information related to the video topic.

7. Generative AI Notes
- Generate AI-assisted notes based on the video summary using a Generative AI model

8. User Interface
- Intuitive and user-friendly design for easy navigation and access to features.

Modules

1. **Download Module**:
   - Enables users to download summaries in various formats (TXT, DOC, PDF) and YouTube videos.
2. **Listen Module**:
   - Allows users to listen to summaries using the SpeechSynthesis API with playback controls.
3. **Translate Module**:
   - Translates summaries into multiple languages using the Google Cloud Translation API.
4. **Home Module**:
   - Serves as the main dashboard for entering video URLs and displaying generated summaries.
5. **Notes Module**:
   - Facilitates creating, editing, and saving personal notes, with AI assistance for note generation.
6. **Save Module**:
   - Manages saving user preferences and data locally or server-side for easy access.

USECASE AND ADVANTAGE

Use Case for You Tube Video Summarizer

1. **Summarizing Video Content:**
   - **Scenario**: A user wants to quickly understand the key points of a long YouTube tutorial.
   - **Action**: The user , retrieves the transcript, and receives a concise summary for efficient learning.
2. **Listening to Summaries:**
   - **Scenario**: A user prefers auditory learning and wants to listen to summaries while multitasking.
   - **Action**: After generating a summary, the user clicks the "Listen" button to hear the text read aloud using the text-to-speech feature.
3. **Translating Summaries:**
   - **Scenario**: A user is a non-native English speaker who wants to understand video content in their language.
   - **Action**: The user selects their preferred language and uses the translate feature to receive a translated summary.
4. **Taking Notes:**
   - **Scenario**: A student is watching a lecture video and wants to jot down important points.
   - **Action**: The user generates a summary, creates personal notes using the notes module, and saves them for future reference.
5. **Downloading Summaries:**
   - **Scenario**: A user wants to keep a record of video summaries for offline access.
   - **Action**: The user downloads the summary in their preferred format (TXT, DOC, PDF) for easy access later.
6. **Downloading YouTube Videos:**
   - **Scenario**: A user wants to save a YouTube video for offline viewing.
   - **Action**: The user clicks the "Download Video" button, and the extension retrieves the video for offline storage (if permissible by YouTube's terms).
7. **Searching for Related Information:**
   - **Scenario**: A user is interested in exploring more about the video's topic after watching it.
   - **Action**: The user uses the related search feature to find additional resources and articles on the subject.

Advantages of YouTube Summarizer

1. **Time Efficiency:**
   - Users can quickly grasp key points from long videos, saving time compared to watching the entire content.
2. **Enhanced Accessibility:**
   - Provides options for listening to summaries and translating them, making content accessible to a wider audience, including those with hearing impairments or non-native speakers.
3. **Convenient Note-taking:**
   - Users can easily create and save notes alongside summaries, facilitating better retention and study practices.
4. **Offline Access:**
   - The ability to download summaries and videos allows users to access content without needing an internet connection.
5. **Comprehensive Learning:**
   - Integrates various features (summarization, translation, note-taking) to support different learning styles and preferences.
6. **Informed Decision-Making:**
   - Users can explore related topics and resources, leading to deeper understanding and informed opinions.
7. **User-Centric Design:**
   - The extension's intuitive interface and multiple functionalities enhance user experience and satisfaction.
8. **Customization Options:**
   - Users can select different voices for text-to-speech and choose their preferred languages for translations.
9. **Streamlined Learning Process:**
   - Combines various educational tools into one platform, making it easier for users to manage their learning resources efficiently.

HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirements

1. **Device:**
   - A computer, laptop, or tablet with a modern web browser.
   - Minimum 2 GB RAM (4 GB recommended).
   - At least 100 MB of free disk space for storage of downloaded files and extension data.

2. **Internet Connection:**
   - Stable internet connection for accessing YouTube and API services.

Software Requirements

1. **Operating System:**
   - Windows, macOS, or Linux.
   -
2. **Web Browser:**
   - Google Chrome (latest version recommended) for running the extension.

3. **User Interface:**
   - No additional software is required; the extension operates entirely within the browser.

Developer Side Requirements :-

Hardware Requirements

1. **Development Machine:**
   - A computer or laptop with at least 8 GB RAM (16 GB recommended for smooth multitasking).
   - Sufficient disk space (minimum 500 MB) for development tools and libraries.

Software Requirements

1. **Operating System:**
   - Windows, macOS, or Linux.
2. **Web Browser:**
   - Google Chrome (latest version) for testing the extension.

3. **Development Environment:**
   - Python: Version 3.6 or higher for backend development.
   - Flask: Python web framework for server-side logic.
   - Text Editor/IDE: (e.g., Visual Studio Code, PyCharm, or Sublime Text) for coding.
4. **Frontend Technologies:**
   - HTML, CSS, JavaScript: For building the extension's user interface.

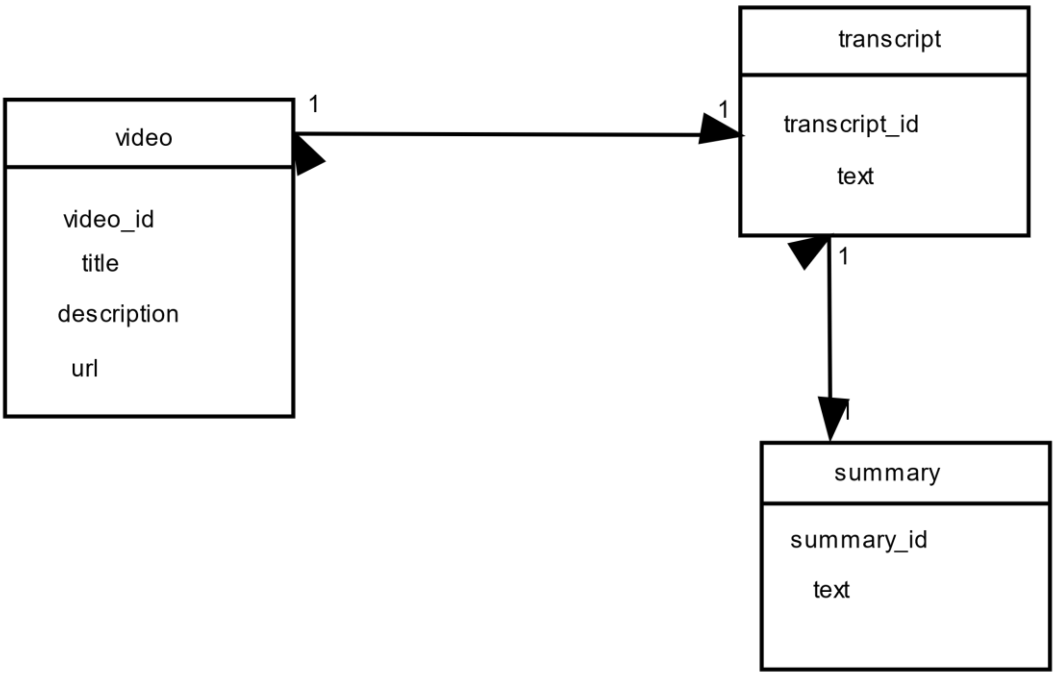
5. **APIs:**
   - **YouTube API**: For fetching video details and transcripts.
   - **Google Cloud Translation API**: For translating summaries.
   - **Generative AI API**: for generating notes based on summaries.
   - **Google Custom Search API**: For searching related topics.

SYSTEM DESIGN

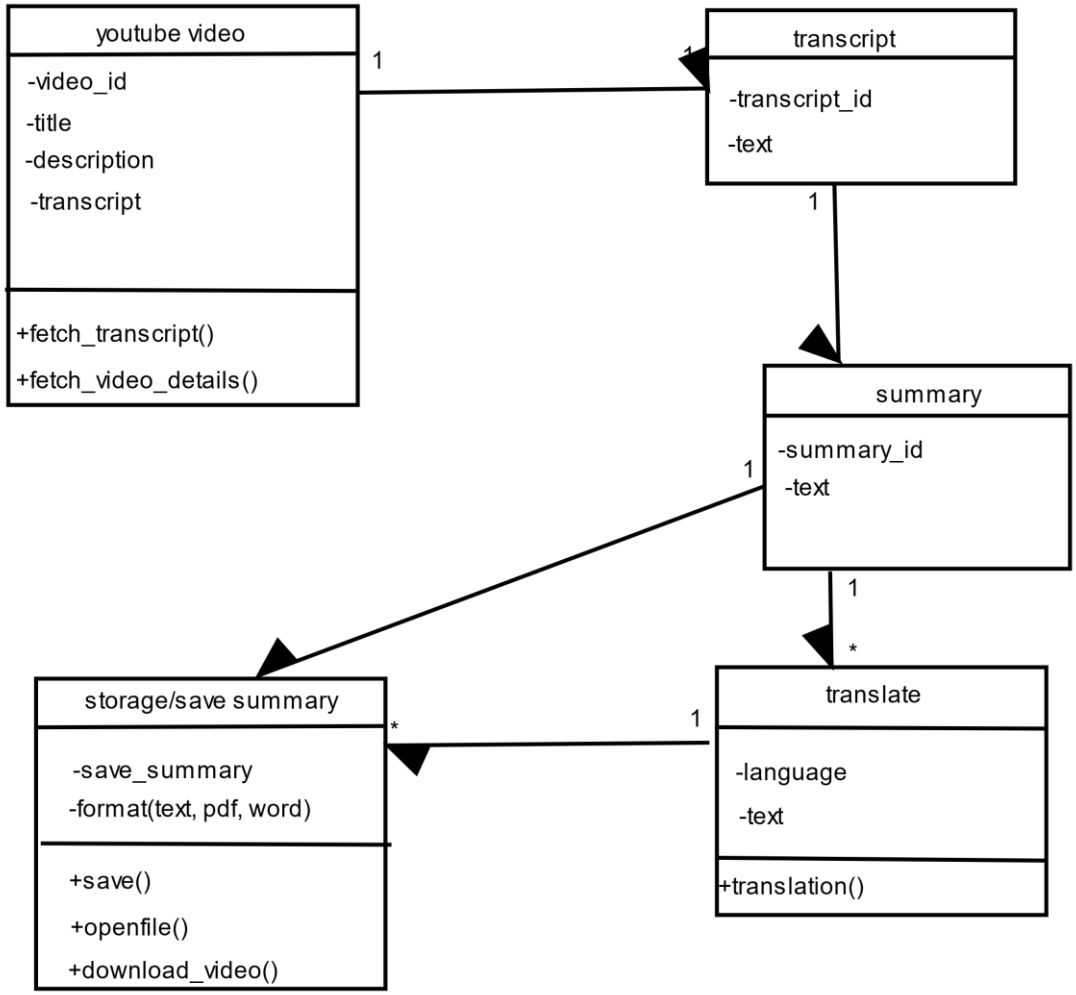
Event Table :

No.	Event	trigger	source	activity	response	destination
1	user request	user initiates a request for video summary	user	user interface display options	system prepares to fetch video details	You Tube video
2	fetch transcript	system receives request for video summary	system	system queries	system retrieves video prompt	transcript
3	summarize transcript	transcript fetched	system	applies algorithm	system generates summary	summary
4	save summary	summary is generated	user	store summary in file(text,doc,pdf format)	file saved successfully	User interface
5	display summary	summary is fetched generated	system	user interface ready to display	Summary displayed	user

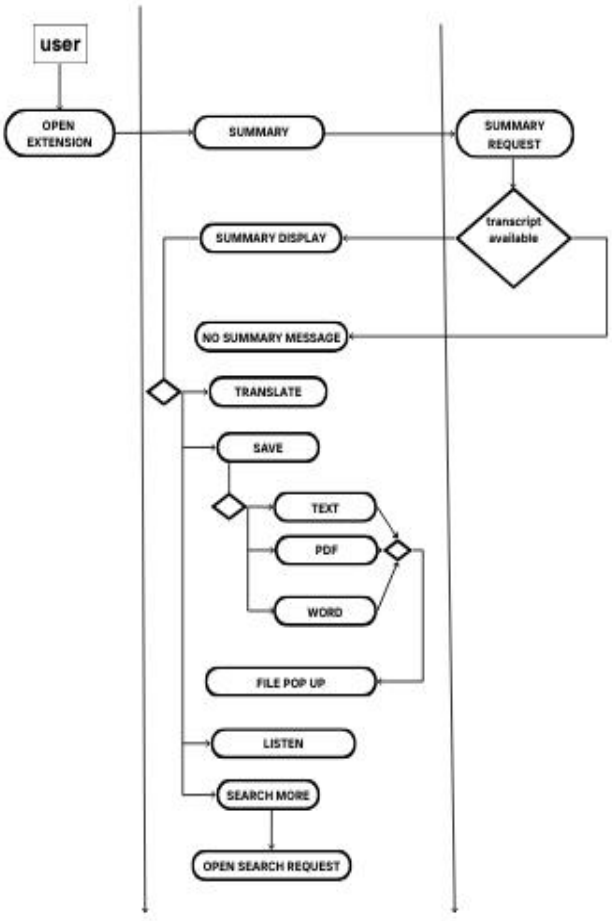
ER Diagram :


transcript
video
1
1
transcript_id
text
video_id
title
description
url
1
summary
summary_id
text

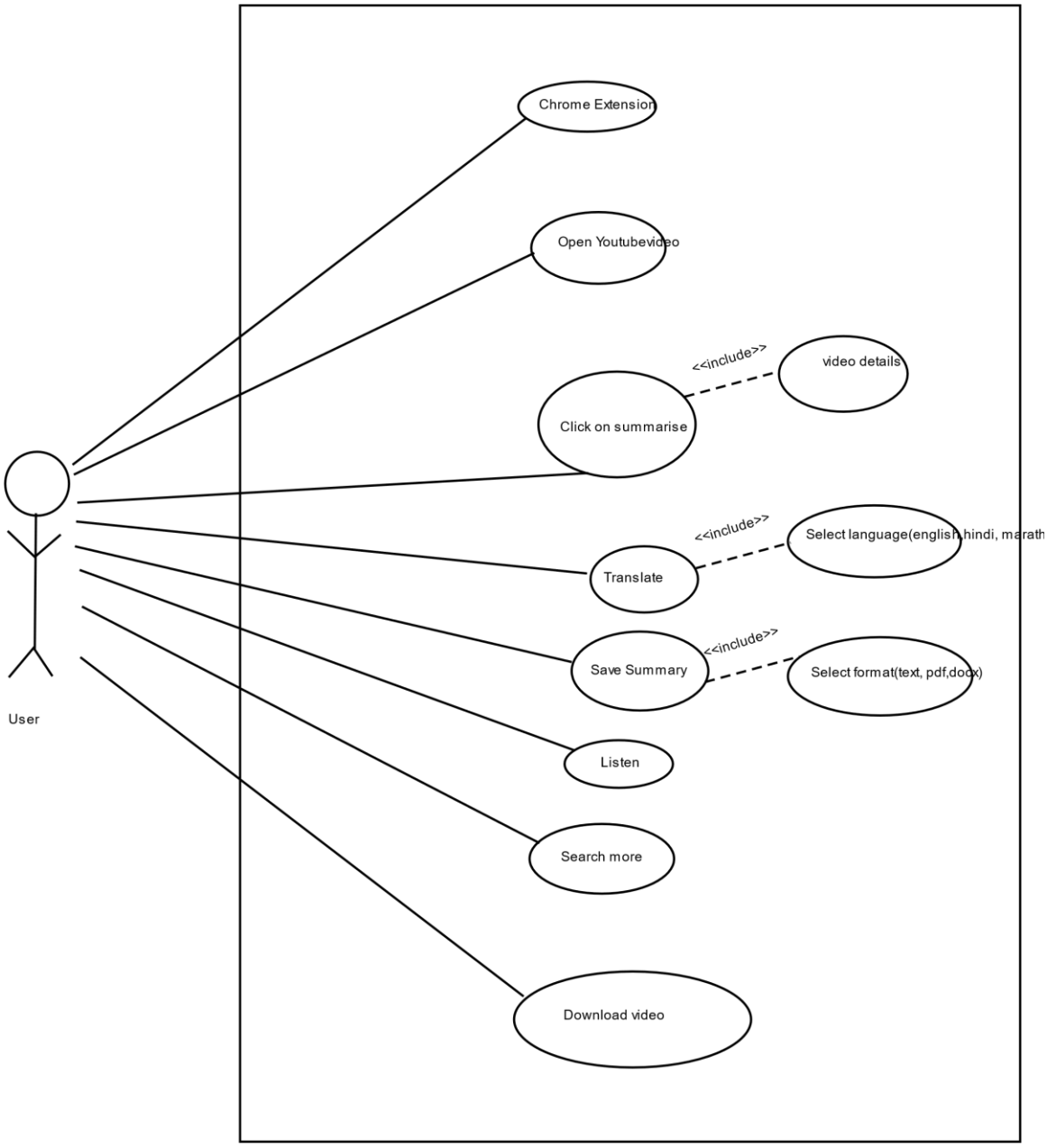
Class Diagram :



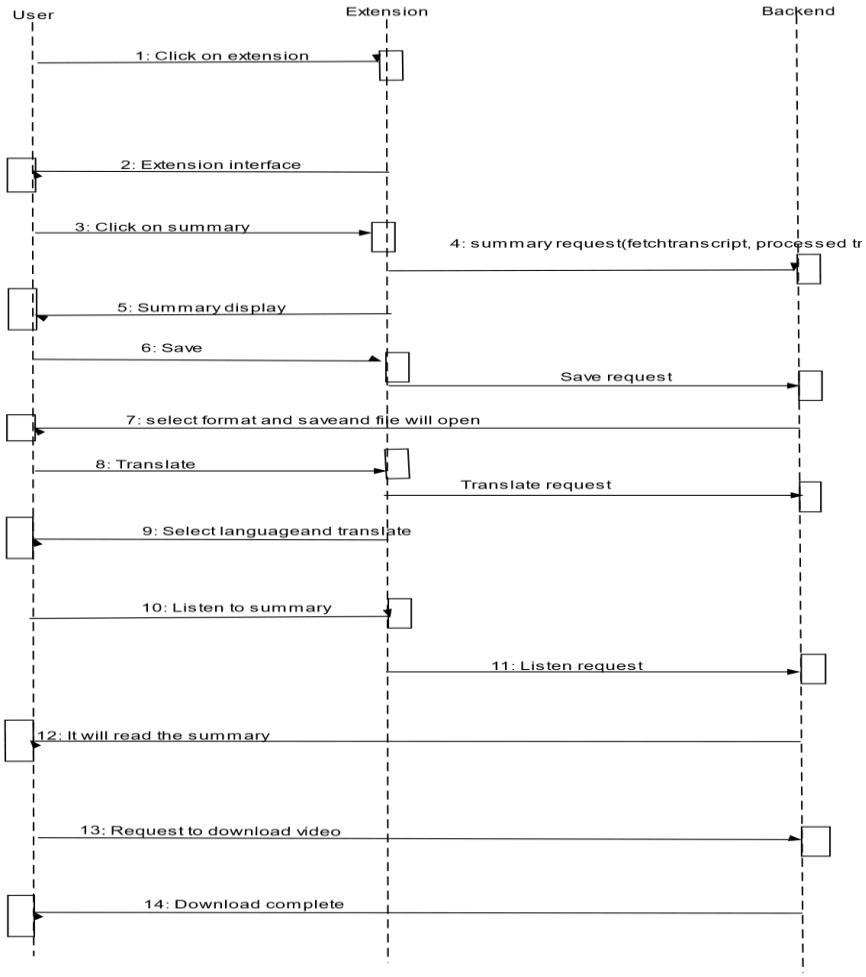
Activity Diagram :



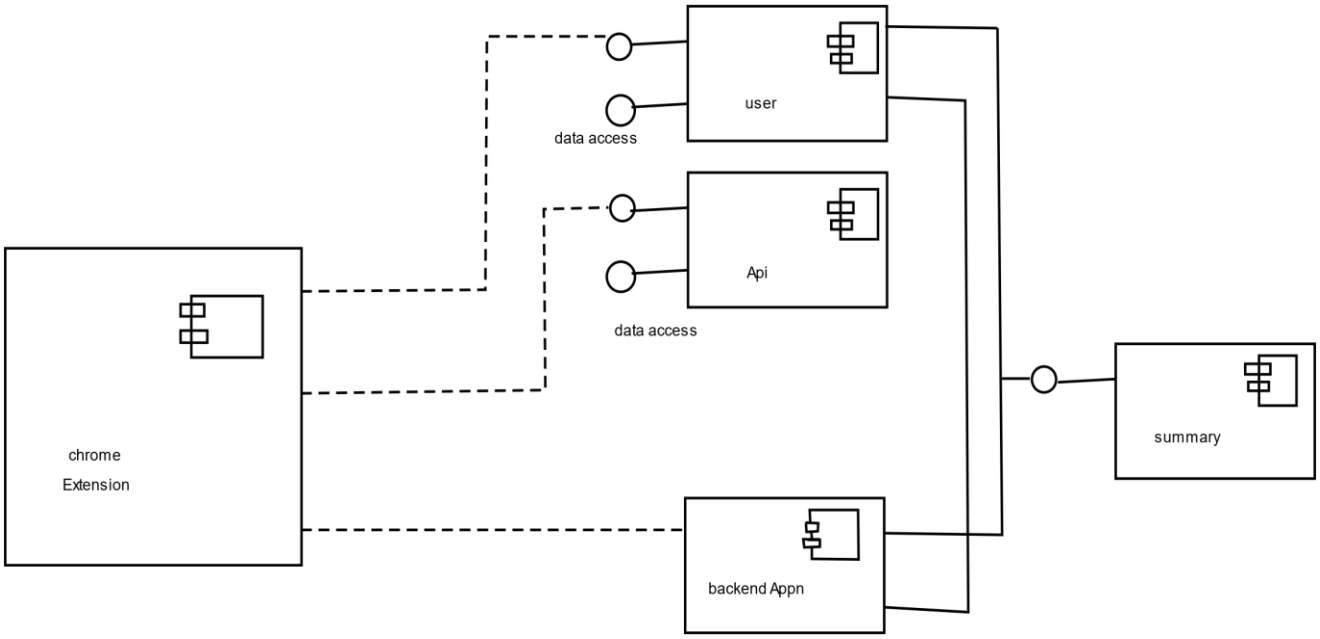
Usecase Diagram :



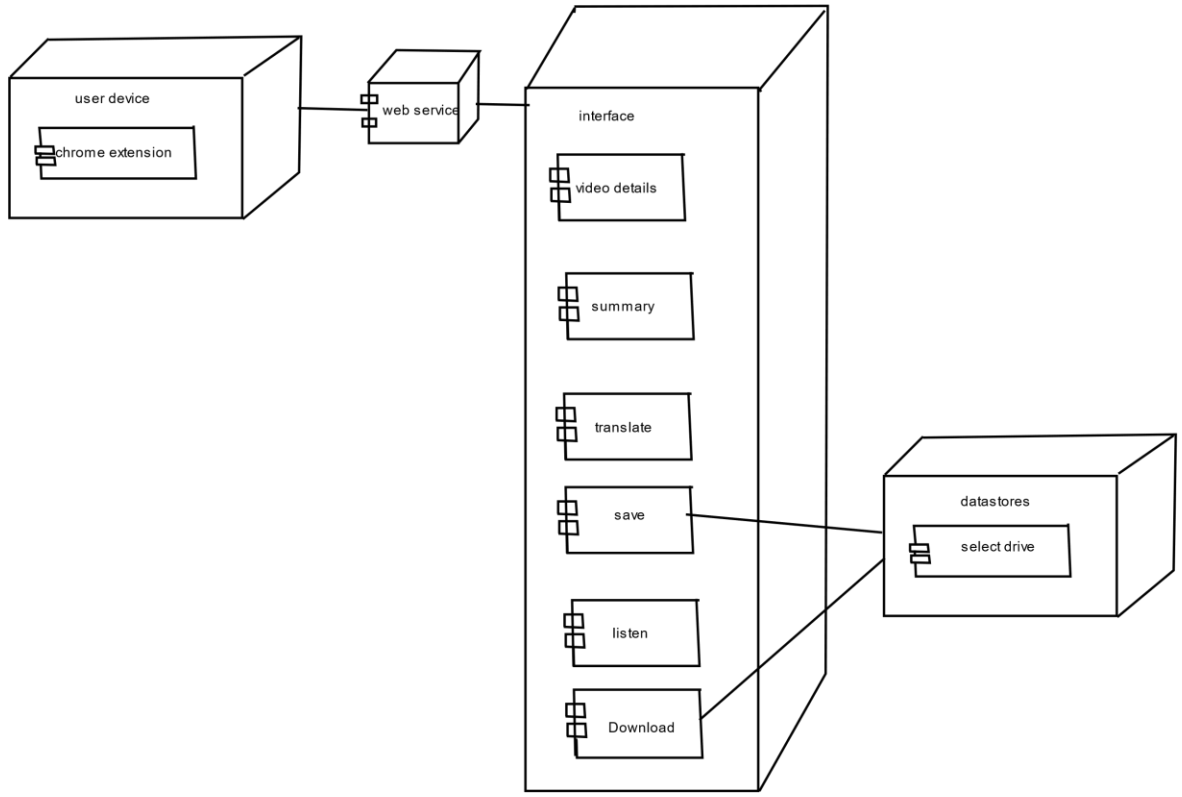
Sequence Diagram :



Component Diagram :



Deployment Diagram :



SYSTEM IMPLEMENTATION

app.py

```
from flask import Flask, request ,send_file ,jsonify
from youtube_transcript_api import YouTubeTranscriptApi, CouldNotRetrieveTranscript,
TranscriptsDisabled, VideoUnavailable
from transformers import pipeline

# **********----------TRANSLATION import Modules----------
**************************************************
# import statement for root window
import tkinter as tk
from tkinter import ttk
# import statement for translation
from googletrans import Translator, LANGUAGES

# **********----------DOWNLOAD import Modules----------
**************************************************
import yt_dlp
from threading import Lock
import os

# **********----------FILE SAVE import Modules----------
**************************************************
from pytube import YouTube
from reportlab.pdfgen import canvas
from reportlab.pdfbase.ttfonts import TTFont
from reportlab.pdfbase import pdfmetrics
from reportlab.lib import colors
from reportlab.lib.pagesizes import letter
from langdetect import detect
from docx import Document

#**********----------NOTES import----------
**************************************************
import google.generativeai as genai
import re
app = Flask(__name__)
progress = 0
progress_lock = Lock()
```

```
translated_text = None
notes= None
@app.get('/summary')
def summary_api():
    global text
    global video_id
    try:
        url = request.args.get('url', '')
        if not url:
            return jsonify({"error": "URL is required"}), 400
        print(type(url))
        print("url===", url)
        video_id = url.split('=')[1]
        try:
            transcript = get_transcript(video_id)
        except (CouldNotRetrieveTranscript, TranscriptsDisabled, VideoUnavailable) as e:
            return "Could not retrieve transcript: could not retrive summary for this video", 400

        summary = get_summary(transcript)
        text = summary
        return  summary, 200

    except Exception as e:
        return  "Could not retrieve transcript: could not retrive summary for this video ", 500


def get_transcript(video_id):
    transcript_list = YouTubeTranscriptApi.get_transcript(video_id)
    transcript = ' '.join([d['text'] for d in transcript_list])
    return transcript

def get_summary(transcript):
    summariser = pipeline('summarization')
    summary = ''
    for i in range(0, (len(transcript)//1000)+1):
        summary_text = summariser(transcript[i*1000:(i+1)*1000])[0]['summary_text']
        summary = summary + summary_text + ' '
    return summary

# ************************************TRANSLATION
FUNCTION*****************************************
@app.get('/translate')
def translate_text():
```

```
    return abc(text)

def abc(txt):
    # root window code
        root = tk.Tk()
        root.title("Language Translator")
        root.geometry('500x300')  # Set the window size
        root.configure(bg="black")

        style = ttk.Style()
        style.theme_use('clam')

        selected_language = tk.StringVar(root)
        selected_language.set("English")  # default value

        # Language list
        languages = ['Marathi', 'Hindi', 'Bengali', 'Telugu', 'Tamil', 'Gujarati' , 'Spanish', 'French', 'German',
'Chinese']

        language_dropdown = ttk.Combobox(root, textvariable=selected_language, values=languages,
state="readonly")
        language_dropdown.pack(pady=20, padx=10)

        label = tk.Label(root, text="Select a language to translate:", font=('times new roman', 15))
        label.config(bg="violet")
        label.pack(pady=(10, 0))

        translate_button = tk.Button(root, text="Translate", command=lambda: [ root.destroy() ])
        translate_button.pack(pady=10, padx=10)
        translate_button.config(bg='aquamarine', fg='black', font=('times new roman', 15, ))
        root.mainloop()

# ******************************translation logic*********************************8
        try:
        # Translation logic
            global translated_text
            translator = Translator()
            if len(txt)>4000:
                midpoint = len(txt) // 2
                split_point = txt.rfind(' ', 0, midpoint)

                if split_point == -1:  # No space found, just split in the middle
                    split_point = midpoint
```

```
    # ********----------Split the text into two parts-------**************************
            part1 = txt[:split_point].strip()
            part2 = txt[split_point:].strip()
            translation1 = translator.translate(part1, dest=selected_language.get())
            translation2 = translator.translate(part2, dest=selected_language.get())
            # global translated_text
            translated_text = "....................................................." + selected_language.get() +
"....................................................."  + "\n\n\n\n" + translation1.text + translation2.text
            return translated_text
        else:
            translation = translator.translate(txt, dest=selected_language.get())

            translated_text = "....................................................." + selected_language.get() +
"....................................................." + "\n\n\n\n\n" +  translation.text
            return translated_text

    except Exception as e:
        return f"Error during translation: {str(e)}", 500

# ********************************************DOWNLOAD
FUNCTION*****************************************************************
@app.get('/download')
def download():
    url = request.args.get('url', '')
    if not url:
        return "No URL provided", 400

    global progress
    progress = 0  # Reseting progress

    ydl_opts = {
        'format': 'bestvideo+bestaudio[ext=m4a]/best',
        'ffmpeg_location': "C:/Users/suraj/Downloads/ffmpeg-2024-08-26-git-98610fe95f-
essentials_build/ffmpeg-2024-08-26-git-98610fe95f-essentials_build/bin/ffmpeg.exe",
        'merge_output_format': 'mp4',
        'outtmpl': 'downloads/%(title)s.%(ext)s',
        'progress_hooks': [progress_hook]  # Add progress hook
    }

    with yt_dlp.YoutubeDL(ydl_opts) as ydl:
        info = ydl.extract_info(url, download=False)
        filename = ydl.prepare_filename(info)
```

```
    file_path=filename
    if os.path.exists(file_path):
        print("File already exists.")
        print("filename",filename)
        return jsonify({"filename":filename,
                        "text":"video already exists"}),800

    with yt_dlp.YoutubeDL(ydl_opts) as ydl:
        info = ydl.extract_info(url, download=True)
        filename = ydl.prepare_filename(info)
        print("filename",filename)
    return jsonify({"filename":filename})
# ************************************FOR PROFRESS
BAR**********************************************************************
@app.route('/progress')
def get_progress():
    global progress
    with progress_lock:
        return jsonify({'progress': progress})

def progress_hook(d):
    global progress
    if d['status'] == 'downloading':
        with progress_lock:
            progress = d.get('downloaded_bytes', 0) / d.get('total_bytes', 1) * 100

#*****************---------NOTES----------
************************************************

prompt="provide notes of given text "
@app.get("/notes")
def notes():
    return gen_notes(text,prompt)
def gen_notes(txt,prompt):
    global notes
    genai.configure(api_key="AIzaSyB3GLDQavn3ntqNyBsneyyb7N8n9r5Gk1k") #api gemini pro
(site=https://aistudio.google.com/app/apikey?_gl=1*7b62cr*_ga*MTc5MTAwMzgyLjE3MjYzODY4Mz
U.*_ga_P1DBVKWT6V*MTcyNjM4NjgzNS4xLjEuMTcyNjM4Njg3MS4yNC4wLjEwNTUzOTUzMz
M. )
    model=genai.GenerativeModel("gemini-pro")
    response=model.generate_content(prompt+txt)
    print(response.text)
    notes= "...................NOTES:-..........." + response.text + "..........."
```

```
    return response.text , 400


# **********************************FILE SAVE
FUNCTION*************************************************************

@app.get("/save")
def save():
    global translated_text
    global notes

    # Root window code
    root = tk.Tk()
    root.title("Format Selector")
    root.geometry('300x200')
    root.configure(bg="black")

    style = ttk.Style()
    style.theme_use('clam')

    selected_format = tk.StringVar(root)
    selected_format.set(".text")  # default value

    formats = ['.text', '.pdf', '.docx']  # Formats list

    language_dropdown = ttk.Combobox(root, textvariable=selected_format, values=formats,
state="readonly")
    language_dropdown.pack(pady=20, padx=10)

    translate_button = tk.Button(root, text="Submit", command=lambda: [root.destroy()])
    translate_button.pack(pady=10, padx=10)
    translate_button.config(bg='aquamarine', fg='black', font=('times new roman', 10, 'bold'))

    label = tk.Label(root, text="Select a format:", font=('times new roman', 15))
    label.config(bg="violet")
    label.pack(pady=(10, 0))

    root.mainloop()

    # Fetching video title for giving name to file
    video_url = "https://www.youtube.com/watch?v=" + video_id
    yt = YouTube(video_url)
    print(yt.title)
```

```
    video_title =  re.sub(r'\W+', '', yt.title)

    # Creating file name
    file_name = ''.join((video_title + selected_format.get()).split())

    # Format conditions
    if selected_format.get() == ".text":
        with open(file_name, "w", encoding="utf-8") as file:
            if text:
                if type(translated_text)!= str and type(notes) != str:
                    file.write(text)
                elif type(translated_text) == str and  type(notes) != str:
                    file.write(text + "\n" + translated_text)
                elif type(translated_text) != str and  type(notes) == str:
                    file.write(text + "\n" + notes)
                elif type(translated_text) == str and type(notes) == str:
                    file.write(text + "\n\n" +translated_text+"\n\n"+ notes)
            else:
                file.close()

    elif selected_format.get() == ".pdf":
        # Save as PDF
        def wrap_text(text, width, font_name, font_size):
            lines = []
            line = ''
            for word in text.split():
                if pdfmetrics.stringWidth(line + ' ' + word, font_name, font_size) <= width:
                    line += ' ' + word
                else:
                    lines.append(line.strip())
                    line = word
            lines.append(line.strip())
            return lines
        font_name="Helvetica"
        if translated_text:
            print("hrllo")
            def get_font_for_language(text):
                language_code = detect(text)
                # selecting language
                if language_code == 'ta':  # Tamil
                    return 'NotoSansTamil',
'C:/Users/suraj/OneDrive/Desktop/Noto_Sans_Tamil/NotoSansTamil-VariableFont_wdth,wght.ttf'
                elif language_code == 'gu': # gujrati
```

```
            return 'Noto Sans Gujarati
Thin','C:/Users/suraj/OneDrive/Desktop/Noto_Sans_Gujarati/NotoSansGujarati-
VariableFont_wdth,wght.ttf'
        elif language_code == "bn": # Bengali
            return 'Noto Sans Bengali
Thin','C:/Users/suraj/OneDrive/Desktop/Noto_Sans_Bengali/NotoSansBengali-
VariableFont_wdth,wght.ttf'
        elif language_code == 'hi':  # Hindi
            return 'Tiro Devanagari
Hindi',  'C:/Users/suraj/OneDrive/Desktop/Tiro_Devanagari_Hindi/TiroDevanagariHindi-Regular.ttf'
        elif language_code == "mr":# Marathi
            return 'Tiro Devanagri
Marathi','C:/Users/suraj/OneDrive/Desktop/Tiro_Devanagari_Marathi/TiroDevanagariMarathi-
Regular.ttf'
        elif language_code == 'te' : #Telugu
            return 'Noto Serif Telugu
Thin','C:/Users/suraj/OneDrive/Desktop/Noto_Serif_Telugu/NotoSerifTelugu-VariableFont_wght.ttf'
        elif language_code == 'zh-cn':  # Chinese
            return 'Noto Sans TC Thin',
"C:/Users/suraj/OneDrive/Desktop/Noto_Sans_SC,Noto_Sans_TC/Noto_Sans_TC/NotoSansTC-
VariableFont_wght.ttf"
        else:  # Default  for unsupported languages
            return 'Helvetica', None

    font_name , font_path =get_font_for_language(translated_text)
    pdfmetrics.registerFont(TTFont(font_name,font_path))

    documentTitle = 'YouTube Summerizer'
    title = 'YouTube Summerizer'
    subTitle = video_title
    if text:
        if type(translated_text)!= str and type(notes) != str:
            textLines = [text]
        elif type(translated_text) == str and  type(notes) != str:
            textLines = [text + "\n\n" + translated_text]

        elif type(translated_text) != str and  type(notes) == str:
            textLines = [text + "\n\n" + notes]
        elif type(translated_text) == str and type(notes) == str:
            textLines = [text + "\n\n" + translated_text + "\n\n" + notes]
    else:
        textLines="SORRY!!!! \n No Summary "
    width, height = letter
```

```
    pdf = canvas.Canvas(file_name, pagesize=letter)
    pdf.setTitle(documentTitle)

    pdf.setFont("Helvetica-Bold", 24)
    pdf.drawCentredString(width / 2, height - 100, title)

    pdf.setFont("Helvetica", 18)
    pdf.drawCentredString(width / 2, height - 150, subTitle)

    pdf.line(30, height - 160, width - 30, height - 160)
    text_y = height - 180
    text_width = width - 60
    font_size = 14
    pdf.setFont(font_name, font_size)

    pdf.setFillColor(colors.black)
    for line in textLines:
        wrapped_lines = wrap_text(line, text_width, font_name, font_size)
        for wrapped_line in wrapped_lines:
            pdf.drawString(40, text_y, wrapped_line)
            text_y -= 20  # Move to the next line
        text_y -= 10  # Add extra space between paragraphs
    pdf.save()

elif selected_format.get() == ".docx":
    # Save as DOCX
    doc = Document()
    if text:
        if type(translated_text)!= str and type(notes) != str:
            doc.add_paragraph(text)
        elif type(translated_text) == str and  type(notes) != str:
            doc.add_paragraph(text + "\n" + str(translated_text))
        elif type(translated_text) != str and  type(notes) == str:
            doc.add_paragraph(text + "\n" + notes)
        elif type(translated_text) == str and type(notes) == str:
            doc.add_paragraph(text + "\n\n" + str(translated_text) + "\n\n" + notes)
    else:
        doc.add_paragraph("SORRY!!!, No Summary")

    doc.save(file_name)

# Check if file exists and return appropriate response
file_path = os.path.abspath(file_name)
```

```
    if os.path.exists(file_path):
        os.startfile(file_path)
        return send_file(file_path, as_attachment=True), 200
    else:
        return jsonify({"error": "File not found"}), 400

if __name__ == '__main__':
    app.run()
```

popup.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Youtube Transcript Summariser</title>
    <style>
        body {
            width: max-content;
            max-width: 700px;
            background-color: black;
        }
        .main{
            display: flex;
            flex-direction: column;
            justify-content: center;
            align-items: center;
        }
        h1{
            margin-top: 50px;
            margin-left: 50px;
            margin-right: 40px;
            /* margin-top: 80px; */
            font-size: 5vh;
            text-align: center;
            color: violet;
        }
        #summarize{
            margin-top: 40px;
            background-color: aquamarine;
            height: 50px;
            width: 150px;
            border-radius: 5px;
            font-size: 5vh;
            font-weight: bold;
```

```
    outline: none;
    border: none;
    cursor: pointer;

}
#summarize:hover{
    transition: 3ms;
    box-shadow: rgb(248, 246, 246) 0px 3px 8px;
}

/* button {
    background-color: red;
    color: white;
    border-radius: 8px;
    width: max-content;
    height: max-content;
    padding: 10px;
    font-size: large;
    margin: auto;
    display: block;
    border-color: coral;
    margin-bottom: 10px;
}
button[disabled] {
    background-color: red;
    color: white;
    border-radius: 8px;
    width: max-content;
    height: max-content;
    padding: 10px;
    font-size: large;
    margin: auto;
    display: block;
    border-color: lightpink;
} */
p {
    font-size: 3vh;
    color: white;
    word-wrap: break-word;
    overflow-wrap: break-word;


}
#output3 {
    font-size: 4vh;
    color: rgb(5, 254, 254);
    word-wrap: break-word;
    overflow-wrap: break-word;
```

```
}

#output4 {
    font-size: 4vh;
    color: rgb(5, 254, 254);
    word-wrap: break-word;
    overflow-wrap: break-word;

}
#output5 {
    font-size: 3vh;
    color: rgb(0, 241, 249);
    word-wrap: break-word;
    overflow-wrap: break-word;

}

h3{
    /* display: none; */
    font-size: 25px;
    font-family: 'Times New Roman', Times, serif;
    left: top;
    top: left;
}
.text-to-speech {
    display: none;
}

.search-More-Btn {
    display: none;
}
.download-Btn {
    display: block;
}
.Save-Btn{
    display: none;
}
.Note-Btn{
    display: none;
}
.Translate-Btn{
    display: none;
}
.download-Btn{
    display: none;
}
.allbtns{
    margin-top: 40px;
    display: flex;
```

```
            flex-wrap: wrap;
            gap: 20px;
            align-items: center;
            justify-content: center;

        }
        .allbtns button{
            background-color: aquamarine;
            height: 80px;
            width: 220px;
            border-radius: 5px;
            font-size: 3vh;
            font-weight: bold;
            outline: none;
            border: none;
            cursor: pointer;

        }
        #downloadProgress{
            height: 30px;
            width: 350px;
        }
        span{
            color: blue;
            font-size: 5vh;
            font-weight: 900;
        }
        @media only screen and (max-width: 800px) {

            h1{
                color: red;
            }
            #summarize{
                width: 200px;
                background-color: plum;
            }
            .allbtns button{
                width: 200px;
                height: 50px;
            }
        }
    </style>
</head>
<body>
    <div class="main">
        <h1>Youtube Transcript summarizer</h1>
        <button id="summarize" type="button">Summarize</button>
        <br/>
        <h3 id ="words_count"> </h3>
```

```
        <p id="output"></p>
        <p id="output2"></p>
        <p id="output3"></p>
        <p id="output4"></p>
        <p id="output5"></p>

        <!-- Add a progress bar element in your HTML -->
        <progress id="downloadProgress" value="0" max="100"></progress>
        <span id="progressPercentage">0%</span>
        <div class="allbtns">
          <button id="textToSpeechBtn" class="text-to-speech" type="button">Listen</button>
          <button id="searchMoreBtn" class="search-More-Btn" type="button">Search More</button>
          <button id="downloadBtn" class="download-Btn" type="button">Download</button>
          <button id="Translate" class="Translate-Btn" type="button">Translate</button>
          <button id="Save"  class="Save-Btn" type="button">Save</button>
          <button id="Notes"  class="Note-Btn" type="button">Notes</button>
        </div>
    </div>

    <script src="popup.js"></script
</body>
</html>
```

Popup.js

```
const btn = document.getElementById("summarize");
const textToSpeechBtn = document.getElementById("textToSpeechBtn");
const searchMoreBtn = document.getElementById("searchMoreBtn");
const Translatebtn = document.getElementById("Translate");
const Savebtn = document.getElementById("Save");
const Notebtn = document.getElementById("Notes");
const downloadBtn = document.getElementById("downloadBtn");
const progressBar = document.getElementById("downloadProgress");
const progressPercentage = document.getElementById("progressPercentage");
const outputParagraph = document.getElementById("output");
const wordsCountHeading = document.getElementById("words_count");

var url;
var video_id;
var video_title;
progressBar.style.display = "none";
progressPercentage.style.display = "none";

btn.addEventListener("click", function() {
    btn.disabled = true;
```

```
    btn.innerHTML = "Summarizing...";
    chrome.tabs.query({currentWindow: true, active: true}, function(tabs){
        url = tabs[0].url;
        console.log("url="+url)


        var xhr = new XMLHttpRequest();
        xhr.open("GET", "http://127.0.0.1:5000/summary?url=" + url, true);
        xhr.onload = function() {
            var text = xhr.responseText;
            console.log(text);

            const p1 = document.getElementById("output");
            if(text=="Could not retrieve transcript: could not retrive summary for this video"){

                p1.innerHTML = text;
                // btn.disabled = false;
                btn.innerHTML = "Summarized";

                // textToSpeechBtn.style.display = "block"; // Show the text to speech button
                searchMoreBtn.style.display = "block"; // Show the search More Btn button
                downloadBtn.style.display = "block"; // Show the download Btn button
                // Savebtn.style.display = "block"; // Show the save Btn button
                // Translatebtn.style.display = "block"; // Show the translate Btn button
                // Notebtn.style.display = "block"; // Show the note Btn button
            }
            else{
                p1.innerHTML = text;
                // btn.disabled = false;
                btn.innerHTML = "Summarized";
                textToSpeechBtn.style.display = "block"; // Show the text to speech button
                searchMoreBtn.style.display = "block"; // Show the search More Btn button
                downloadBtn.style.display = "block"; // Show the download Btn button
                Savebtn.style.display = "block"; // Show the save Btn button
                Translatebtn.style.display = "block"; // Show the translate Btn button
                Notebtn.style.display = "block"; // Show the note Btn button

            }
            updateWordsCount();
        }
        xhr.send();

    });
});


//*********----------FUNCTION FOR LISTEN ----------********** */

textToSpeechBtn.addEventListener("click", function() {
```

```
    const textToSpeak = document.getElementById("output").textContent;
    const utterance = new SpeechSynthesisUtterance(textToSpeak);
    const defaultVoice = speechSynthesis.getVoices().find(voice => voice.default);
    if (defaultVoice) {
        utterance.voice = defaultVoice;
    }
    speechSynthesis.speak(utterance);
});

//*********----------FUNCTION FOR SEARCH MORE BUTTON----------********** */

// fetching video Id
searchMoreBtn.addEventListener("click",function () {
    console.log(url)
    var s = url.match (/[?&]v=([^&]+)/);
    if (s) {
         video_id=s[1];
        fetchVideoTitle(video_id);
    } else {
        console.error("Invalid YouTube URL:", url);
        return null;
    }
});


// FETCHING VIDEO TITLE
function fetchVideoTitle(video_id) {
    var API_KEY="AIzaSyCKLTZ0JiROCVda01VfF2SbuwPNWv-kLVw"; // for video  title fetch api
(https://console.cloud.google.com/apis/credentials?project=driven-strength-420608)

fetch(`https://www.googleapis.com/youtube/v3/videos?id=${video_id}&part=snippet&key=${API_KEY
}`)
    .then(response => response.json())
    .then(data => {
         video_title = data.items[0].snippet.title;
        console.log("Video Title:", video_title);
        searchOnGoogle(video_title);
        // return video_title

    })
    .catch(error => {
        console.error("Error fetching video title:", error);
    });
}

// Function to perform a Google search using the Google Custom Search API
function searchOnGoogle(query) {
    var searchQuery = encodeURIComponent(query);
    var apiKey = 'AIzaSyDA9a78xOJhuMgqGzp31qArMeSJsK8Nt_I'; // Google Custom Search API key
```

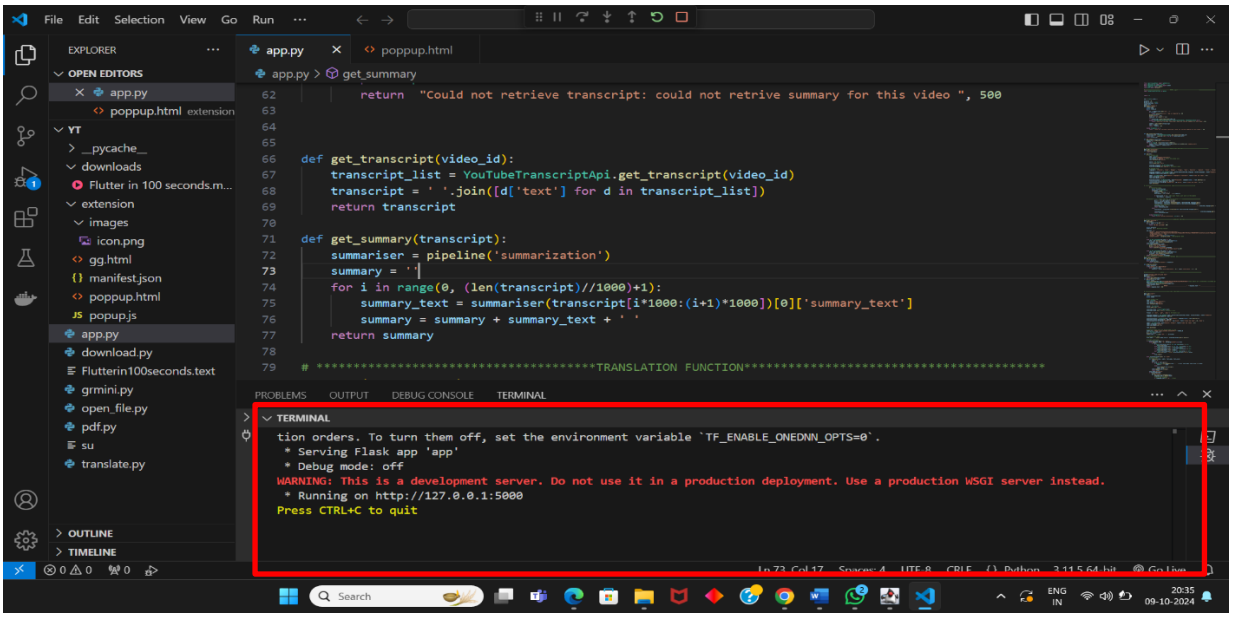
```
    var cx = '115f2634d35ab4406'; // Custom Search Engine ID
(https://programmablesearchengine.google.com/controlpanel/overview?cx=115f2634d35ab4406)
    var apiUrl =
`https://www.googleapis.com/customsearch/v1?q=${searchQuery}&key=${apiKey}&cx=${cx}`;
    console.log(apiUrl);

    fetch(apiUrl)
        .then(response => response.json())
        .then(data => {
            if (data.items && data.items.length > 0) {
                var firstResultUrl = data.items[1].link;
                console.log(firstResultUrl);
                window.open(firstResultUrl, '_blank');
            } else {
                console.error('No search results found.');
            }
        })
        .catch(error => {
            console.error('Error performing Google search:', error);
        });
}


// function fro wrod count
function countWords(text) {
    var words = text.split(/\s+/);
    return words.length;
}

// Function to update the words count heading
function updateWordsCount() {
    var textContent = outputParagraph.textContent || outputParagraph.innerText;
    var wordsCount = countWords(textContent);
    wordsCountHeading.textContent = "Summary in " + wordsCount + " words";
}

//**********----------TRANSLATION----------********** */

Translatebtn.addEventListener("click", function() {
    var xhr = new XMLHttpRequest();
    xhr.open("GET", "http://127.0.0.1:5000/translate", true);
    xhr.onload = function() {
        var text1=xhr.responseText;
        console.log(text1)
        const p2=document.getElementById("output2");
        p2.innerHTML = text1;

        textToSpeechBtn.style.display = "block"; // Show the text to speech button
        searchMoreBtn.style.display = "block"; // Show the search More Btn button
```

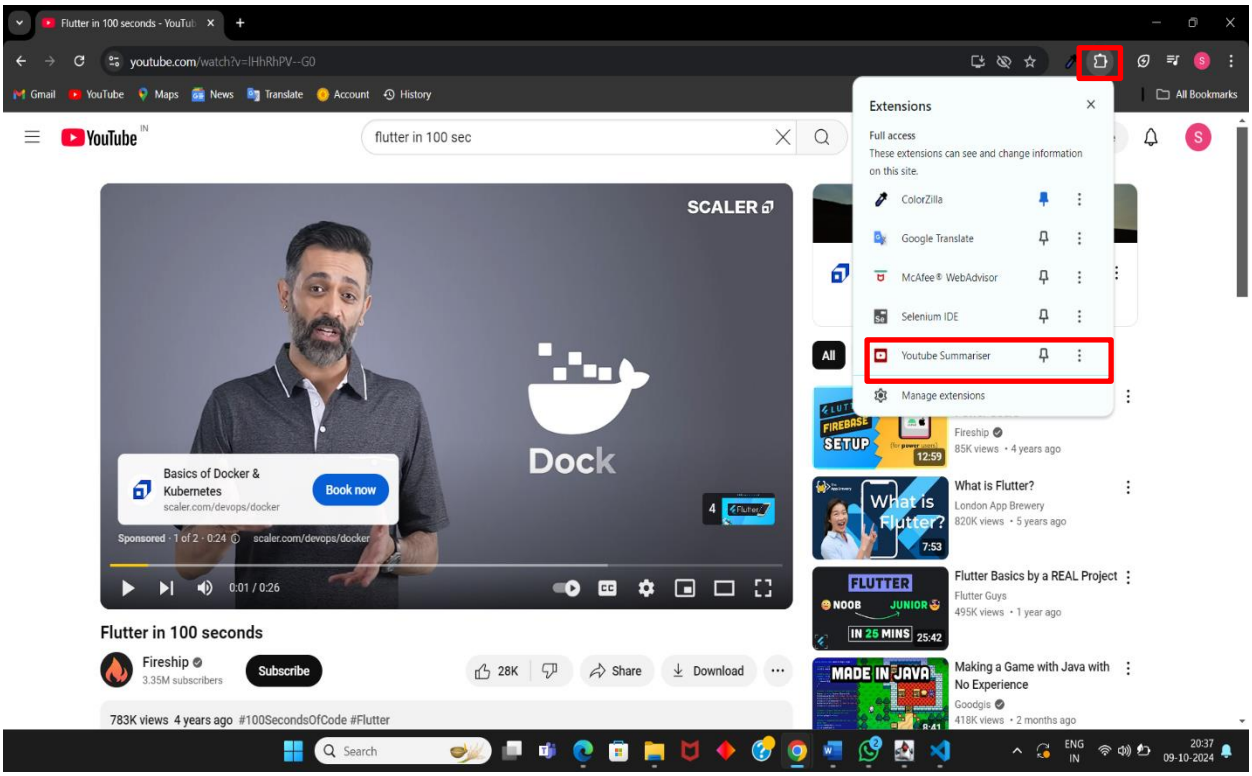
```
        downloadBtn.style.display = "block"; // Show the download Btn button
        Savebtn.style.display = "block"; // Show the save Btn button
        Translatebtn.style.display = "block"; // Show the translate button
        Notebtn.style.display = "block"; // Show the note button
    };
    xhr.send();
});

//**************------------DOWNLOAD----------**************** */

downloadBtn.addEventListener("click", function() {
    // Start the download
    downloadBtn. disabled= true;
    progressBar.style.display = "block";
    progressPercentage.style.display = "block";
    var xhr = new XMLHttpRequest();
    xhr.open("GET", "http://127.0.0.1:5000/download?url=" + encodeURIComponent(url), true);
    xhr.onload = function() {
        var res=JSON.parse(xhr.responseText).text
        if (res==="video already exists") {
            var response = JSON.parse(xhr.responseText);
            var filename = response.filename.toString();
            var p3 =document.getElementById("output3");
            p3.innerHTML= "Video Already Downloaded. "+"<br>Path:- "+ filename;
            progressBar.style.display = "none";
            progressPercentage.style.display = "none";


        }else {
            var response = JSON.parse(xhr.responseText);
            var filename = response.filename.toString();
            var p4 =document.getElementById("output4");
            p4.innerHTML= "Downloaded file: " + filename;
            console.error("Error during download:", res);
        }
    };
    xhr.send();


//**************------------POLL PROGRESS----------**************** */

    var progressInterval = setInterval(function() {
        var progressXhr = new XMLHttpRequest();
        progressXhr.open("GET", "http://127.0.0.1:5000/progress", true);
        progressXhr.onload = function() {
            if (progressXhr.status === 200) {
                var progress = JSON.parse(progressXhr.responseText).progress;
                progressBar.value = progress;
                progressPercentage.textContent = Math.round(progress) + "%"; // Update percentage text
```

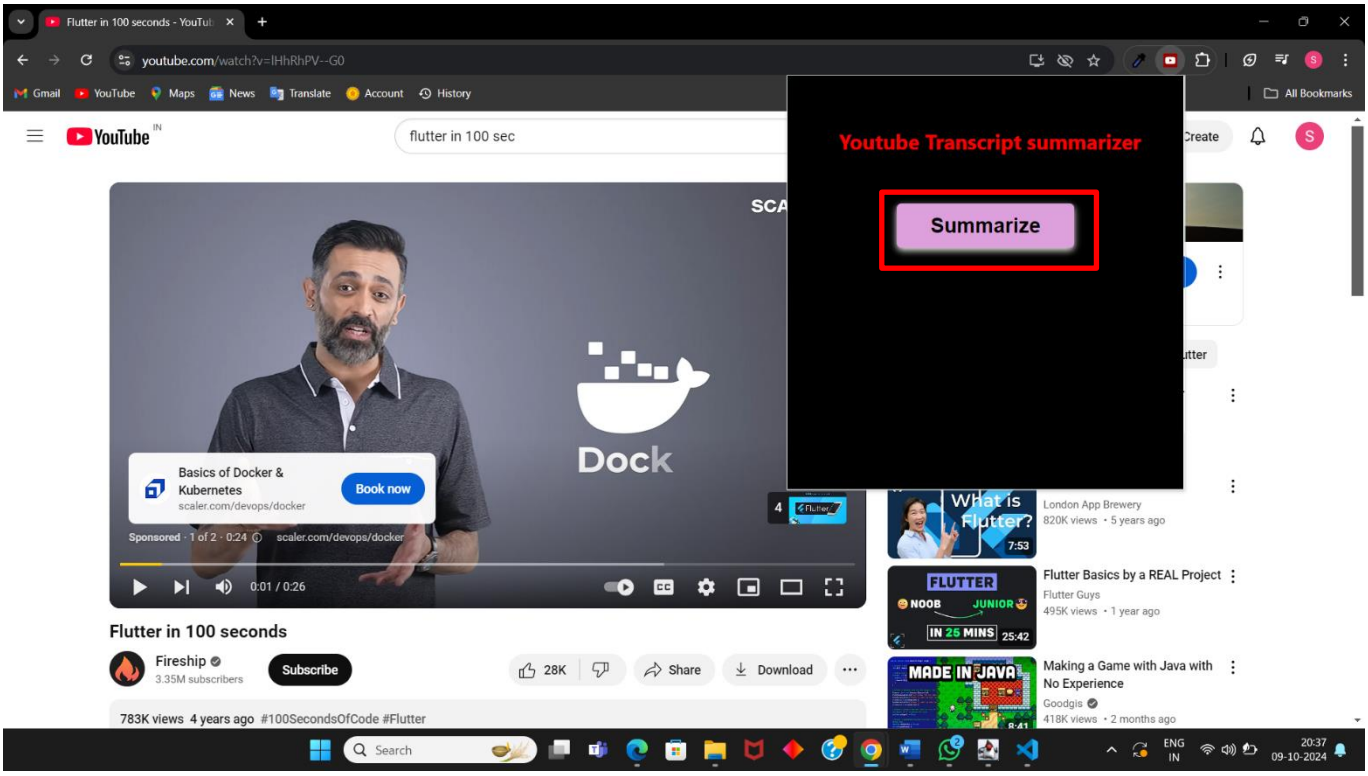
```
            if (progress >= 100) {
                clearInterval(progressInterval);
                // downloadBtn.status.disabled;
            }
        }
    };
    progressXhr.send();
}, 1000); // Poll every 1 second
});



// ***************---------FILE SAVE----------*******************
Savebtn.addEventListener("click",function(){
    var xhr = new XMLHttpRequest();
    xhr.open("GET","http://127.0.0.1:5000/save",true)
    xhr.onload=function(){
      console.log("heelo");

        if (xhr.status === 200) {
            console.log("File saved successfully.");
        } else {
            console.error("Error saving file:", xhr.statusText);
        }
    };
    xhr.send();
})


Notebtn.addEventListener("click",function(){
    var xhr = new XMLHttpRequest();
    xhr.open("GET","http://127.0.0.1:5000/notes",true)
    xhr.onload=function(){
        console.log("hii");
        console.log(xhr.response);

         if (xhr.response ) {
            function formatTextToHtml(text) {
                // Replace double asterisks with <strong> for bold text
                let formattedText = text.replace(/\*\*(.*?)\*\*/g, '<strong>$1</strong>');
                // Replace asterisks with <br> for new lines inside a paragraph
                formattedText = formattedText.replace(/\n?\* (.*?)\n/g, '<br> * $1<br>');

                // Wrap paragraphs with <p> tags
                formattedText = formattedText.split('\n\n').map(para => `<p>${para}</p>`).join('');

                return formattedText;
```

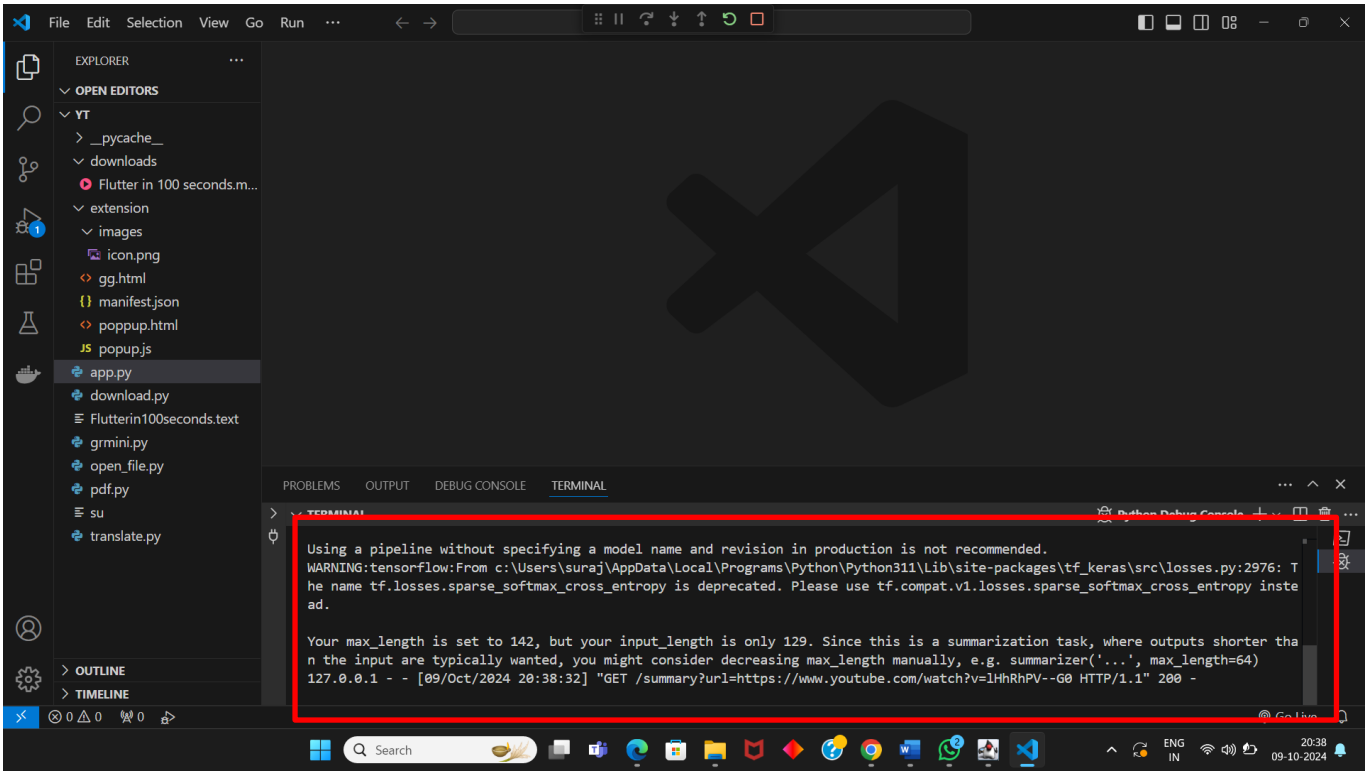
```
        }
        var p5=document.getElementById("output5");
        p5.innerHTML=formatTextToHtml(xhr.response);
        console.log("notes displays successfully.");
      } else {
        var p5=document.getElementById("output5");
        p5.innerHTML="SOORY!!, Unable to create Notes";
         console.error("Error displays notes:", xhr.statusText);
      }
  };
  xhr.send();
})
```

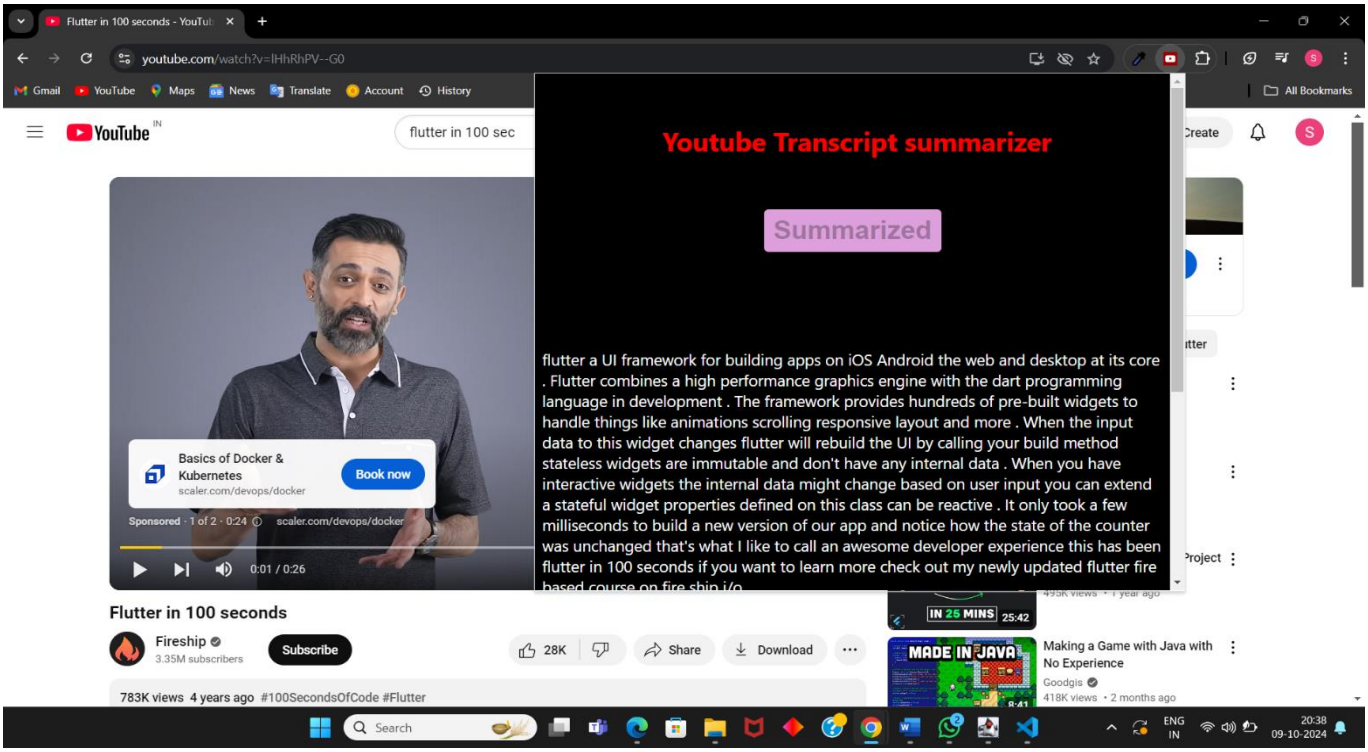
USER INTERFACE

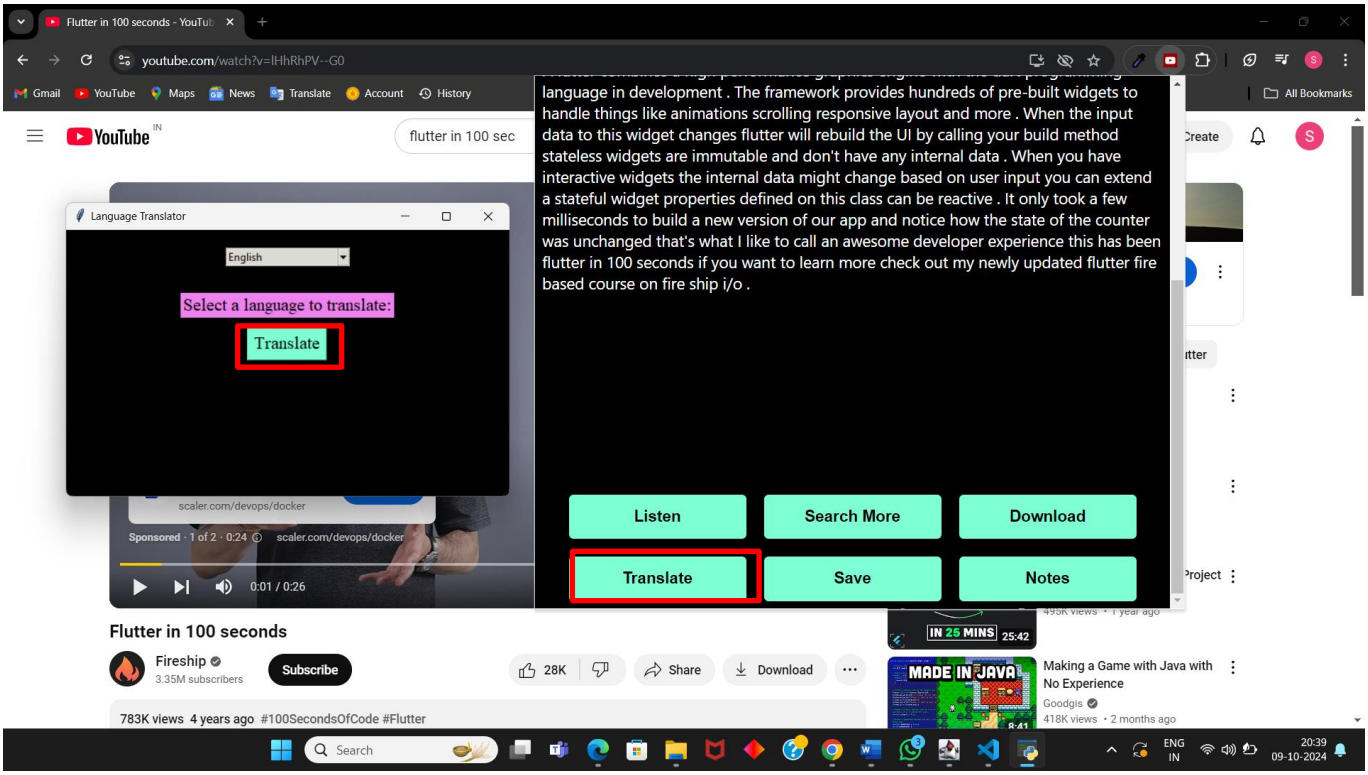


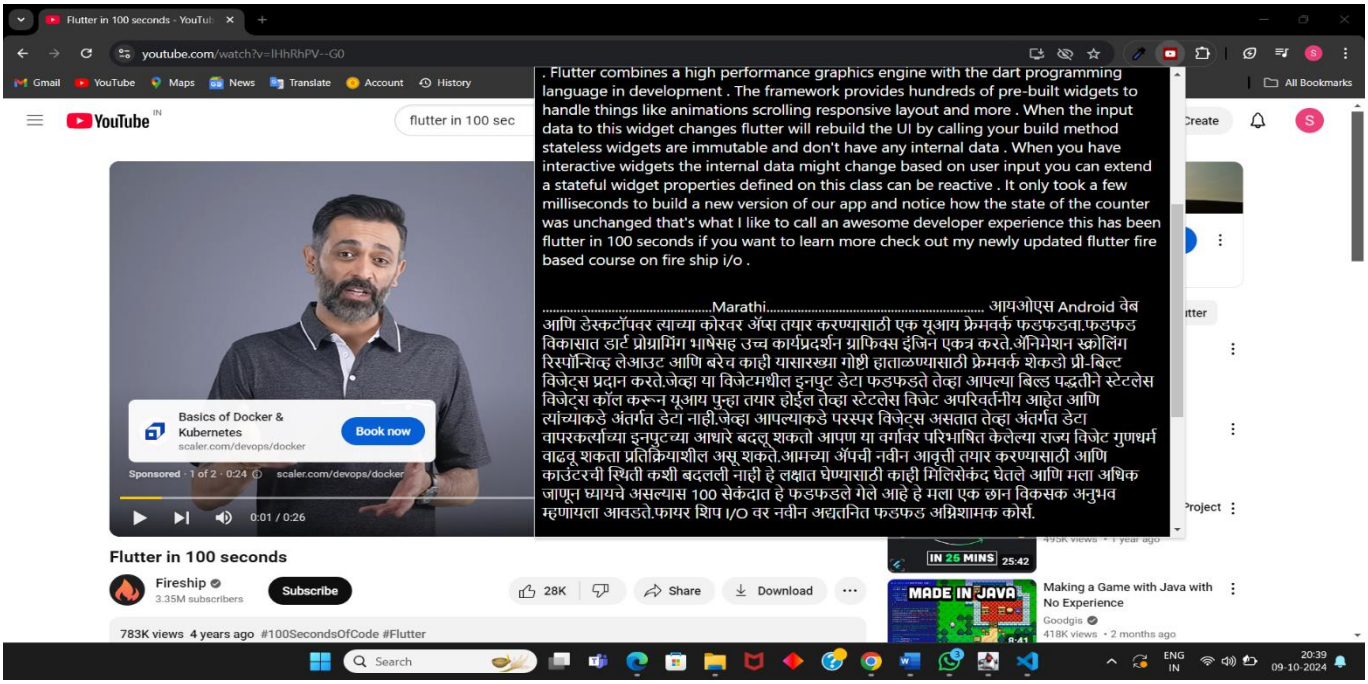


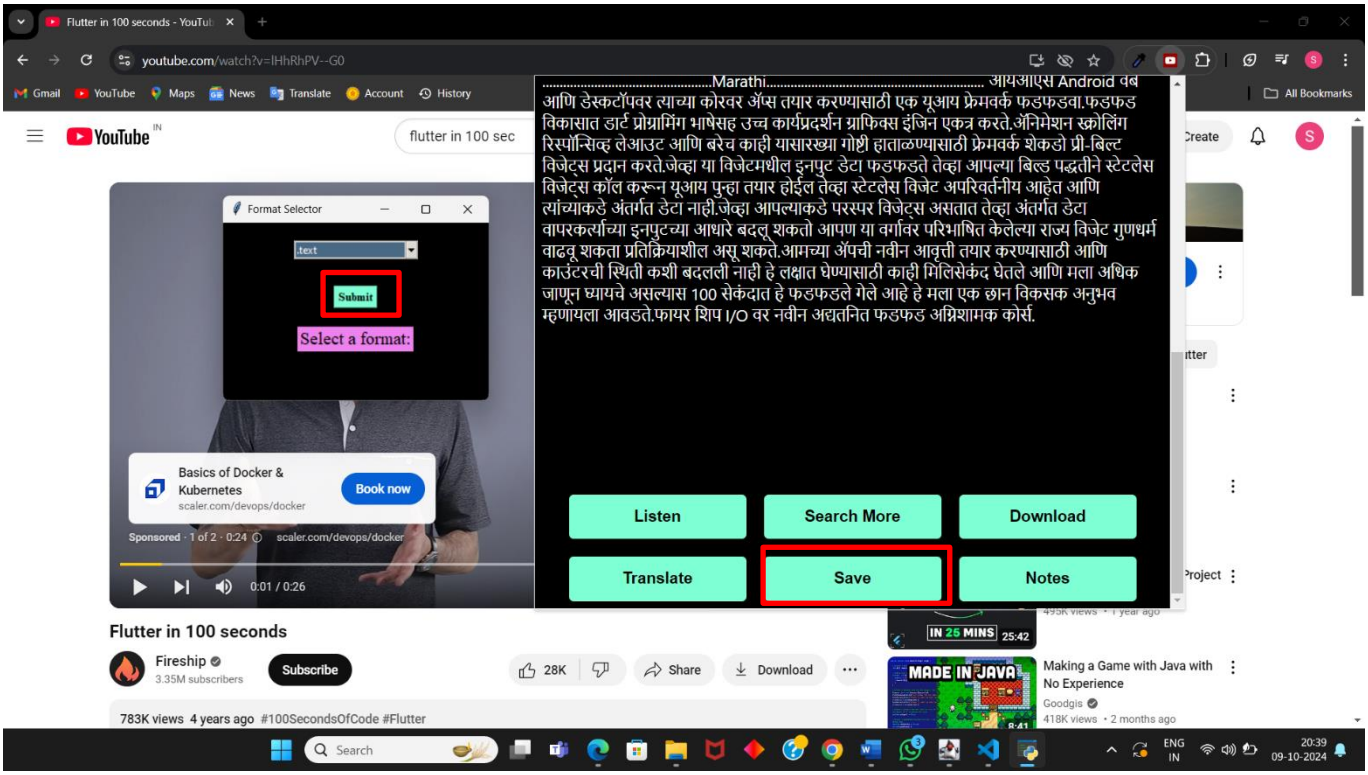

Youtube Transcript summarizer
Summarize
flutter in 100 sec
Flutter in 100 seconds
Fireship
3.35M subscribers
Subscribe
783K views 4 years ago #100SecondsOfCode #Flutter
Basics of Docker & Kubernetes
Book now
scaler.com/devops/docker
Flutter Basics by a REAL Project
Flutter Guys
495K views • 1 year ago
Making a Game with Java with No Experience
Goodgis
418K views • 2 months ago

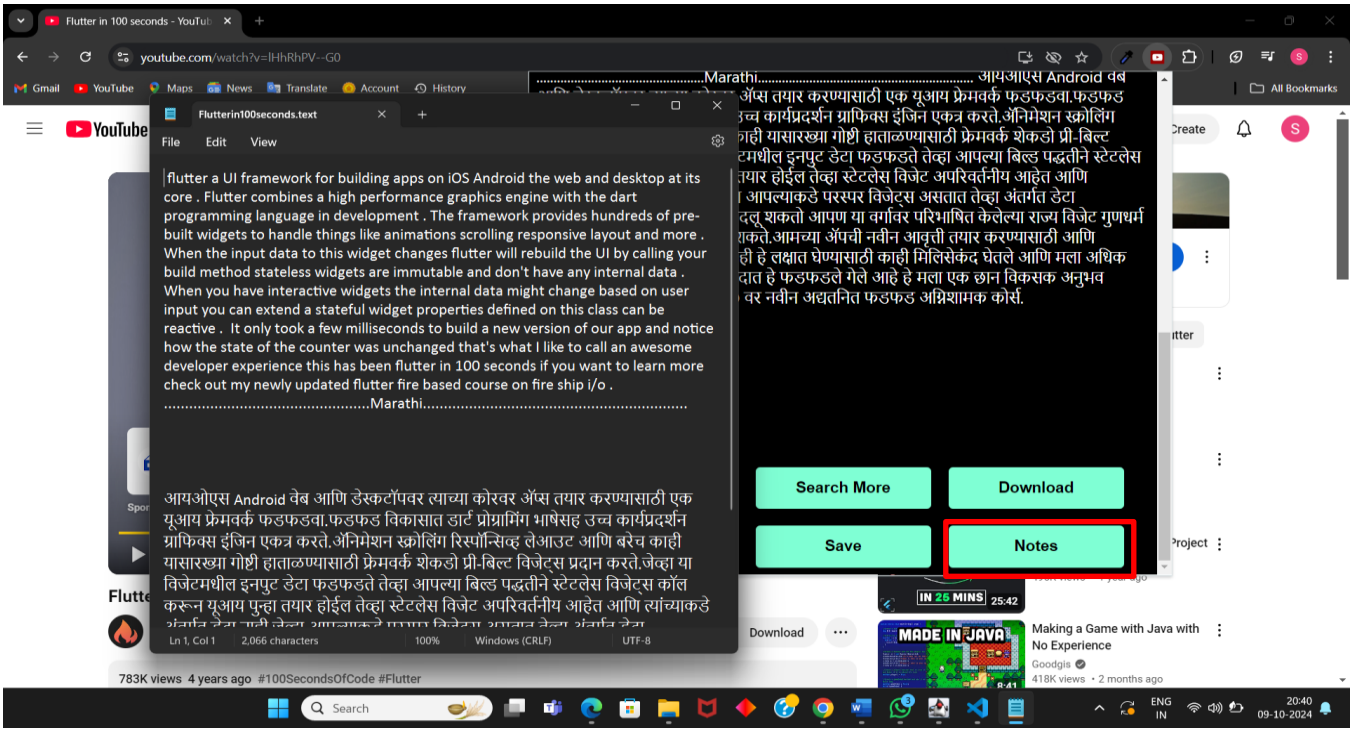

EXPLORER
OPEN EDITORS
YT
app.py
download.py
Flutterin100seconds.text
grmini.py
open_file.py
pdf.py
su
translate.py
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Using a pipeline without specifying a model name and revision in production is not recommended.
WARNING:tensorflow:From c:\Users\suraj\AppData\Local\Programs\Python\Python311\Lib\site-packages\tf_keras\src\losses.py:2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cross_entropy instead.
Your max_length is set to 142, but your input_length is only 129. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider decreasing max_length manually, e.g. summarizer('...', max_length=64)
127.0.0.1 - - [09/Oct/2024 20:38:32] "GET /summary?url=https://www.youtube.com/watch?v=lHhRhPV--G0 HTTP/1.1" 200 -

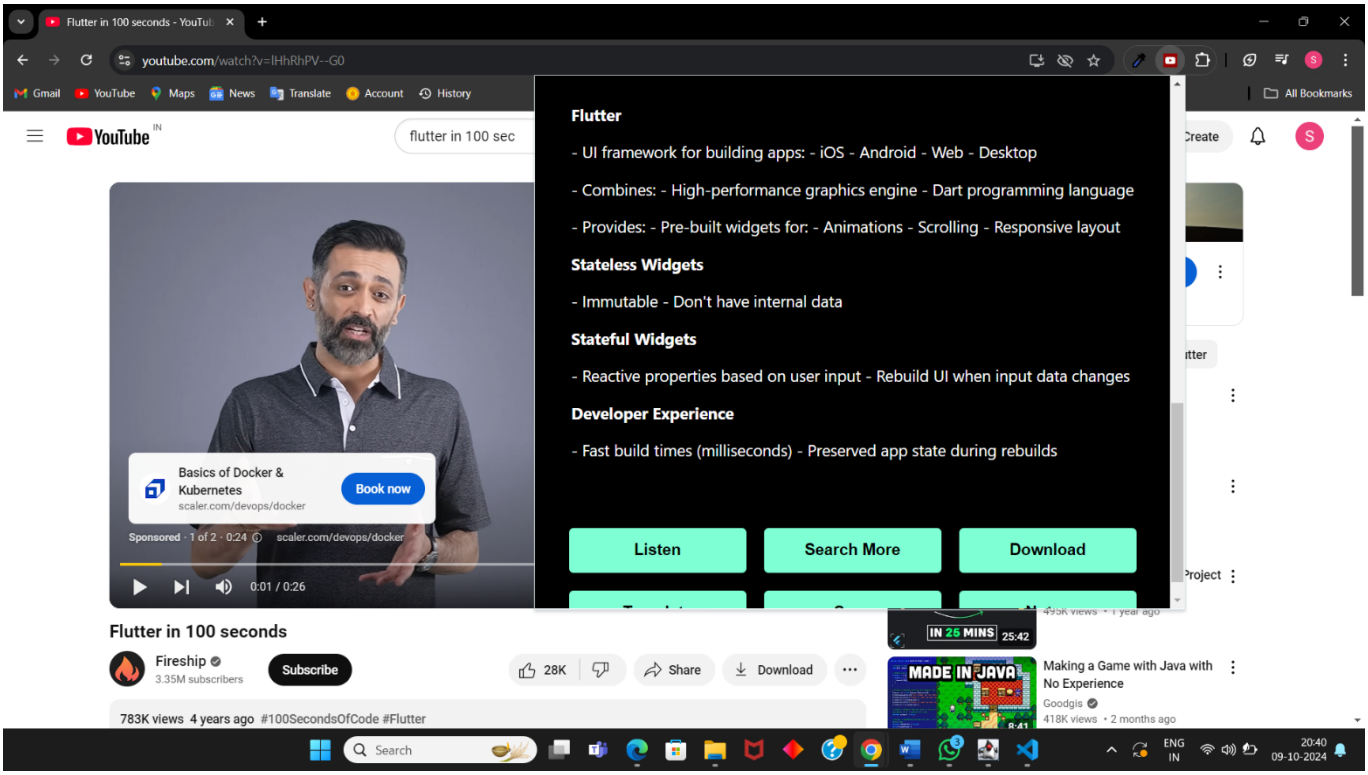

Youtube Transcript summarizer
Summarized
flutter a UI framework for building apps on iOS Android the web and desktop at its core . Flutter combines a high performance graphics engine with the dart programming language in development . The framework provides hundreds of pre-built widgets to handle things like animations scrolling responsive layout and more . When the input data to this widget changes flutter will rebuild the UI by calling your build method stateless widgets are immutable and don't have any internal data . When you have interactive widgets the internal data might change based on user input you can extend a stateful widget properties defined on this class can be reactive . It only took a few milliseconds to build a new version of our app and notice how the state of the counter was unchanged that's what I like to call an awesome developer experience this has been flutter in 100 seconds if you want to learn more check out my newly updated flutter fire
Flutter in 100 seconds
Fireship
Subscribe
Making a Game with Java with No Experience

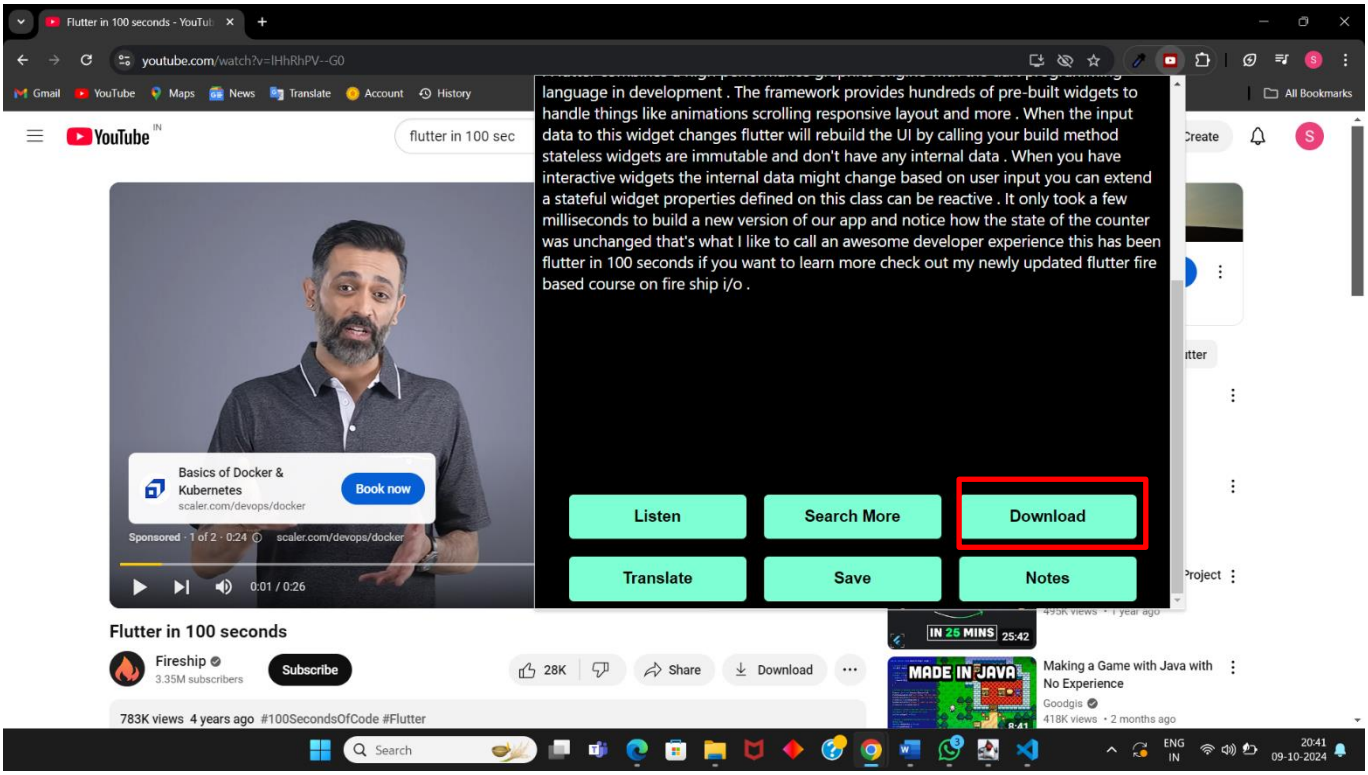

language in development . The framework provides hundreds of pre-built widgets to handle things like animations scrolling responsive layout and more . When the input data to this widget changes flutter will rebuild the UI by calling your build method stateless widgets are immutable and don't have any internal data . When you have interactive widgets the internal data might change based on user input you can extend a stateful widget properties defined on this class can be reactive . It only took a few milliseconds to build a new version of our app and notice how the state of the counter was unchanged that's what I like to call an awesome developer experience this has been flutter in 100 seconds if you want to learn more check out my newly updated flutter fire based course on fire ship i/o .
Language Translator
English
Select a language to translate:
Translate
Listen
Search More
Download
Translate
Save
Notes
Flutter in 100 seconds
Fireship
Subscribe
Making a Game with Java with No Experience

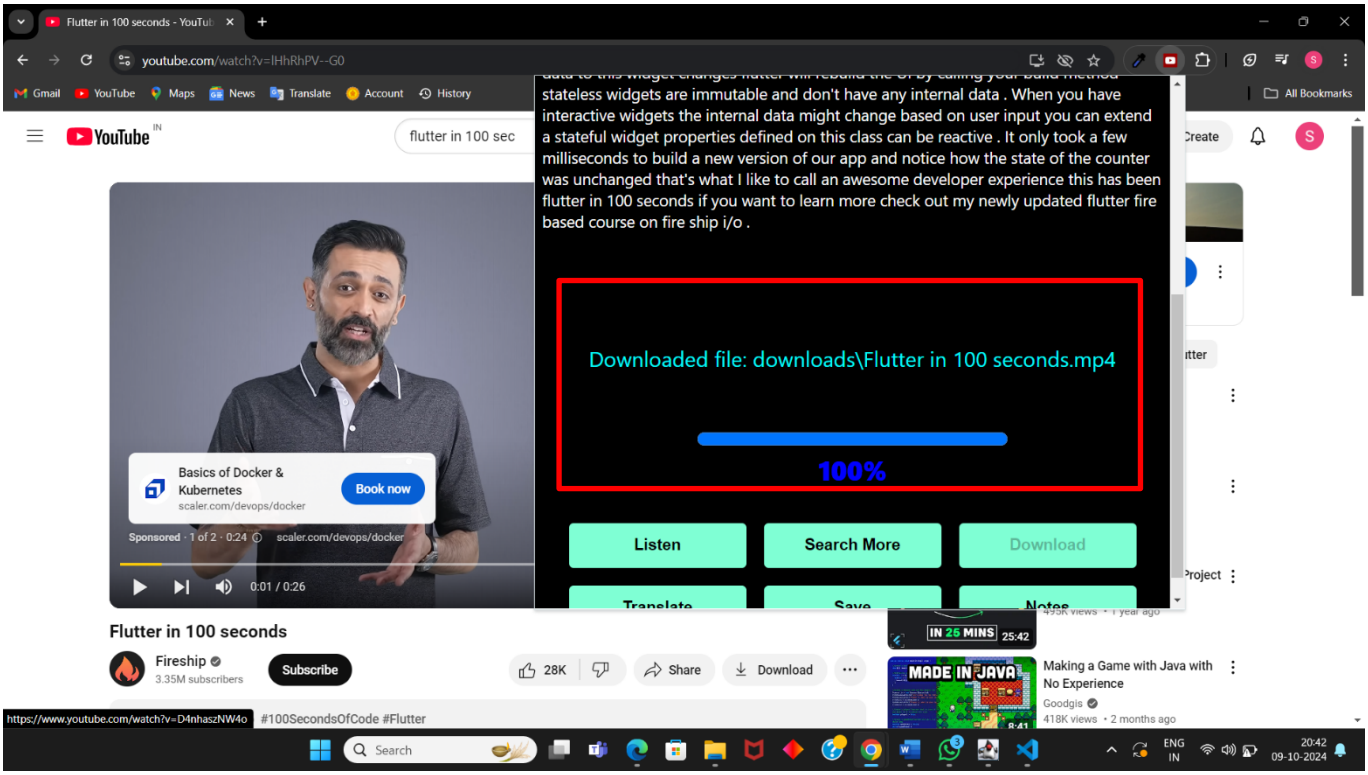

. Flutter combines a high performance graphics engine with the dart programming language in development . The framework provides hundreds of pre-built widgets to handle things like animations scrolling responsive layout and more . When the input data to this widget changes flutter will rebuild the UI by calling your build method stateless widgets are immutable and don't have any internal data . When you have interactive widgets the internal data might change based on user input you can extend a stateful widget properties defined on this class can be reactive . It only took a few milliseconds to build a new version of our app and notice how the state of the counter was unchanged that's what I like to call an awesome developer experience this has been flutter in 100 seconds if you want to learn more check out my newly updated flutter fire based course on fire ship i/o .
.....Marathi..... आयओएस Android वेब आणि डेस्कटॉपवर त्याच्या कोरवर ॲप्स तयार करण्यासाठी एक यूआय फ्रेमवर्क फडफडवा.फडफड विकासात डार्ट प्रोग्रामिंग भाषेसह उच्च कार्यप्रदर्शन ग्राफिक्स इंजिन एकत्र करते.ॲनिमेशन स्क्रोलिंग रिस्पॉन्सिव्ह लेआउट आणि बरेच काही यासारख्या गोष्टी हाताळण्यासाठी फ्रेमवर्क शेकडो प्री-बिल्ट विजेट्स प्रदान करते.जेव्हा या विजेटमधील इनपुट डेटा फडफडते तेव्हा आपल्या बिल्ड पद्धतीने स्टेटलेस विजेट्स कॉल करून यूआय पुन्हा तयार होईल तेव्हा स्टेटलेस विजेट अपरिवर्तनीय आहेत आणि त्यांच्याकडे अंतर्गत डेटा नाही.जेव्हा आपल्याकडे परस्पर विजेट्स असतात तेव्हा अंतर्गत डेटा वापरकर्त्याच्या इनपुटच्या आधारे बदलू शकतो आपण या वर्गावर परिभाषित केलेल्या राज्य विजेट गुणधर्म वाढवू शकता प्रतिक्रियाशील असू शकते.आमच्या ॲपची नवीन आवृत्ती तयार करण्यासाठी आणि काउंटरची स्थिती कशी बदलली नाही हे लक्षात घेण्यासाठी काही मिलिसेकंद घेतले आणि मला अधिक जाणून घ्यायचे असल्यास 100 सेकंदात हे फडफडले गेले आहे हे मला एक छान विकसक अनुभव म्हणायला आवडते.फायर शिप I/O वर नवीन अद्यतनित फडफड अग्निशामक कोर्स.
Basics of Docker & Kubernetes
scaler.com/devops/docker
Book now
Sponsored · 1 of 2 · 0:24
scaler.com/devops/docker
0:01 / 0:26
Flutter in 100 seconds
Fireship
3.35M subscribers
Subscribe
28K
Share
Download
783K views 4 years ago #100SecondsOfCode #Flutter
IN 25 MINS 25:42
Making a Game with Java with No Experience
Goodgis
418K views · 2 months ago


.....Marathi..... आयओएस Android वेब आणि डेस्कटॉपवर त्याच्या कोरवर ॲप्स तयार करण्यासाठी एक यूआय फ्रेमवर्क फडफडवा.फडफड विकासात डार्ट प्रोग्रामिंग भाषेसह उच्च कार्यप्रदर्शन ग्राफिक्स इंजिन एकत्र करते.ॲनिमेशन स्क्रोलिंग रिस्पॉन्सिव्ह लेआउट आणि बरेच काही यासारख्या गोष्टी हाताळण्यासाठी फ्रेमवर्क शेकडो प्री-बिल्ट विजेट्स प्रदान करते.जेव्हा या विजेटमधील इनपुट डेटा फडफडते तेव्हा आपल्या बिल्ड पद्धतीने स्टेटलेस विजेट्स कॉल करून यूआय पुन्हा तयार होईल तेव्हा स्टेटलेस विजेट अपरिवर्तनीय आहेत आणि त्यांच्याकडे अंतर्गत डेटा नाही.जेव्हा आपल्याकडे परस्पर विजेट्स असतात तेव्हा अंतर्गत डेटा वापरकर्त्याच्या इनपुटच्या आधारे बदलू शकतो आपण या वर्गावर परिभाषित केलेल्या राज्य विजेट गुणधर्म वाढवू शकता प्रतिक्रियाशील असू शकते.आमच्या ॲपची नवीन आवृत्ती तयार करण्यासाठी आणि काउंटरची स्थिती कशी बदलली नाही हे लक्षात घेण्यासाठी काही मिलिसेकंद घेतले आणि मला अधिक जाणून घ्यायचे असल्यास 100 सेकंदात हे फडफडले गेले आहे हे मला एक छान विकसक अनुभव म्हणायला आवडते.फायर शिप I/O वर नवीन अद्यतनित फडफड अग्निशामक कोर्स.
Format Selector
.text
Submit
Select a format:
Listen
Search More
Download
Translate
Save
Notes
Basics of Docker & Kubernetes
scaler.com/devops/docker
Book now
Sponsored · 1 of 2 · 0:24
scaler.com/devops/docker
0:01 / 0:26
Flutter in 100 seconds
Fireship
3.35M subscribers
Subscribe
28K
Share
Download
783K views 4 years ago #100SecondsOfCode #Flutter
IN 25 MINS 25:42
Making a Game with Java with No Experience
Goodgis
418K views · 2 months ago


Flutterin100seconds.text
File Edit View
flutter a UI framework for building apps on iOS Android the web and desktop at its core . Flutter combines a high performance graphics engine with the dart programming language in development . The framework provides hundreds of pre-built widgets to handle things like animations scrolling responsive layout and more . When the input data to this widget changes flutter will rebuild the UI by calling your build method stateless widgets are immutable and don't have any internal data . When you have interactive widgets the internal data might change based on user input you can extend a stateful widget properties defined on this class can be reactive . It only took a few milliseconds to build a new version of our app and notice how the state of the counter was unchanged that's what I like to call an awesome developer experience this has been flutter in 100 seconds if you want to learn more check out my newly updated flutter fire based course on fire ship i/o .
.....Marathi.....
आयओएस Android वेब आणि डेस्कटॉपवर त्याच्या कोरवर ॲप्स तयार करण्यासाठी एक यूआय फ्रेमवर्क फडफडवा.फडफड विकासात डार्ट प्रोग्रामिंग भाषेसह उच्च कार्यप्रदर्शन ग्राफिक्स इंजिन एकत्र करते.ॲनिमेशन स्क्रोलिंग रिस्पॉन्सिव्ह लेआउट आणि बरेच काही यासारख्या गोष्टी हाताळण्यासाठी फ्रेमवर्क शेकडो प्री-बिल्ट विजेट्स प्रदान करते.जेव्हा या विजेटमधील इनपुट डेटा फडफडते तेव्हा आपल्या बिल्ड पद्धतीने स्टेटलेस विजेट्स कॉल करून यूआय पुन्हा तयार होईल तेव्हा स्टेटलेस विजेट अपरिवर्तनीय आहेत आणि त्यांच्याकडे
Ln 1, Col 1 2,066 characters 100% Windows (CRLF) UTF-8
.....Marathi..... आयओएस Android वेब
ॲप्स तयार करण्यासाठी एक यूआय फ्रेमवर्क फडफडवा.फडफड
उच्च कार्यप्रदर्शन ग्राफिक्स इंजिन एकत्र करते.ॲनिमेशन स्क्रोलिंग
काही यासारख्या गोष्टी हाताळण्यासाठी फ्रेमवर्क शेकडो प्री-बिल्ट
टमधील इनपुट डेटा फडफडते तेव्हा आपल्या बिल्ड पद्धतीने स्टेटलेस
तयार होईल तेव्हा स्टेटलेस विजेट अपरिवर्तनीय आहेत आणि
आपल्याकडे परस्पर विजेट्स असतात तेव्हा अंतर्गत डेटा
दलू शकतो आपण या वर्गावर परिभाषित केलेल्या राज्य विजेट गुणधर्म
शकते.आमच्या ॲपची नवीन आवृत्ती तयार करण्यासाठी आणि
ही हे लक्षात घेण्यासाठी काही मिलिसेकंद घेतले आणि मला अधिक
दात हे फडफडले गेले आहे हे मला एक छान विकसक अनुभव
वर नवीन अद्यतनित फडफड अग्निशामक कोर्स.
Search More
Download
Save
Notes
783K views 4 years ago #100SecondsOfCode #Flutter
IN 25 MINS 25:42
Making a Game with Java with No Experience
Goodgis
418K views • 2 months ago


Flutter
- UI framework for building apps: - iOS - Android - Web - Desktop
- Combines: - High-performance graphics engine - Dart programming language
- Provides: - Pre-built widgets for: - Animations - Scrolling - Responsive layout
Stateless Widgets
- Immutable - Don't have internal data
Stateful Widgets
- Reactive properties based on user input - Rebuild UI when input data changes
Developer Experience
- Fast build times (milliseconds) - Preserved app state during rebuilds
Listen
Search More
Download
Basics of Docker & Kubernetes
scaler.com/devops/docker
Book now
Sponsored · 1 of 2 · 0:24
0:01 / 0:26
Flutter in 100 seconds
Fireship
3.35M subscribers
Subscribe
28K
Share
Download
783K views 4 years ago #100SecondsOfCode #Flutter
IN 25 MINS 25:42
Making a Game with Java with No Experience
Goodgis
418K views • 2 months ago


language in development . The framework provides hundreds of pre-built widgets to handle things like animations scrolling responsive layout and more . When the input data to this widget changes flutter will rebuild the UI by calling your build method stateless widgets are immutable and don't have any internal data . When you have interactive widgets the internal data might change based on user input you can extend a stateful widget properties defined on this class can be reactive . It only took a few milliseconds to build a new version of our app and notice how the state of the counter was unchanged that's what I like to call an awesome developer experience this has been flutter in 100 seconds if you want to learn more check out my newly updated flutter fire based course on fire ship i/o .
Listen
Search More
Download
Translate
Save
Notes
Flutter in 100 seconds
Fireship
3.35M subscribers
Subscribe
28K
Share
Download
783K views 4 years ago #100SecondsOfCode #Flutter
Making a Game with Java with No Experience
Basics of Docker & Kubernetes
Book now


stateless widgets are immutable and don't have any internal data . When you have interactive widgets the internal data might change based on user input you can extend a stateful widget properties defined on this class can be reactive . It only took a few milliseconds to build a new version of our app and notice how the state of the counter was unchanged that's what I like to call an awesome developer experience this has been flutter in 100 seconds if you want to learn more check out my newly updated flutter fire based course on fire ship i/o .
Downloaded file: downloads\Flutter in 100 seconds.mp4
100%
Listen
Search More
Download
Flutter in 100 seconds
Fireship
3.35M subscribers
Subscribe
28K
Share
Download
https://www.youtube.com/watch?v=D4nhaszNW4o #100SecondsOfCode #Flutter
Making a Game with Java with No Experience
Basics of Docker & Kubernetes
Book now

REFERENCES

1. https://chatgpt.com/
2. https://thepythoncode.com/article/text-summarization-using-huggingface-transformers-python
3. https://betterprogramming.pub/the-ultimate-guide-to-building-a-chrome-extension-4c01834c63ec
4. https://medium.com/coding-in-simple-english/how-to-create-chrome-extension-7dd396e884ef
5. https://pypi.org/project/youtube-transcript-api/
6. https://docs.python.org/3/library/os.html
7. https://www.youtube.com/watch?v=qkR06BJj30A