

# **1. INTRODUCTION**

## **1.1. AIM OF THE PROJECT**

The aim of the Macaulay Education Digitalized Video Application project is to make learning better by using computers to change how we learn. It wants to make school subjects more interesting by turning them into fun videos. This project also wants to fix problems in the current way of learning, like not teaching enough about morals and having a hard time sharing knowledge. By doing this, it hopes that everyone, no matter where they are, can easily learn important things. This project also wants to keep up with new ways of teaching so that learning is always fun and easy for everyone.

## **1.2. OBJECTIVE OF THE PROJECT**

The objectives for Macaulay Education Digitalized Video Application encompass DALL-E and ChatGPT APIs with Flutter and Firebase to enhance user experience. Use DALL-E for image generation and ChatGPT for natural language processing to enrich content. Develop a seamless user interface with Flutter for good navigation. Utilize Firebase for efficient data storage and retrieval. Aim for a comprehensive platform revolutionizing education, providing engaging learning experiences worldwide. Continuously iterate and improve to remain innovative in the dynamic digital landscape.

## **2. SYSTEM ANALYSIS**

### **2.1. FEASIBILITY STUDY**

The Macaulay Education Digitalized Video Application looks promising but needs thorough evaluation in technical, financial, operational, market, legal, ethical, and resource aspects. Using advanced technologies like AI, Flutter, and Firebase, it faces challenges in smooth integration and user engagement. Financially, it requires investment in development and revenue models. Operational success relies on user experience and responsiveness. Market analysis is vital for acceptance, and legal compliance is necessary. Adequate resources and planning are crucial for execution. Overall, strategic planning is needed for successful implementation.

### **2.2. EXISTING SYSTEM**

- ChatGPT AI model by OpenAI, generates human-like text responses, enhances natural language understanding and conversation.
- DALL-E is an AI model developed by OpenAI capable of generating images from textual descriptions.

### **2.3. PROPOSED SYSTEM**

- Our proposed system is a text-to-video application, leveraging ChatGPT and DALL-E AI APIs for AI-driven conversion. ChatGPT generates text responses from user queries, while DALL-E AI converts them into images.
- These images are aligned and given voice-over, then assembled into videos using Python libraries. The frontend is developed using Flutter, accessed through the Dart language, with Firebase serving as the backend for user data storage

### **3. SYSTEM REQUIREMENTS**

#### **3.1. HARDWARE REQUIREMENTS**

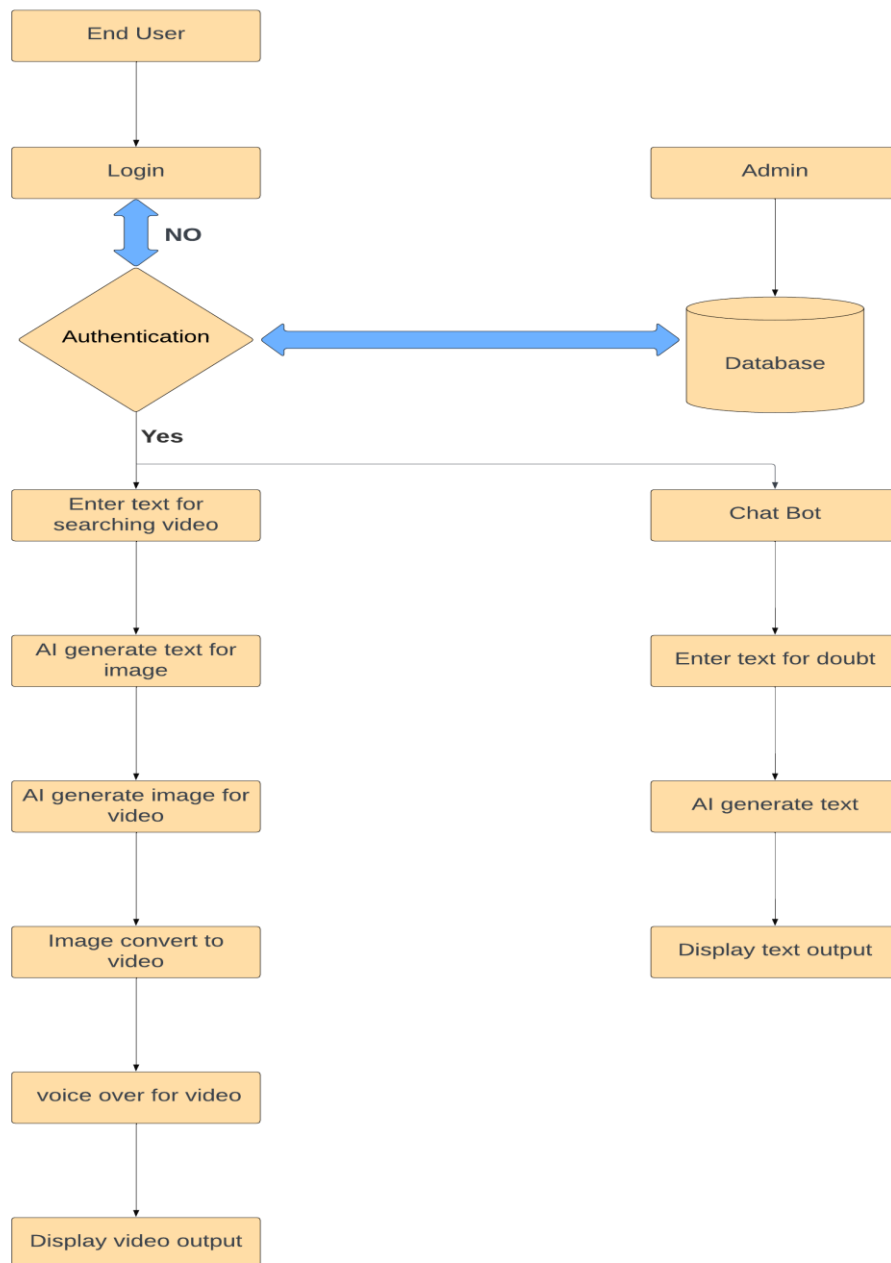
- Processor : Intel i5 10th Gen
- RAM : 8 GB
- Hard Disk : 250 GB
- Graphics Card : Nvidia Geforce GTX 1650

#### **3.2. SOFTWARE REQUIREMENTS**

- Operating system : Windows10
- Front-end Framework : Flutter (Dart programming)
- Language : Python
- Back-end : Firebase

## 4. SYSTEM DESIGN

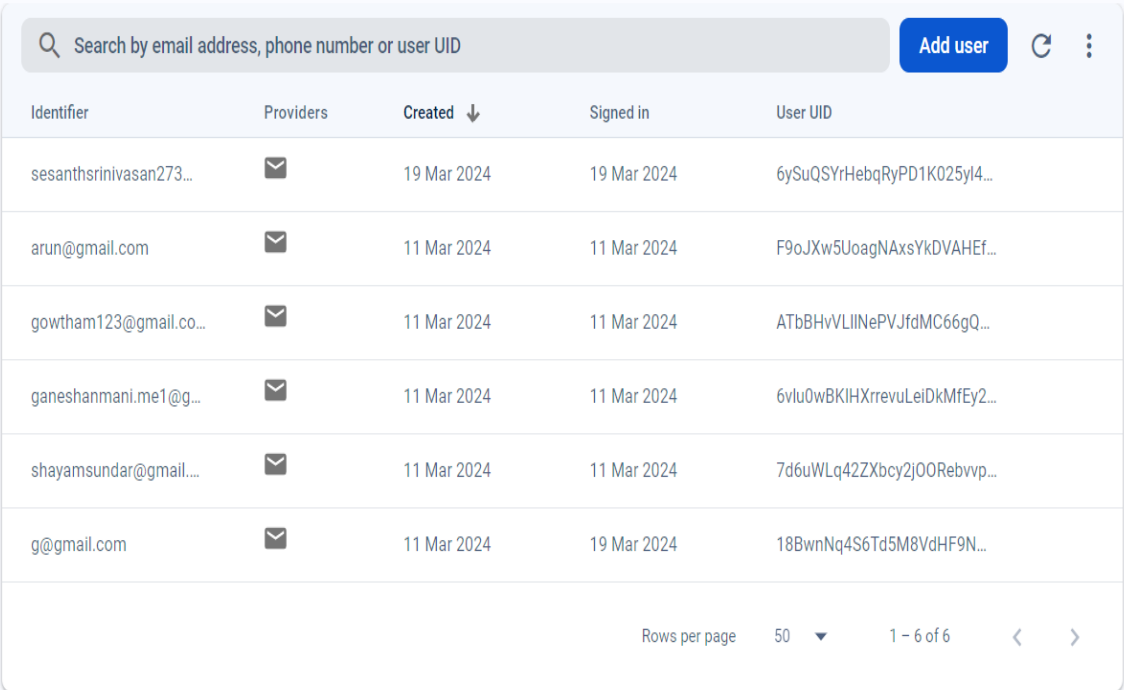
### 4.1. SYSTEM ARCHITECTURE









## 4.2. DATABASE DESIGN

### FIREBASE AUTHENTICATION

- Firebase Authentication aims to make building secure authentication systems easy, while improving the sign-in and onboarding experience for end users.



Identifier	Providers	Created ↓	Signed in	User UID
sesanthsrinivasan273...		19 Mar 2024	19 Mar 2024	6ySuQSYrHebqRyPD1K025yl4...
arun@gmail.com		11 Mar 2024	11 Mar 2024	F9oJXw5UoagNAXsYkDVAHEf...
gowtham123@gmail.co...		11 Mar 2024	11 Mar 2024	ATbBHvVLIINePVJfdMC66gQ...
ganeshanmani.me1@g...		11 Mar 2024	11 Mar 2024	6vlu0wBKIHXrrevuLeiDkMfEy2...
shayamsundar@gmail....		11 Mar 2024	11 Mar 2024	7d6uWLq42ZXbcy2j00Rebvvp...
g@gmail.com		11 Mar 2024	19 Mar 2024	18BwnNq4S6Td5M8VdHF9N...

Rows per page 50 1 – 6 of 6

## **5. MODULES DESCRIPTION**

### **1) Searching for video through text :**

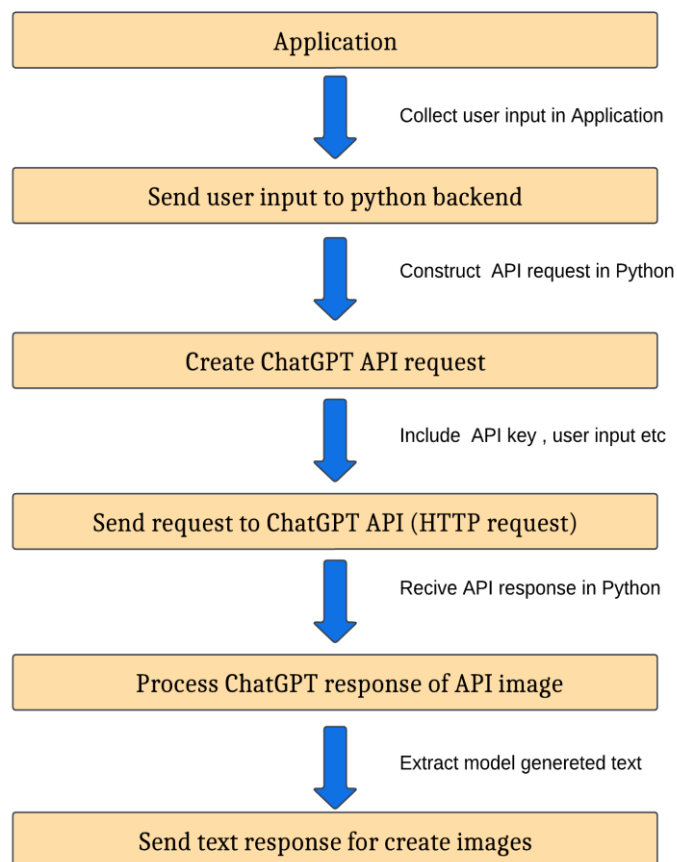
- i. AI generate text for image
- ii. AI generate image for video
- iii. Image convert to video
- iv. Subtitle and voice over for video

### **2) Chat Bot**

## 1) Searching for video through text:

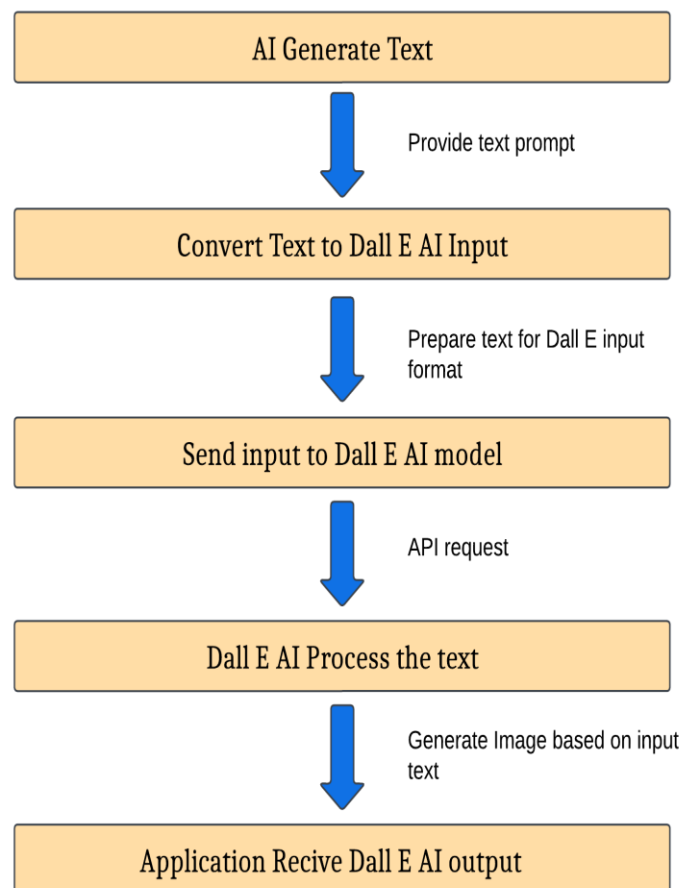
### i. AI generate text for image:-

This module interacts with the ChatGPT by using the open AI API key .It sends prompt from user and generates text based response by sending prompt to ChatGPT and receiving relevant response for creating image.



## ii. AI generate image for video:

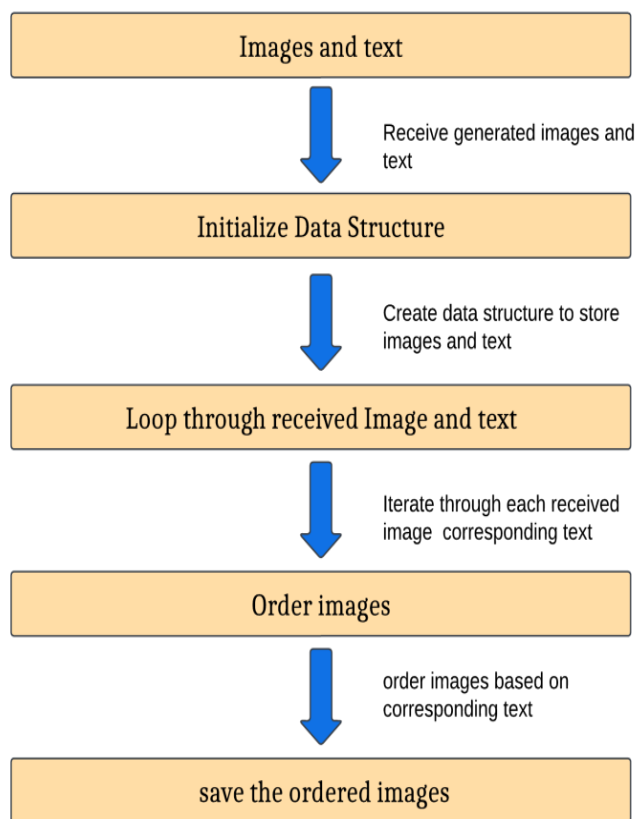
Text based response from ChatGPT is converted to images using Dall E AI. It is done by sending prompt to Dall E using its API key and Dall E AI generates Images based on the prompt.





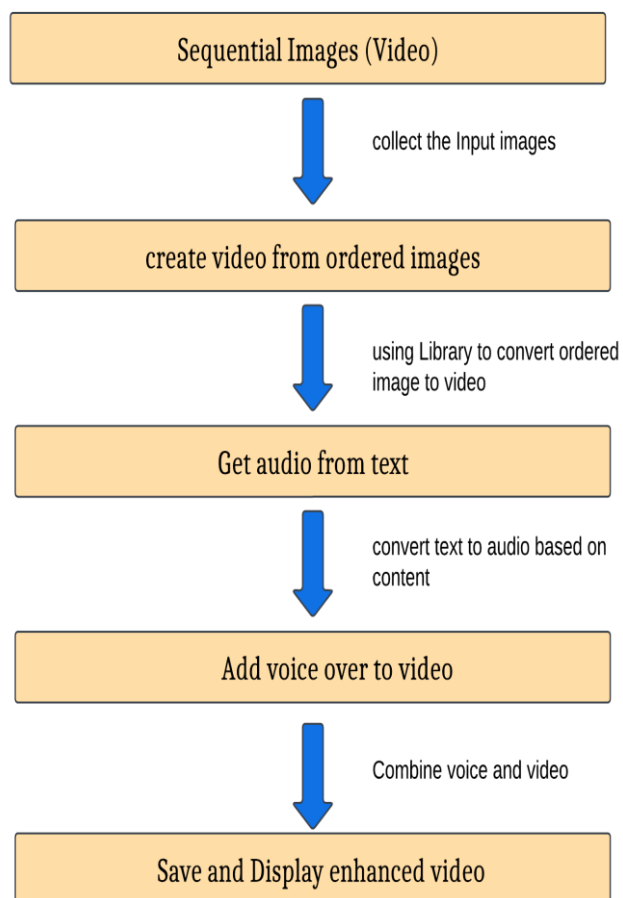
### iii. Image convert to video:

This module process images generated by the Dall E AI model, aligning them sequentially based on their corresponding text description upon receiving the images and text, Program initializes data structure, iterates through each pair and orders the images.



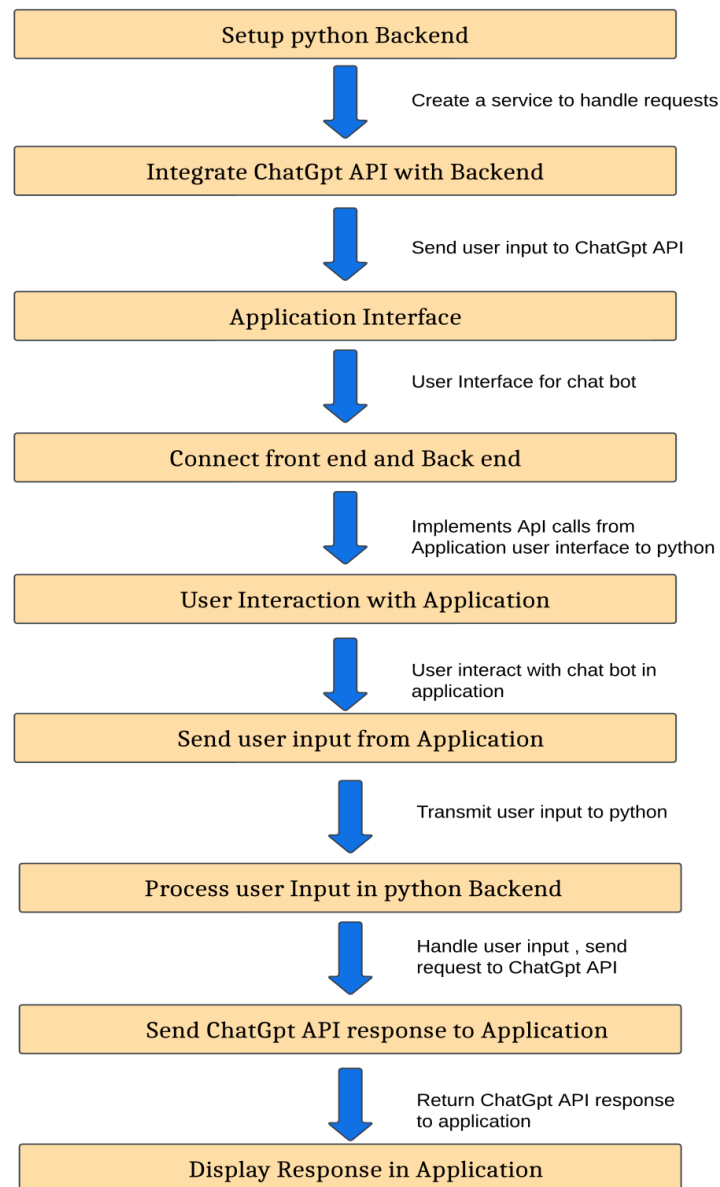
#### iv. Voice over for video:

This module enhances video by automatically generating voice over Using with python. This module convert's sequence of image into a good enhances multimedia content.



## 2) Chat Bot:

A user can clarify the doubt using in build chat bot. This module get's user question and Clarifies with text based response using ChatGPT API key.



## 6. SOURCE CODE

**Video :**

**Main.py:**

```
import time

import firebase_admin

from firebase_admin import credentials, storage, db

import image_to_video as gg

import os

from openai import OpenAI

import cv2

import numpy as np

import requests

import pyttsx3

from moviepy.editor import VideoFileClip, AudioFileClip

client = OpenAI(api_key="sk-
pJOR78UDHo38DBmXdIUhT3BibkFJ2h8le9Oobgulmi3biX3X")

def text_generation(PROMPT, maxtokens=200):#generates text based response

    response = client.completions.create(

        model="gpt-3.5-turbo-instruct",

        prompt=PROMPT,

        max_tokens=maxtokens

    )
```

```

output = response.choices[0].text.strip()

return output

def imgtext_generation(PROMPT, maxtokens=170):#generates prompt for
generating images

# Generate completion

response = client.completions.create(

    model="gpt-3.5-turbo-instruct",

    prompt=PROMPT+"(based on the text specified give me a prompt to generate
related images(only reply prompt that generate images make it easy) ) ",

    max_tokens=maxtokens

)

imgoutput = response.choices[0].text.strip()

return imgoutput

def image_creation(PROMPT, no_of_img=5):#generates images

image_urls = []

response = client.images.generate(

    model="dall-e-2",

    prompt=PROMPT,

    size="1024x1024",

    quality="standard",

    n=no_of_img

)

for data in response.data:

```

```

        image_urls.append(data.url)

    return image_urls

def img_to_url(url):    #gets images from url

    for idx, image_url in enumerate(image_urls):

        # Fetch the image from the URL

        response = requests.get(image_url)

        # Check if the request was successful

        if response.status_code == 200:

            # Decode the image content

            image_data = response.content

            # Convert the image data into a NumPy array

            np_array = np.frombuffer(image_data, np.uint8)

            # Decode the NumPy array into an OpenCV image object

            opencv_image = cv2.imdecode(np_array, cv2.IMREAD_COLOR)

            # Save the image

            name = f"image_{idx}.jpg"

            cv2.imwrite(f"M:/login_page_official/new_in_tamil/VideoSenderReceiver/images/{name}", opencv_image)

            # Display the image

            #cv2.imshow("Image", opencv_image)

            cv2.waitKey(0)

            cv2.destroyAllWindows()

def text_to_speech(text, output_file):# generates speech

```

```

engine = pyttsx3.init()

engine.setProperty("speed",100)

# Save the speech as an audio file

engine.save_to_file(text, output_file)

engine.runAndWait()

def merge_audio_video(video_file, audio_file, output_file):#mearges audio and video
file

# Load the video clip

video_clip = VideoFileClip(video_file)

# Load the audio clip

audio_clip = AudioFileClip(audio_file)

# Set the audio of the video clip to the loaded audio clip

if audio_clip.duration > video_clip.duration:

    audio_clip = audio_clip.subclip(0, video_clip.duration)

elif audio_clip.duration < video_clip.duration:

    video_clip = video_clip.subclip(0, audio_clip.duration)

video_clip = video_clip.set_audio(audio_clip)

# Write the merged video file

video_clip.write_videofile(output_file, codec="libx264", audio_codec="aac")

#video_clip.show()

# Initialize Firebase Admin SDK

cred = credentials.Certificate("credentials.json")

```

```

firebase_admin.initialize_app(cred, {'storageBucket': 'mar10-
89ff8.appspot.com','databaseURL': 'https://mar10-89ff8-default-
rtbd.firebaseio.com/'})

refQ = db.reference('Video/SendQueryFromFlutter/Question')

while True:

    if refQ.get() != "":

        QueryText = refQ.get()

        # input for video generator code --- > QueryText

        print("Question : ", QueryText)

        userprompt = QueryText

        text_response = text_generation(userprompt) #text based on query

        print(text_response)

        img_prompt = imgtext_generation(text_response) # generate image prompt

        print(img_prompt)

        image_urls = image_creation(img_prompt, 5) # gets img from url

        print(image_urls)

        img_to_url(image_urls)

        img_to_videoobj = gg.img2vvideo()

        img_to_videoobj.img_to_video()

        text_to_speech(text_response,
"M:/login_page_official/new_in_tamil/VideoSenderReceiver/audio/output_audio.mp3"
)

merge_audio_video("M:/login_page_official/new_in_tamil/VideoSenderReceiver/vide
o/fun.avi","M:/login_page_official/new_in_tamil/VideoSenderReceiver/audio/output_a

```



```
udio.mp3","M:/login_page_official/new_in_tamil/VideoSenderReceiver/videoaudio/fish.mp4")
```

```
    #.sleep(5) #take it after
```

```
    #for firebase storage
```

```
    bucket = firebase_admin.storage.bucket()
```

```
    # Path to the local file you want to upload
```

```
    local_file_path =
```

```
"M:/login_page_official/new_in_tamil/VideoSenderReceiver/videoaudio/fish.mp4"
```

```
    # Destination path in Firebase Storage
```

```
    destination_blob_name = "test/fish.mp4"
```

```
    # Upload the local file to Firebase Storage
```

```
    blob = bucket.blob(destination_blob_name)
```

```
    blob.upload_from_filename(local_file_path)
```

```
    print("File uploaded to Firebase Storage.")
```

```
    refQ = db.reference('Video/SendQueryFromFlutter/Question')
```

```
    QueryText = refQ.set("")
```

```
else:
```

```
    print("no data initialized")
```

```
time.sleep(5)
```

**image\_to\_video.py:**

```
import cv2
```

```
from PIL import Image
```

```
import os
```

```

import pytsx3

from moviepy.editor import VideoFileClip, AudioFileClip

class img2vvideo:

    def img_to_video(self):

        os.chdir("M:/login_page_official/new_in_tamil/VideoSenderReceiver/images")

        path=os.getcwd()

        #os.remove("D:\output\fun.avi")

        video_output_path="M:/login_page_official/new_in_tamil//VideoSenderReceiver/video"

        video_name="fun.avi"

        video_output_fullpath=os.path.join(video_output_path,video_name)

        images_name=os.listdir()

        imges=[]

        for i in images_name:

            g=os.path.join(path,i)

            imges.append(g)

        im_obj=cv2.imread(imges[0])

        height,width=im_obj.shape[:2]

        tuple1=(width,height)

        print((height,width))

        #fourcc = cv2.VideoWriter_fourcc(*'mp4v')

        video=cv2.VideoWriter(video_output_fullpath,0,0.1,tuple1)

        for i in imges:

```

```
video.write(cv2.imread(i))
```

```
video.release()
```

## **ChatBot :**

### **Main.py :**

```
import firebase_admin
```

```
from firebase_admin import credentials
```

```
from firebase_admin import db
```

```
import time
```

```
from openai import OpenAI
```

```
client = OpenAI(api_key="sk-  
pJOR78UDHo38DBmXdIUhT3BibkFJ2h8le9Oobgulmi3biX3X")
```

```
def text_generation(PROMPT, maxtokens=200):#generates text based response
```

```
    response = client.completions.create(
```

```
        model="gpt-3.5-turbo-instruct",
```

```
        prompt=PROMPT,
```

```
        max_tokens=maxtokens
```

```
    )
```

```
    output = response.choices[0].text.strip()
```

```
    return output
```

### **# Fetch the service account key JSON file path**

```
cred = credentials.Certificate("credentials.json") #change
```

### **# Initialize the Firebase app with the service account credentials #change**

```
firebase_admin.initialize_app(cred, {
```

```

'databaseURL': 'https://mar10-89ff8-default-rtdb.firebaseio.com/'

})

# Reference to your Firebase database

refQ = db.reference('ChatBot/sendTextFromFlutterForChatBot/TextQuery')
#question #change

refA = db.reference('ChatBot/receiveTextFromPythonForChatbot/QueryAnswer')
#answer

while True:

    if refQ.get() != "":

        QueryText = refQ.get()

        print("Question : "+ QueryText)

        chat_gpt_answer = text_generation(QueryText)

        QueryAnswer = refA.set(chat_gpt_answer)

        refA1 = db.reference('ChatBot/receiveTextFromPythonForChatbot/QueryAnswer')
#answer

        QueryAnswer1 = refA1.get()

        print(QueryAnswer1)

        refQ = db.reference('ChatBot/sendTextFromFlutterForChatBot/TextQuery')
#question

        QueryText = refQ.set("")

    else:

        print("no data initilize")

    time.sleep(5)

```

## Front - end :

### Main.dart:

```
import 'package:firebase_core/firebase_core.dart';

import 'package:flutter/material.dart';

import 'package:get/get_navigation/src/root/get_material_app.dart';

import 'package:login_signup/firebase_options.dart';

import 'package:login_signup/screens/sign_in.dart';

Future<void> main() async {

  WidgetsFlutterBinding.ensureInitialized();

  await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);

  runApp(const MyApp());

}

class MyApp extends StatefulWidget {

  const MyApp({super.key});

  @override

  State<MyApp> createState() => _MyAppState();

}

class _MyAppState extends State<MyApp> {

  @override

  Widget build(BuildContext context) {

    return GetMaterialApp(home: AppSignInScreen(),

    );

  }}

}}
```

## Signin.dart :

```
import 'package:flutter/material.dart';

import 'package:login_signup/authmanagement/authmanage.dart';

import 'package:login_signup/screens/forgot_password.dart';

import 'package:login_signup/screens/sign_up.dart';

class AppSignInScreen extends StatefulWidget {

  const AppSignInScreen({super.key});

  @override

  State<AppSignInScreen> createState() => _AppSignInScreenState();

}

class _AppSignInScreenState extends State<AppSignInScreen> {

  bool textVisible = true;

  final _signInFormKey = GlobalKey<FormState>();

  TextEditingController username = TextEditingController();

  TextEditingController userpass = TextEditingController();

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Form( key: _signInFormKey, child: SafeArea(child: ListView(children: [

        SizedBox(height: 300,width: 500, child: Image.asset("assets/png_images/sign-
in-img.png")),Padding( padding: EdgeInsets.all(20.0), child: Text("Sign In", style:
TextStyle(color: Colors.black, fontSize: 25,), ),),Padding(padding: const
EdgeInsets.all(20.0),child: TextFormField( controller: username,validator:(username)
{ if (username!.isEmpty && username != null)
```

```

return "Enter the Username"  }

return null;

},

decoration: InputDecoration(hintText: "Enter email address", labelText: "Username",

labelStyle: TextStyle(color: Colors.black), prefixIcon: Icon(Icons.person,

color: Colors.blue, ), focusedBorder: OutlineInputBorder( borderRadius:

BorderRadius.circular(20), borderSide: BorderSide(color: Colors.black), ),

border: OutlineInputBorder( borderRadius: BorderRadius.circular(20),

borderSide: BorderSide(color: Colors.black), ), ), ), Padding( padding: const

EdgeInsets.all(20.0), child: TextFormField(controller: userpass, validator: (userpass) {

if (userpass!.isEmpty && userpass != null) {

return "Enter the Password";

}

return null; }, obscureText: textVisible, decoration: InputDecoration( hintText: "Enter

your password", labelText: "Password", labelStyle: TextStyle(color: Colors.black),

prefixIcon: Icon(Icons.password, color: Colors.blue, ), suffixIcon: IconButton(

onPressed: () { setState(() { textVisible = !textVisible; }); }, icon: textVisible? Icon

Icons.visibility_off, color: Colors.blue, ): Icon( Icons.visibility, color: Colors.blue, ), ),

focusedBorder: OutlineInputBorder(borderRadius: BorderRadius.circular(20),

borderSide: BorderSide(color: Colors.black), ), border: OutlineInputBorder(

borderRadius: BorderRadius.circular(20), borderSide: BorderSide(color:

Colors.black), ), ), ), Padding(padding: const EdgeInsets.only(top: 10, bottom: 10, left:

20, right: 20, ), child: ElevatedButton(onPressed: () async {

if (_signInFormKey.currentState!.validate()) {

String? error = await AuthManage().login(username.text, userpass.text, );

```

```

if (error != null) {

  showDialog(context: context, builder: (context) {return AlertDialog(backgroundColor:
Colors.white,title: Text("Error"),content: Text(error),actions: [ TextButton(onPressed:
() {Navigator.pop(context);},child: Text("OK"),),],);},)} },style:
ElevatedButton.styleFrom(backgroundColor: Colors.blue,foregroundColor:
Colors.white,shape: RoundedRectangleBorder( borderRadius:
BorderRadius.circular(20.0), ), ),child: Text("Login",style: TextStyle(fontSize: 18.0),
),), ),Padding(padding: const EdgeInsets.only(right: 20, left: 20), child: Row(
mainAxisAlignment: MainAxisAlignment.end,children: [ TextButton(
child: Text("Forgot Password ?"),onPressed: () {Navigator.push( context,
MaterialPageRoute(builder: (context) => AppForgotPassword(),),);},),],),),
Row( mainAxisAlignment: MainAxisAlignment.center,children: [Text("New User?"),
TextButton( onPressed: () { Navigator.push( context,MaterialPageRoute(builder:
(context) => AppSignUpScreen(), ), },child: Text("Signup Now"),),],),),);}}

```

## Signup.dart:

```

import 'package:flutter/material.dart';

import 'package:login_signup/authmanagement/authmanage.dart';

import 'package:login_signup/screens/sign_in.dart';

class AppSignUpScreen extends StatefulWidget {

  const AppSignUpScreen({super.key});

  @override

  State<AppSignUpScreen> createState() => _AppSignUpScreenState();

}

class _AppSignUpScreenState extends State<AppSignUpScreen> {

  final _signupFormKey = GlobalKey<FormState>();

```



```

TextEditingController username = TextEditingController();

TextEditingController userMobile = TextEditingController();

TextEditingController userpass = TextEditingController();

TextEditingController userconfirmPass = TextEditingController();

bool passTextVisible = true;

bool cpassTextVisible = true;

@override

Widget build(BuildContext context) {

return Scaffold(appBar: AppBar(),backgroundColor: Colors.white,body: SafeArea(

child: Form(key: _signupFormKey, child: ListView( children: [SizedBox(

height: 200, width: 500,child: Image.asset("assets/png_images/sign_up_img.png")),

Padding(padding: EdgeInsets.all(20.0),child: Text( "Sign Up",style: TextStyle(color:

Colors.black,fontSize: 25,),),),Padding(padding: const EdgeInsets.only(left: 20, right:

20, bottom: 10),child: TextFormField(controller: username,validator: (username) {

if (username != null && username.isEmpty) {

return "Enter your email";

}

return null;

},decoration: InputDecoration(hintText: "Enter Username",labelText: "Username",

labelStyle: TextStyle(color: Colors.black),prefixIcon: Icon(Icons.person,color:

Colors.blue,), focusedBorder: OutlineInputBorder(borderRadius:

BorderRadius.circular(20),borderSide: BorderSide(color: Colors.black),),

border: OutlineInputBorder(borderRadius: BorderRadius.circular(20), borderSide:

BorderSide(color: Colors.black),),),),Padding(padding: const EdgeInsets.only(left:

20, right: 20, bottom: 10),child: TextFormField(keyboardType:TextInputType.number,

```

```

maxLength: 10,controller: userMobile,validator: (mobilenos) {

  if (mobilenos != null && mobilenos.isEmpty) {

    return "Enter your 10 digit Mobile Number";

  } return null;

},decoration: InputDecoration(hintText: "Enter your Mobile Number",labelText:
"Mobile Number",labelStyle: TextStyle(color: Colors.black),prefixIcon: Icon(
Icons.phone,color: Colors.blue,), focusedBorder: OutlineInputBorder(borderRadius:
BorderRadius.circular(20),borderSide: BorderSide(color: Colors.black),), border:
OutlineInputBorder(borderRadius: BorderRadius.circular(20), borderSide:
BorderSide(color: Colors.black),),),),Padding(padding: const EdgeInsets.only(left:
20, right: 20, bottom: 10),child: TextFormField(obscureText: passTextVisible,

maxLength: 6,controller: userpass,validator: (userpass) {

  if (userpass != null && userpass.isEmpty) {

    return "Enter your 6 digit password";

  } return null;

},decoration: InputDecoration(hintText: "Enter your password",labelText:
"Password",labelStyle: TextStyle(color: Colors.black), prefixIcon: Icon(
Icons.password,color: Colors.blue,), suffixIcon: IconButton(onPressed: () {

setState(() {passTextVisible = !passTextVisible; });},icon: passTextVisible ? Icon(
Icons.visibility_off,color: Colors.blue,): Icon( Icons.visibility, color: Colors.blue,)),
focusedBorder: OutlineInputBorder(borderRadius:
BorderRadius.circular(20),borderSide: BorderSide(color: Colors.black),),border:
OutlineInputBorder( borderRadius: BorderRadius.circular(20),borderSide:
BorderSide(color: Colors.black),),),),),Padding(padding: const EdgeInsets.only(left:
20, right: 20, bottom: 10),child: TextFormField( obscureText: cpassTextVisible,

maxLength: 6,controller: userconfirmPass,validator: (userconfirmPass) {

  if (userconfirmPass != null && userconfirmPass.isEmpty) {

```

```

return "Enter your confirm 6 digit password";

} return null;

},decoration: InputDecoration(hintText: "Enter your Confirm password", labelText:
"Confirm Password",labelStyle: TextStyle(color: Colors.black), prefixIcon: Icon(
Icons.password, color: Colors.blue,),suffixIcon: IconButton( onPressed: () {
setState(() {

        cpassTextVisible = !cpassTextVisible;

});}, icon: cpassTextVisible ? Icon(Icons.visibility_off,color: Colors.blue):
Icon(Icons.visibility,color: Colors.blue,),), focusedBorder: OutlineInputBorder(

borderRadius: BorderRadius.circular(20), borderSide: BorderSide(color:
Colors.black),),border: OutlineInputBorder(borderRadius: BorderRadius.circular(20),
borderSide: BorderSide(color: Colors.black),),),),Padding(padding: const
EdgeInsets.only(bottom: 5, left: 20, right: 20),child: ElevatedButton(

onPressed: () async {

    if (_signupFormKey.currentState!.validate()) {

        String? error = await AuthManage().signup(username.text.trim(), userpass.text);

        if (error != null) {

            showDialog(context: context,builder: (context) {return AlertDialog(
backgroundColor: Colors.white,title: Text("Error"),content: Text(error), actions: [
TextButton(onPressed: () {Navigator.pop(context);},child: Text("OK"),),],); },)},
style: ElevatedButton.styleFrom(backgroundColor: Colors.blue,foregroundColor:
Colors.white, shape: RoundedRectangleBorder(borderRadius:
BorderRadius.circular(20.0),),),child: Padding( padding: const EdgeInsets.all(10.0),
child: Text("Sign Up",style: TextStyle(fontSize: 18.0),),),),Row(

mainAxisAlignment: MainAxisAlignment.center,children: [Text("Already having
account?"),TextButton(onPressed: () Navigator.push(context, MaterialPageRoute
(builder: (context) => AppSignInScreen()),),),child: Text("SignIn"),),],),),),)},

```

## Authmanagement.dart:

```
import 'package:firebase_auth/firebase_auth.dart';

import 'package:get/get.dart';

import 'package:login_signup/pages/home.dart';

class AuthManage {

  final FirebaseAuth _auth = FirebaseAuth.instance;

  Future<String?> signup(String useremailSU, String userpassSU) async {

    try {

      await _auth.createUserWithEmailAndPassword(

        email: useremailSU, password: userpassSU);

      print("sign up complete");

      Get.offAll(Home());

      return null; // No error

    } catch (e) {

      print("-----error occur in sign up-----");

      print(e);

      return e.toString(); // Return error message

    }

  }

  Future<String?> login(String useremailL, String userpassL) async {

    try {

      await _auth.signInWithEmailAndPassword(

        email: useremailL, password: userpassL);

      print("sign in or login is complete");
```

```

    Get.offAll(Home());

    return null; // No error

  } catch (e) {

    print("-----error occur in sign in -----");

    print(e);

    return e.toString(); // Return error message

  }}}

```

### **Home.dart:**

```

import 'package:flutter/material.dart';

import 'package:get/get.dart';

import 'package:login_signup/pages/nav_bar.dart';

class Home extends StatefulWidget {

  const Home({super.key});

  @override

  State<Home> createState() => _HomeState();

}

class _HomeState extends State<Home> {

  @override

  Widget build(BuildContext context) {

    return MaterialApp(

      home: Scaffold(body: NavBar()),

    );

  }}

```

## Nav\_bar.dart:

```
import 'package:flutter/material.dart';

import 'package:login_signup/pages/about.dart';

import 'package:login_signup/pages/chatbot_search.dart';

import 'package:login_signup/pages/search_box.dart';

import 'package:login_signup/screens/sign_in.dart';

class NavBar extends StatefulWidget {

  const NavBar({super.key});

  @override

  State<NavBar> createState() => _NavBarState();

}

class _NavBarState extends State<NavBar> {

  int _selectedIndex = 0;

  final List<Widget> _tabs = [

    const Padding(padding: EdgeInsets.all(16.0),child: ChatGPTTextField(), ),

    const Padding(padding: EdgeInsets.all(16.0),

      child: ChatBotSearch(), ), ],

  void _onItemTapped(int index) {

    setState(() {

      _selectedIndex = index;

    });

  }

  @override

  Widget build(BuildContext context) {
```

```

    return Scaffold(resizeToAvoidBottomInset: false,backgroundColor: const
Color.fromARGB(255, 207, 222, 229),appBar: AppBar(title: Padding(padding: const
EdgeInsets.only(left: 100),child: Text('M E D V'), ),backgroundColor: const
Color.fromARGB(255, 0, 139, 253),foregroundColor: Colors.white,elevation: 0),

    drawer: Drawer(child: Container(color: Color.fromARGB(255, 101, 170, 226),

    child: ListView(children: [DrawerHeader(child: Center(child: Text("M E D V", style:
TextStyle(color: Colors.white,fontSize: 50,fontWeight: FontWeight.bold,)),),),ListTile(

leading: Icon(Icons.home,color: const Color.fromARGB(255, 236, 218, 218)),title:
Text( "Home",style: TextStyle(fontSize: 20, color: Colors.white)),, onTap: () {

Navigator.push(context, MaterialPageRoute(builder: (context) {return NavBar();

}),;)),ListTile(leading: Icon(Icons.logout_outlined,color: const Color.fromARGB(255,
236, 218, 218)),,title: Text("Logout",style: TextStyle(fontSize: 20, color: Colors.white),
),onTap: () {Navigator.push( context,MaterialPageRoute(builder: (context) {

return AppSignInScreen(); })),;)),ListTile(leading: Icon(Icons.help_rounded,

color: const Color.fromARGB(255, 236, 218, 218), ),title: Text( "About",style:
TextStyle(fontSize: 20, color: Colors.white)),onTap: () { Navigator.push(context,

MaterialPageRoute(builder: (context) {

    return AboutApp();

}), );}),),), ),

    body: _tabs[_selectedIndex], bottomNavigationBar: BottomNavigationBar(

    backgroundColor: const Color.fromARGB(255, 0, 139, 253),currentIndex:
_selectedIndex,onTap: _onItemTapped,elevation: 0,selectedItemColor: const
Color.fromARGB(255, 207, 222,229),

// Change the color of the selected item icon and label

selectedLabelStyle: const TextStyle(color: Color.fromARGB(255, 207, 222,229)),

items: const [BottomNavigationBarItem(icon: Icon(Icons.ondemand_video_rounded,

```

```

color: Colors.white,),label: "Video",),BottomNavigationBarItem(icon: Icon(Icons.chat,
color: Colors.white,),label: "Chat Bot",),]),),);

}}

```

### **Search\_box.dart :**

```

import 'package:firebase_core/firebase_core.dart';

import 'package:firebase_database/firebase_database.dart';

import 'package:firebase_storage/firebase_storage.dart';

import 'package:flutter/material.dart';

import 'package:flick_video_player/flick_video_player.dart';

import 'package:video_player/video_player.dart';

import 'package:login_signup/firebase_options.dart';

class ChatGPTTextField extends StatefulWidget {

  const ChatGPTTextField({super.key});

  @override

  _ChatGPTTextFieldState createState() => _ChatGPTTextFieldState();

}

class _ChatGPTTextFieldState extends State<ChatGPTTextField> {

  late FlickManager flickManager;

  bool _isVideoVisible = false;bool _isFetchingVideo = false;

  final textController = TextEditingController();String urlIdel = "";String urlOfVideo = "";

  final _formKey = GlobalKey<FormState>();

  @override

  void initState() {

```



```

    super.initState(); deleteVideoInFirebaseStorage();

  }

  @override

  void dispose() {

    flickManager.dispose(); super.dispose();

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold( resizeToAvoidBottomInset: false, backgroundColor: const
Color.fromARGB(255, 207, 222, 229),body: Formkey: _formKey, // Set form key
child: Column(children: [ Container(decoration: BoxDecoration(color:
Colors.grey[100],borderRadius: BorderRadius.circular(20.0)),),child: Row(children:
[Expanded(child: Padding(padding: EdgeInsets.symmetric(horizontal: 16.0),

child: TextFormField( controller: textController,validator: (value) {

if (value == null || value.isEmpty) {

    return 'Please enter some text';

  } return null;

},decoration: InputDecoration(hintText: 'Type to Search....', border:
InputBorder.none,),),),), IconButton( onPressed: () async {

if (_formKey.currentState!.validate()) {

  try {

    final databaseReference =FirebaseDatabase.instance.ref("Video/");

    await databaseReference.child("SendQueryFromFlutter").set({'Question':
textController.text});

    fetchUrlOfVideo();

```

```

    } catch (e) { } }

    ),icon: const Icon(Icons.send),color: const Color.fromARGB(255, 26, 136, 226),
  ),],),),const SizedBox(height: 100,),
if (_isFetchingVideo)
  Column(children: [ const SizedBox(height: 10),const CircularProgressIndicator(),
    const Text('AI Preparing Video...'),],)
else if (_isVideoVisible)
  Container(height: 300,width: MediaQuery.of(context).size.width,
    child: FlickVideoPlayer(flickManager: flickManager,), ),],),),);}

// delete video and store user query in firebase realtime db
deleteVideoInFirebaseStorage() async {
  setState(() {
    _isVideoVisible = false;
  });
  try {
    final ref = FirebaseStorage.instance.ref("test/").child("fish.mp4");
    urlDel = (await ref.getDownloadURL()).toString();
    FirebaseStorage.instance.refFromURL(urlDel).delete();
  } catch (e) {
    print(e);
  } }

// check file exists or not

```

```

Future<bool> checkFileisExits() async {

  await Future.delayed(Duration(seconds: 60));

  try {

    final ref = FirebaseStorage.instance.ref("test/");

    await ref.child("fish.mp4").getDownloadURL(); return true;

  } catch (e) { fetchUrlOfVideo();

    return false;

  } }

fetchUrlOfVideo() async {

  setState(() {

    _isVideoVisible = false; _isFetchingVideo = true;

  });

  bool fileExists = await checkFileisExits();

  if (fileExists == true) {

    try {

      final ref = FirebaseStorage.instance.ref("test/").child("fish.mp4");

      urlOfVideo = await ref.getDownloadURL();

      initializeVideo();

    } catch (e) {print(e); }

  }

  // display video

  Future<void> initializeVideo() async {

```

```

setState(() { _isFetchingVideo = true;});

try {

  final videoPlayerController = VideoPlayerController.network(urlOfVideo);

  flickManager = FlickManager(videoPlayerController: videoPlayerController,

  );} catch (e) {  print(e);  }

setState(() {

  _isVideoVisible = true;_isFetchingVideo = false;

  } ); }}

```

### **Chat\_bot\_search.dart:**

```

import 'package:firebase_database/firebase_database.dart';

import 'package:flutter/material.dart';

class ChatBotSearch extends StatefulWidget {

  const ChatBotSearch({super.key});

  @override

  State<ChatBotSearch> createState() => _ChatBotSearchState();

}

class _ChatBotSearchState extends State<ChatBotSearch> {

  String answer = "";  bool isLoading = false;

  final databaseReference = FirebaseDatabase.instance.ref("ChatBot");

  final databaseReferenceFetch = FirebaseDatabase.instance

  .ref("ChatBot").child("receiveTextFromPythonForChatbot/QueryAnswer");

  @override

  void initState() {

```

```

    super.initState(); answer = "";
  }

  @override
  Widget build(BuildContext context) {
    final textController = TextEditingController();

    return Column(crossAxisAlignment: CrossAxisAlignment.start, children: [
      Container(decoration: BoxDecoration(color: Colors.grey[100],
        borderRadius: BorderRadius.circular(20.0)), child: Row(children: [Expanded(
        child: Padding(padding: EdgeInsets.symmetric(horizontal: 16.0), child: TextField(
        controller: textController, decoration: InputDecoration(hintText: 'Clear your
        Doubts...', border: InputBorder.none, )),),),),
      IconButton(onPressed: () async {
        setState(() {isLoading = true;}); try {await databaseReferenceFetch.set("");
        await databaseReference.child("sendTextFromFlutterForChatBot")
        .set({'TextQuery': textController.text});
      } catch (e) {
        print("Error sending text: $e");}
      try {
        databaseReferenceFetch.onValue.listen((event) {
          answer = event.snapshot.value.toString();
          setState(() {
            if (answer != "") {
              isLoading = false;

```

```

    });}); } catch (e) {

    print("Error receiving text: $e");} },icon: Icon(Icons.send),color: const
    Color.fromARGB(255, 26, 136, 226),

    ),], ), ),SafeArea(child: const SizedBox( height: 20, ),),Expanded(child: isLoading?
    Center(child: Column(children: const [CircularProgressIndicator(),SizedBox(height:
    10), Text("Fetching Data..."),], ),): Scrollbar(child: SingleChildScrollView(

    child: Padding(padding:const EdgeInsets.only(left: 20, right: 20, top: 20),

    child: Text(answer,textAlign: TextAlign.justify,style: TextStyle( fontSize: 16.0,

    ), ),

    ),),

    ),),

    ],);

    }

    }

```

## **7. SYSTEM TESTING AND MAINTENANCE**

**The various levels of testing are**

1. White Box Testing
2. Black Box Testing
3. Unit Testing
4. Functional Testing
5. Performance Testing
6. Configuring testing
7. Validation Testing

### **1. White Box Testing**

This corresponds to testing the internal workings of your machine learning models. It involves examining the logic and structure of the code implementing the models. For example, test generating the right information based on the user prompt.

### **2. Black Box Testing**

In the context of machine learning, this would involve testing the overall functionality of your model without inspecting its internal structure. You're concerned with whether the model produces the expected output given certain inputs, similar to how you evaluate the functionality of a software application. For example, test the application's user interface to ensure it correctly displays the video.

### **3. Unit testing**

Unit testing in machine learning involves testing individual components or functions of your model in isolation. For example, you might test the accuracy of a specific algorithm or evaluate the performance of a particular preprocessing step.

#### **4. Functional testing**

This aligns with testing the functional aspects of your machine learning model. For example, verify if the model behaves as expected when given different inputs, checking if it accurately predicts the user questions or not.

#### **5. Performance testing**

Performance testing in machine learning involves assessing how well your model performs in terms of responsiveness and stability under various workloads or datasets. You'll evaluate metrics such as accuracy, precision and recall of the model's performance. For example, time of video generate by AI.

#### **6. Configuration testing**

This corresponds to testing how changes in configurations, such as hyperparameters or feature selection methods, affect the performance and behavior of your machine learning model.

#### **7. Verification and validation**

Validation testing in machine learning refers to assessing whether your model meets the requirements and specifications defined for its intended purpose. This includes validating the accuracy and reliability of the model's predictions. For example, authentication.



## **8. SYSTEM IMPLEMENTATION**

### **1) User Authentication:**

This system ensures that only authorized users can access your system. By integrating user authentication mechanisms, such as login screens and secure password handling, we can enhance the security and trustworthiness of application. Users must authenticate themselves before they can interact with the system's features, protecting sensitive data and maintaining user privacy. users can access and interact with the system, enhancing security and trust.

### **2) Text Input Processing:**

This system involves integrating the Chat GPT 4 model into the system to process text inputs provided by users. Instead of traditional text processing techniques, the Chat GPT 4 model is employed to understand prompts or questions entered by users and generate relevant text-based responses. The generated text response may include explanations, descriptions, or relevant information related to the user's query.

### **3) AI-Powered Image Suggestions:**

This system utilizes DALL-E to generate relevant images based on the text input provided by chat gpt 3.5 model. The AI analyzes the text to understand its context and then recommends visually appealing and contextually relevant images that can be incorporated into the generated videos. By assisting users in selecting suitable visuals, this feature enhances the quality and effectiveness of the generated video content.

#### **4) Voice - over Narration:**

This system adds voice-over narration to the generated videos, enhancing clarity and engagement for viewers. By integrating voice-over capabilities, users can include audio commentary or explanations that complement the visual content of the videos. Voice narration improves the accessibility of the content and provides an immersive experience for viewers, making the videos more informative and engaging.

#### **5) Firebase Storage Integration:**

This system uses Firebase Storage to securely store the generated videos. By utilizing Firebase's reliable and scalable cloud storage infrastructure, we can ensure that the videos are safely stored and easily accessible to users across different devices. Firebase Storage provides robust security features and seamless integration with our application, making it an ideal solution for storing large multimedia files like videos.

#### **6) Real-time Database Updates:**

Implement real-time database updates using Firebase Realtime Database or Firestore to track video uploads and metadata changes, providing instant feedback to users.

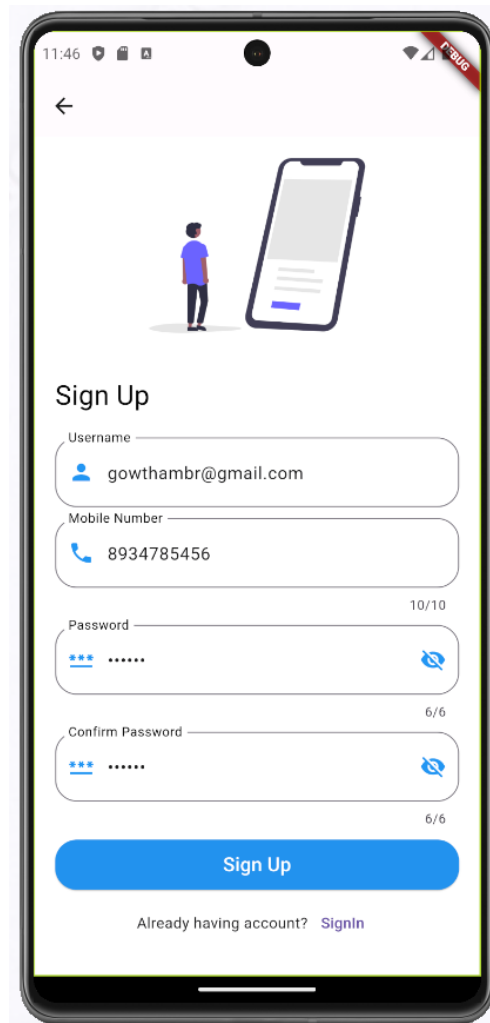
#### **7) User-friendly Interface:**

This system focuses on designing visually appealing interface for our application. A user-friendly interface makes it easy for users to input text prompts, navigate through the app, and access its features. By prioritizing usability and aesthetics, we enhance the overall user experience and increase user engagement with our application

## 9. SCREEN SHOTS

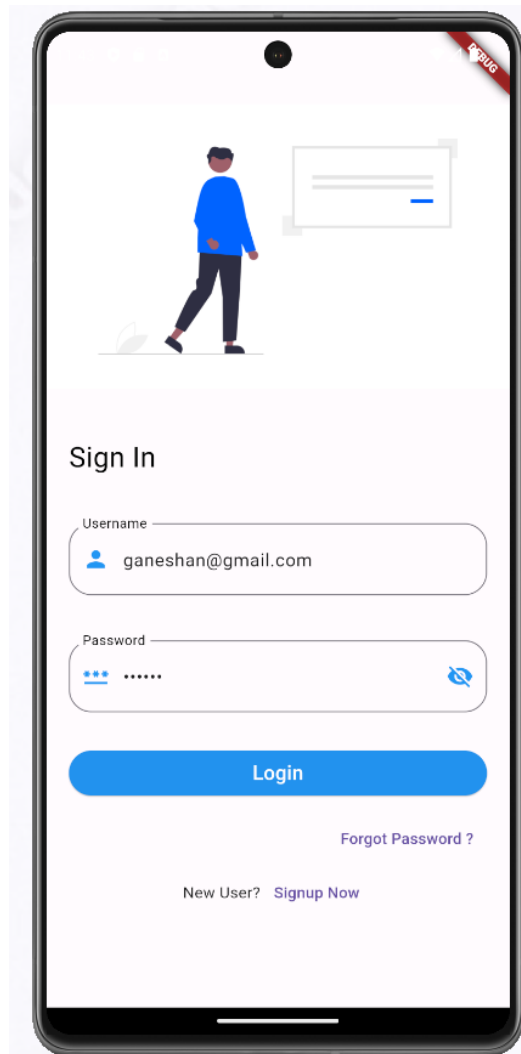
### STEP 1:

Create an account on the signup page.



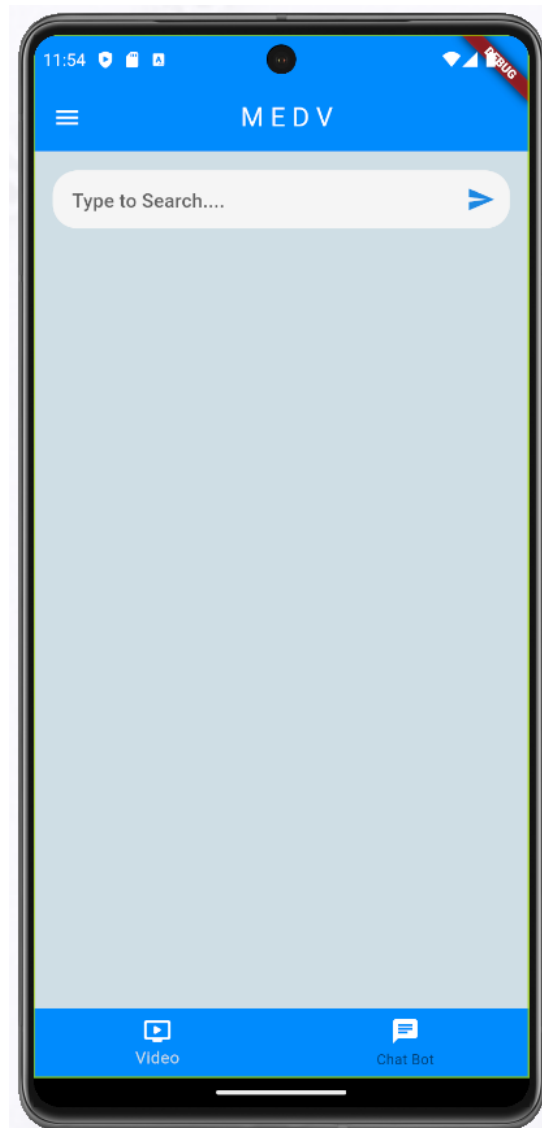
## STEP 2:

If you already have an account, sign in.



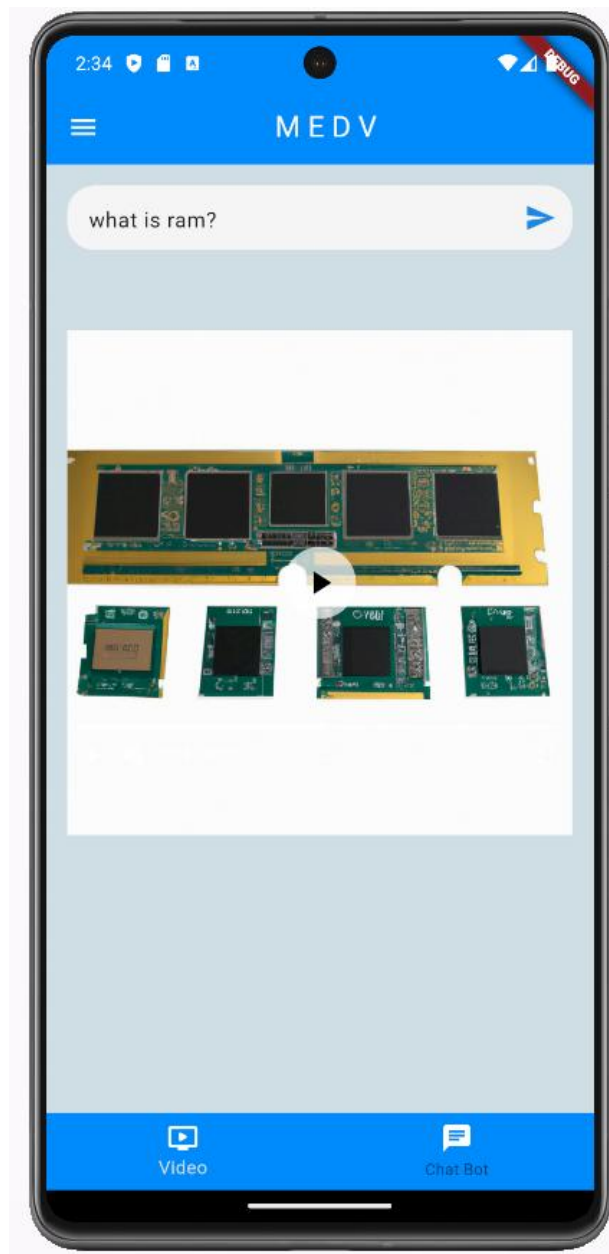
### STEP 3

The Home page will open with two icons in navigation bar : one for generating video and the other for generating text based on the user's query.



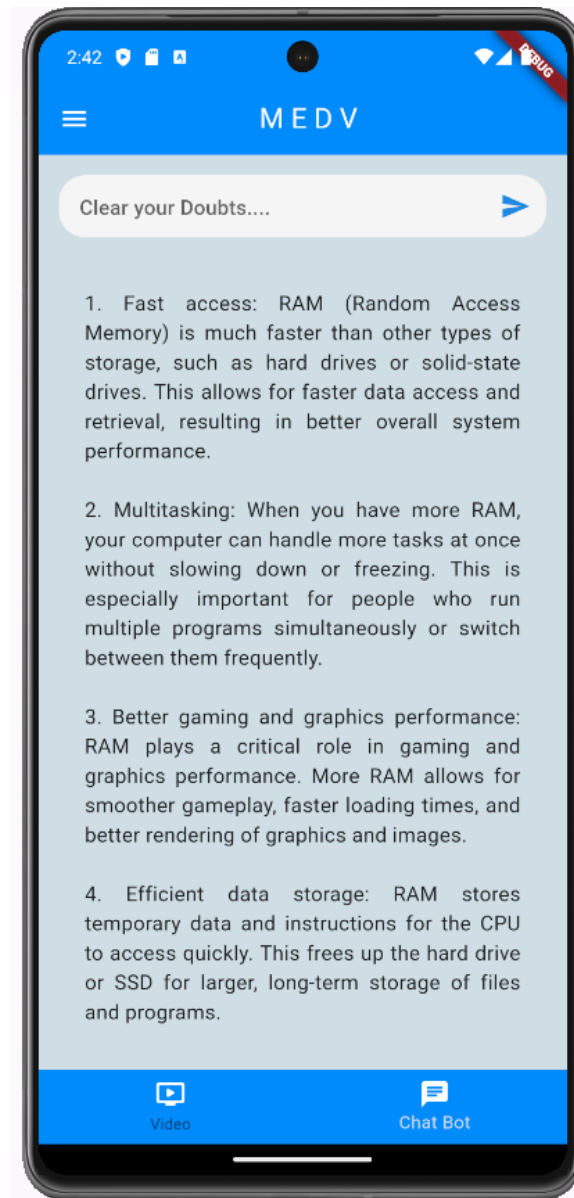
#### STEP 4:

After the user give a prompt in the search bar, the AI will generate the video after a few seconds.



## STEP 5:

Click on the chatbot icon , user's can clarify doubts through a chatbot.



## 10. ESTIMATION OF COST

MATERIAL	COST	TOTAL
API KEY	500	3000
REPORT PREPARATION	200	1000
BROWSING	1000	2000
	<b>TOTAL</b>	6000



## 11. CONCLUSION

- The Macaulay Education Digitalized Video Application aims to improve how students learn. Traditional education systems in India have some flaws, like not covering moral science well and not being very engaging. This new project solves these issues by turning written lessons into fun video lessons. Students can type in text, and the app will turn it into a video they can watch.
- The goal is to make learning enjoyable and accessible for everyone, no matter where they are. By using technology like AI for text and image conversion, the app offers a richer learning experience. Importantly, all video content is generated solely by AI, without human intervention.
- In the future, this project's Chatbot is currently text-based. In the future, this project's Chatbot is currently text-based. However, after the next version, it will display videos and also improve the quality of educational videos..Overall, this project wants to make education better by using technology to make learning more fun and inclusive.

## 12. BIBLIOGRAPHY

### References

- ChatGPT was created by OpenAI. OpenAI was co-founded by Ilya Sutskever, Greg Brockman, John Schulman, and Wojciech Zaremba, with Sam Altman later joining as the CEO. The invention of ChatGPT can be attributed to the team of researchers and engineers at OpenAI, led by Ilya Sutskever and Dario Amodei. The company launched ChatGPT on November 30, 2022.
- David, Emilia (20 September 2023). "OpenAI releases third version of DALL•E". DALL•E, DALL•E 2, and DALL•E 3 are text-to-image models developed by OpenAI using deep learning methodologies to generate digital images.
- Sanket Shah, Pankit Chedda and Harsh Vakharia are the Co-founders of In Video. In Video, the AI-powered video editing platform, was launched on April 10th, 2019. The platform aims to make professional-looking video creation accessible to everyone by leveraging AI technology

## 13. USER MANUAL

Welcome to **MACAULAY EDUCATION DIGITALIZED VIDEO**, here you can know full specification about the system.

These user manual guide you to use the application.

- Open the MEDV application on your mobile.
- Log in with your email ID and password.
- Then, the home page will open.
- The home page contains two icons: one for searching for videos and the other for the chatbot.
- By default, the search for video page is open.
- Users can search for videos through the search bar; after a few minutes, it displays the video. For example, Explain about the ram?
- If users have any doubts, they can clarify them using the chatbot, which provides text-based responses. For example, What are the types of ram?