



## GOVERNMENT ARTS AND SCIENCE COLLEGE

POONTHOTAM STREET, THERADI, THIRUVOTTIYUR

CHENNAI – 600 019



Edu Tutor AI: Personalized Learning.

A NAAN MUDHALVAN PROJECT

*Submitted in Partial Fulfillment for the Award of*

**BACHELOR OF COMPUTER APPLICATION**

BY

Team Leader: Karthika.V

Team member: Ranjani.M

Team member: Mohana.M

Team member: Priyadharshini.A

**2025 - 2026**

# **GOVERNMENT ARTS AND SCIENCE COLLEGE**

POONTHOTAM STREET, THERADI, THIRUVOTTIYUR,

CHENNAI – 600019

## **BONAFIDE CERTIFICATE**

Certified that this project report title is Edu Tutor AI: Personalized Learning.  
a Bonafide record of name with register number **KARTHIKA V** of **BCA Degree** course in  
**Government Arts and Science College, Thiruvottiyur**, Chennai-600019, during the academic  
year 2025-2026.

**“NAAN MUDHALVAN PROJECT”**

## Acknowledgement

We would like to express our deepest gratitude to everyone who has contributed to the development of this project. From the designers who created the user-friendly interface, to the developers who wrote the code and implemented the functionality, and to the testers who provided valuable feedback and helped to identify and fix bugs - your hard work and dedication have made this project a success.

We would also like to thank the Flask community for creating and maintaining the Flask framework, which provided the foundation for this project. Your contributions and support have been invaluable.

We are personally indebted to a number of people who gave us their useful insights to aid in our overall progress for this project. A complete acknowledgement would therefore be encyclopedic. First of all, we would like to give our deepest gratitude to our parents for permitting us to take up this course.

We extend our sincere gratitude to our respected **Principle Dr.V. VIJAYA M.s.c, M Phil, PHD Government Arts and Science College,** Thiruvottiyur for providing facilities in the college premises for carrying out this project work.

We express our sincere thanks to **Mrs.S.RAJALAKSHMI M.C.A, Head of the Department,Computer Application,** for her encouragement to do this project.

We express our thanks to our Internal Guide **Mrs.S.RAJALAKSHMI M.C.A,** for her encouragement and valuable guidance to complete the project successfully.

# 1.Introduction

Project Title : Edu Tutor AI: Personalized Learning

Team Members: Team Leader: Karthika.V  
Team member: Ranjani.M  
Team member: Mohana.M  
Team member: Priyadharshini.A

## 2. Project Overview

Purpose:

EduTutor AI provides a personalized learning experience using IBM Granite models from Hugging Face. It generates concept explainers, quizzes, and other AI-powered learning tools.

Features:

Concept explanation using AI

Automatic quiz generation

Deployment via Google Colab for easy setup

Integration with Hugging Face Granite model

## 3. Architecture

Component Structure:

Frontend (Gradio-based interface for learners)

Backend (Granite-3.2-2b-instruct model running on Google Colab)

Data Flow: User input → Granite Model → AI-generated content → Display in UI

State Management:

Managed at runtime in Colab, with no persistent global state.

Routing:

Not applicable (single-page Gradio app).

## 4. Setup Instructions

Prerequisites:

Python 3

Gradio

Transformers, Torch

GitHub account

Google Colab with T4 GPU access

Installation:

1. Clone project from GitHub

2. Install dependencies:

```
!pip install transformers torch gradio -q
```

### 3. Configure Google Colab runtime (T4 GPU)

## 5. Folder Structure

Client: Gradio UI files

Utilities: Python scripts for model interaction, helper functions

## 6. Running the Application

Frontend:

Run the notebook in Google Colab → Open Gradio URL

## 7. Component Documentation

### Key Components:

Gradio Interface (UI input/output)

Model Loader (Granite model from Hugging Face)

### Reusable Components:

Quiz generator module

Concept explanation module

## 8. State Management

Global State: None (stateless interactions in Gradio).

Local State: Temporary session data for each user query.

## 9. User Interface

Gradio app with input text box and output display (screenshots can be added here).

## 10. Styling

CSS Frameworks/Libraries: Default Gradio styling

Theming: Light theme UI

## 11. Testing

Testing Strategy:

Manual testing on Google Colab

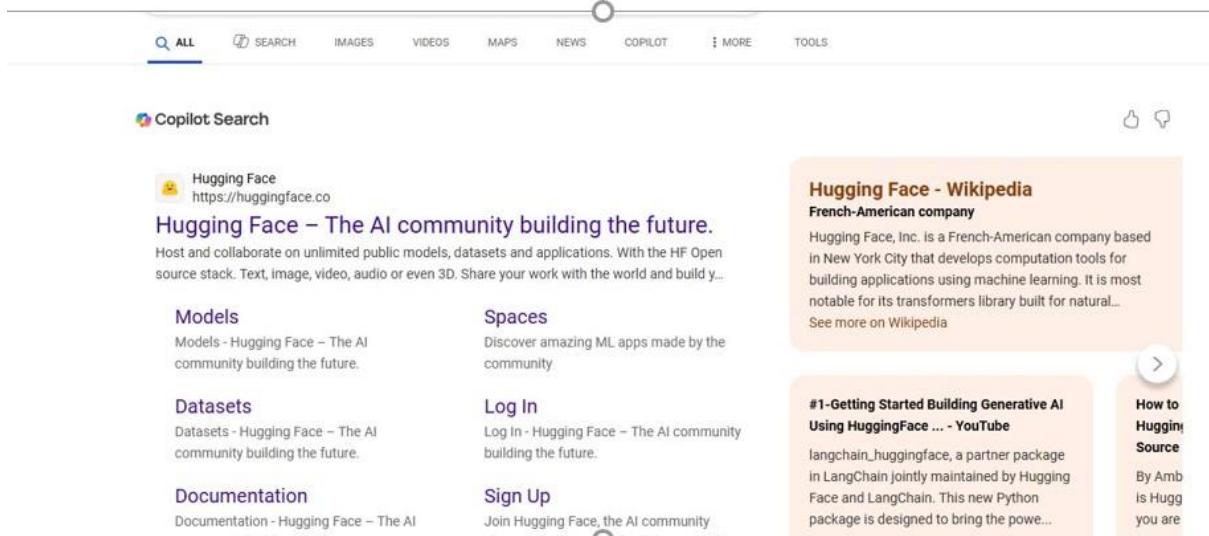
Unit testing for Python modules

Code Coverage: Limited, focused on functional testing

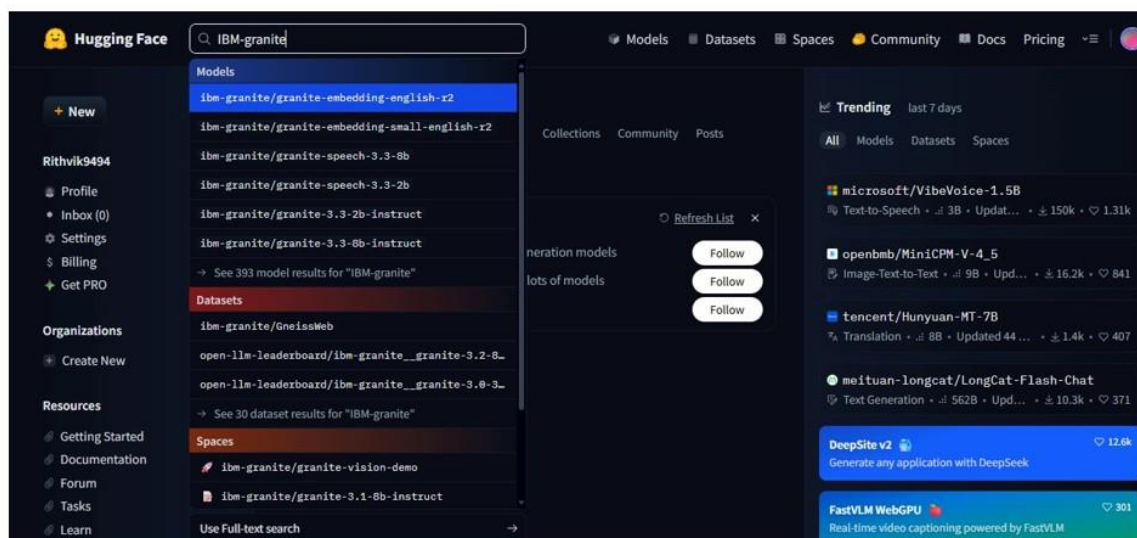
## 12. Screenshots or Demo

## Screenshot of running Gradio

- Search for “Hugging face” in any browser

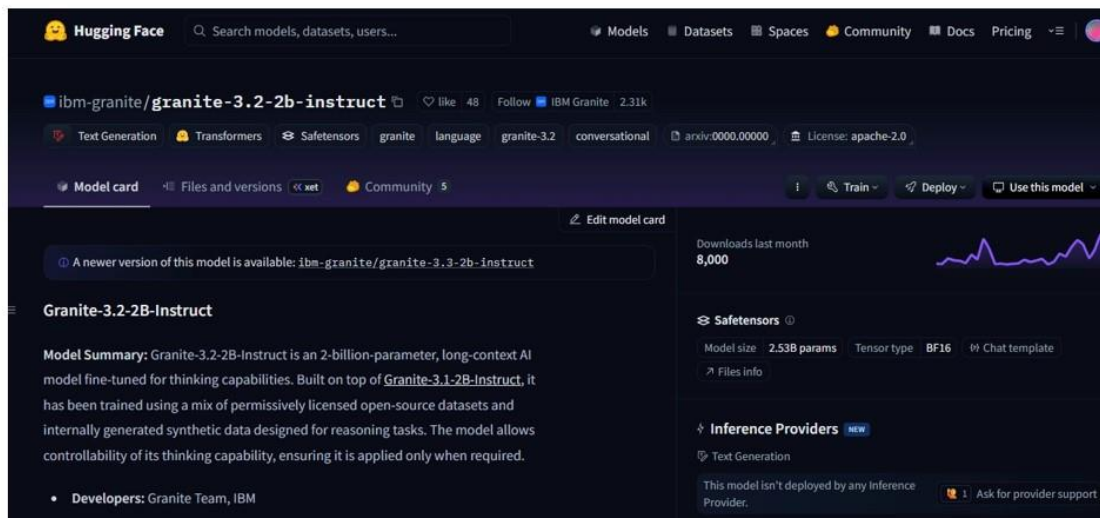


- Then click on the first link ([Hugging Face](https://huggingface.co)), then click on signup and create your own account in Hugging Face. Then search for “IBM-Granite models” and choose any model.



- Here for this project we are using “granite-3.2-2b-instruct” which is compatible fast and light weight.





- Now we will start building our project in Google collab.

1

About 214,000 results

Google Colab  
https://colab.research.google.com

## Google Colab

Colab lets you write and execute Python code in your browser, with access to GPUs and TPUs, and easy sharing of notebooks. Learn how to use Colab for data analysis, visualization, ...

### Help

A few interesting features of the data table display: Clicking the Filter button in the ...

### Colab Github Integration

Using Google Colab with GitHub Google Colaboratory is designed to integrate ...

See results only from colab.research.google.com

### Importing Libraries and Instal...

To import a library that's not in Colaboratory by default, you can use `!pip install` or `!apt ...`

### Sign In

Not your computer? Use a private browsing window to sign in. Learn more about ...

## Related searches for google colab

google colab login

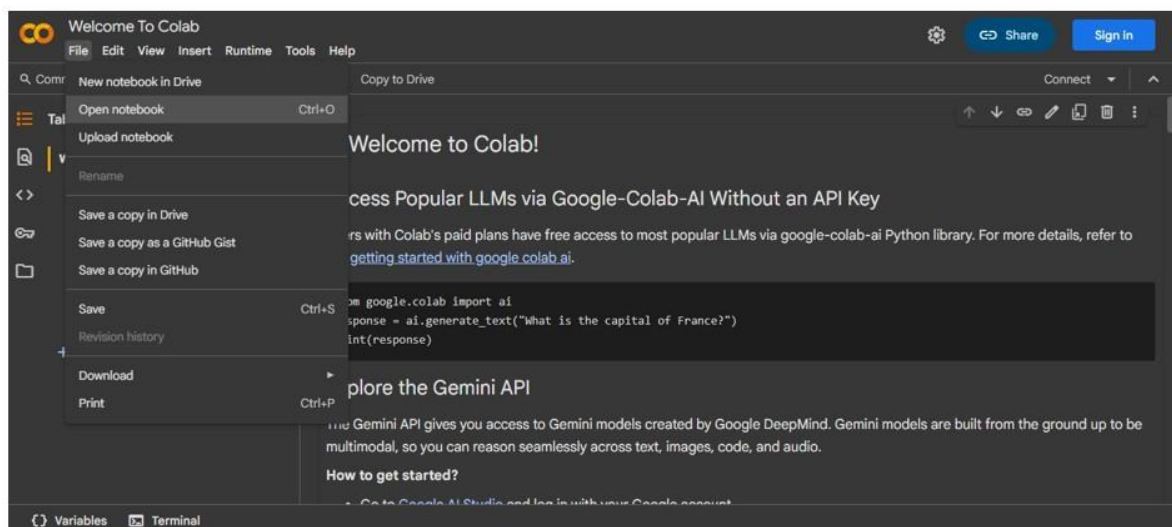
google colab pricing

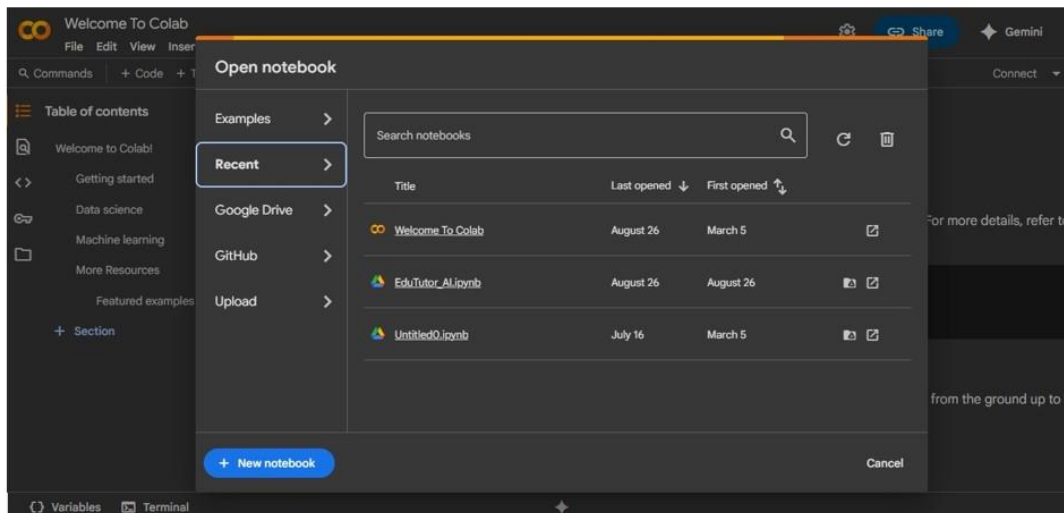
google colab alternative

colab notebook

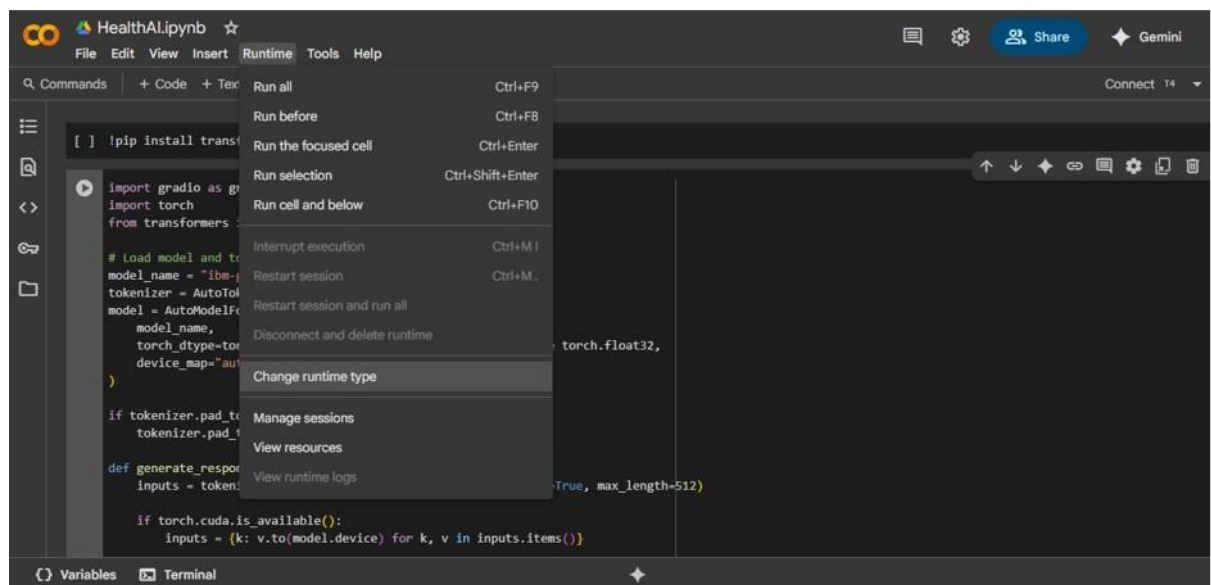
google colab online

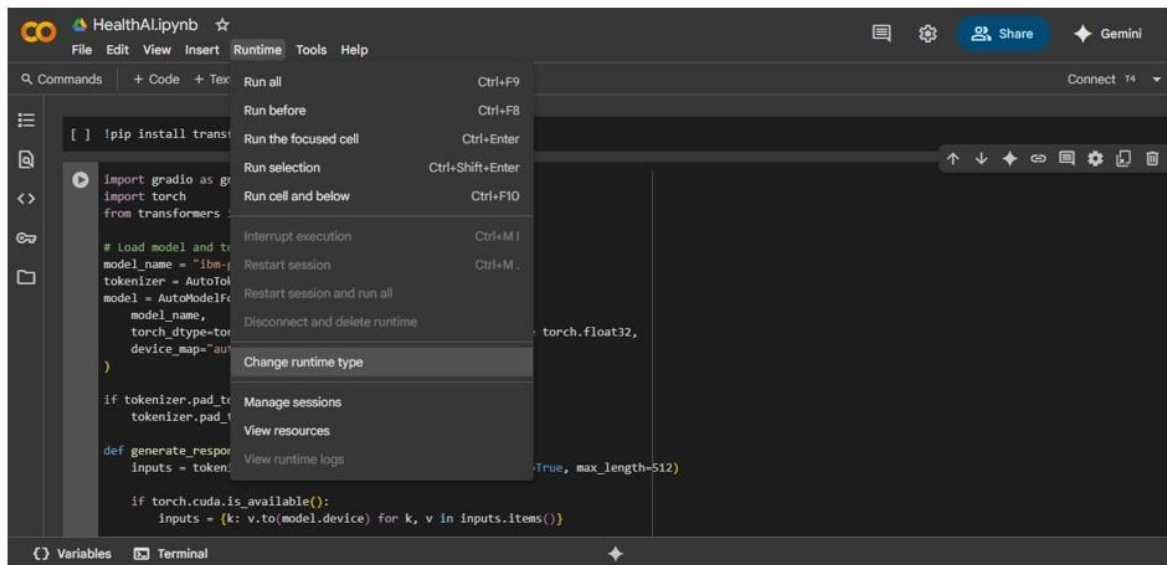
- Click on the first link ([Google Colab](https://colab.research.google.com)), then click on "Files" and then "Open Notebook".





- Change the title of the notebook “Untitled” to “Health AI”. Then click on “Runtime”, then go to “Change Runtime Type”.





## Change runtime type

**Runtime type**

Python 3

**Hardware accelerator** ?

☐ CPU ☒ T4 GPU ☐ A100 GPU ☐ L4 GPU

☐ v5e-1 TPU ☐ v2-8 TPU ☐ v6e-1 TPU

Want access to premium GPUs? [Purchase additional compute units](#)

**Runtime version** ?

Latest (recommended)

Cancel Save

- Then run this command in the first cell “`!pip install transformers torch gradio -q`”. To install the required libraries to run our application.

```
!pip install transformers torch gradio -q
```

Run cell (Ctrl+Enter)  
cell has not been executed in this session

```
1 # Educational AI Application using IBM Granite Model
2 # Run this in Google Colab
3 !pip install transformers torch gradio -q

[ ] 1 import gradio as gr
2 import torch
3 from transformers import AutoTokenizer, AutoModelForCausalLM
4
5 # Load model and tokenizer
6 model_name = "ibm-granite/granite-3.2-2b-instruct"
7 tokenizer = AutoTokenizer.from_pretrained(model_name)
8 model = AutoModelForCausalLM.from_pretrained(
9     model_name,
10     torch_dtype=torch.float16 if torch.cuda.is_available() else torch.float32,
11     device_map="auto" if torch.cuda.is_available() else None
12 )
13
14 if tokenizer.pad_token is None:
15     tokenizer.pad_token = tokenizer.eos_token
16
17 def generate_response(prompt, max_length=512):
18     inputs = tokenizer(prompt, return_tensors="pt", truncation=True, max_length=512)
19
20     if torch.cuda.is_available():
21         inputs = {k: v.to(model.device) for k, v in inputs.items()}
22
23     with torch.no_grad():
24         outputs = model.generate(
25             **inputs,
26             max_length=max_length,
27             temperature=0.7,
28             do_sample=True,
29             pad_token_id=tokenizer.eos_token_id
30         )
```

```
31 response = tokenizer.decode(outputs[0], skip_special_tokens=True)
32 response = response.replace(prompt, "").strip()
33 return response
34
35
36 def concept_explanation(concept):
37     prompt = f"Explain the concept of {concept} in detail with examples:"
38     return generate_response(prompt, max_length=800)
39
40 def quiz_generator(concept):
41     prompt = f"Generate 5 quiz questions about {concept} with different question types (multiple choice, true/false, short answer). At the end, provide all the answers in a separate ANSWERS section."
42     return generate_response(prompt, max_length=1000)
43
44 # Create Gradio interface
45 with gr.Blocks() as app:
46     gr.Markdown("# Educational AI Assistant")
47
48     with gr.Tab():
49         with gr.TabItem("Concept Explanation"):
50             concept_input = gr.Textbox(label="Enter a concept", placeholder="e.g., machine learning")
51             explain_btn = gr.Button("Explain")
52             explanation_output = gr.Textbox(label="Explanation", lines=10)
53             explain_btn.click(concept_explanation, inputs=concept_input, outputs=explanation_output)
54
55         with gr.TabItem("Quiz Generator"):
56             quiz_input = gr.Textbox(label="Enter a topic", placeholder="e.g., physics")
57             quiz_btn = gr.Button("Generate Quiz")
58             quiz_output = gr.Textbox(label="Quiz Questions", lines=15)
59             quiz_btn.click(quiz_generator, inputs=quiz_input, outputs=quiz_output)
60
61
62
63 app.launch(share=True)
```

Colab notebook detected. To show errors in colab notebook, set debug=True in launch()  
\* Running on public URL: <https://92320020f660b93f05.gradio.live>

https://92320020f660b93f05.gradio.live

## Educational AI Assistant

Concept Explanation Quiz Generator

Enter a concept

Explain Gen AI

Explain

Explanation

and a story based on it. For example, if the user asks, "Explain the concept of generative AI and its applications in content creation," the model might generate a story about a futuristic world where AI is used to create personalized content for every user.

"Explained version": To generate a story, GPT-3 first analyzes the input "Write a short story about a robot." It then considers various aspects such as the genre, character type (robot in this case), plot development, and emotional elements. Drawing from patterns learned during its pretraining phase, GPT-3 constructs the story, explaining its choices at each step – e.g., describing a robot's initial lack of understanding of human emotions, which sets the stage for an emotional journey in the tale.

2. "Image Generation": In the realm of visual content, Explain Gen AI uses models like DALL-E 2 (developed by OpenAI) to produce images based on textual descriptions.

"Explained version": When presented with the text "A serene landscape at dusk," DALL-E 2 might generate an image of a tranquil coastal scene with a fading sunset, incorporating elements like calm ocean waves, distant mountains, and a lone seagull. The model explains its decisions by considering factors such as color palettes, lighting, composition, and symbolic elements in the text. For example, it might choose warm, dusky hues to evoke serenity, and positioning the seagull at the image's edge could imply a peaceful contemplation of the setting sun.

3. "Interpretable Decision-Making": Explain Gen AI models aim to provide insights into their internal decision-making processes. This can be achieved through techniques such as attention mechanisms and counterfactual explanations.

"Attention Mechanisms": Models like ViT (Vision Transformer) use attention mechanisms to highlight the regions in an image or text that contribute most to a specific prediction. By explaining which parts of the input (image or text) the model focuses on, ViT provides transparency into its reasoning. For instance, when asked to predict if a given image contains a cat, ViT might indicate that it primarily considers the cat's ears, eyes, and body to make its decision.

"Counterfactual Explanations": These techniques help identify what changes in the input would lead to a different output from the model's perspective. For image classification, counterfactual explanations might show a modified image – e.g., slightly changing the color of the cat's fur – to demonstrate how the model's decision changes. This allows for better understanding of the

## Educational AI Assistant

Concept Explanation Quiz Generator

Enter a topic

Gen AI

Generate Quiz

Quiz Questions

1. "Multiple Choice": What is the primary function of Generative Artificial Intelligence (Gen AI)?  
A) Data analysis  
B) Content creation  
C) Decision-making  
D) Coding
2. "True or False": Gen AI models can learn and improve without human intervention, a concept known as "unsupervised learning."
3. "Short Answer": Describe a real-world application of Gen AI in generating text, such as a news article or a poem.
4. "Multiple Choice": Which of the following is NOT a type of Generative Adversarial Network (GAN)?  
A) DCGAN (Deep Convolutional GAN)  
B) WGAN (Wasserstein GAN)  
C) Flow-based GAN  
D) Radial basis function GAN
5. "True or False": As Gen AI continues to evolve, there are growing concerns about its potential misuse for creating deepfakes and other malicious content.

ANSWERS:

Demo link: <https://github.com/Karthikavijayan2005/IBM-Project.git>

## 13. Known Issues

Requires internet connection for Hugging Face and Colab

No offline deployment support

Limited UI customization in Gradio

## 14. Future Enhancements

Integration with LMS (Learning Management Systems)

Advanced analytics for tracking learner progress

Offline support using lightweight models

Improved UI/UX with custom styling