Project Title

Edu Tutor AI: Personalized Learning

Project Overview

Team Leader : GOWRI SANKAR C

Team member : RAMSUBIRAMANI K

Team member : SAKTHI V

Team member : SANJAY S

Project Overview

Purpose

This project delivers a user-friendly educational assistant that leverages large language models (LLMs) to help learners deepen understanding of academic concepts and practice knowledge using auto-generated quizzes. The system employs IBM's Granite-3.2-2B-Instruct LLM for robust natural language understanding and generation.

Primary Users: Students, educators, self-learners.

Features

Concept Explanation:

Key Point: Interactive academic guidance

Functionality: Accepts topic queries and returns detailed, example-backed explanations.

Quiz Generator:

Key Point: Custom knowledge assessment

Functionality: Produces five quiz questions across diverse formats (multiple choice, true/false, short answer) and provides answers for self-evaluation.

Gradio Interface:

Key Point: Intuitive web-based usage

Functionality: Employs Gradio’s block and tab system for easy navigation, instant feedback, and real-time model interaction.

Architecture

Frontend (Gradio)

The frontend is built using Gradio’s Blocks system for a modular, tabbed layout:

Concept explanation and quiz generation are presented as separate tabs.

Users input concepts or topics using textboxes; buttons trigger the model for output.

Outputs are displayed as formatted text in expandable boxes for ease of reading.

Backend (Transformers + PyTorch)

The backend loads IBM’s Granite-3.2-2B-Instruct model using the HuggingFace Transformers library.

Model loading is device-aware, supporting GPU acceleration for performance.

Core backend functions include text prompt formulation, model inference with defined parameters (temperature, sampling), and post-processing for clarity.

Key Files and Functions

Main Script: Contains model loading, Gradio app definition, and launch commands.

Model Inference Functions:

Generate_response: Generates responses from user prompts.

Concept_explanation: Reformulates user’s topic for detailed explanation.

Quiz_generator:

Creates quizzes and answer sections.

Setup Instructions

Prerequisites

Python 3.9 or above

PyTorch (with CUDA support for GPUs recommended)

Transformers (latest)

Gradio (latest

Sufficient VRAM (8GB+) for LLM inference

Internet access to download models

Installation Process

Install required packages:

Text

```
Pip install torch torchvision torchaudio
Pip install transformers
Pip install gradio
```

Save the provided script.

Optionally, export environment variables for GPU acceleration.

Running the Application

Start the application with:

Python

```
Python <script_name>.py
```

The Gradio interface will launch locally (default: http://localhost:7860) with an optional public “share” link for broader access.

Use the “Concept Explanation” and “Quiz Generator” tabs according to need.

User Interface

Sidebar (auto-generated by Gradio) for navigation.

Tabs: Concept Explanation, Quiz Generator.

Form Controls: Textbox (input), Button (trigger), Textbox (output)

Styling: Minimal, responsive, mobile and desktop friendly.

Testing

Unit Testing: Verify each function handles input and returns non-empty outputs.

Manual Testing: Try a range of topics, edge-cases (empty/complex queries), toggle between CPU/GPU settings.

Performance depends on model size and hardware; expect latency on CPU.

API Documentation

Not exposed as API endpoints in this basic form, but can be modularized into FastAPI/REST for integration as seen in similar architectures.

Authentication

Intended for open, demo usage. Production deployments can layer authentication using Gradio’s integration with web frameworks or external services.

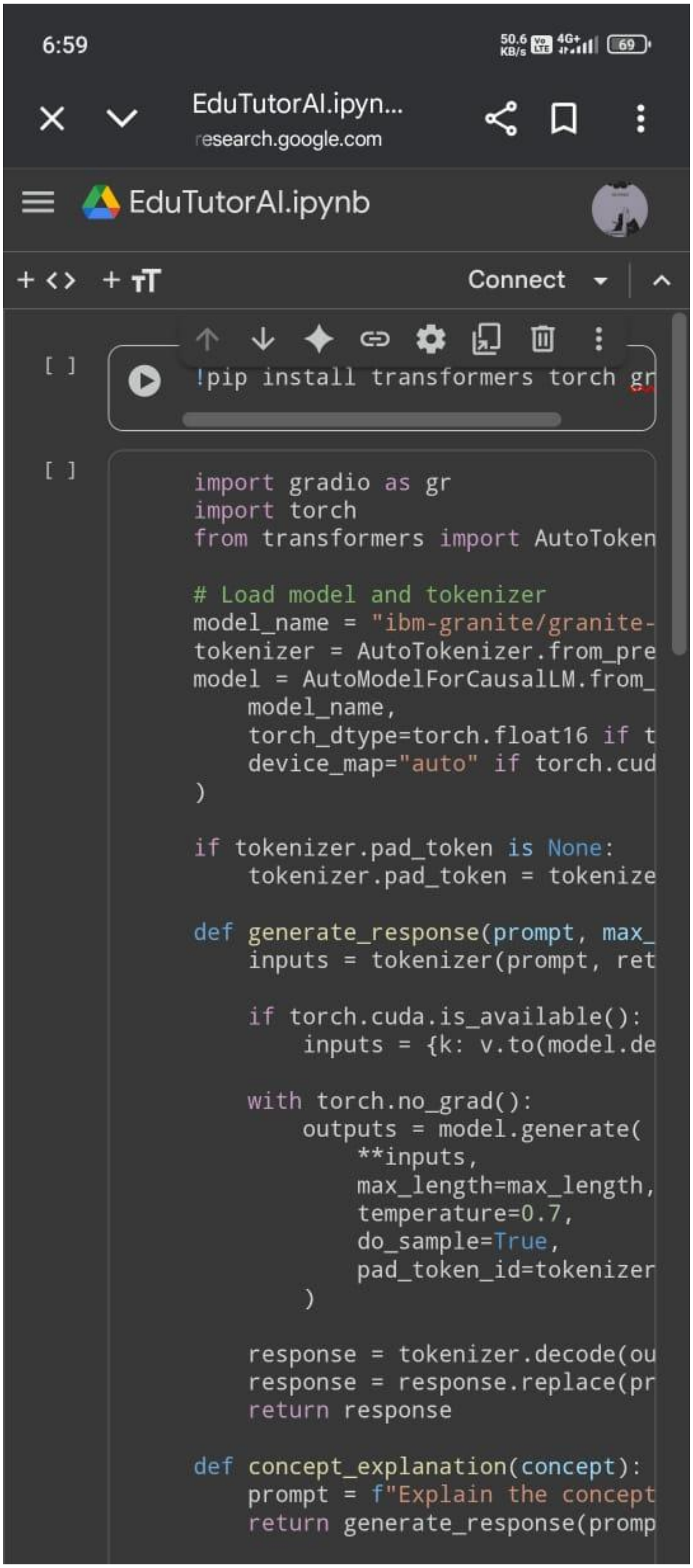
Future Enhancements

Add document upload for context-based explanations.

Integrate role-based access for teacher/student features.

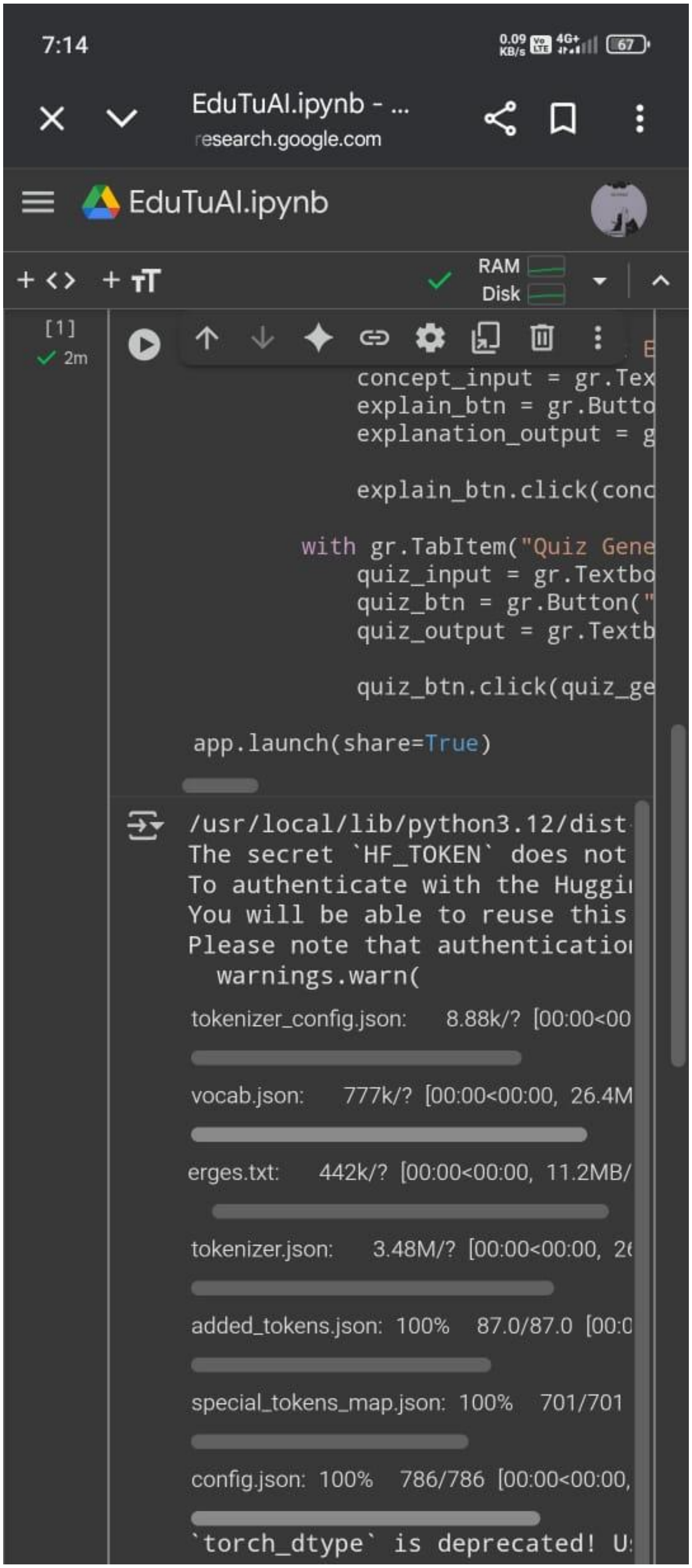
Expand LLM options and offer multi-language support.

Add analytics and progress-tracking features.



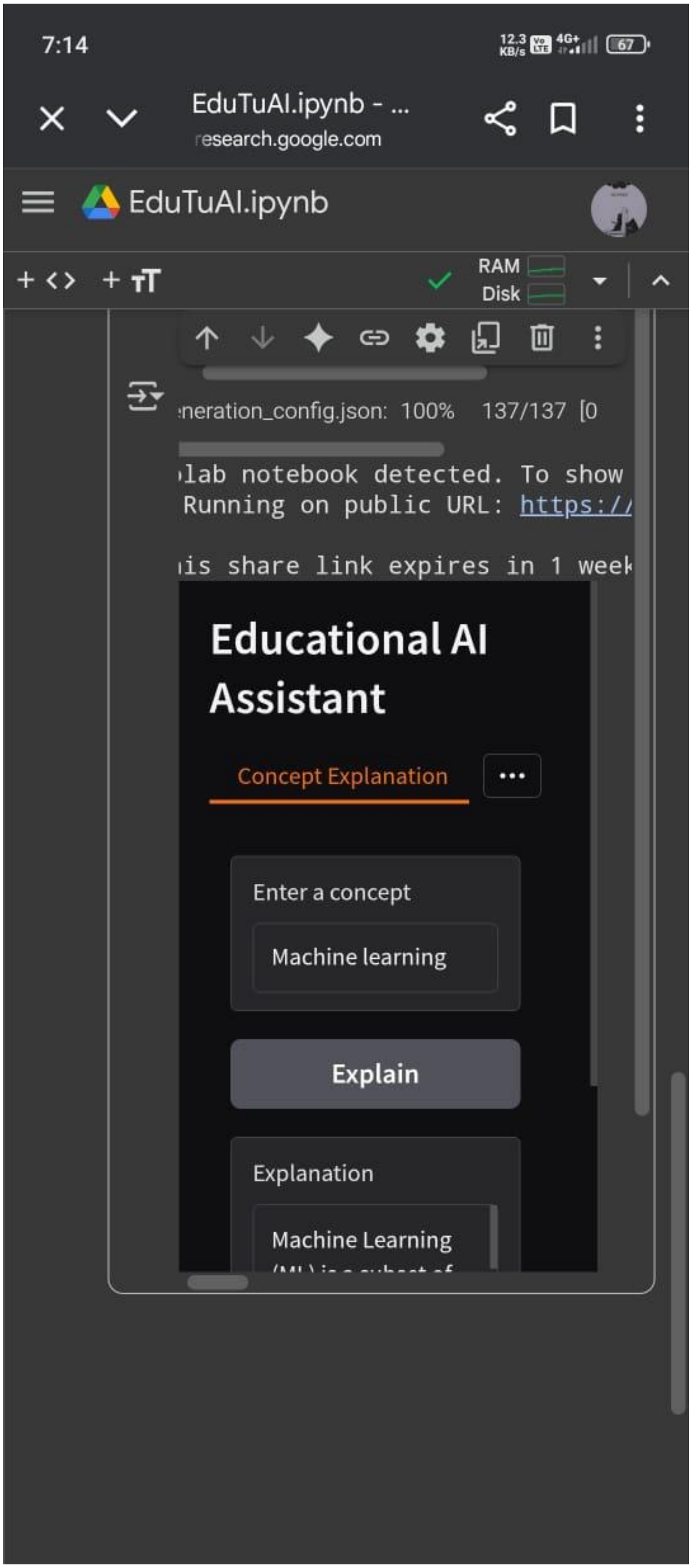
```
!pip install transformers torch gr
```

```
import gradio as gr
import torch
from transformers import AutoToken

# Load model and tokenizer
model_name = "ibm-granite/granite-
tokenizer = AutoTokenizer.from_pre
model = AutoModelForCausalLM.from_
    model_name,
    torch_dtype=torch.float16 if t
    device_map="auto" if torch.cud
)

if tokenizer.pad_token is None:
    tokenizer.pad_token = tokenize

def generate_response(prompt, max_
    inputs = tokenizer(prompt, ret

    if torch.cuda.is_available():
        inputs = {k: v.to(model.de

    with torch.no_grad():
        outputs = model.generate(
            **inputs,
            max_length=max_length,
            temperature=0.7,
            do_sample=True,
            pad_token_id=tokenizer
        )

    response = tokenizer.decode(ou
    response = response.replace(pr
    return response

def concept_explanation(concept):
    prompt = f"Explain the concept
    return generate_response(promp
```



```
            concept_input = gr.Tex
            explain_btn = gr.Butto
            explanation_output = g

            explain_btn.click(conc

        with gr.TabItem("Quiz Gene
            quiz_input = gr.Textbo
            quiz_btn = gr.Button("
            quiz_output = gr.Textb

            quiz_btn.click(quiz_ge

app.launch(share=True)
```

```
/usr/local/lib/python3.12/dist
The secret `HF_TOKEN` does not
To authenticate with the Huggi
You will be able to reuse this
Please note that authenticatio
  warnings.warn(
tokenizer_config.json:  8.88k/? [00:00<00
vocab.json:  777k/? [00:00<00:00, 26.4M
erges.txt:  442k/? [00:00<00:00, 11.2MB/
tokenizer.json:  3.48M/? [00:00<00:00, 2(
added_tokens.json: 100%  87.0/87.0 [00:0
special_tokens_map.json: 100%  701/701
config.json: 100%  786/786 [00:00<00:00,
`torch_dtype` is deprecated! U:
```



```
ıneration_config.json: 100%   137/137 [0
ılab notebook detected. To show
Running on public URL: https://
ıis share link expires in 1 week
```

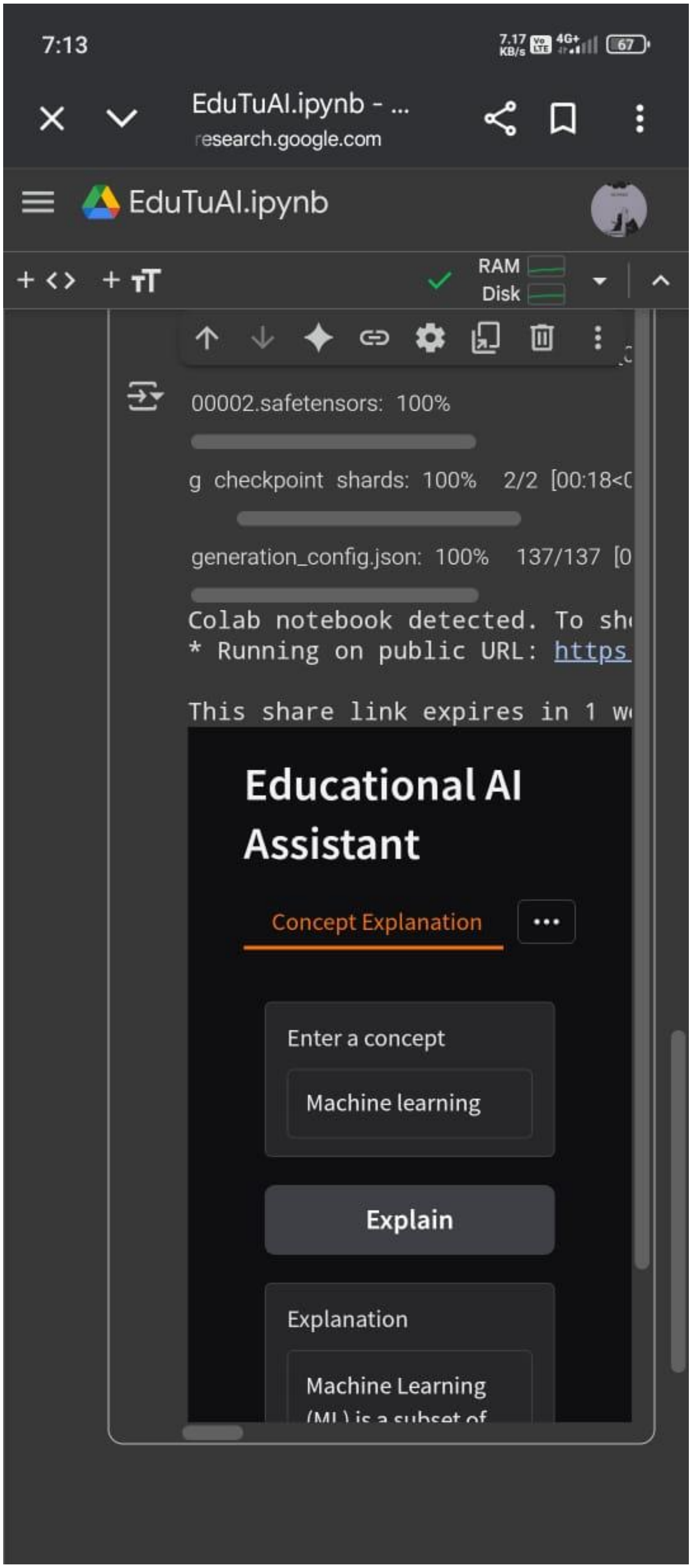
Educational AI Assistant

Enter a concept

Machine learning

Explanation

Machine Learning
(ML) is a subset of



```
00002.safetensors: 100%
g checkpoint shards: 100%   2/2 [00:18<(
generation_config.json: 100%   137/137 [0
Colab notebook detected. To sh
* Running on public URL: https
This share link expires in 1 w
```

Educational AI Assistant

Enter a concept

Machine learning

Explanation

Machine Learning (ML) is a subset of