

Edu Tutor AI: Personalized Learning

Generative AI with IBM



Team Id :

NM2025TMID02806

Team Leader :

A. Vaishnavi

Team Members :

K. Vardhini
J. Jayalakshmi
B.G.Sujith

Email Id :

- vaishnavibca23.krmmc@gmail.com
- kalajayaram472@gmail.com
- kvardhinibca23.krmmc@gmail.com
- sudharsri9080@gmail.com

Project Description:

EduTutor AI uses the Granite model from Hugging Face to create simple, personalized learning tools like concept explainers, quizzes generator and add more functionalities that you like. This project is deployed in Google Colab using Granite for low setup effort and reliable performance.

Pre-requisites:

1. Gradio Framework Knowledge : [Gradio Documentation](#)
2. IBM Granite Models (Hugging Face) : [IBM Granite models](#)
3. Python Programming Proficiency : [Python Documentation](#)
4. Version Control with Git : [Git Documentation](#)
5. Google Collab's T4 GPU Knowledge : [Google collab](#)

Project Workflow:

Activity-1: Exploring Naan Mudhalavan Smart Interz Portal.

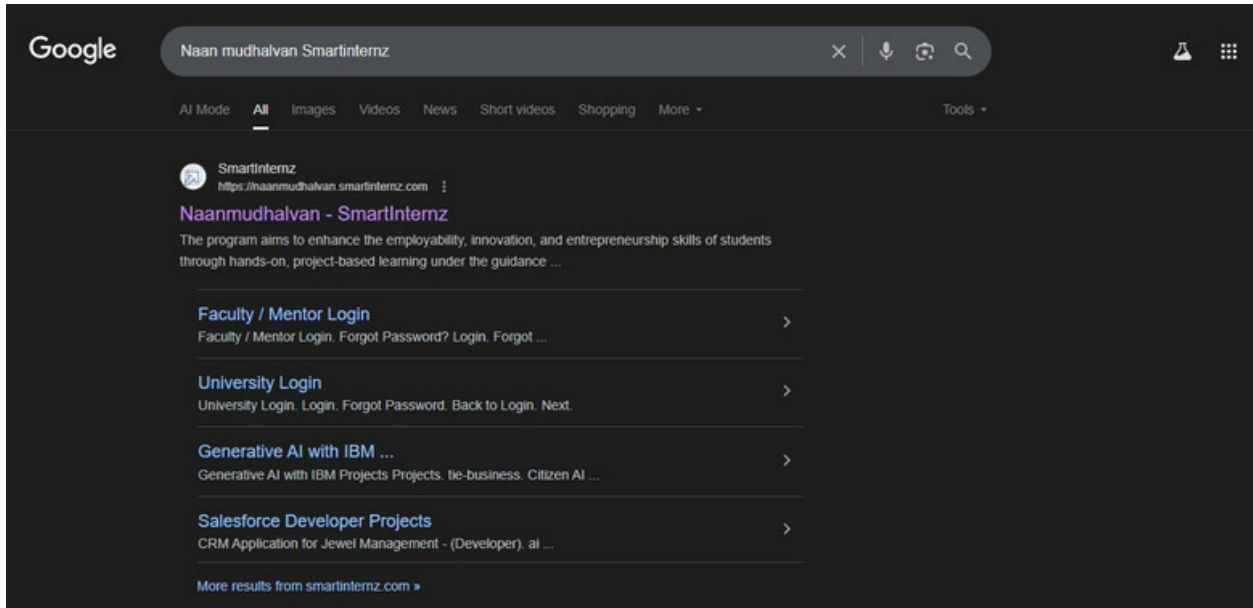
Activity-2: Choosing a IBM Granite Model From Hugging Face.

Activity-3: Running Application In Google Colab.

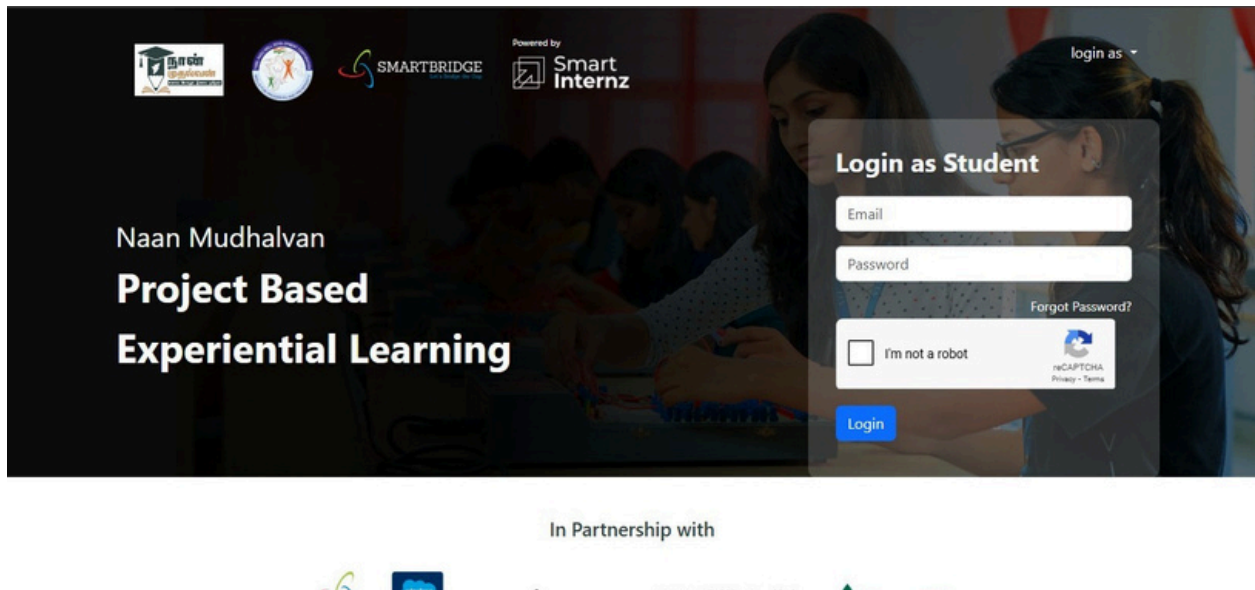
Activity-4: Upload your Project in Github.

Activity-1: Exploring Naan Mudhalavan Smart Interz Portal.

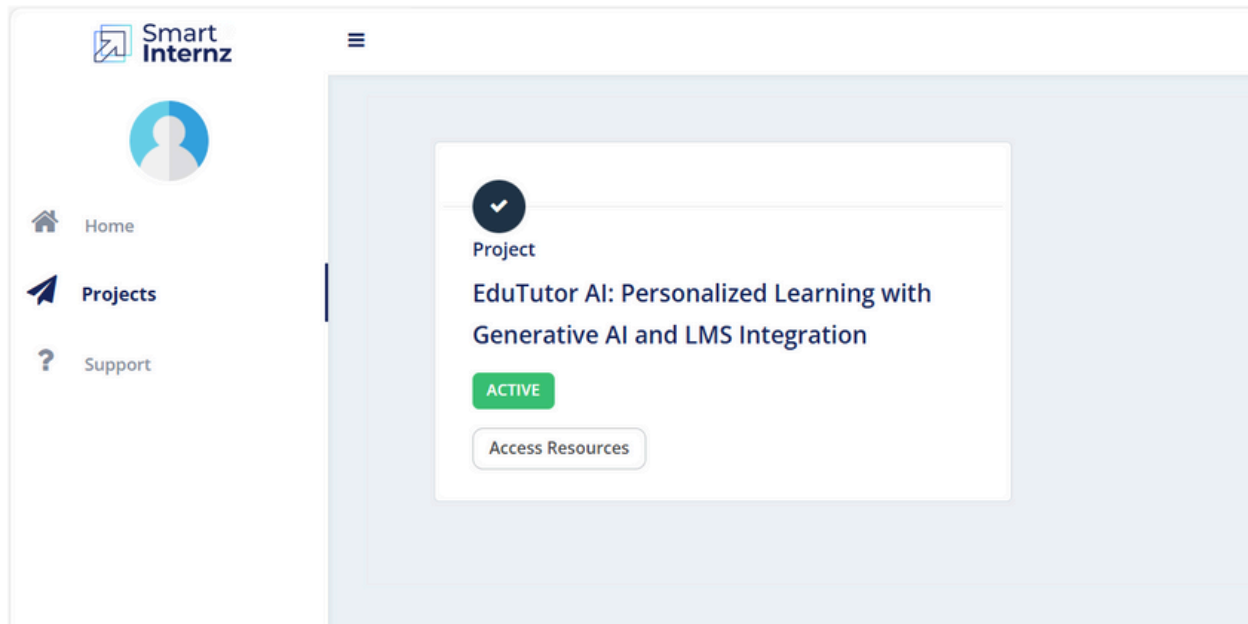
- Search for “Naan Mudhalavan Smart Interz” Portal in any Browser.



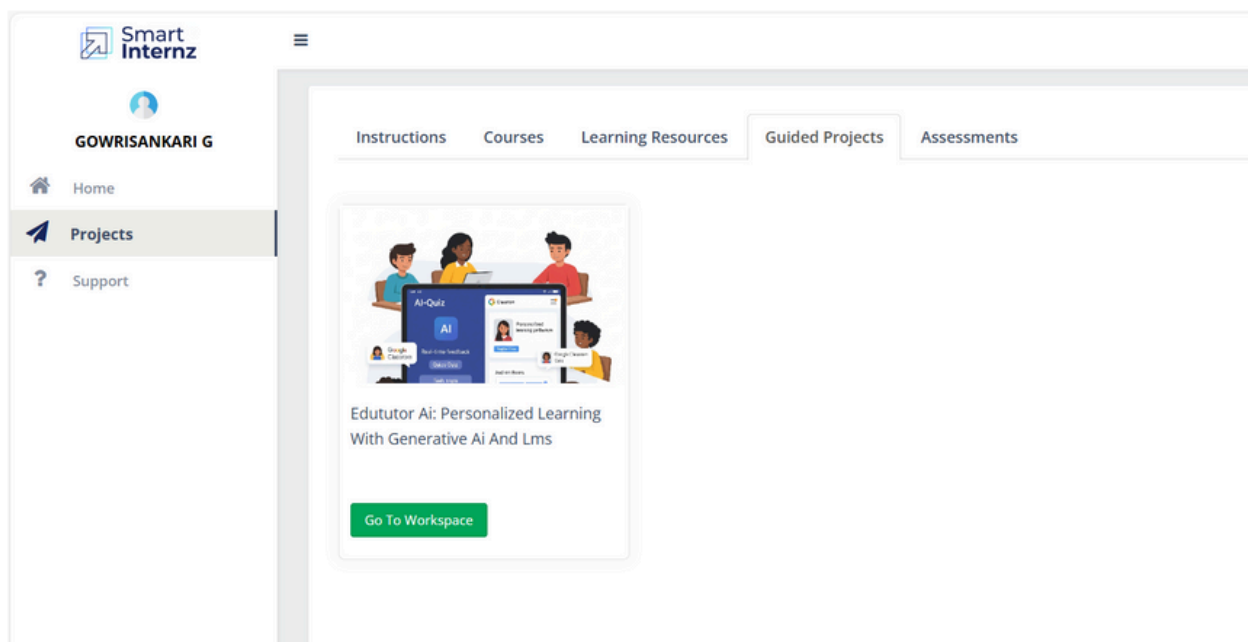
- Then Click on the first link. ([Naanmudhalvan Smartinternz](https://naanmudhalvan.smartinternz.com)) Then login with your details.



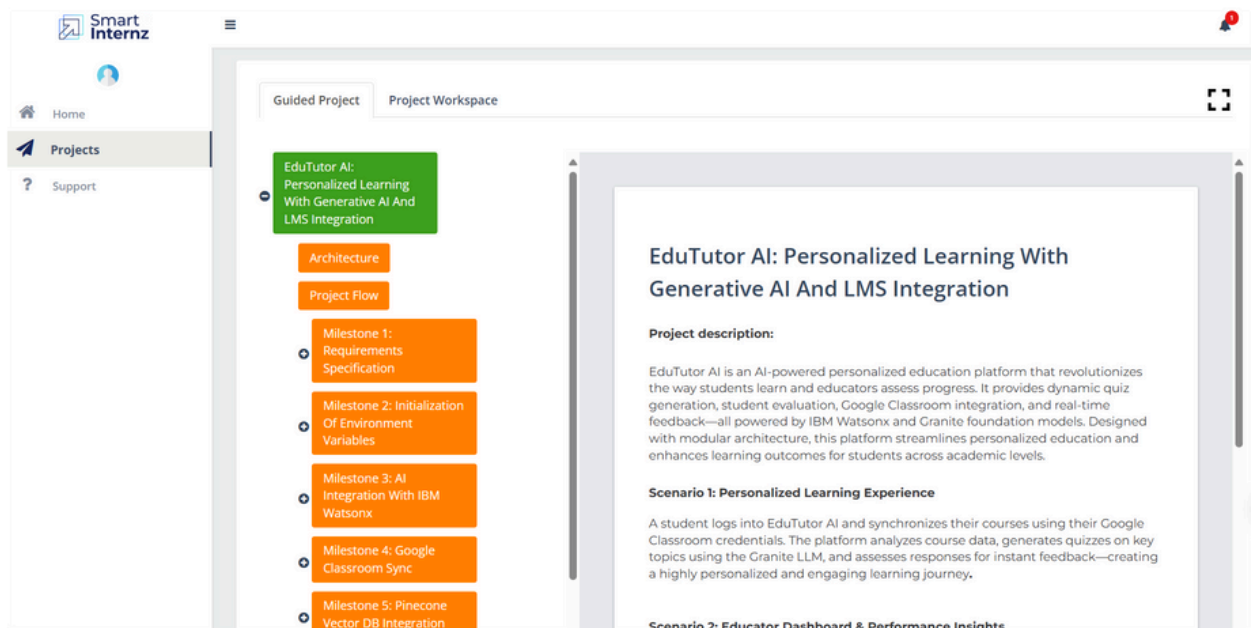
- Then you will be redirected to your account then click on “Projects” Section. There you can see which project you have enrolled in here it is “EduTutor AI”.



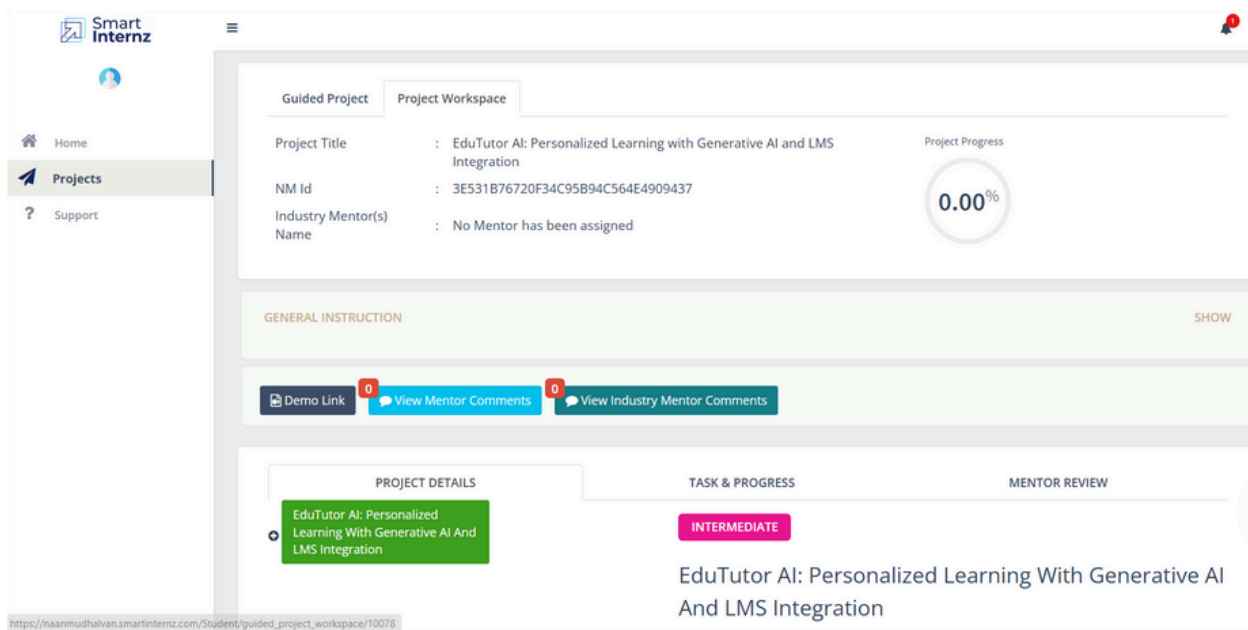
- Then click on “Access Resources” and go to the “Guided Project” Section.



- Click on the “Go to workspace” section. Then you can find the detailed explanation of Generative AI Project using IBM Watsonx API key.



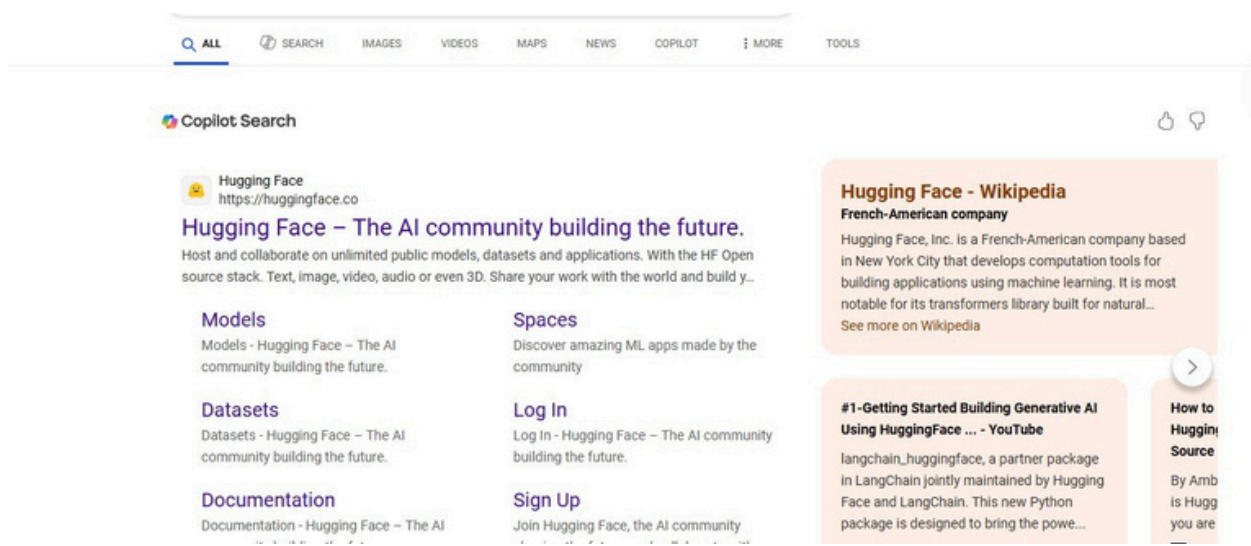
- Click on “Project Workspace”, there you can find your project progress and Place to upload “Demo link”.



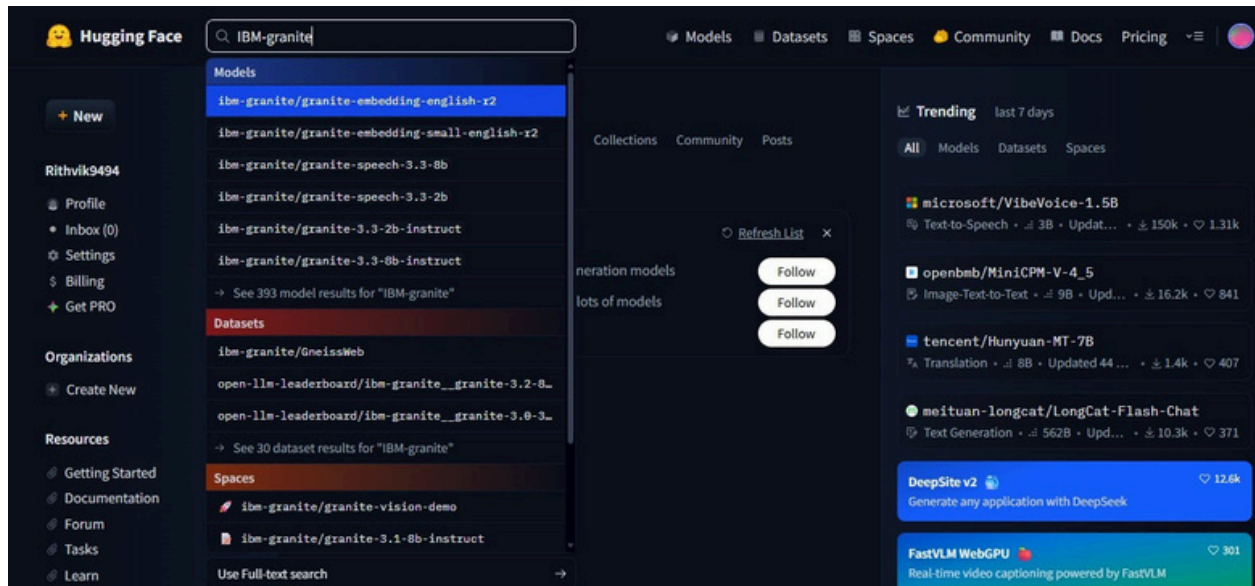
- Now we have gone through portal understanding, now let's find a IBM granite model from hugging face to integrate in our project.

Activity-2: Choose a IBM Granite model From Hugging Face.

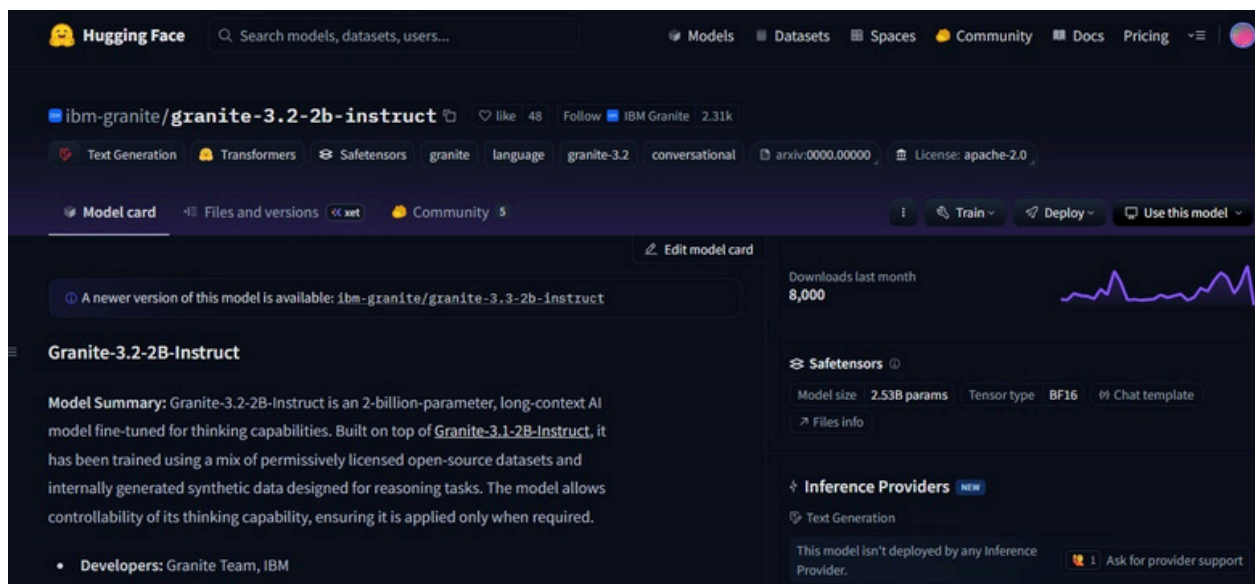
- Search for “Hugging face” in any browser.



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



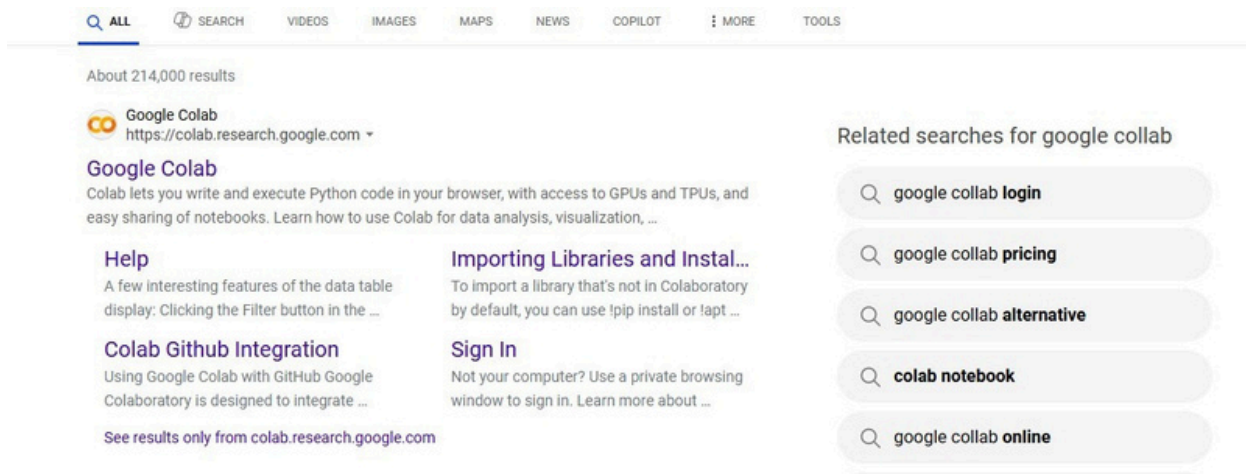
- Here for this project we are using “granite-3.2-2b-instruct” which is compatible fast and light weight.



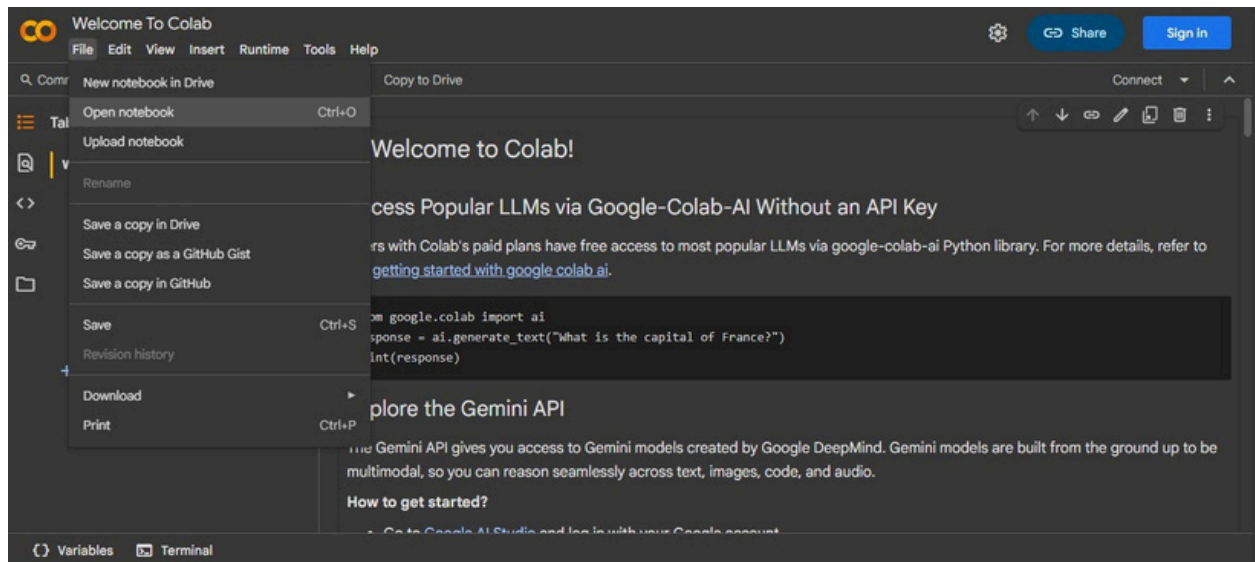
- Now we will start building our project in Google collab.

Activity-3: Running Application in Google Collab.

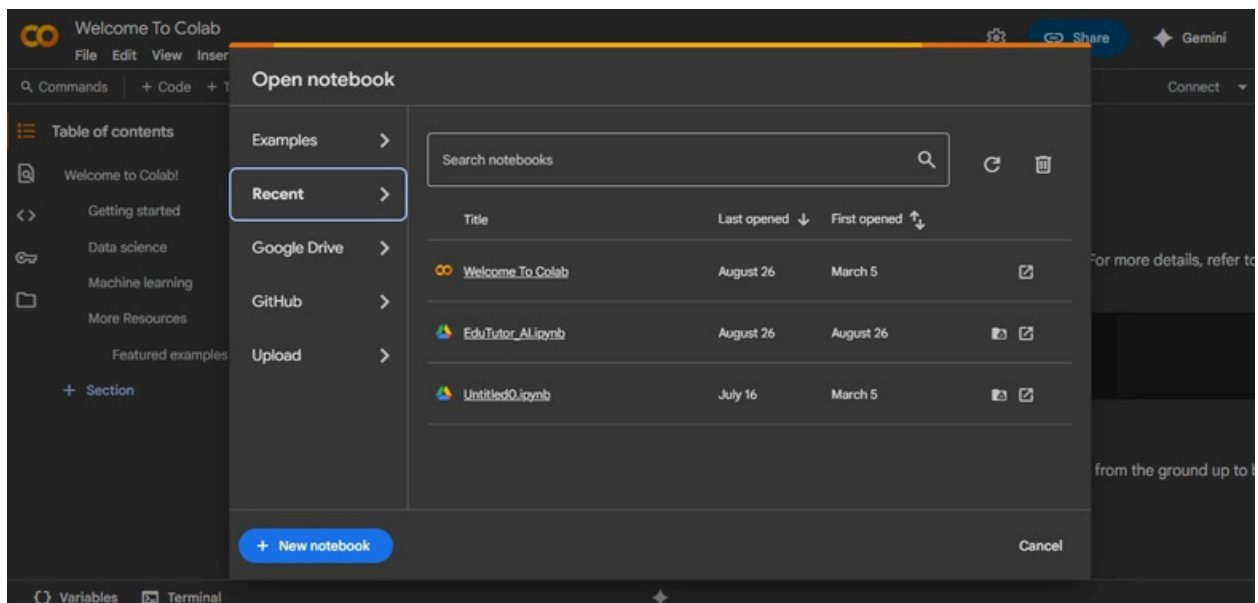
Search for “Google collab” in any browser.



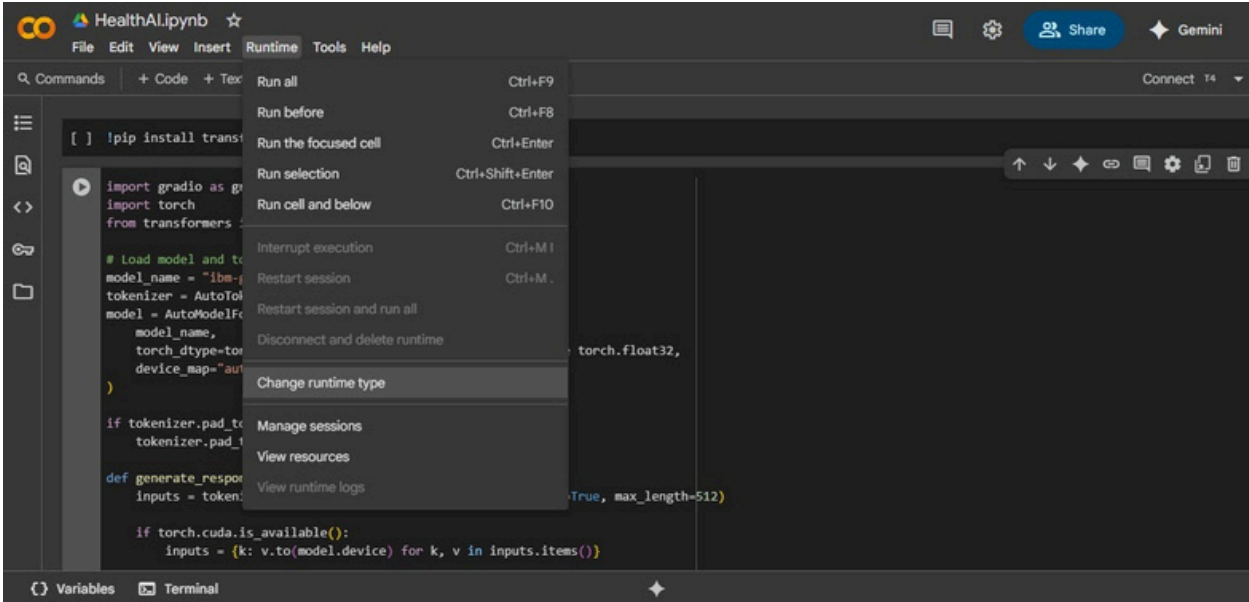
- Click on the first link ([Google Colab](https://colab.research.google.com)), then click on “Files” and then “Open Notebook”.



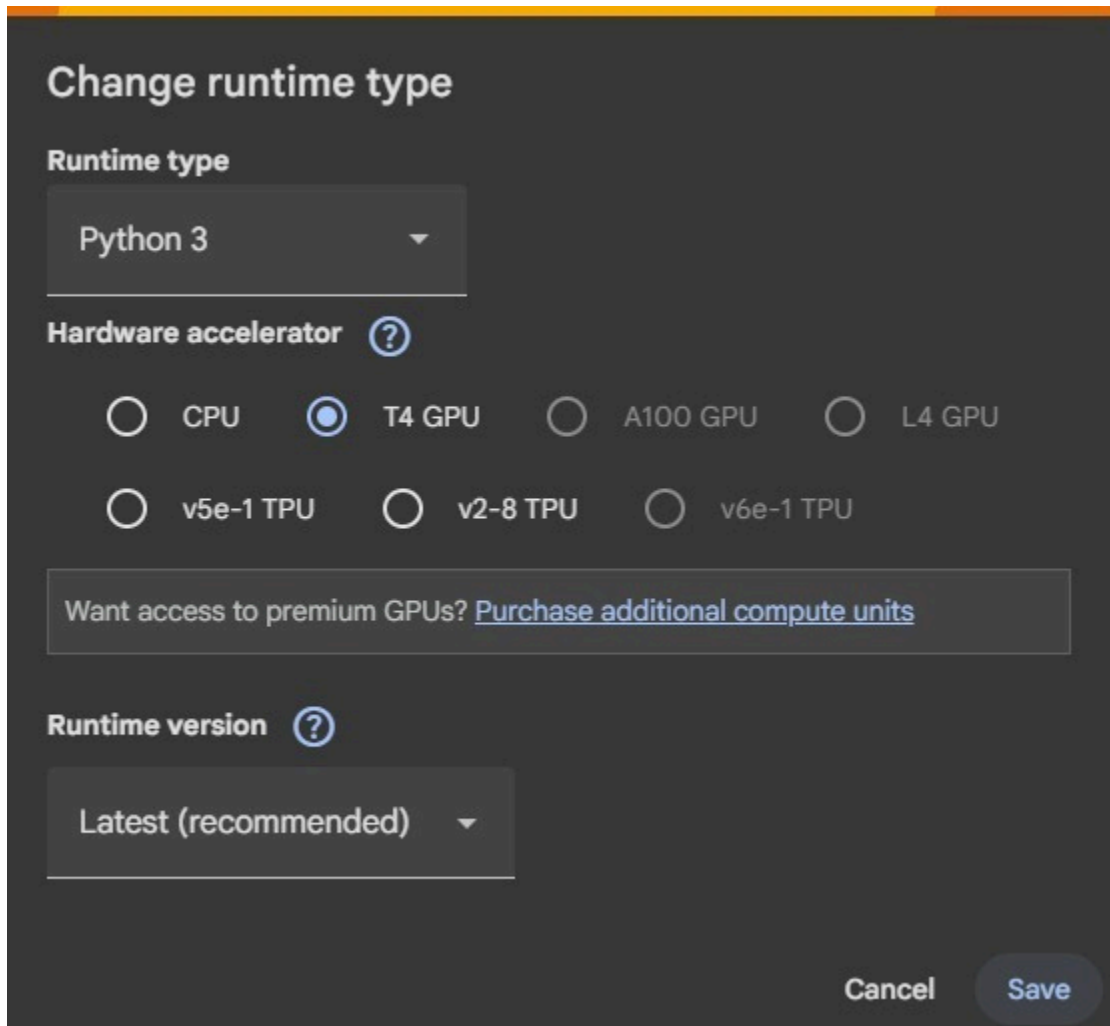
- Click on “New Notebook”



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



- Choose “T4 GPU” and click on “Save”



The image shows a 'Change runtime type' dialog box with a dark background. At the top, the title 'Change runtime type' is in white. Below it, the 'Runtime type' is set to 'Python 3' in a dropdown menu. The 'Hardware accelerator' section has a help icon and several radio button options: 'CPU', 'T4 GPU' (which is selected), 'A100 GPU', 'L4 GPU', 'v5e-1 TPU', 'v2-8 TPU', and 'v6e-1 TPU'. Below these options is a text box that says 'Want access to premium GPUs? [Purchase additional compute units](#)'. The 'Runtime version' section has a help icon and a dropdown menu set to 'Latest (recommended)'. At the bottom right, there are 'Cancel' and 'Save' buttons.

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

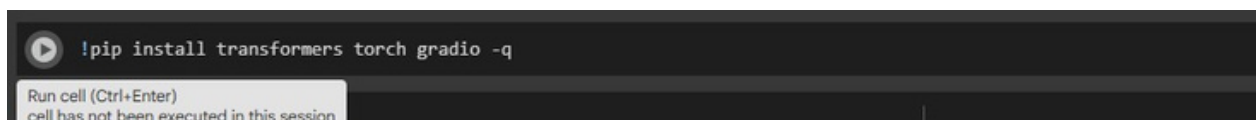
Want access to premium GPUs? [Purchase additional compute units](#)

Runtime version ?

Latest (recommended)

Cancel Save

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



The image shows a code cell interface. At the top, there is a play button icon and the command `!pip install transformers torch gradio -q`. Below the code, there is a button that says 'Run cell (Ctrl+Enter)'. At the bottom, there is a status bar that says 'cell has not been executed in this session'.

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

Run cell (Ctrl+Enter)

cell has not been executed in this session

- Then run the rest of the code in the next cell.

```
EduTutorAlIpynb ☆ Changes will not be saved
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text ▶ Run all Copy to Drive
Connect 14

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         )
```

```
EduTutorAlIpynb ☆ Cannot save changes
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text ▶ Run all Copy to Drive
RAM 100% Disk 100%

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.Tabs():
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
54             explain_btn.click(concept_explanation, inputs=concept_input, outputs=explanation_output)
55
56         with gr.TabItem("Quiz Generator"):
57             quiz_input = gr.Textbox(label="Enter a topic", placeholder="e.g., physics")
58             quiz_btn = gr.Button("Generate Quiz")
59             quiz_output = gr.Textbox(label="Quiz Questions", lines=15)
60
61             quiz_btn.click(quiz_generator, inputs=quiz_input, outputs=quiz_output)
62
63 app.launch(share=True)
```

● You can find the code here in this [link: EduTutor AI Code](#)

OUTPUT:

- Now you can see our model is being Downloaded and the application is

g.

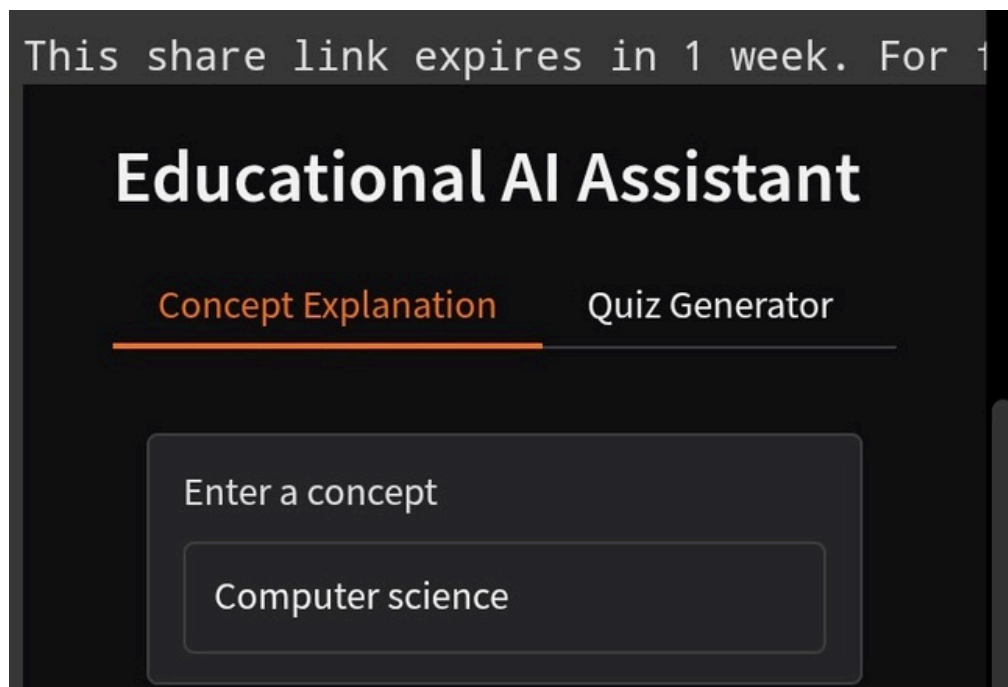
```
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret 'HF_TOKEN' does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart your session.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
warnings.warn(

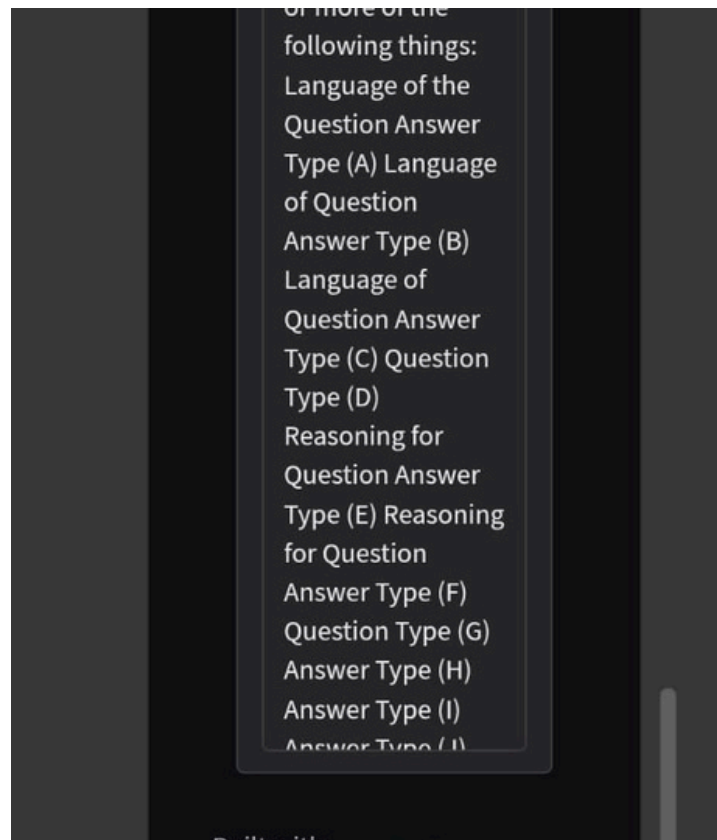
tokenizer_config.json: 8.88k/? [00:00<00:00, 666kB/s]
vocab.json: 777k/? [00:00<00:00, 30.9MB/s]
merges.txt: 442k/? [00:00<00:00, 23.4MB/s]
tokenizer.json: 3.48M/? [00:00<00:00, 84.3MB/s]
added_tokens.json: 100% [00:00<00:00, 87.0kB/s]
special_tokens_map.json: 100% [00:00<00:00, 50.9kB/s]
config.json: 100% [00:00<00:00, 48.8kB/s]
model.safetensors.index.json: 29.8k/? [00:00<00:00, 2.54MB/s]
Fetching 2 files: 100% [02:21<00:00, 141.8kB/s]
model-00001-of-00002.safetensors: 100% [02:21<00:00, 50.7MB/s]
model-00002-of-00002.safetensors: 100% [00:02<00:00, 37.0MB/s]
Loading checkpoint shards: 100% [00:25<00:00, 10.58kB/s]
generation_config.json: 100% [00:00<00:00, 10.5kB/s]
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://92320020f660b93f05.gradio.live
This share link expires in 1 week. For free permanent hosting and GPU upgrades, run 'gradio deploy' from the terminal in the working directory to deploy to Hugging Face Spaces (https://huggingface.co/spaces)
```

- Click on the URI to open the Gradio Application click on the link.

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

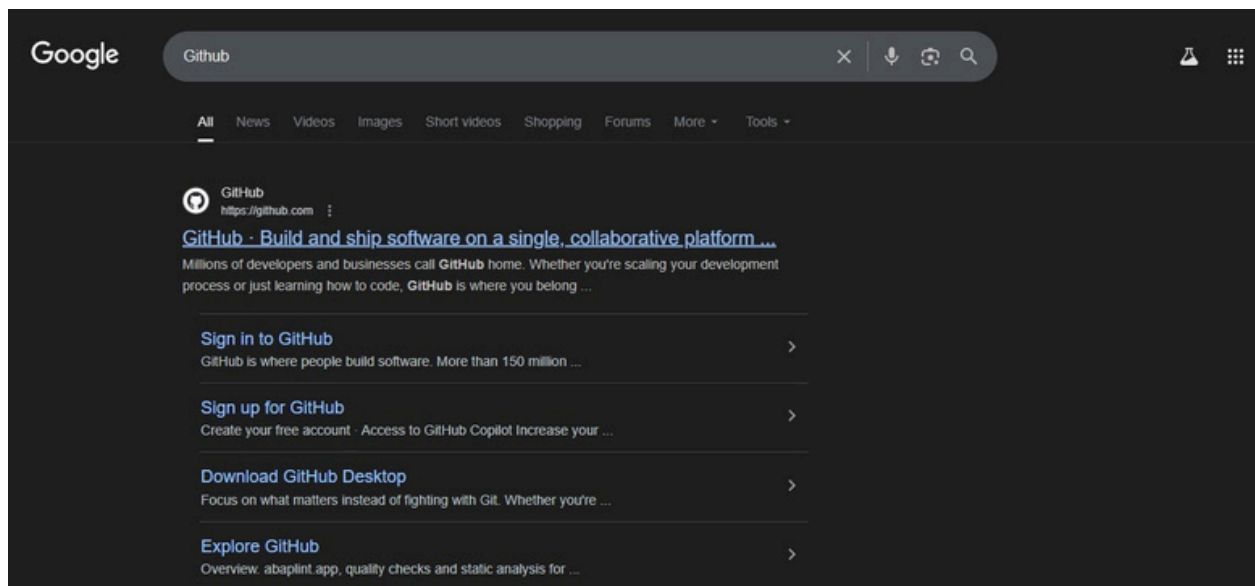
- You can View the Application is running in the other tab.



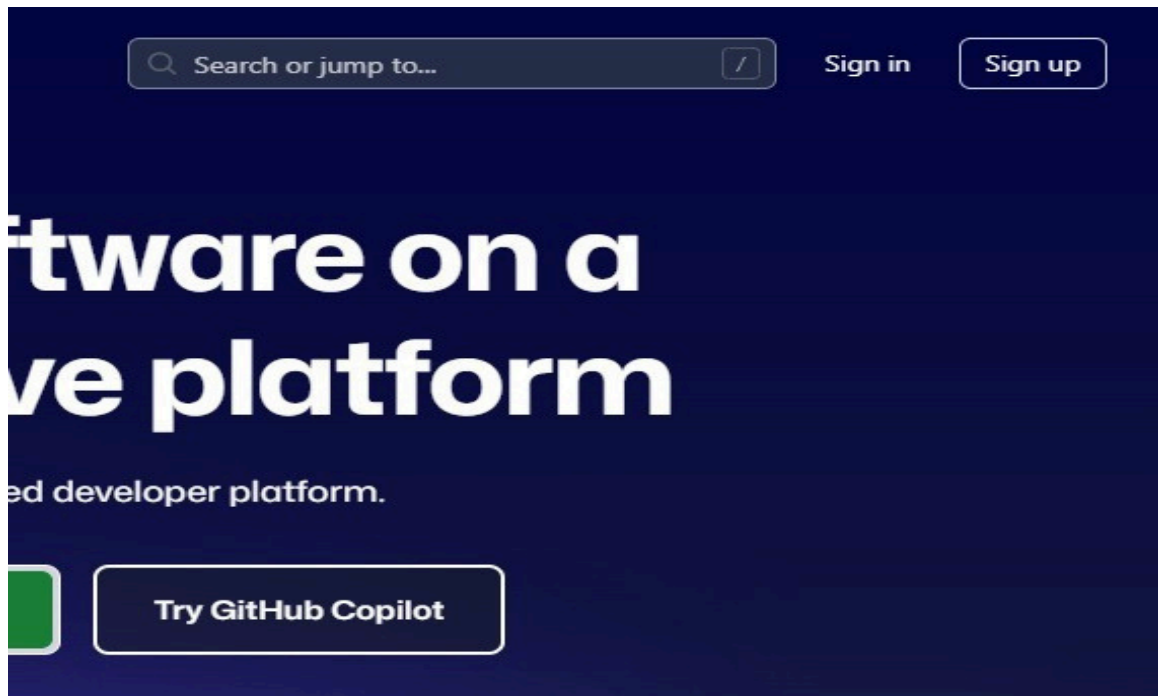


Activity-4: Upload Your Project in GitHub.

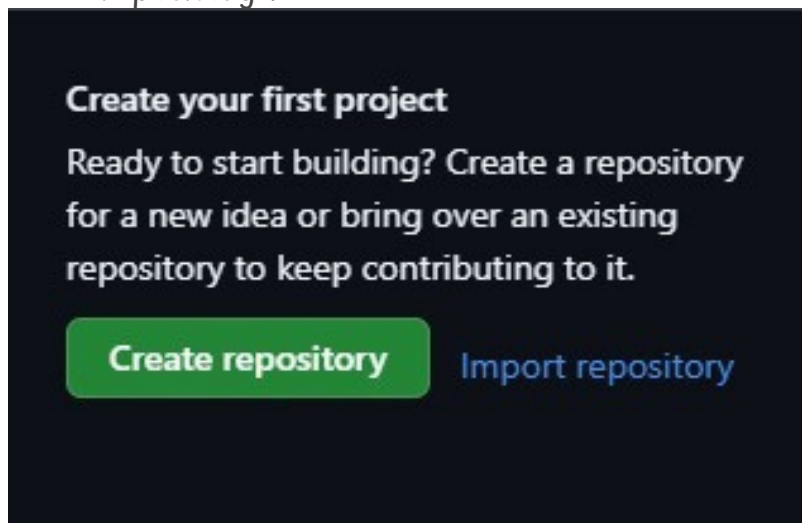
- Search for “GitHub” in any browser, then click on the first link ([GitHub](https://github.com)).



- Then click on “Signup” and create your own account in GitHub. If you already have an account click on “Sign in”



- Click on “Create repository”.



- In “General” Name your repo. (Here I have given “IBM-Project” as my repo name and it is available)

Create a new repository
Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).
Required fields are marked with an asterisk (*).

1 General

Owner * padamavathikonakala-design / **Repository name *** IBM-Project
IBM-Project is available.

Great repository names are short and memorable. How about [sturdy-octo-guacamole](#)?

Description

0 / 350 characters

2 Configuration

Choose visibility *
Choose who can see and commit to this repository Public

Add README
READMEs can be used as longer descriptions. [About READMEs](#) Off

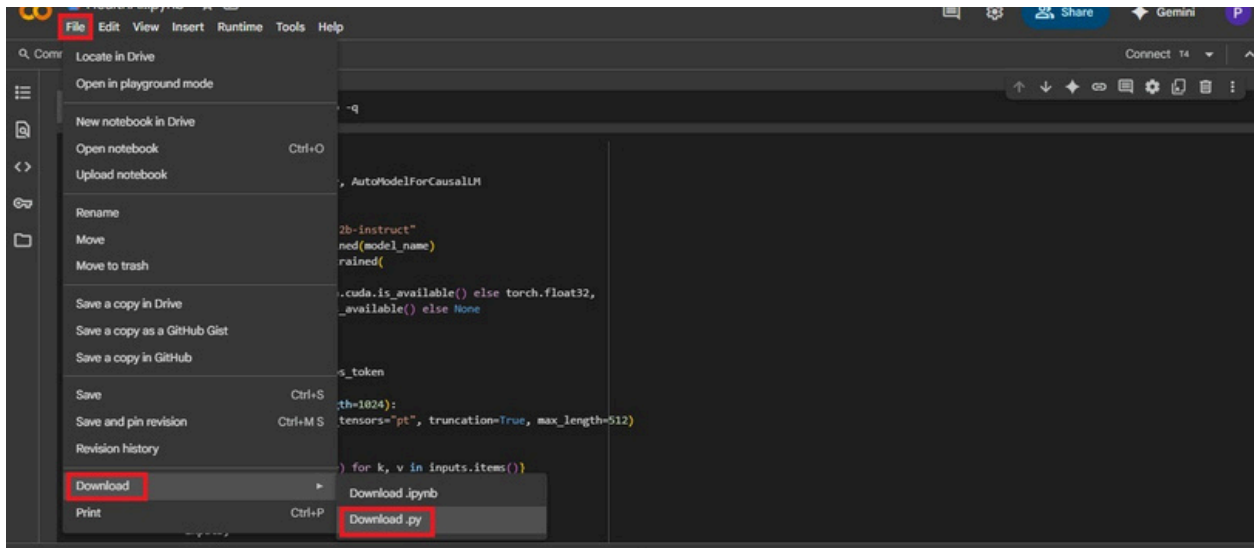
- In “Configurations” Turn On “Add readme” file Option.

2 Configuration

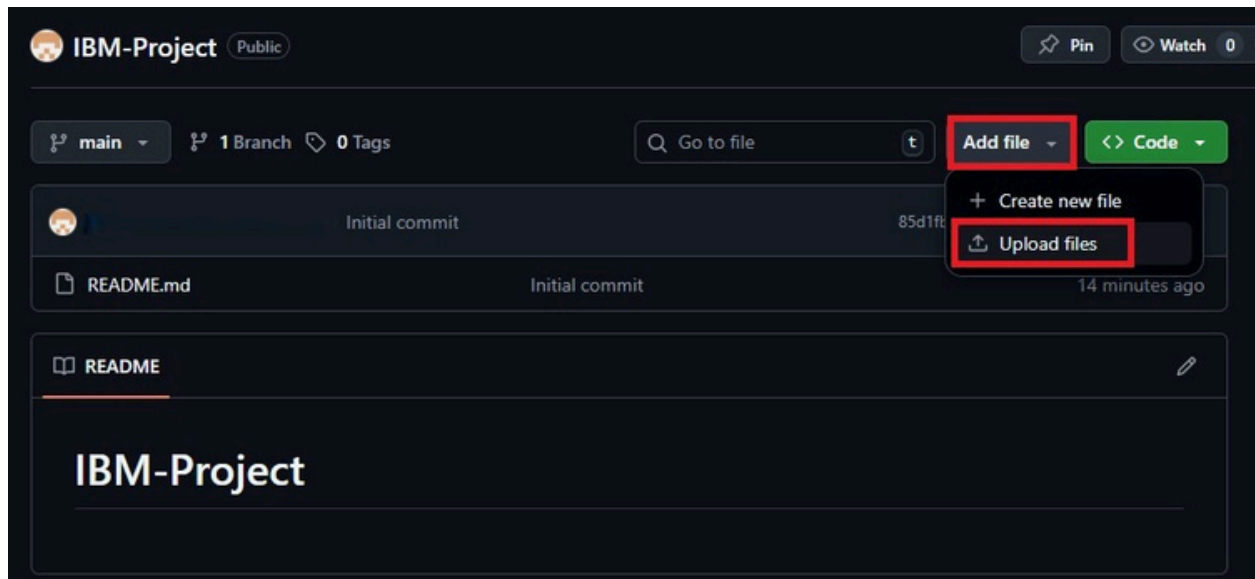
Choose visibility *
Choose who can see and commit to this repository Public

Add README
READMEs can be used as longer descriptions. [About READMEs](#) On

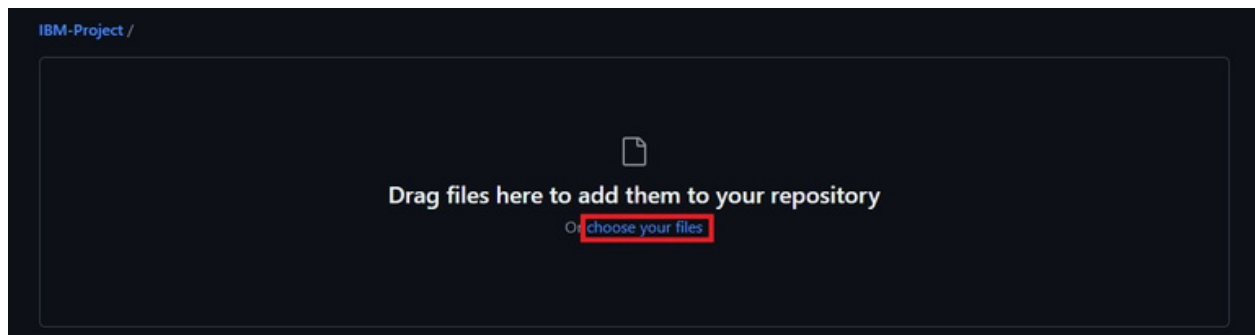
- Now Download your code from Google collab by Clicking on “File”, then Goto “Download” then download as “.py”.



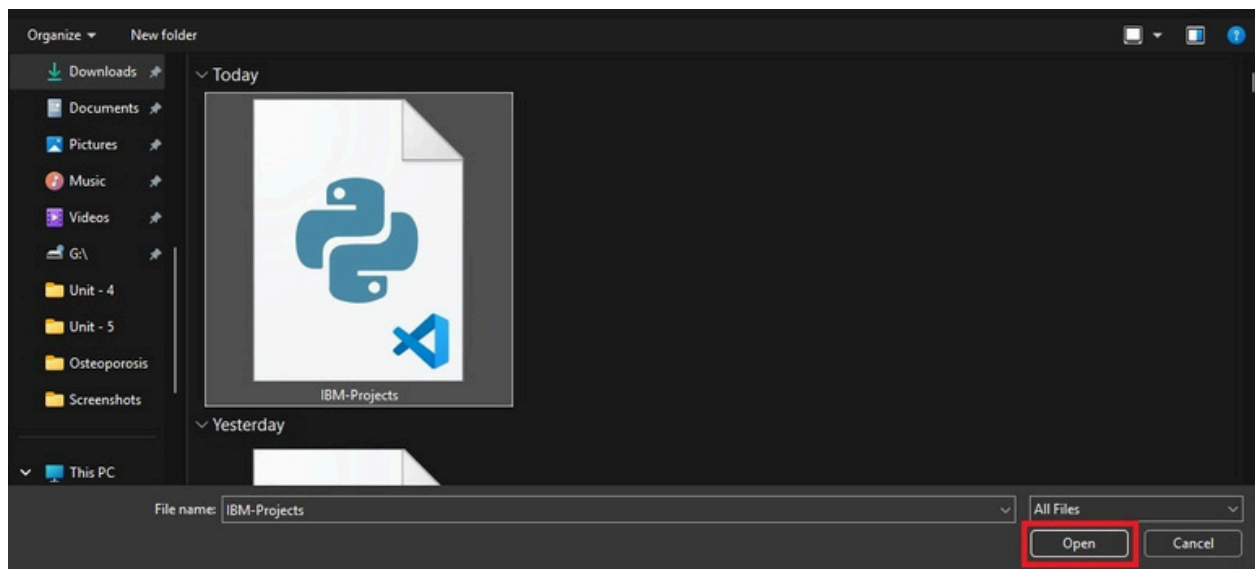
- Then your repository is created, then Click on “Add file” Option. Then Click “Upload files” to upload your files.



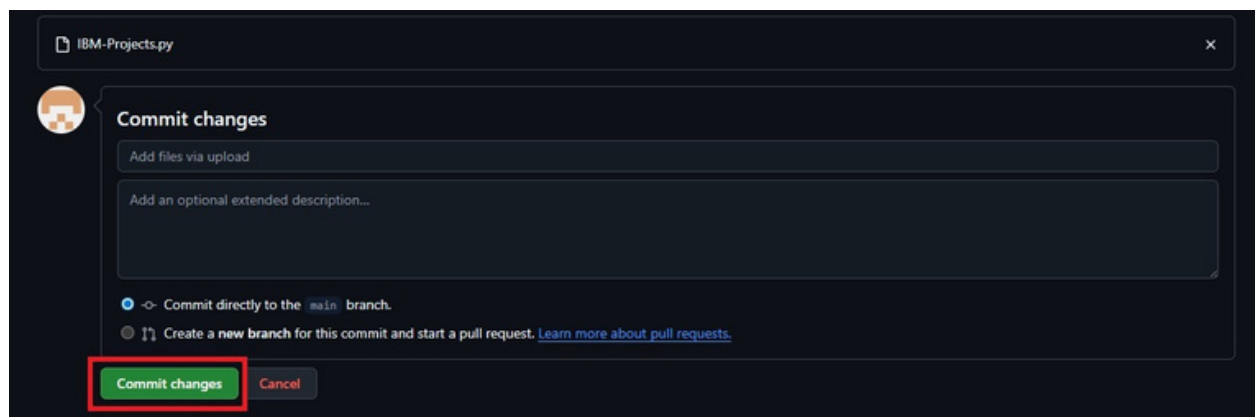
- Click on “choose your files”.



- Choose your project file and click on “Open”.



- After your file has Uploaded Click on “Commit changes”.



Conclusion EduTutor AI is a revolutionary educational tool that harnesses the power of AI to provide personalized learning experiences, transforming the way students learn and interact with complex concepts.