LAB

Write prompts to generate specific outputs from LLMs

IBM **SkillsBuild**

Introduction

In this lab, you will write and refine prompts to generate relevant output from large language models (LLMs).

Software requirements

To complete this lab, you don’t have to install any software on your computer. You only require a Google account to access Google Colab and a Replicate account to access and use various artificial intelligence (AI) models.

Objective

After completing this lab, you should be able to:

- Write prompts to generate specific outputs from LLMs

Lab steps

This lab requires you to complete the following procedures:

1. Create a GitHub account
2. Create a Replicate account
3. Sign up for Google Colab
4. Load the Jupyter notebook and initialize the model
5. Load the input dataset and write a baseline prompt
6. Refine the prompt using task-specific details
7. Refine the prompt using additional details

Estimated duration to complete

20 minutes

Scenario

Background information

Codal Technologies is a mid-sized information technology (IT) services firm that manages a variety of projects for global retail companies. With multiple projects running simultaneously, the project management team needs clear and consistent weekly updates to stay aligned. These updates provide a structured view of what was completed, what's upcoming, and any emerging issues. This helps managers track progress accurately, identify risks early, allocate resources effectively, and ensure smooth, on-time delivery for clients.

Challenge

You are a project intern at Codal Technologies and prepare weekly project updates for the project management team. However, the information you receive is scattered across Slack messages, Jira tickets, meeting notes, and long email threads. It is often mixed with highly technical updates from clients and developers. These inputs are inconsistent, incomplete, and confusing, making it difficult to create professional and structured updates.

Solution

To improve efficiency and accuracy, you begin using IBM Granite via Replicate to generate project updates. After some experimentation, you've realized that the quality of the project updates depends on how clearly and precisely you guide the AI model. So, now, you've decided to refine your prompts step-by-step to generate consistent and professional-quality updates.

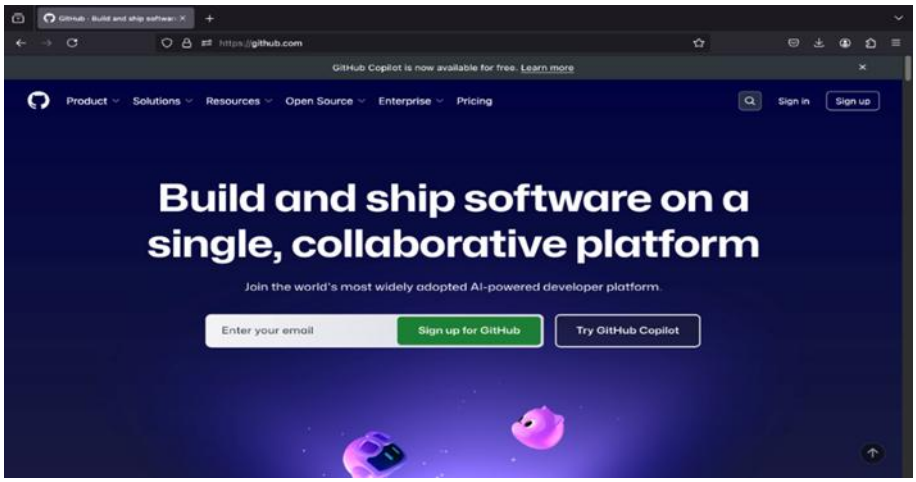
Create a GitHub account

Overview

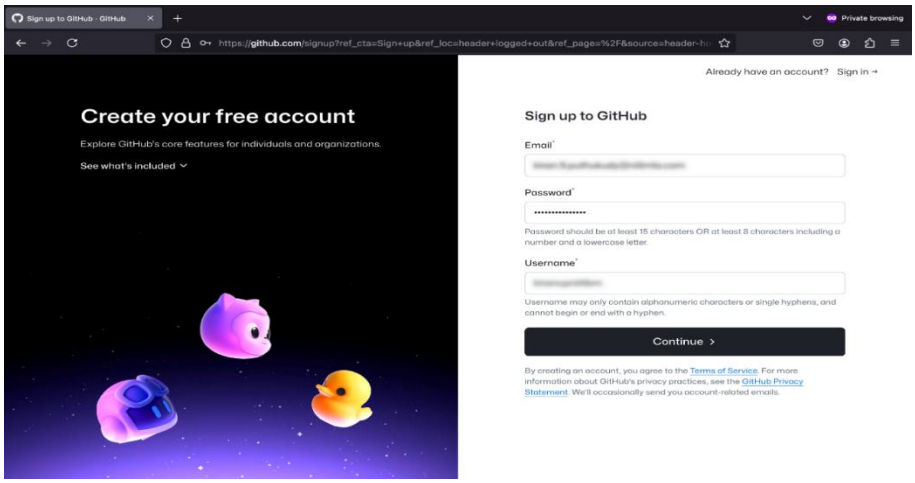
In this procedure, you'll set up a GitHub account. GitHub is a platform that helps developers store, manage, and share code, while also supporting collaboration through tools such as version control, bug tracking, and task management. Setting up a GitHub account provides access to the Replicate platform, which is required to complete the lab efficiently.

Instructions

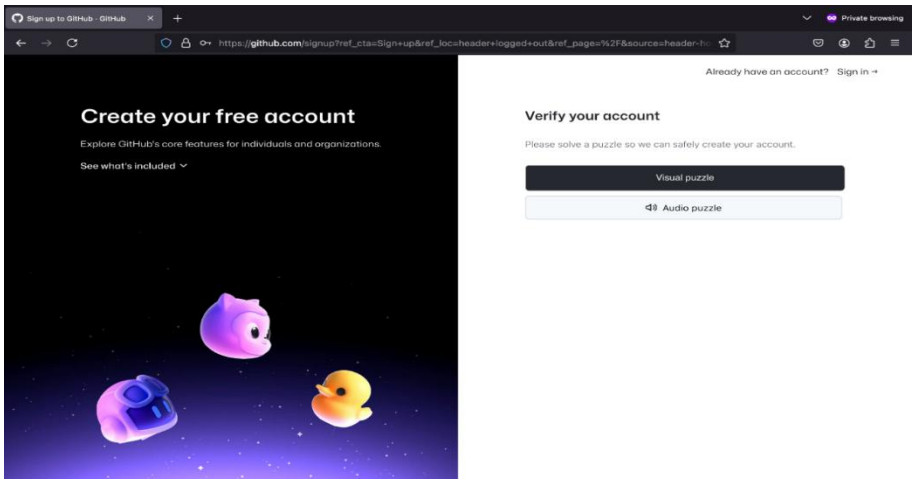
1. To create a GitHub account, go to the GitHub website.
2. Select the **Sign up for GitHub** button, as shown in the following screenshot.



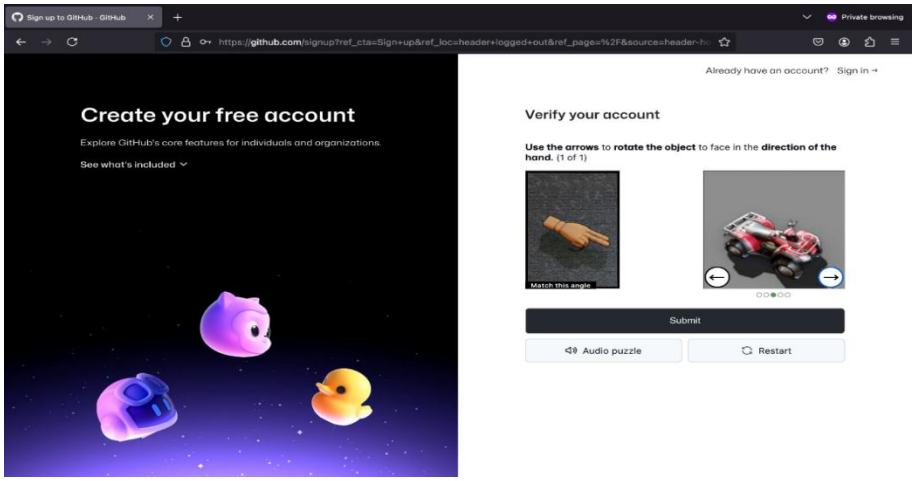
3. The following screenshot shows the "Sign up to GitHub" page. Enter your details in the **Email**, **Password**, and **Username** fields. Then, select **Continue**.



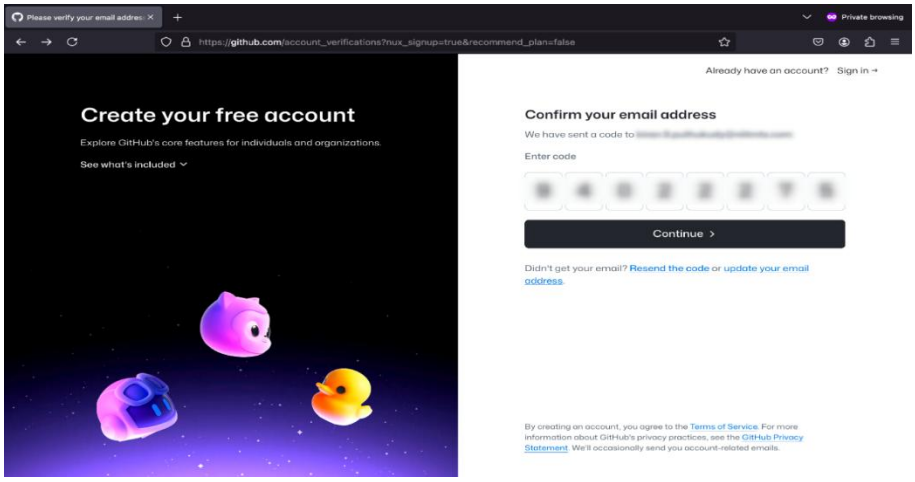
4. To verify your account, select the **Visual puzzle** button.



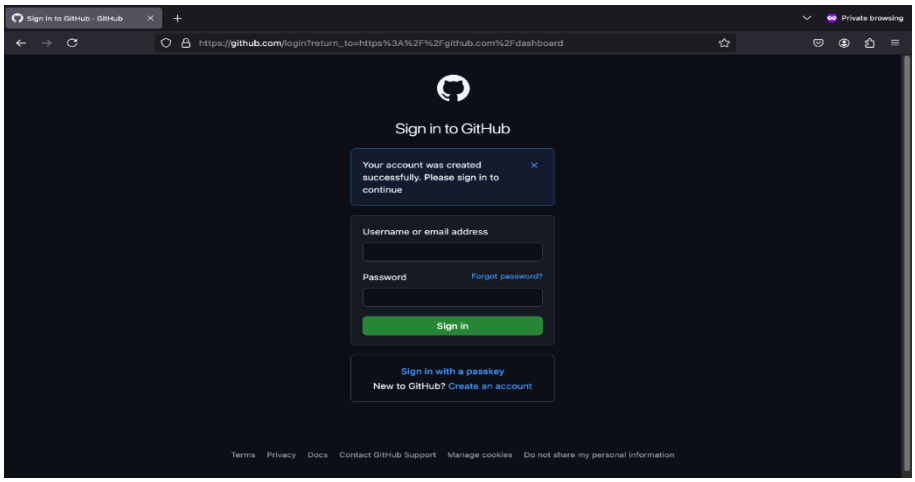
5. Solve the visual puzzle and select **Submit**.



6. To confirm your email, enter the confirmation code sent to your registered email in the **Enter code** field and select **Continue**.

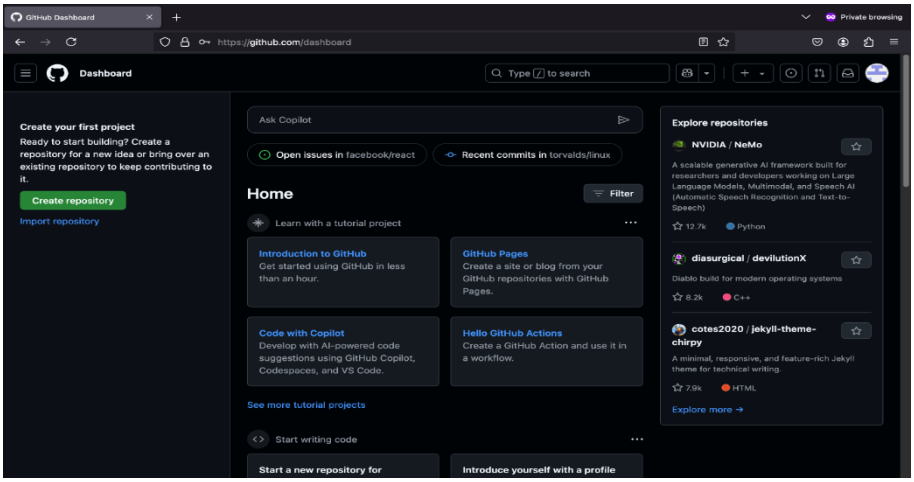


7. After your GitHub account has been successfully created, a confirmation message is displayed as shown in the following screenshot.



8. To sign in to your account, enter your credentials in the **Username or email address** and **Password** fields, and then select **Sign in**.

9. After you sign in, the GitHub dashboard displays as shown in the following screenshot.



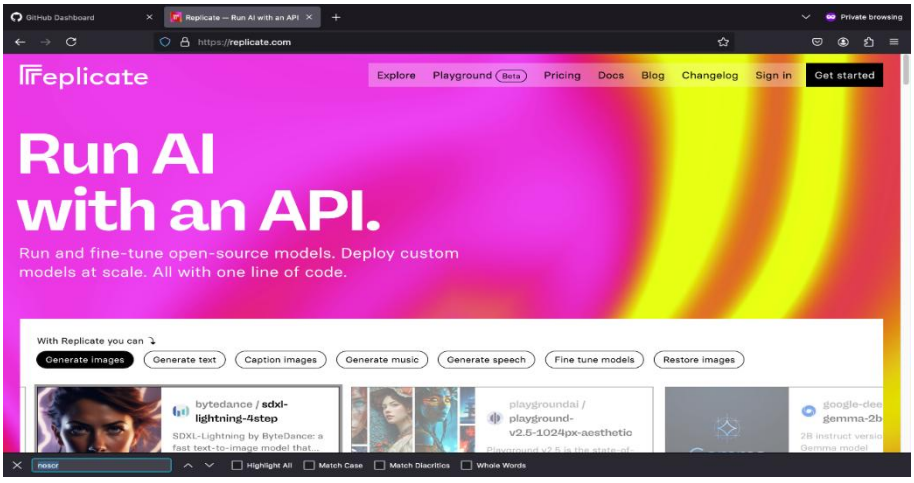
Create a Replicate account

Overview

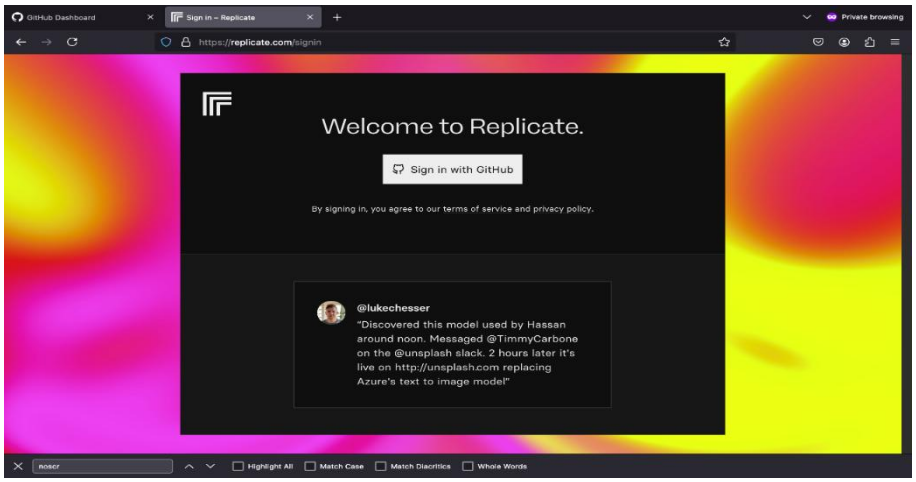
In this procedure, you'll use your GitHub account to register for a Replicate account. Replicate is a cloud-based platform that lets you use AI models such as IBM Granite without requiring advanced hardware. As part of this procedure, you'll create a Replicate token, a secure access key that allows the lab environment to authenticate with Replicate and run models from your account in Google Colab.

Instructions

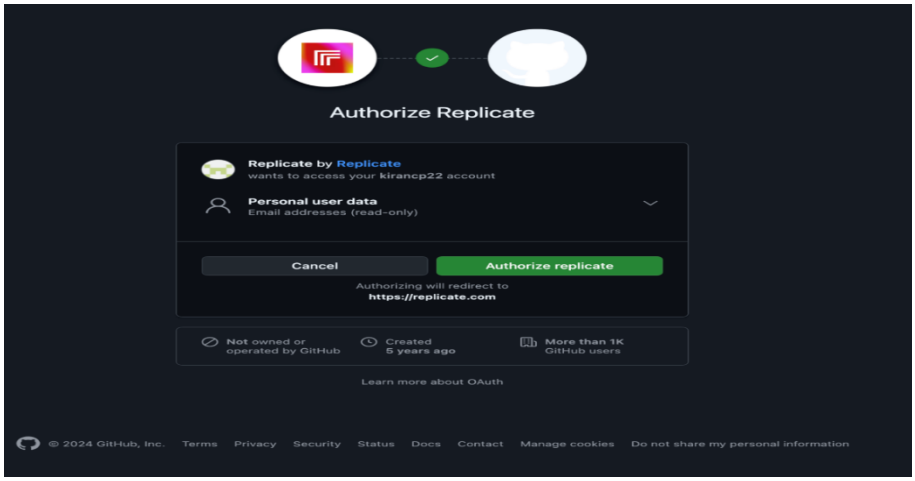
1. Go to the Replicate website.
2. Select **Get started**, as shown in the following screenshot.



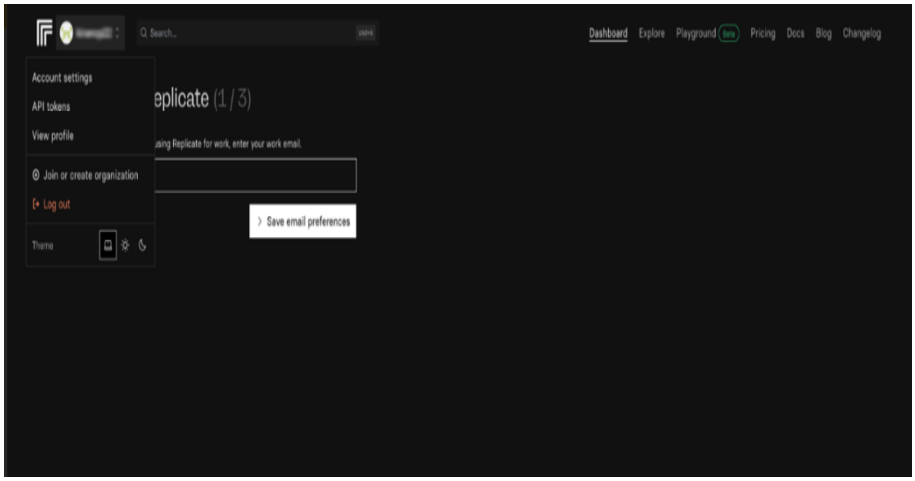
3. Select the **Sign in with GitHub** button, as shown in the following screenshot.



4. Select **Authorize replicate** to continue, as shown in the following screenshot.

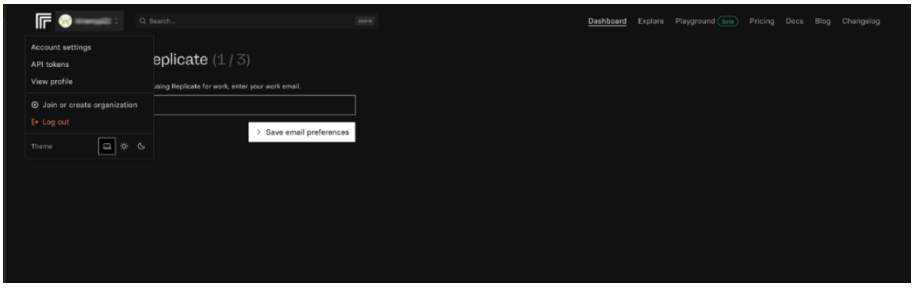


5. After your Replicate account is created, the Replicate dashboard displays as shown in the following screenshot. To create a Replicate token, select the **Account settings** option from the navigation menu.

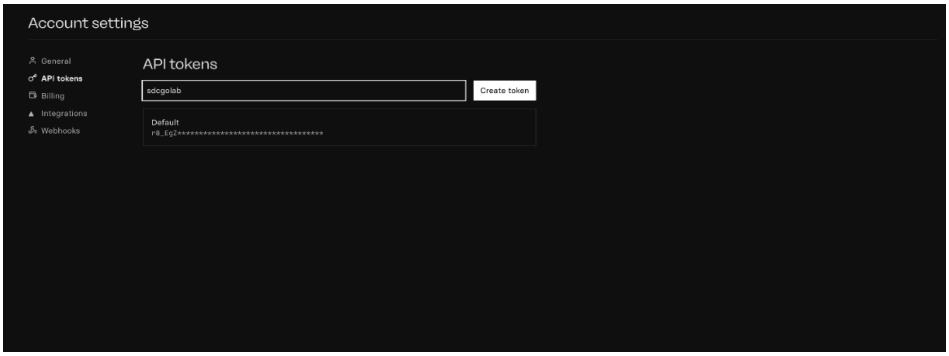


6. The following screenshot shows the Replicate dashboard. Select **API tokens** from the “Account settings” panel.

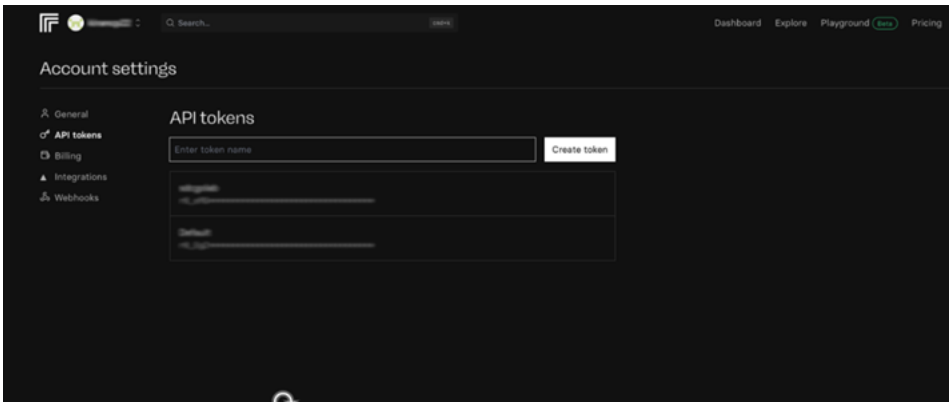
.



7. Enter a name for the token in the **API tokens** field and then select **Create token**, as shown in the following screenshot.

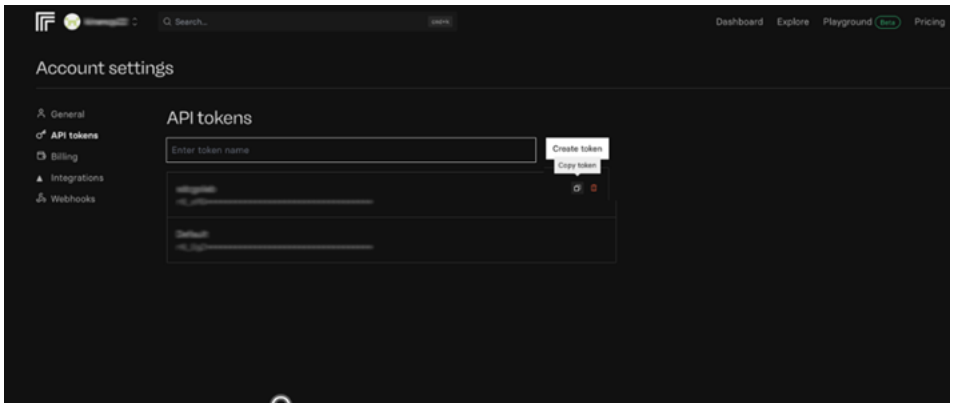


8. After your API token is created, it displays on the “Account settings” page as shown in the following screenshot.



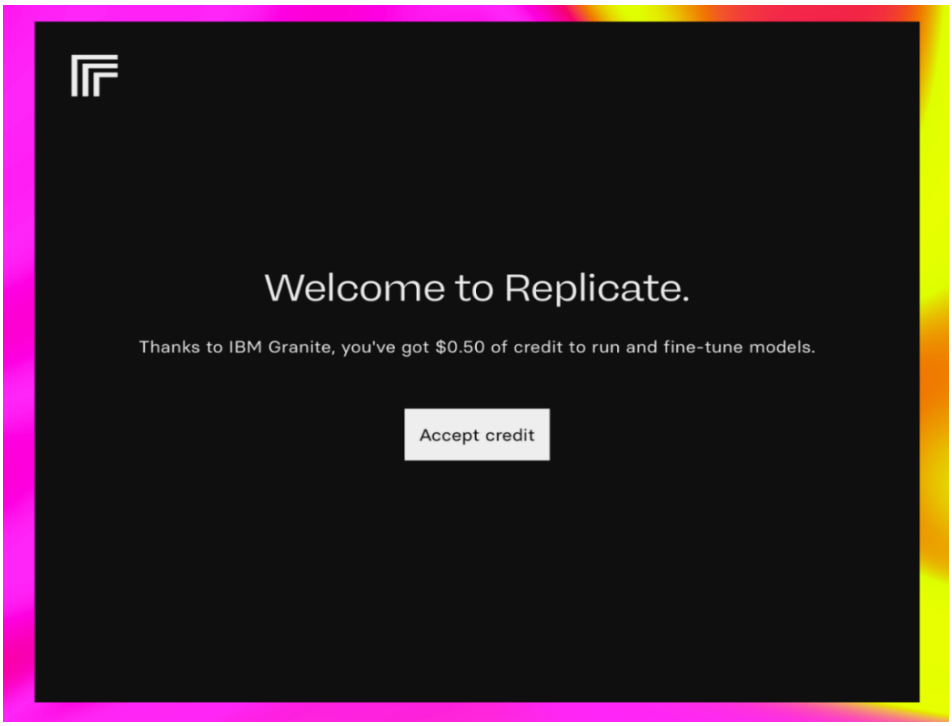
9. To copy the Replicate API, select the **Copy token** icon, as shown in the following screenshot.

   Note: Save the Replicate token because you’ll need it to authenticate with the Replicate API when running code in the Google Colab environment later in this lab.

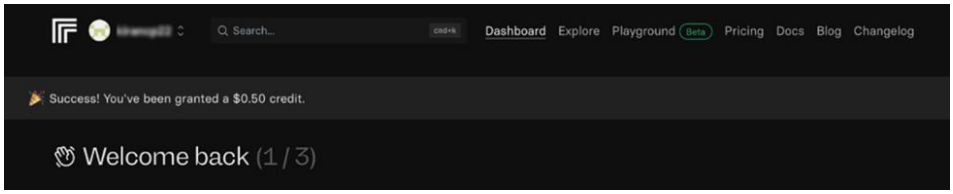


10. To run the lab without interruptions, you'll require some Replicate credits. Go to the Replicate invite link to claim your free $0.50 credit.

11. To claim the amount, select **Accept credit**.



12. A confirmation message is displayed indicating that a $0.50 credit has been added to your Replicate account as shown in the following screenshot.



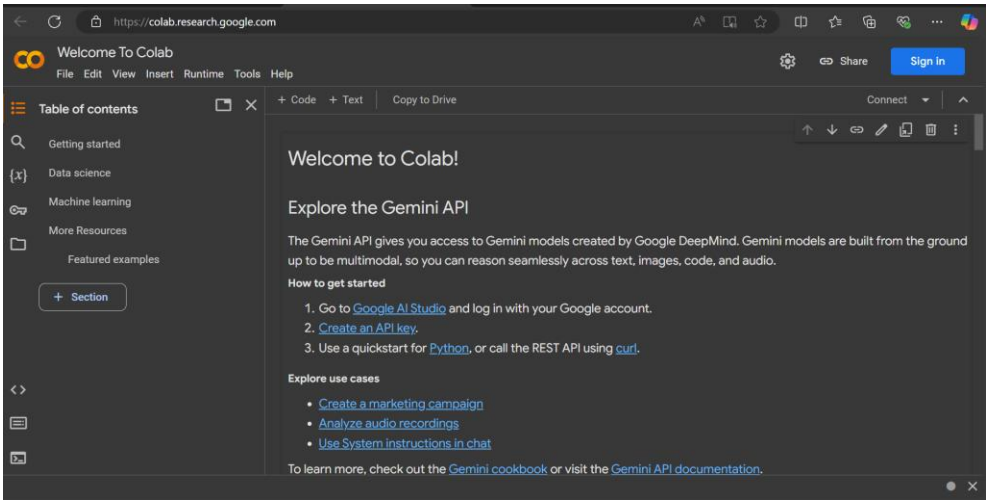
Sign up for Google Colab

Overview

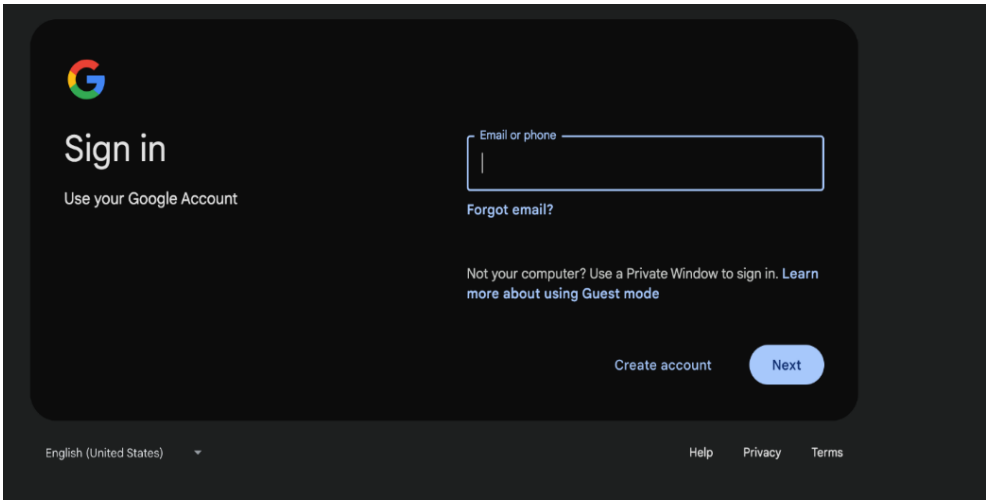
In this procedure, you'll set up a Google Colab account. Google Colab is a free cloud platform that lets you run code in notebooks, which are commonly used for machine learning, data science, and AI tasks. A Google Colab account allows you to install and use the tools needed to complete this lab.

Instructions

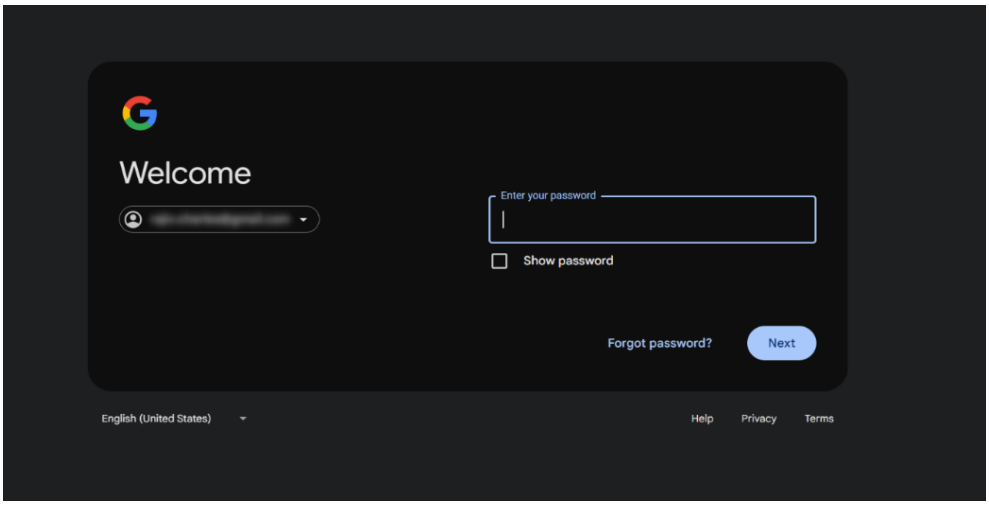
1. To sign up, go to the Google Colab website.
2. Select **Sign in**, as shown in the following screenshot.



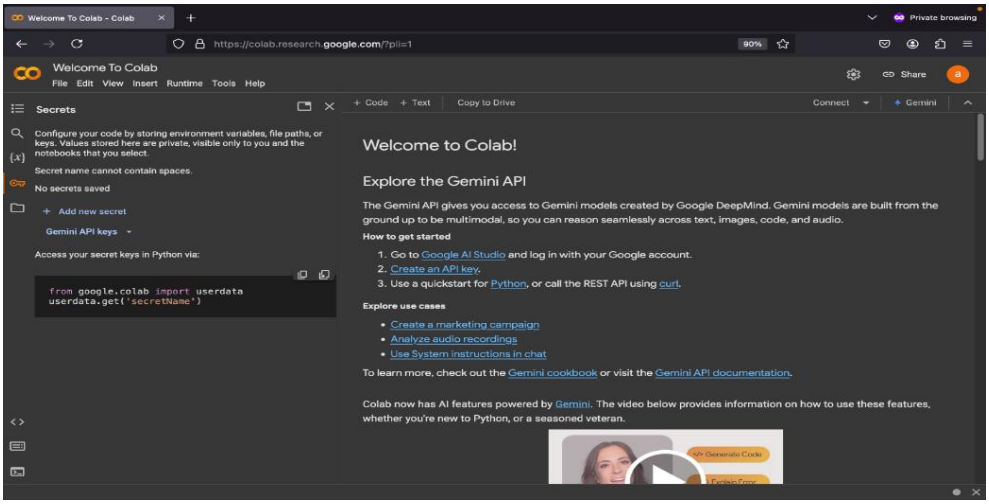
3. Enter your email or phone number in the **Email or phone** field, and then select **Next**. The following screenshot shows the Google sign-in page.



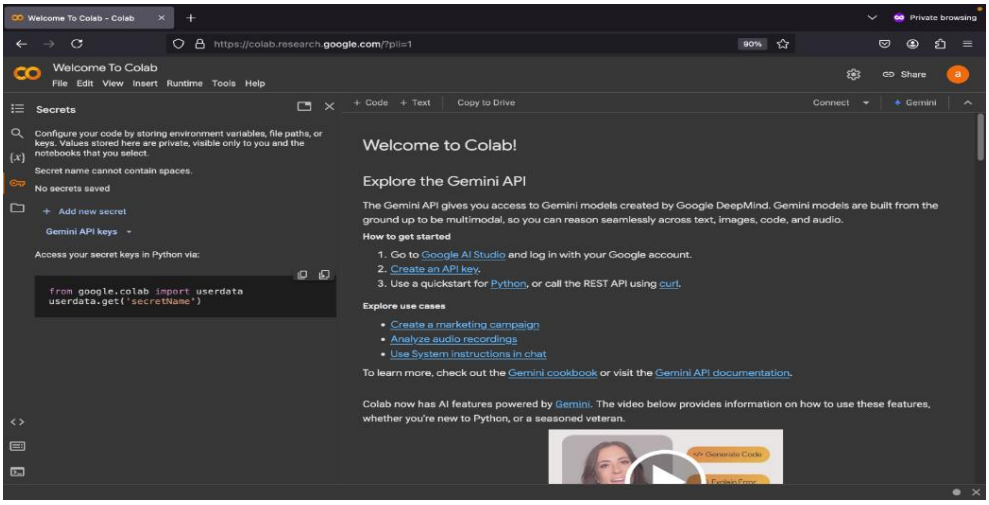
4. Enter your password in the **Enter your password** field, and then select **Next**.



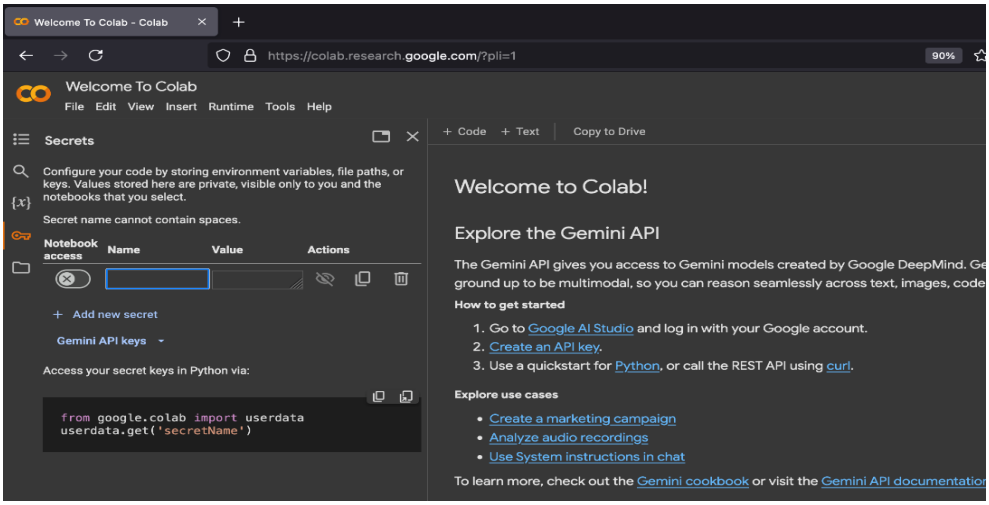
5. To store your Replicate API token, select the key icon from the Secrets tab on the Welcome to Colab page, as shown in the following screenshot.



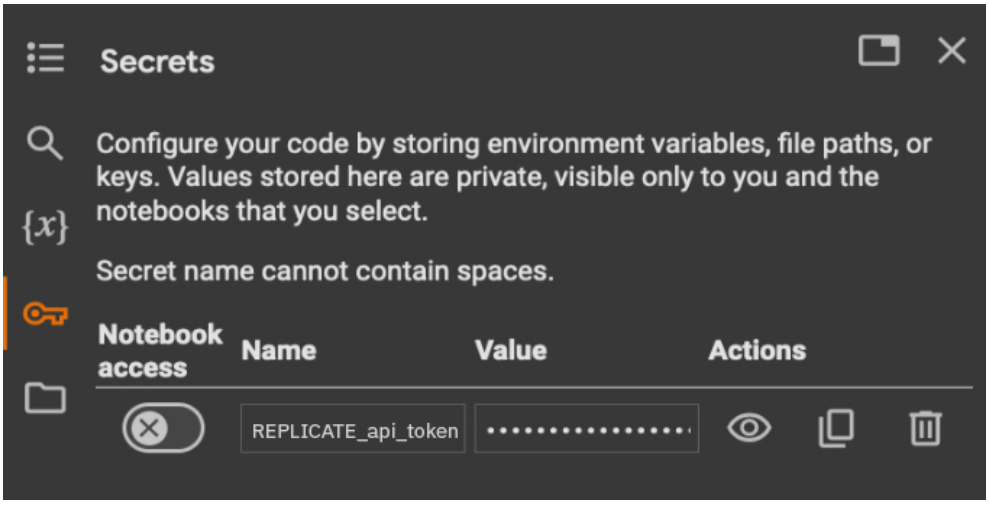
6. Select **Add new secret**.



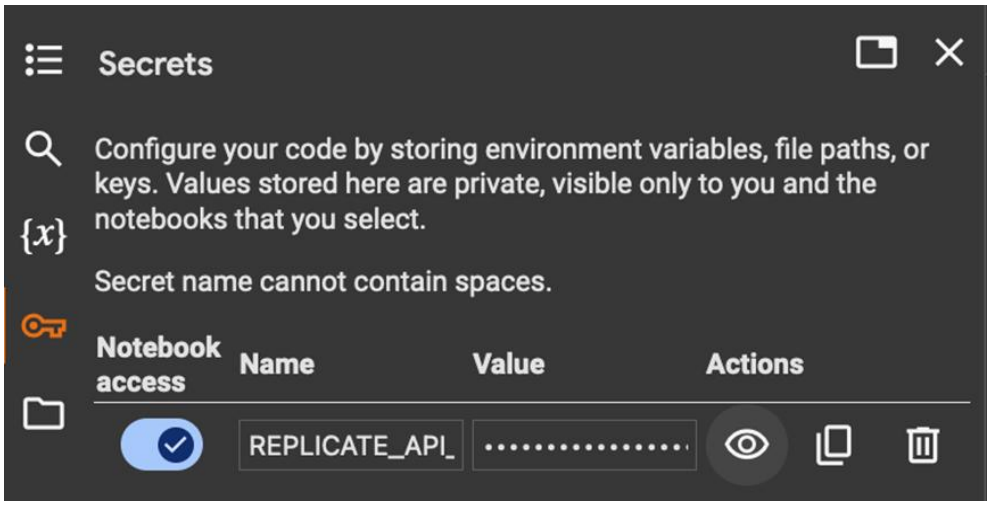
7. Type `REPLICATE_api_token` in the **Name** field.



8. Paste the Replicate API token you copied into the **Value** field, as shown in the following screenshot.



9. Set the **Notebook access** switch on, as shown in the following screenshot. Then, select the close icon to exit the configuration.



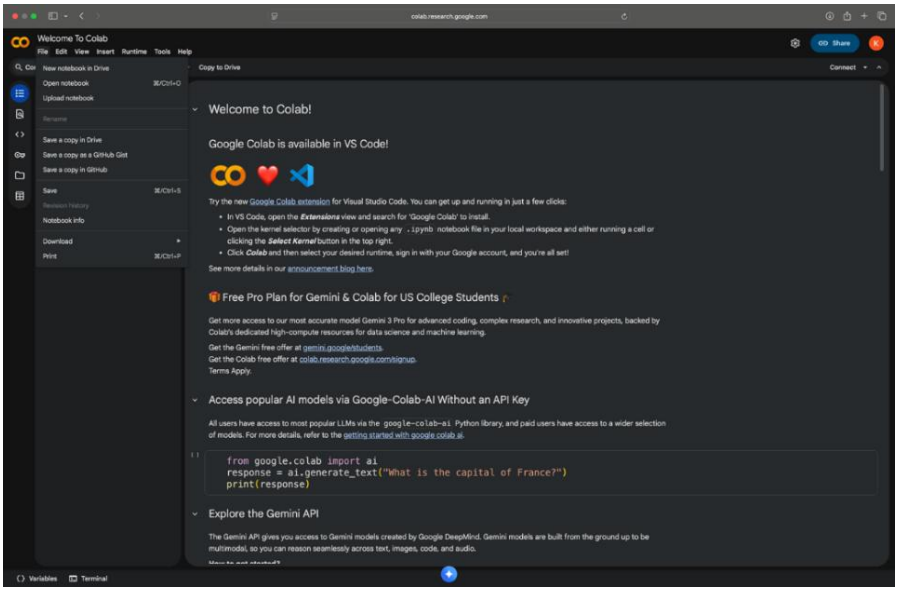
Load the Jupyter notebook and initialize the model

Overview

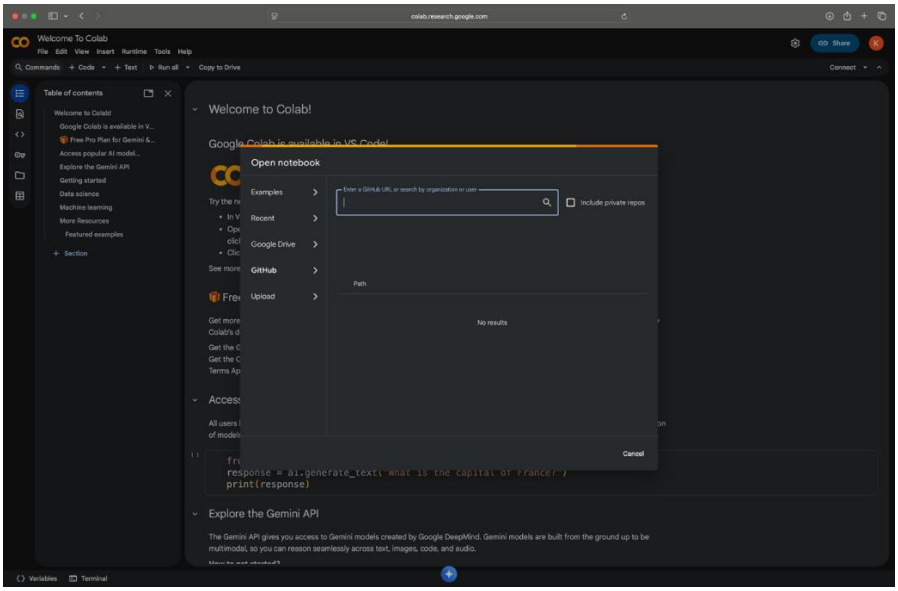
In this procedure, you'll load the Jupyter notebook and initialize the AI model. This sets up your environment before you can interact with the AI model.

Instructions

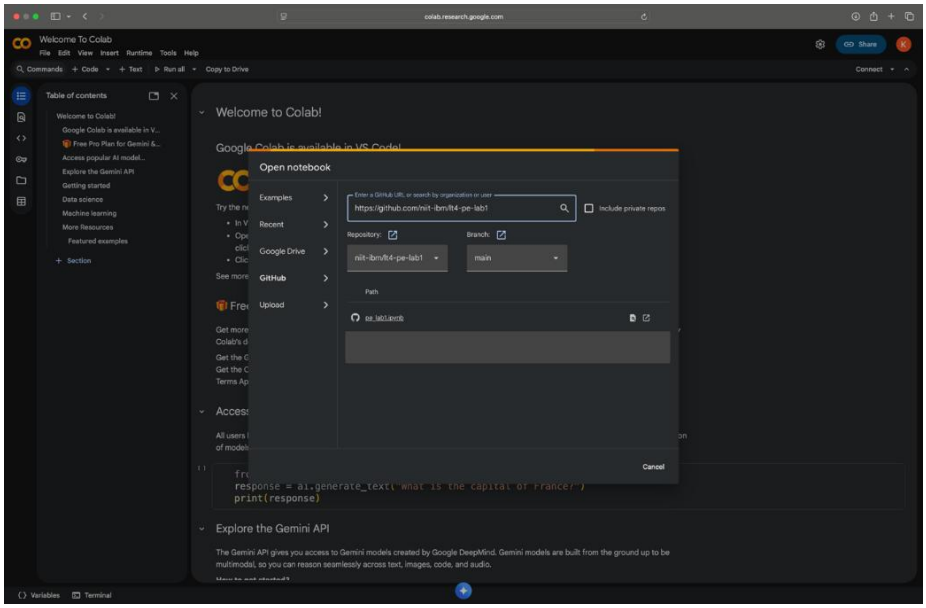
1. In your Google Colab workspace, select **File > Open notebook**.



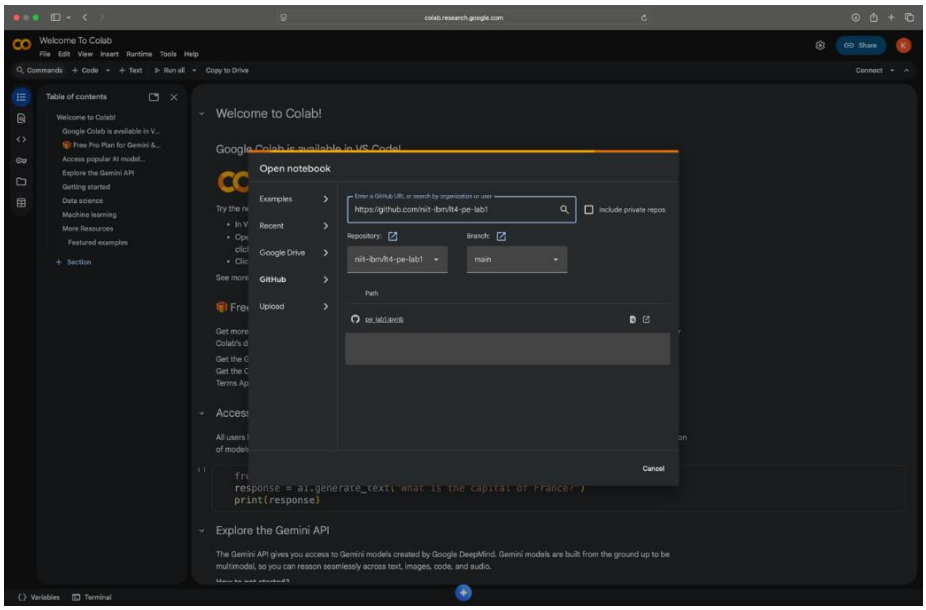
2. From the navigation menu, select **GitHub**.



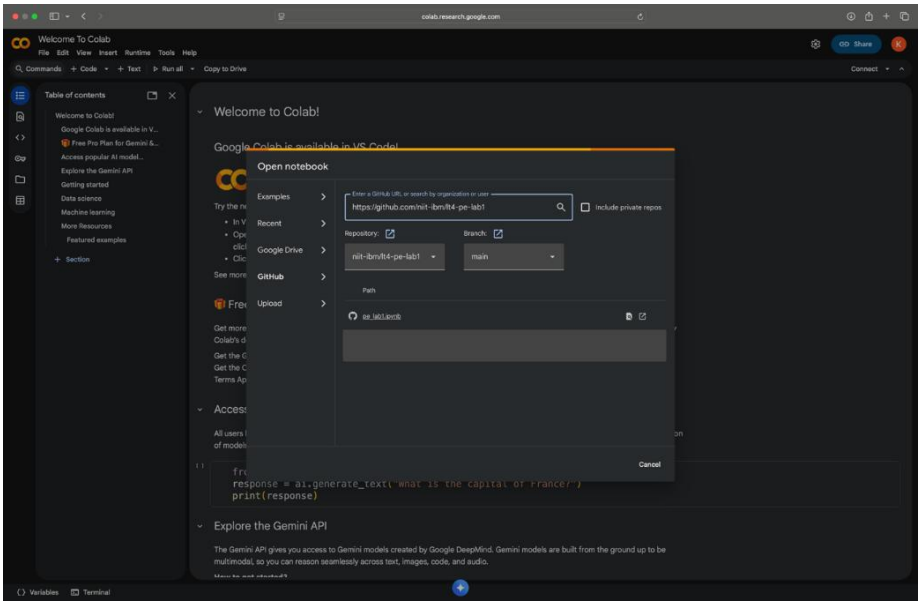
3. In the **Enter a GitHub URL or search by organization or user** field, copy and paste the following URL: https://github.com/niit-ibm/lt4-pe-lab1 and then, select the **search** icon.



4. In the Branch section, select **main**.



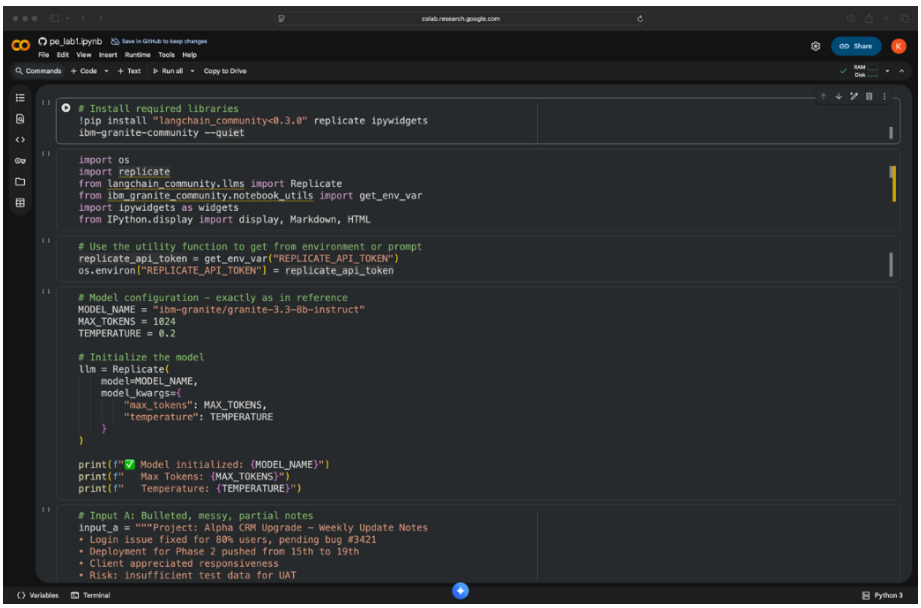
5. Select the **pe_lab1.ipynb** notebook in the Path section to open the notebook in Google Colab.



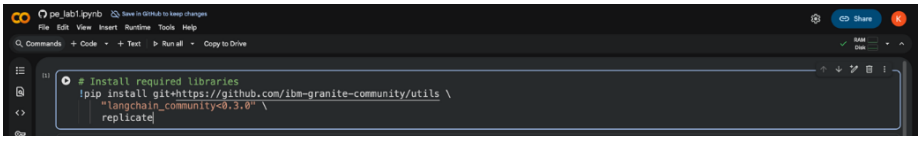
Result: A notebook titled **pe-lab1** that you need to use for this lab activity opens in the Colab Workspace. Note that each row in the notebook is referred to as a **cell**.

6. To run the code in the Jupyter notebook, you need to start a new instance in the Google Colab runtime. For this, select the **Connect** menu on the Google Colab navigation bar, and then, select the **Connect to a hosted runtime** option.

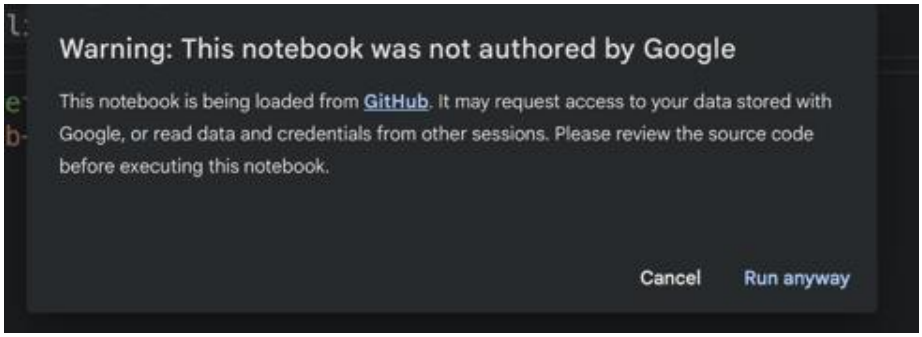
Result: A green checkmark indicates that you have successfully connected to the hosted runtime.



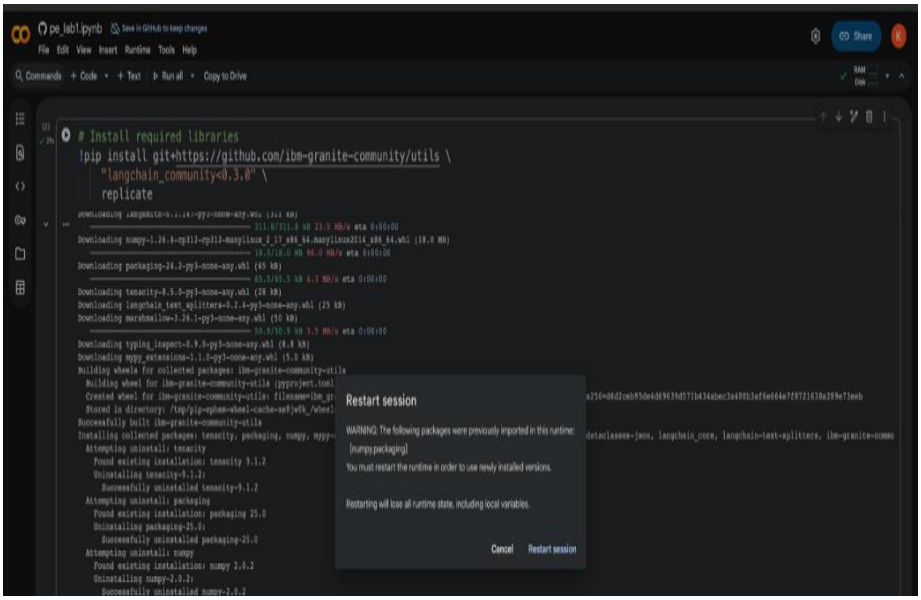
7. In the first cell of the notebook, select the **Play** button to install the required libraries from the Granite suite.



8. Select **Run anyway** to proceed loading the required libraries.

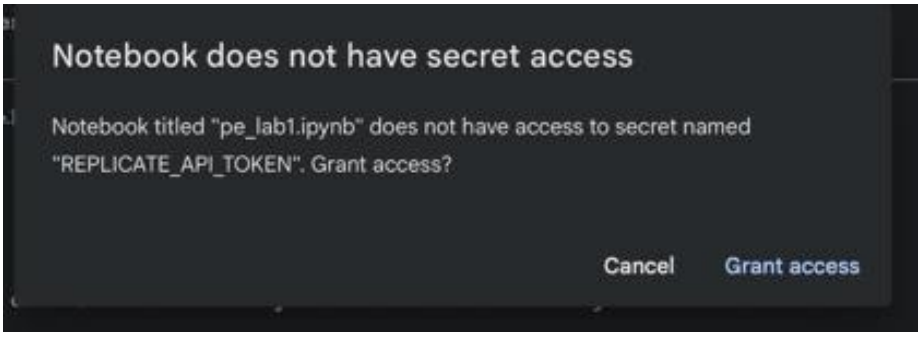


9. Once installation is complete you are prompted to restart the runtime session. Select **Restart session**.



10. Select **Run** to import the libraries.

11. From the next cell, select **Grant access** to let the model access the Replicate API Token from the Secrets.



12. The following message displays at the bottom of the cell: REPLICATE_API_TOKEN loaded from Google Colab secret.

13. Select the **Play** button to initialize the granite-3.3-8B-instruct model.

Load the input dataset and write a baseline prompt

Overview

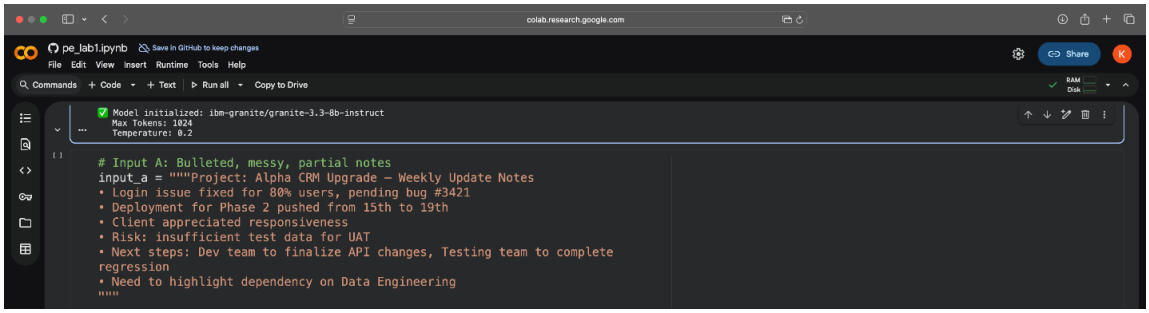
In this procedure, you'll upload the input dataset and write a baseline prompt or the initial instruction. You'll use the zero-shot prompting technique, a technique that includes providing minimal detail to generate an output from the LLM.

Instructions

1. Begin by uploading the inputs related to the project updates you've received from project leads. These inputs provide all the relevant project information to the Granite model. Without these inputs, the model might generate inconsistent or inaccurate project summaries.

   To do this, copy the following code, paste it into a new code cell, and then select the **Play** icon to execute it.

```
# Input A: Bulleted, messy, partial notes
Input_a= """Project: Alpha CRM Upgrade – Weekly Update Notes
Login issue fixed for 80% users, pending bug #3421
Deployment for Phase 2 pushed from 15th to 19th
Client appreciated responsiveness
Risk: insufficient test data for UAT
Next steps: Dev team to finalize API changes, Testing team to
complete regression
Need to highlight dependency on Data Engineering
"""
```



2. Next, enter the baseline prompt. To do this, copy the following code, paste it into a new code cell, and then select the **Play** icon to execute it.

```
    # Direct prompt
direct_prompt = "Write a 120-word weekly project update."
```

```
print("TASK 3: Direct Prompting")
print("*"*60)
print(f"Prompt: {direct_prompt}\n")

# Get response from model
direct_prompt_response = llm.invoke(direct_prompt)

print("Response:")
print("*"*60)
print(direct_prompt_response)
print("*"*60)
```

The following screenshot shows the code pasted into the code cell.

Result: After you enter the baseline prompt, the LLM generates an output as shown in the following screenshot.

Note that the output is insufficient; the model has not generated a structured project update. So, you will now refine the prompt.

Refine the prompt using task-specific details

Overview

In this procedure, you'll give the model task-specific prompts, that is, instructions designed to guide LLMs to perform a defined task. These prompts have a defined focus and often include additional context or examples to help the AI models generate accurate and relevant responses.

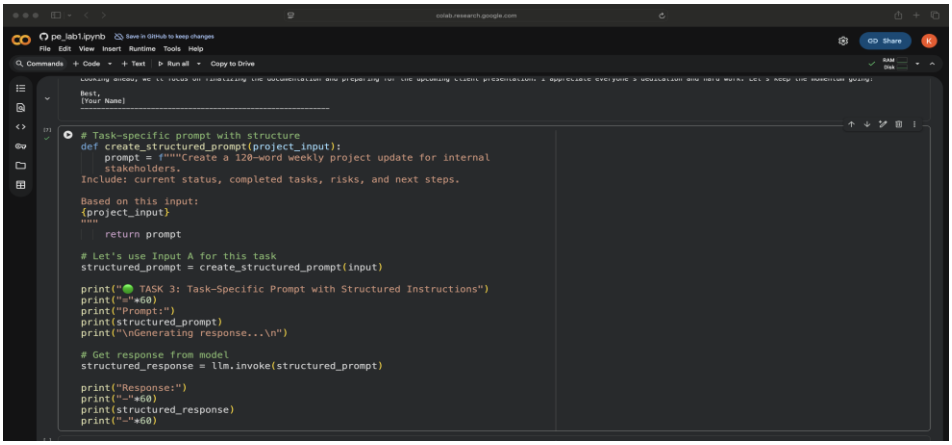
Instructions

1. Now refine the baseline prompt by adding some task-specific details such as the "current status" of the project and "risks" related to the project. To do this, copy the following code, paste it into a new code cell, and then select the **Play** icon to execute it.

```
# Task-specific prompt with structure
def create_structured_prompt(project_input):
    prompt = f"""Create a 120-word weekly project update for
internal
    stakeholders.
    Include: current status, completed tasks, risks, and next
steps.
   Based on this input:
    {project_input}
    """
    return prompt
# Let's use Input A for this task
structured_prompt = create_structured_prompt(input)

print("TASK 3: Task-Specific Prompt with Structured
Instructions")
print("="*60)
print("Prompt:")
print(structured_prompt)
print("\nGenerating response...\n")

# Get response from model
structured_response = llm.invoke(structured_prompt)

print("Response:")
print("="*60)
```

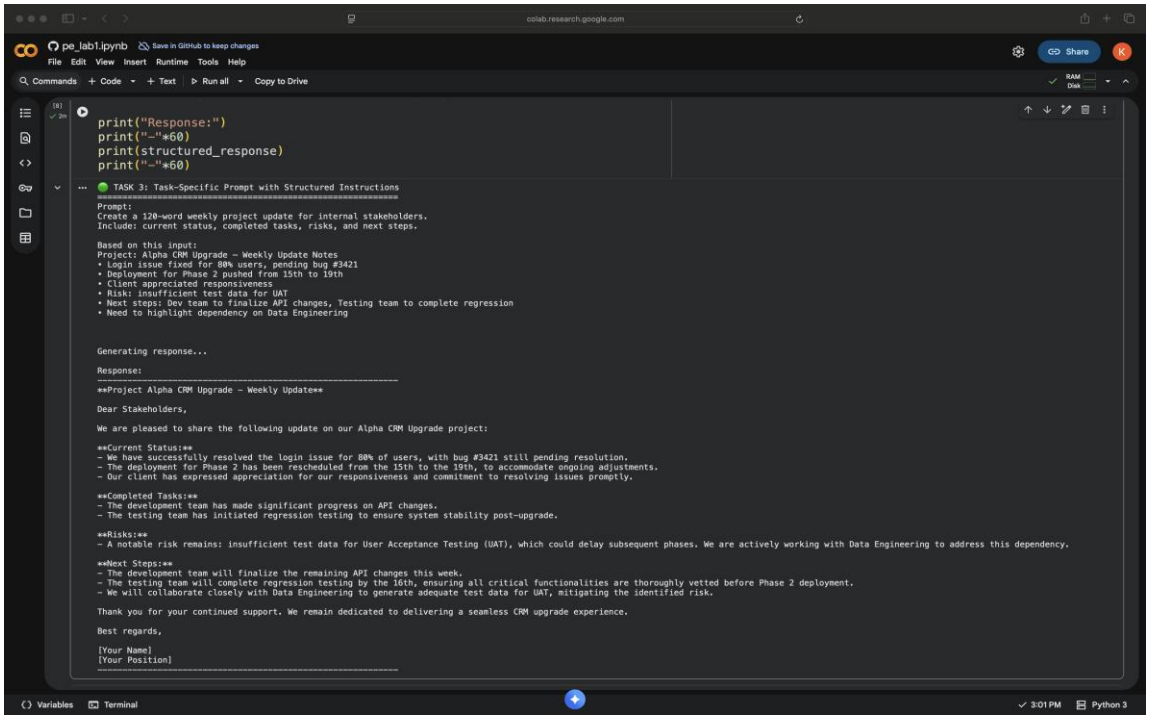
```
print(structured_response)
print("="*60)
```

Result: The following screenshot shows the code pasted into the code cell.



After you enter the refined prompt, the LLM generates an output as shown in the following screenshot.

Observe that the response is now more organized. But does it have the professional tone expected in stakeholder updates?



Refine the prompt using additional details

Overview

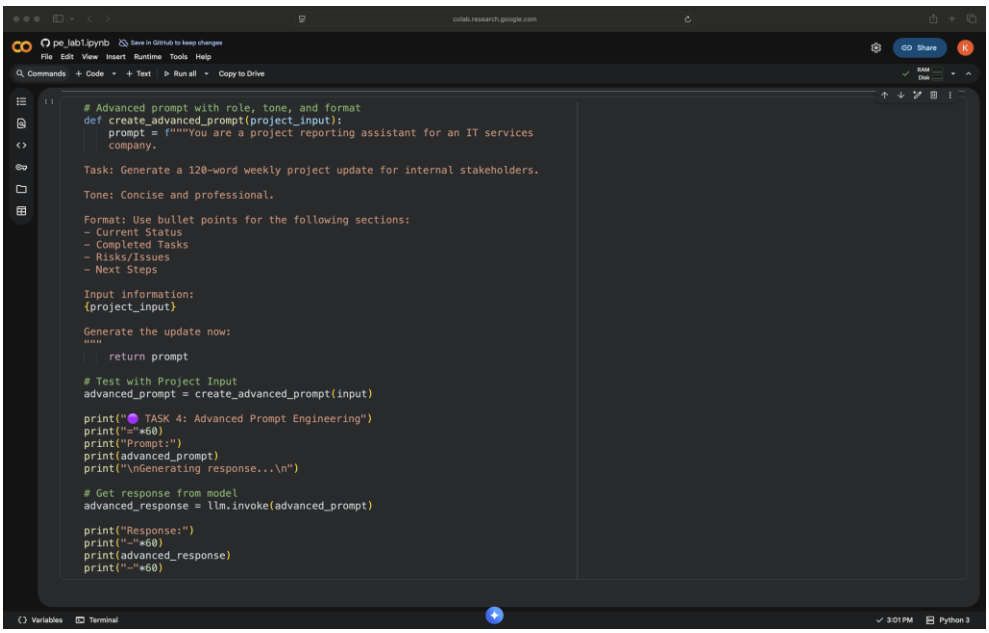
In this procedure, you'll further refine your prompt by adding additional details to define the tone and specify a structure of the required output.

Instructions

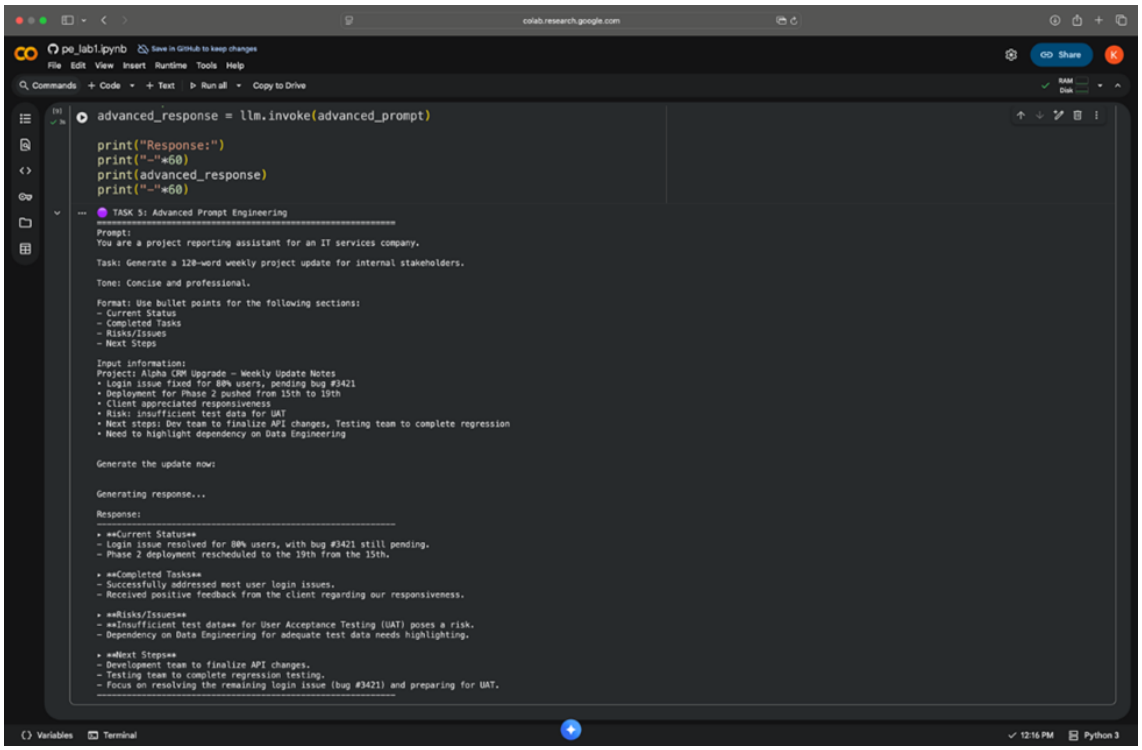
1. To refine the prompt, copy the following code, paste it into a new code cell, and then select the **Play** icon to execute it.

```
# Advanced prompt with role, tone, and format

def create_advanced_prompt(project_input):
    prompt = f"""You are a project reporting assistant for an IT
services
    company.
    Task: Generate a 120–word weekly project update for internal
stakeholders.
    Tone: Concise and professional.
    Format: Use bullet points for the following sections:
    – Current Status
    – Completed Tasks
    – Risks/Issues
    – Next Steps

    Input information:
    {project_input}

    Generate the update now:
    """
    return prompt

# Test with Project Input
advanced_prompt = create_advanced_prompt(input)
print("TASK 4: Advanced Prompt Engineering")
print("="*60)
print("Prompt:")
print(advanced_prompt)
print("\nGenerating response...\n")

# Get response from model
advanced_response = llm.invoke(advanced_prompt)
print("Response:")
print("="*60)
```

```
print(advanced_response)
print("="*60)
```



Result: The LLM generates an output as shown in the following screenshot. Observe that the response provides an update which is both professional and structured. The update includes the complete and relevant details of the status, completed tasks, risk, and next steps of the project.



Conclusion

Congratulations! You have learned how to write and refine prompts to generate specific, professional, and structured output from LLMs. By combining task description, context, output format, role, and tone, you produced polished, stakeholder-ready project updates. You have helped project management by providing them with consistent and systematically organized project updates.