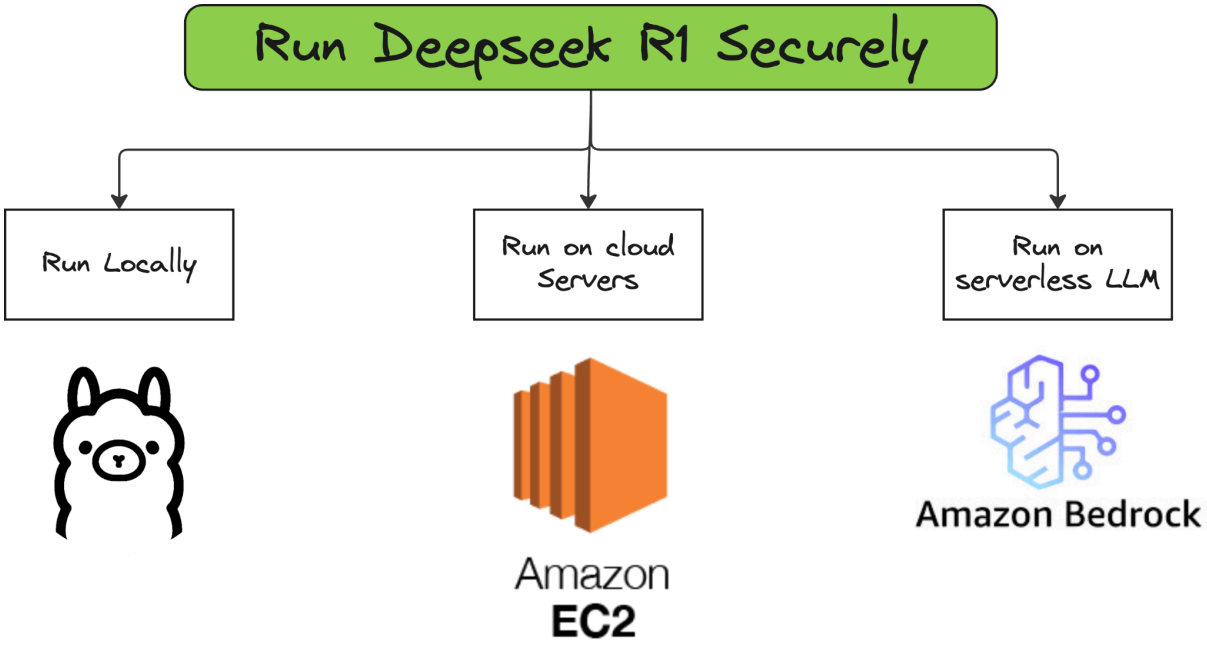
Run Deepseek R1 Securely

Introduction **Large Language Models (LLMs)** like **DeepSeek-R1** are transforming AI, but cloud-based APIs often come with costs, latency, and privacy concerns. Running models locally gives you full control, privacy, and customization. In this guide, you'll learn how to install and use DeepSeek-R1 locally using tools like Ollama and Python.

Why should we run DeepSeek R1 locally?

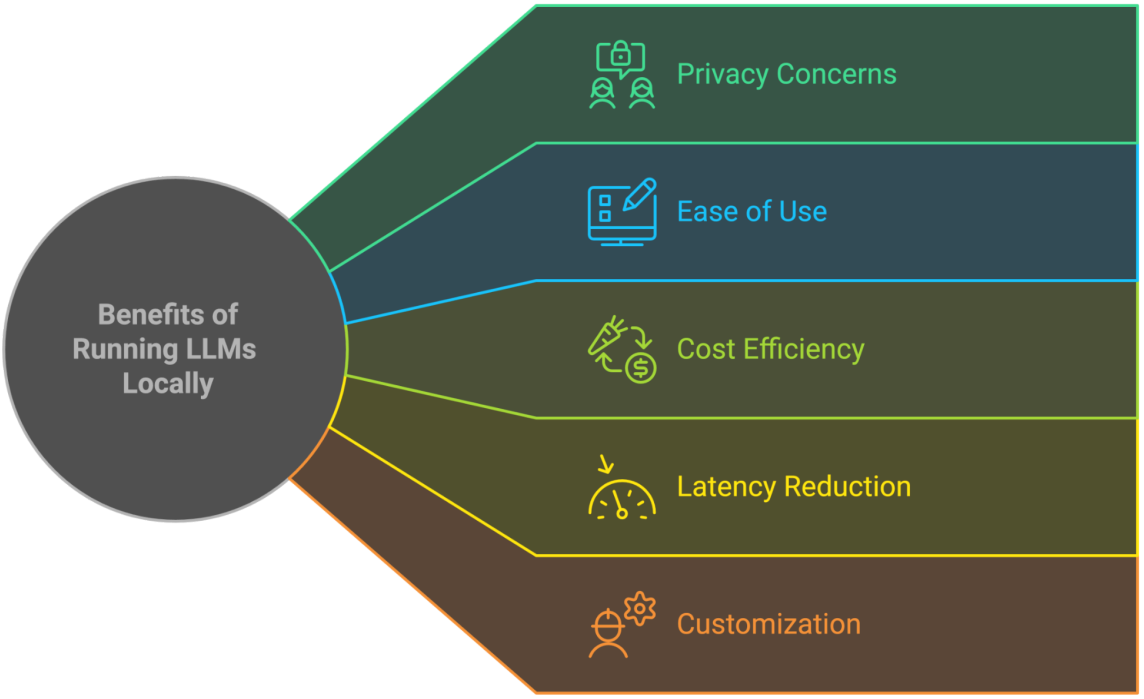
- Privacy & Data Security
- No Internet Dependency
- Performance & Latency
- Cost Savings
- Customization & Control
- Open-Source & Community Support

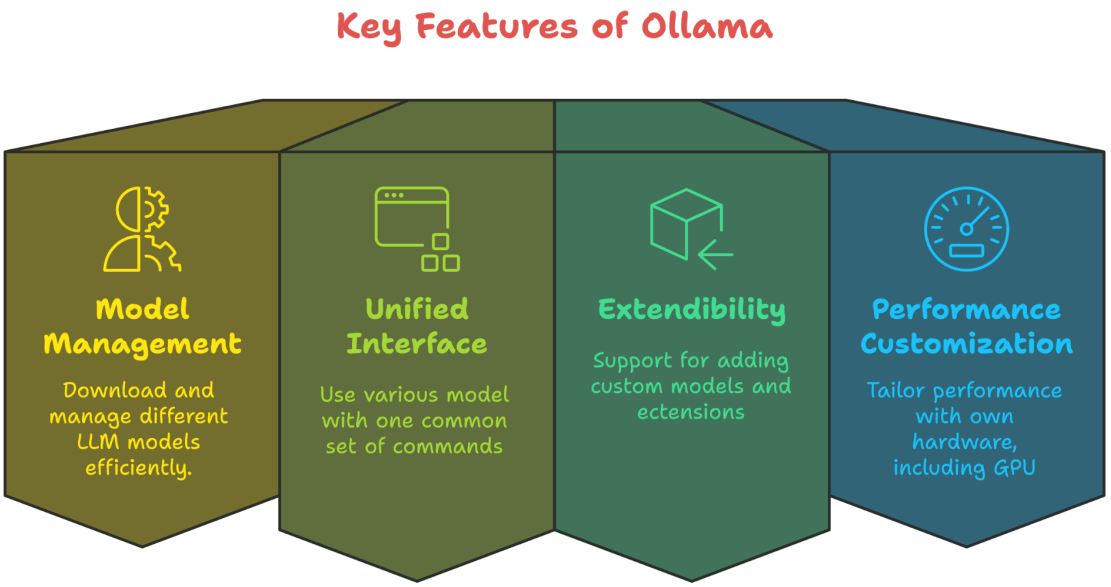


Vizuara AI Lab

Pritam Kudale

1. Run locally



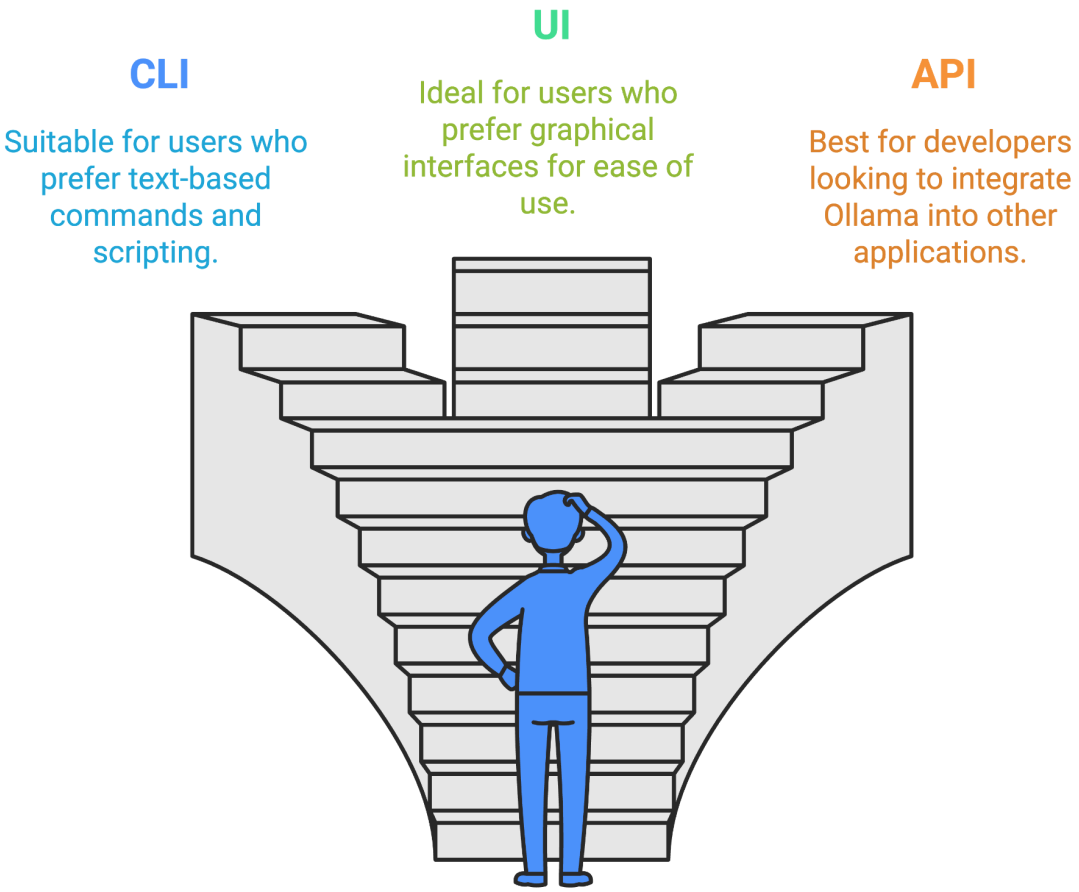


Prerequisites

Hardware:

- 16GB+ RAM (32GB recommended for larger models).
- GPU (NVIDIA with CUDA support) for faster inference

How to interact with Ollama and its models?



Step 1: Install Ollama (Common for all)

Ollama is a platform that enables efficient model execution on local devices.

For macOS & Linux

Open the terminal and run the following command:

```
curl -fsSL https://ollama.com/install.sh | sh
```

For Windows

Download and install Ollama from the official website: https://ollama.com/download

1.1 Run Deepseek Using Command Line Interface (CLI)

Step 2: Download the DeepSeek Model

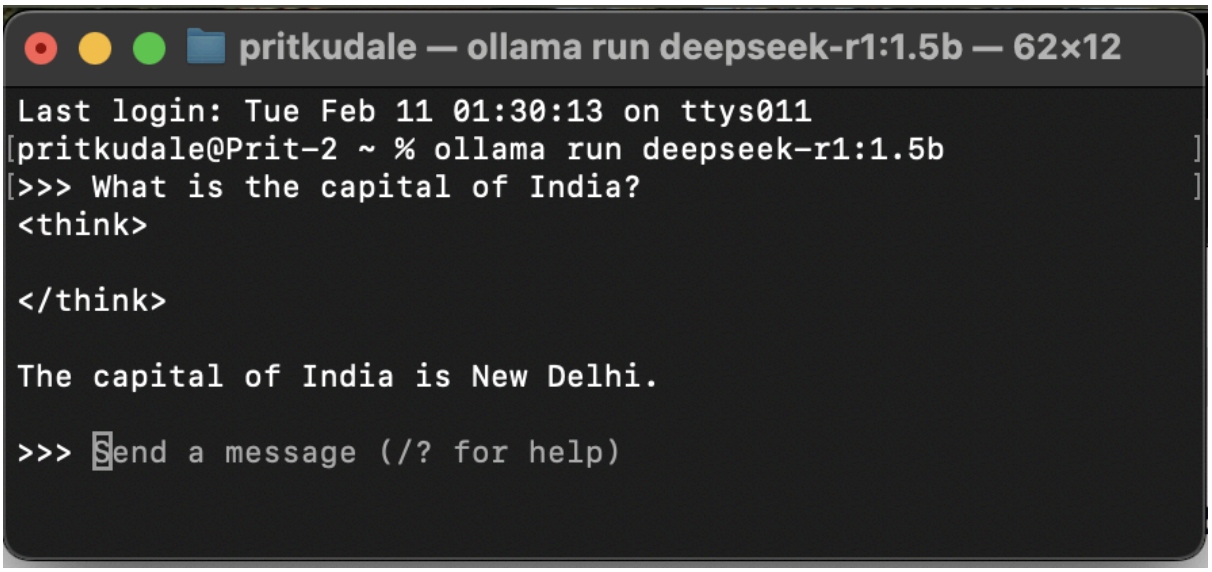
Once Ollama is installed, fetch the DeepSeek model by running in Terminal/PowerShell:

```
ollama run deepseek-r1:1.5b
```

Select a model based on hardware:

1.5B =~ 3.5GB RAM
7B =~ 16GB RAM
8B =~ 18GB RAM
14B =~ 32GB RAM
70B =~ 161GB RAM
671B =~ 1342GB RAM

Run Inference:



Reference Video: https://youtu.be/YFRch6ZaDeI

1.2 Run Deepseek Using User Interface (UI)

Step 2: Installation via Python pip 🐍

Open WebUI can be installed using pip, the Python package installer. Before proceeding, ensure you're using Python 3.11 to avoid compatibility issues.

1. Install Open WebUI: Open your terminal and run the following command to install Open WebUI:

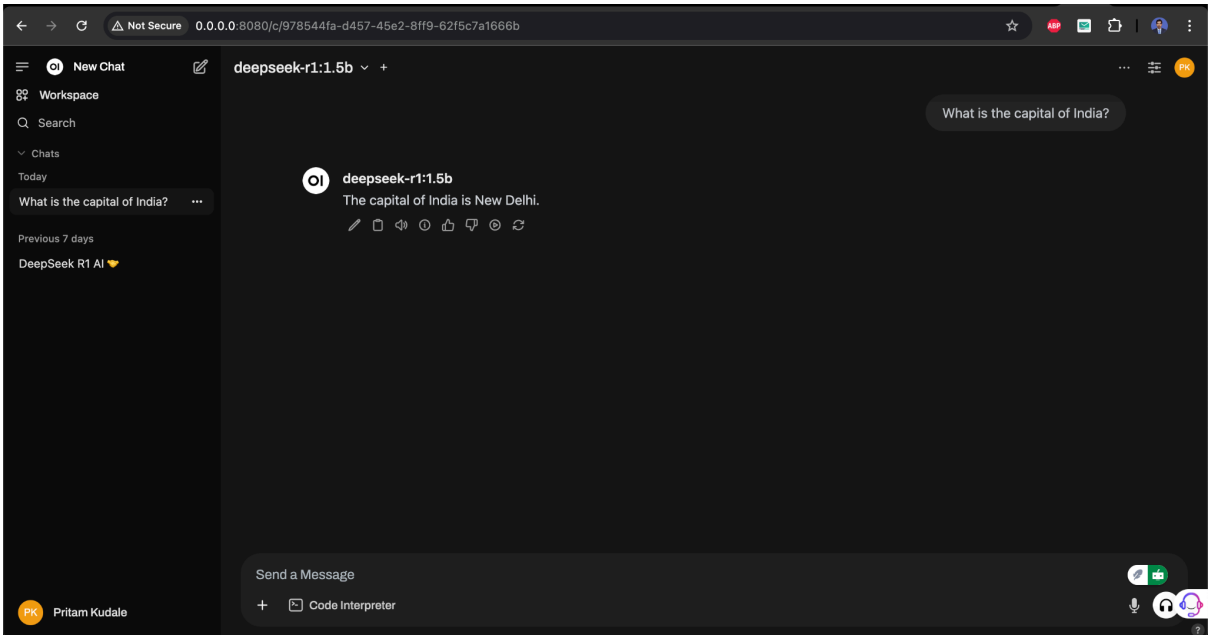
```
pip install open-webui
```

2. Running Open WebUI: After installation, you can start Open WebUI by executing:

```
open-webui serve
```

This will start the Open WebUI server, which you can access at

```
http://localhost:8080
```



Reference Video: https://youtu.be/YFRch6ZaDeI

1.3 Run Deepseek Using API

Step 2: Create a Python Program

```
#apifile.py
import requests
import json
url = 'http://localhost:11434/api/generate'
data = {
   "model": "deepseek-r1:1.5b",
   "prompt": "Tell me a short story.",}
response = requests.post(url, json=data, stream=True)
# check the response status
if response.status_code == 200:
   print("Generated Text:", end=" ", flush=True)
   # Iterate over the streaming response
   for line in response.iter_lines():
       if line:
           # Decode the line and parse the JSON
           decoded_line = line.decode("utf-8")
           result = json.loads(decoded_line)
           # Get the text from the response
           generated_text = result.get("response", "")
           print(generated_text, end="", flush=True)
else:
   print("Error:", response.status_code, response.text)
```

Step 3: Run in Terminal

```
python3 apifile.py
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   JUPYTER   COMMENTS

(venv) pritkudale@Prit-2 Ollama % python3 apifile.py
Generated Text: <think>

</think>

The capital of India is New Delhi.%
(venv) pritkudale@Prit-2 Ollama %
```

Reference Video: https://youtu.be/JiFeB2Q43hA

1.4 Run Deepseek Using API—Python Library

Step 2: Install necessary library

```
Pip install ollama
```

Step 3: Run the following code (Non-streaming Response)

```
res = ollama.chat(
   model="deepseek-r1:1.5b",
   messages=[
       {'role': 'user', 'content': 'What is the
capital of India?'}
   ]
)
print(res['message']['content'])
```

```
<think>

</think>

The capital of India is Delhi.
```

Step 3: Run the following code (Streaming Response)

```
res = ollama.chat(
   model="deepseek-r1:school",
   messages=[
       {'role': 'user', 'content': 'explain the gravity'}
   ],
   stream=True,
)
for chunk in res:
  print(chunk["message"]["content"], end="", flush=True)
```

Reference Video: https://youtu.be/JiFeB2Q43hA

2. Run Deepseek Using Amazon Bedrock

Step 1: Install necessary library

```
!pip install boto3
```

Step 2 : import necessary library

```
from huggingface_hub import snapshot_download
import boto3
import os
```

Step 3 : Choose Suitable model and download to Colab (Faster compared to local download)

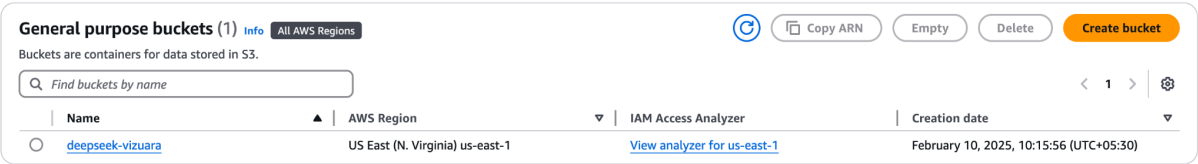
Choose model from https://huggingface.co/

- I used Deepseek R1 Distill-Llama-8B
- AWS supports Llama, Multimodal Llama, Mistral, Mixtral, and Flan

```
model_id='deepseek-ai/DeepSeek-R1-Distill-Llama-8B'
model_path = snapshot_download(repo_id=model_id,
local_dir='DeepSeek-R1-Distill-Llama-8B')
```

Step 4 : Link Colab file to AWS server

Create a bucket in AWS S3



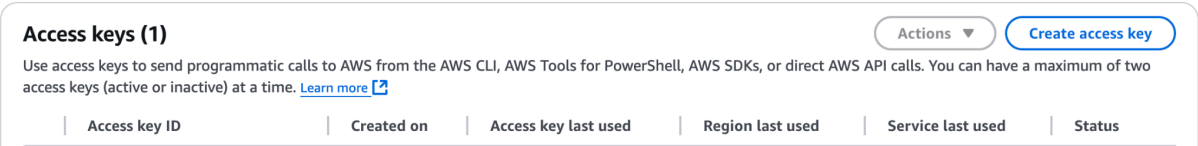
Get your bucket name: (replace with yours)

```
bucket_name = 'deepseek-vizuara'
```

Step 5 : Link Colab file to AWS server

Get your AWS credential from

IAM -> Security credentials



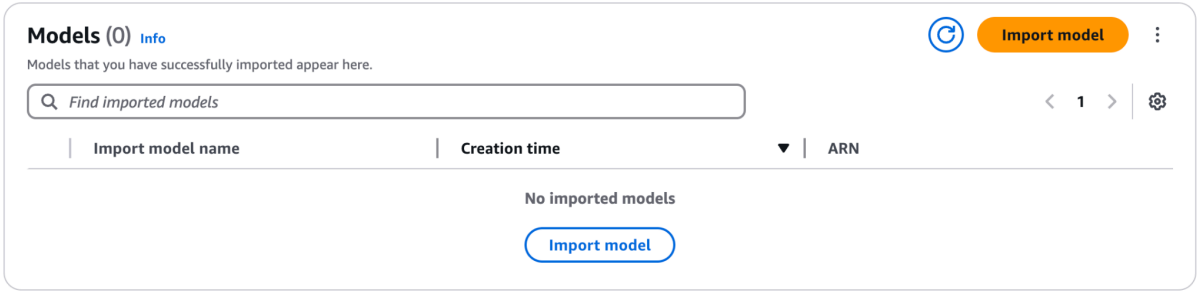
Credential Required: AWS_Access_key, AWS_Secret_key

Store in colab Secrets to access it securely

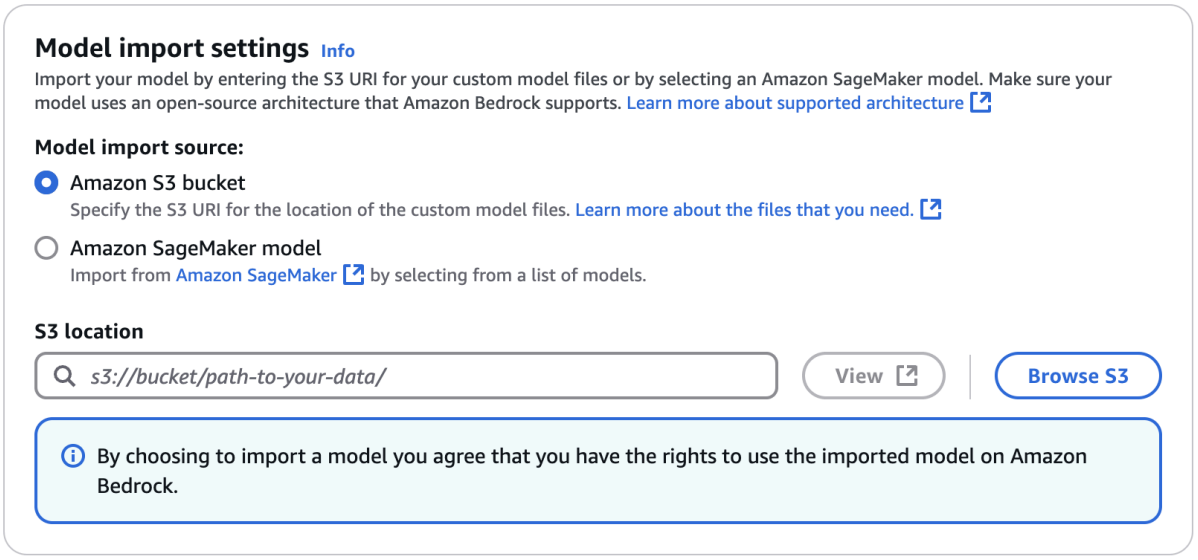
```
from google.colab import userdata
aws_access_key_id = userdata.get('AWS_Access_key')
aws_secret_access_key = userdata.get('AWS_Secret_key')
s3_client =
boto3.client('s3',region_name='us-east-1',aws_access_key_id=a
ws_access_key_id,
aws_secret_access_key=aws_secret_access_key)
bucket_name = 'deepseek-vizuara'
local_dir = 'DeepSeek-R1-Distill-Llama-8B'
for root,dir,files in os.walk(local_dir):
    for file in files:
        local_path=os.path.join(root,file)
        s3_key = os.path.relpath(local_path,local_dir)

s3_client.upload_file(local_path,bucket_name,local_dir+'/'+s3
_key)
        print(f'Uploaded {file} to
s3://{bucket_name}/{os.path.join(local_dir,file)}')
```

Step 6 : Import model in AWS Bedrock

Amazon Bedrock -> Imported models



- Select Import model and fill in the accessory names
- Import the model using S3 bucket
- Keep remaining option as default
- Hit import model button



Step 7 : Use in playground or use in python

```
import boto3
import json
model_arn = #Add your model arn
prompt = "What is the capital of France?"
brt =
boto3.client(service_name='bedrock-runtime',region_name='us-east-1
',aws_access_key_id=aws_access_key_id,
aws_secret_access_key=aws_secret_access_key)
body = json.dumps({
    'prompt': prompt,
    'max_tokens_to_sample': 4000 })
response = brt.invoke_model_with_response_stream(
    modelId=model_arn,
    body=body)
stream = response.get('body')
if stream:
    for event in stream:
        chunk = event.get('chunk')
        if chunk:
print(json.loads(chunk.get('bytes').decode())['generation'],end=''
)
```

Reference Video: https://youtu.be/WzzMgvbSKtU