Comprehensive Local LLM and Chatbot Development Project

Part 1: Local LLM Setup and Interaction

1.1 Set up Llama3 -

1. From GitHub repository - https://github.com/ollama/ollama?tab=readme-ov-file - installed for Mac
2. In terminal - installed the Ollama -prompt- brew install ollama
3. To verify the installation - ollama list -

NAME ID SIZE MODIFIED
llama3.1:latest 42182419e950 4.7 GB 5 minutes ago

4. Verifying the installation and running a simple query - ollama run llama2 "Hellow, how are you?"

Response - Hello! I'm just an AI assistant, so I don't have feelings or emotions like humans do. However, I'm here to help you with any questions or tasks you may have. How can I assist you today?

1.2 Choose and Set Up a Second LLM - Mistral

1. cd "/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2"
2. source myenv/bin/activate
3. ollama pull mistral

Model Ollama llama3 - Commands for llama3

- cd "/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2"
- source myenv/bin/activate
- ollama pull llama3
- ls
- cd src
- ls
- python llama3_script.py
- deactivate

1.3 Command-Line Interaction Using Curl
Responses received & Screeshots

a. Send a Simple Query

llama3_script.py UNREGISTERED

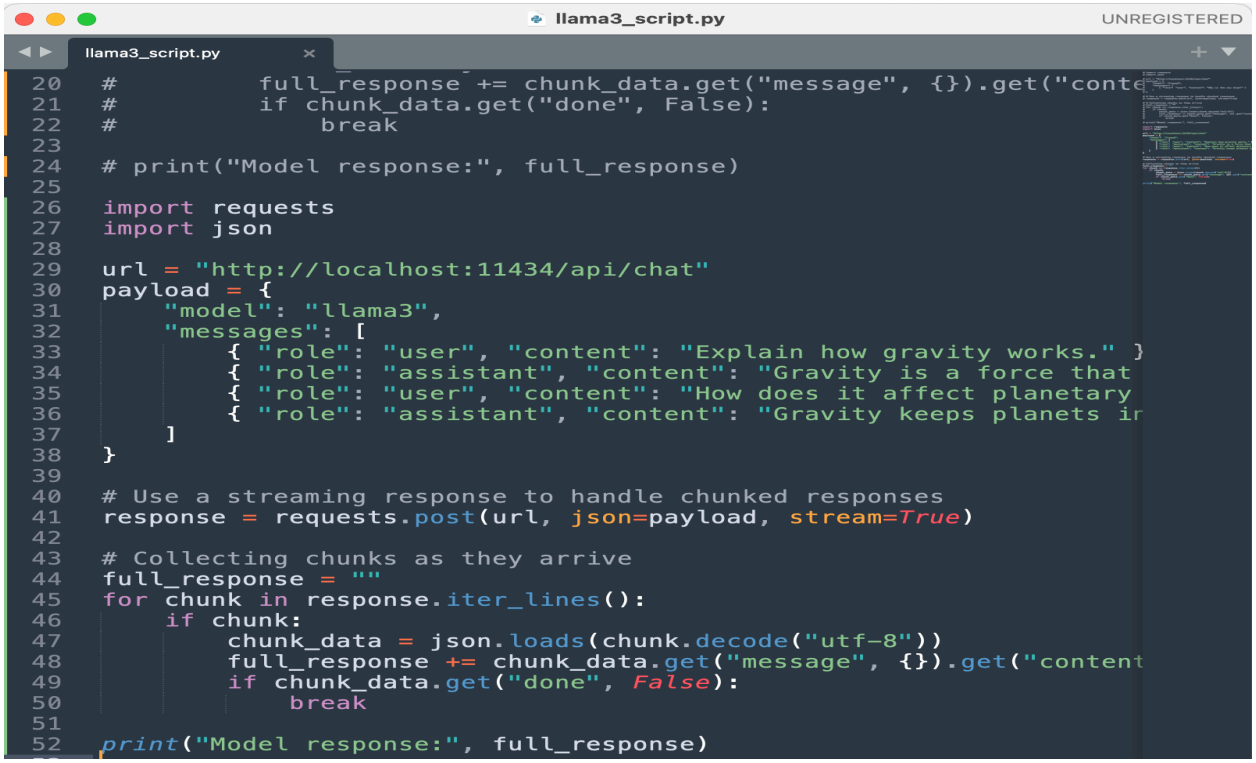
llama3_script.py

```python
import requests
import json

url = "http://localhost:11434/api/chat"
payload = {
    "model": "llama3",
    "messages": [
        { "role": "user", "content": "Why is the sky blue?" }
    ]
}

# Use a streaming response to handle chunked responses
response = requests.post(url, json=payload, stream=True)

# Collecting chunks as they arrive
full_response = ""
for chunk in response.iter_lines():
    if chunk:
        chunk_data = json.loads(chunk.decode("utf-8"))
        full_response += chunk_data.get("message", {}).get("content
        if chunk_data.get("done", False):
            break

print("Model response:", full_response)
```

src — -zsh — 159×54

```
Last login: Sat Sep 28 17:00:10 on ttys000
nehagopinath@Nehas-MacBook-Air ~ % -     cd "/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2"
nehagopinath@Nehas-MacBook-Air Assignment 2 % source myenv/bin/activate

(myenv) nehagopinath@Nehas-MacBook-Air Assignment 2 % ollama pull llama3

pulling manifest
pulling 6a0746a1ec1a... 100% ████████████████  4.7 GB
pulling 4fa551d4f938... 100% ████████████████   12 KB
pulling 8ab4849b038c... 100% ████████████████  254 B
pulling 577073ffcc6c... 100% ████████████████  110 B
pulling 3f8eb4da87fa... 100% ████████████████  485 B
verifying sha256 digest
writing manifest
success
(myenv) nehagopinath@Nehas-MacBook-Air Assignment 2 % ls
myenv   src
(myenv) nehagopinath@Nehas-MacBook-Air Assignment 2 % cd src
(myenv) nehagopinath@Nehas-MacBook-Air src % ls
llama3_script.py        llms_cript.py
(myenv) nehagopinath@Nehas-MacBook-Air src % python llama3_script.py
/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2/myenv/lib/python3.9/site-packages/urllib3/__init__.py:35: NotOpenSSLWarning: urllib3 v2 only supports
 OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Model response: What a great question!

The short answer is that the sky appears blue because of a phenomenon called scattering, which occurs when sunlight interacts with tiny molecules of gases in t
he Earth's atmosphere.

Here's a more detailed explanation:

1. **Sunlight**: When the sun emits light, it produces a broad spectrum of colors, including all the colors of the rainbow (red, orange, yellow, green, blue, i
ndigo, and violet).
2. **Atmospheric gases**: The Earth's atmosphere is made up of various gases, such as nitrogen (N2), oxygen (O2), and trace amounts of other gases like carbon
dioxide (CO2) and water vapor (H2O). These gases are much smaller than the wavelength of light.
3. **Scattering**: When sunlight enters the Earth's atmosphere, it encounters these tiny gas molecules. The shorter, blue wavelengths of light are scattered mo
re intensely by the smaller molecules than the longer, red wavelengths. This is because the smaller molecules are more effective at scattering the shorter wave
lengths.
4. **Blue dominance**: As a result of this scattering, the blue light is distributed evenly throughout the sky, giving it its characteristic blue color. The ot
her colors, like red and orange, continue to travel in a more direct path to our eyes, appearing less intense and more focused towards the sun.
5. **Atmospheric conditions**: The exact shade of blue can vary depending on atmospheric conditions, such as:
        * Aerosols: Tiny particles in the air, like pollutants or dust, can scatter light in different ways, changing the apparent color of the sky.
        * Water vapor: High humidity can cause more scattering, leading to a deeper blue or even a purplish hue.
        * Dust and pollution: These can reduce the amount of blue light that reaches our eyes, making the sky appear more hazy or gray.

In summary, the sky appears blue because the shorter wavelengths of sunlight are scattered by tiny atmospheric molecules, giving them an even distribution thro
ughout the sky. The color we see is a result of this scattering and the specific conditions in the atmosphere.
(myenv) nehagopinath@Nehas-MacBook-Air src %
```

b. Send a Complex, Multi-turn Conversation




```
(myenv) nehagopinath@Nehas-MacBook-Air src % python llama3_script.py
/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2/myenv/lib/python3.9/site-packages/urllib3/__init__.py:35: NotOpenSSLWarning: urllib3 v2 only supports
 OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Model response:  Here's why:

1. **Attraction**: Every planet, moon, and even you have mass (amount of matter). According to the theory of gravity proposed by Sir Isaac Newton, any two obje
cts with mass attract each other.
2. **Force**: The force of gravity between two objects depends on their masses and the distance between them. This force is proportional to the product of thei
r masses and inversely proportional to the square of the distance between them (F ∝ m1 × m2 ÷ r^2).
3. **Orbital motion**: When a planet, like Earth, has enough mass and speed, it moves into orbit around its star (like the Sun). Gravity pulls the planet towar
ds the star, but the planet's velocity keeps it from falling in.
4. **Balancing forces**: The force of gravity pulling the planet towards the star is balanced by the centrifugal force, which pushes the planet away from the c
enter due to its orbital motion. This balance keeps the planet moving in a stable orbit.
5. **Shape of orbits**: Gravity also determines the shape of planetary orbits. For example, planets with low velocities will have elliptical orbits (more curve
d), while those with higher velocities will have more circular orbits.

In summary, gravity plays a crucial role in keeping planets in orbit around stars by balancing the force pulling them towards the star with their orbital motio
n and velocity.

Would you like to know more about gravity or planetary motion?
(myenv) nehagopinath@Nehas-MacBook-Air src %
```

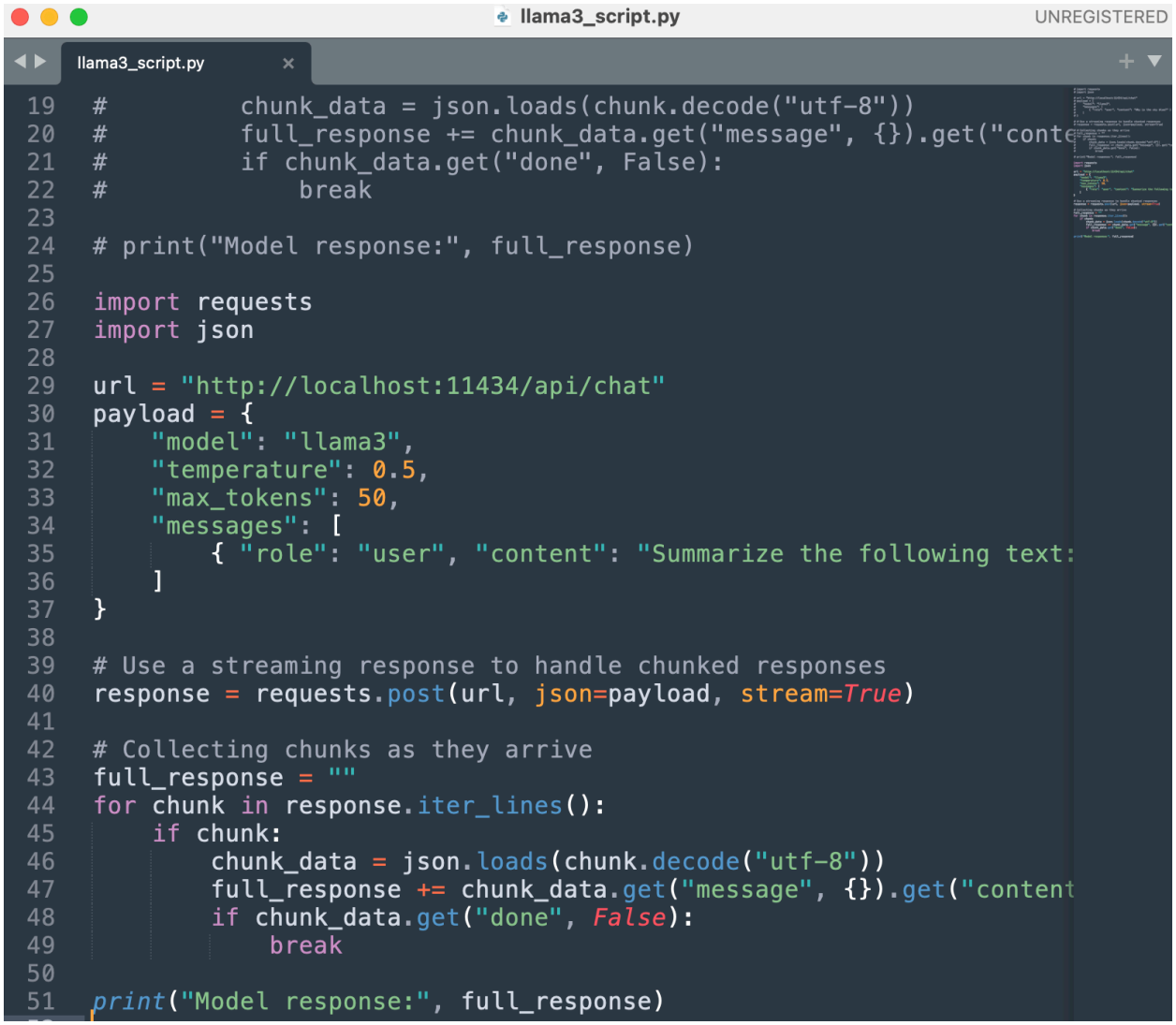
c. Experiment with Different Parameters (e.g., temperature, max_tokens)

llama3_script.py — UNREGISTERED

```python
#         full_response += chunk_data.get("message", {}).get("conte
#         if chunk_data.get("done", False):
#             break

# print("Model response:", full_response)

import requests
import json

url = "http://localhost:11434/api/chat"
payload = {
    "model": "llama3",
    "temperature": 0.7,
    "max_tokens": 100,
    "messages": [
        { "role": "user", "content": "Tell me a short story about a
    ]
}

# Use a streaming response to handle chunked responses
response = requests.post(url, json=payload, stream=True)

# Collecting chunks as they arrive
full_response = ""
for chunk in response.iter_lines():
    if chunk:
        chunk_data = json.loads(chunk.decode("utf-8"))
        full_response += chunk_data.get("message", {}).get("content
        if chunk_data.get("done", False):
            break

print("Model response:", full_response)
```

```
(myenv) nehagopinath@Nehas-MacBook-Air src % python llama3_script.py
/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2/myenv/lib/python3.9/site-packages/urllib3/__init__.py:35: NotOpenSSLWarning: urllib3 v2 only supports
 OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Model response: In the land of Everia, where the sun dipped into the horizon and painted the sky with hues of crimson and gold, there lived a brave knight name
d Sir Valoric. He was known throughout the realm for his unwavering courage in the face of danger, his unshakeable honor, and his unyielding commitment to just
ice.

One day, a dragon named Tharros began to terrorize the countryside, breathing fire that burned crops and villages alike. The people of Everia lived in fear of
the beast's wrath, and none dared to stand against it.

Sir Valoric, however, was not one to shy away from battle. He donned his armor, grasped his sword, and set out to face the dragon. As he approached Tharros' la
ir, the air grew thick with heat and smoke, and the ground trembled beneath his feet.

The knight dismounted his horse and walked forward, his eyes fixed on the beast. "I am Sir Valoric, defender of the realm," he declared, his voice steady and s
trong. "I come to vanquish you, Tharros, and restore peace to our land."

Tharros laughed, a deafening roar that shook the earth. Flames danced in its eyes as it took a step closer, its claws snapping with deadly precision.

Sir Valoric drew his sword, the blade shining like a beacon of hope. With a mighty cry, he charged forward, striking true and unleashing a torrent of attacks a
gainst the dragon. The battle raged on for what seemed like hours, the two combatants clashing in a whirlwind of steel and scales.

In the end, it was Sir Valoric who emerged victorious, his sword raised high as Tharros lay defeated at his feet. As the people of Everia celebrated their deli
verance, the knight stood tall, his armor dented but unbroken, his heart filled with pride and his spirit unshaken.

From that day on, Sir Valoric was hailed as a hero, his name whispered in reverence throughout the land. And though Tharros never again threatened the people o
f Everia, the bravery and honor of Sir Valoric remained an inspiration to all who knew him.
(myenv) nehagopinath@Nehas-MacBook-Air src %
```

d. Test the Model's Performance on Specific Tasks




```
import requests
import json

url = "http://localhost:11434/api/chat"
payload = {
  "model": "llama3",
  "temperature": 0.5,
  "max_tokens": 50,
  "messages": [
    { "role": "user", "content": "Summarize the following text: 'Artificial intelligence is
rapidly growing and changing the landscape of various industries. It is used in fields
```

```
such as healthcare, finance, and education to improve decision-making and
efficiency.'" }
  ]
}

# Use a streaming response to handle chunked responses
response = requests.post(url, json=payload, stream=True)

# Collecting chunks as they arrive
full_response = ""
for chunk in response.iter_lines():
  if chunk:
    chunk_data = json.loads(chunk.decode("utf-8"))
    full_response += chunk_data.get("message", {}).get("content", "")
    if chunk_data.get("done", False):
      break

print("Model response:", full_response)
```

```
(myenv) nehagopinath@Nehas-MacBook-Air src % python llama3_script.py
/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2/myenv/lib/python3.9/site-packages/urllib3/__init__.py:35: NotOpenSSLWarning: urllib3 v2 only supports
 OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Model response: Here is a summary of the text:

Artificial intelligence (AI) is transforming many industries by enhancing decision-making and efficiency. Specifically, it's being applied in areas like health
care, finance, and education.
(myenv) nehagopinath@Nehas-MacBook-Air src %
```

Second Model Mistral - Commands for Mistral

- cd "/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2"
- source myenv/bin/activate
- ollama pull mistral
- ls
- cd src
- ls
- python Mistral_llms_cript.py
- deactivate

Responses received & Screeshots

a. Send a Simple Query

```
curl http://localhost:11434/api/chat -d '{
 "model": "mistral",
 "messages": [
  { "role": "user", "content": "What is the capital of Japan?" }
 ]
}'
```

```
(myenv) nehagopinath@Nehas-MacBook-Air Assignment 2 % curl http://localhost:11434/api/chat -d '{
  "model": "mistral",
  "messages": [
    { "role": "user", "content": "What is the capital of Japan?" }
  ]
}'
```


```
{"model":"mistral","created_at":"2024-09-28T22:22:28.208921Z","message":{"role":"assistant","content":" The"},"do
ne":false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.26048Z","message":{"role":"assistant","content":" capital"},
"done":false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.323267Z","message":{"role":"assistant","content":" of"},"don
e":false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.377065Z","message":{"role":"assistant","content":" Japan"},"
done":false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.433364Z","message":{"role":"assistant","content":" is"},"don
e":false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.486225Z","message":{"role":"assistant","content":" Tokyo"},"
done":false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.537638Z","message":{"role":"assistant","content":"."},"done"
:false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.592059Z","message":{"role":"assistant","content":" It"},"don
e":false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.647179Z","message":{"role":"assistant","content":"'"},"done"
:false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.69869Z","message":{"role":"assistant","content":"s"},"done":
false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.750523Z","message":{"role":"assistant","content":" one"},"do
ne":false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.802524Z","message":{"role":"assistant","content":" of"},"don
e":false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.85552Z","message":{"role":"assistant","content":" the"},"don
e":false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.907743Z","message":{"role":"assistant","content":" "},"done"
:false}
{"model":"mistral","created_at":"2024-09-28T22:22:28.958817Z","message":{"role":"assistant","content":"4"},"done"
:false}
{"model":"mistral","created_at":"2024-09-28T22:22:29.01248Z","message":{"role":"assistant","content":"7"},"done":
false}
{"model":"mistral","created_at":"2024-09-28T22:22:29.06686Z","message":{"role":"assistant","content":" pre"},"don
e":false}
{"model":"mistral","created_at":"2024-09-28T22:22:29.121381Z","message":{"role":"assistant","content":"fect"},"do
ne":false}
```

a. Send a Simple Query

Mistral_llms_cript.py — UNREGISTERED

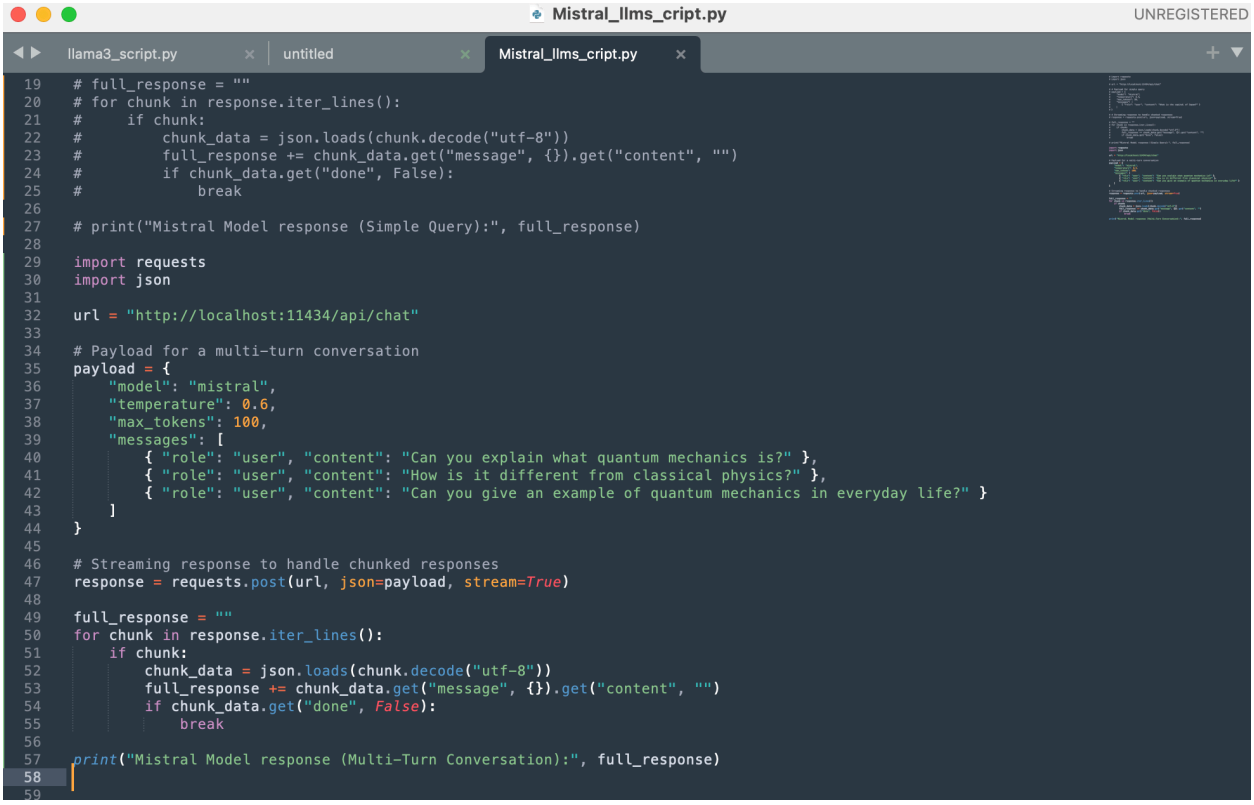
llama3_script.py | untitled | Mistral_llms_cript.py

```python
import requests
import json

url = "http://localhost:11434/api/chat"

# Payload for simple query
payload = {
    "model": "mistral",
    "temperature": 0.5,
    "max_tokens": 50,
    "messages": [
        { "role": "user", "content": "What is the capital of Japan?" }
    ]
}

# Streaming response to handle chunked responses
response = requests.post(url, json=payload, stream=True)

full_response = ""
for chunk in response.iter_lines():
    if chunk:
        chunk_data = json.loads(chunk.decode("utf-8"))
        full_response += chunk_data.get("message", {}).get("content", "")
        if chunk_data.get("done", False):
            break

print("Mistral Model response (Simple Query):", full_response)
```

src — -zsh — 113×54

```
nehagopinath@Nehas-MacBook-Air src % -  cd "/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2"
nehagopinath@Nehas-MacBook-Air Assignment 2 % source myenv/bin/activate
(myenv) nehagopinath@Nehas-MacBook-Air Assignment 2 % ollama pull mistral

pulling manifest
pulling ff82381e2bea... 100% ▕████████████████████████████████████████████████▏ 4.1 GB
pulling 43070e2d4e53... 100% ▕████████████████████████████████████████████████▏  11 KB
pulling 491dfa501e59... 100% ▕████████████████████████████████████████████████▏ 801 B
pulling ed11eda7790d... 100% ▕████████████████████████████████████████████████▏  30 B
pulling 42347cd80dc8... 100% ▕████████████████████████████████████████████████▏ 485 B
verifying sha256 digest
writing manifest
success
(myenv) nehagopinath@Nehas-MacBook-Air Assignment 2 % ls
myenv   src
(myenv) nehagopinath@Nehas-MacBook-Air Assignment 2 % cd src
(myenv) nehagopinath@Nehas-MacBook-Air src % ls
Mistral_llms_cript.py    llama3_script.py        llms_cript.py           myenv
(myenv) nehagopinath@Nehas-MacBook-Air src % python Mistral_llms_cript.py
/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2/myenv/lib/python3.9/site-packages/urllib3/__init__.py:3
5: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'Libre
SSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Mistral Model response (Simple Query):  The capital city of Japan is Tokyo. However, Tokyo is not the constitutio
nal capital of Japan; instead, Kyoto was its imperial capital from 794 to 1868. In 1868, after the Meiji Restorat
ion, the administrative capital was moved to Edo (now Tokyo). The imperial capital remained in Kyoto until 1945 w
hen it was moved back to Tokyo. Tokyo serves as both a political and economic hub for Japan.
(myenv) nehagopinath@Nehas-MacBook-Air src %
```

b. Send a Complex, Multi-Turn Conversation




```python
# full_response = ""
# for chunk in response.iter_lines():
#     if chunk:
#         chunk_data = json.loads(chunk.decode("utf-8"))
#         full_response += chunk_data.get("message", {}).get("content", "")
#         if chunk_data.get("done", False):
#             break

# print("Mistral Model response (Simple Query):", full_response)

import requests
import json

url = "http://localhost:11434/api/chat"

# Payload for a multi-turn conversation
payload = {
    "model": "mistral",
    "temperature": 0.6,
    "max_tokens": 100,
    "messages": [
        { "role": "user", "content": "Can you explain what quantum mechanics is?" },
        { "role": "user", "content": "How is it different from classical physics?" },
        { "role": "user", "content": "Can you give an example of quantum mechanics in everyday life?" }
    ]
}

# Streaming response to handle chunked responses
response = requests.post(url, json=payload, stream=True)

full_response = ""
for chunk in response.iter_lines():
    if chunk:
        chunk_data = json.loads(chunk.decode("utf-8"))
        full_response += chunk_data.get("message", {}).get("content", "")
        if chunk_data.get("done", False):
            break

print("Mistral Model response (Multi-Turn Conversation):", full_response)

```

```
(myenv) nehagopinath@Nehas-MacBook-Air src % python Mistral_llms_cript.py
/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2/myenv/lib/python3.9/site-packages/urllib3/__init__.py:3
5: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'Libre
SSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Mistral Model response (Multi-Turn Conversation):  Quantum mechanics is a fundamental theory in physics that prov
ides a description of the physical properties of nature at the scale of atoms and subatomic particles. It is most
 successfully used to explain chemical bonds, molecular structure, and phenomena associated with light (electroma
gnetic radiation).

The key difference between quantum mechanics and classical physics lies in how they treat energy levels and parti
cles' behavior. In classical physics, particles are considered as continuous objects that can be found anywhere w
ithin a certain volume of space, while in quantum mechanics, particles can only exist in discrete, quantized ener
gy levels and their positions are probabilistic.

For example, consider the famous double-slit experiment in quantum mechanics. If you shine single photons (partic
les of light) through two slits and observe them on a screen, they will create an interference pattern that seems
 to suggest they pass through both slits simultaneously — a phenomenon known as wave-particle duality. However, i
f you add a detector to determine which slit the particle passes through, the particles start behaving like ordin
ary particles and form two distinct bands on the screen. This demonstrates the principle of observation affecting
 reality (also called the observer effect) that is unique to quantum mechanics.

As for an example of quantum mechanics in everyday life, consider lasers — devices that produce coherent light by
 stimulating emission of photons from atoms. The behavior of these photons is governed by quantum mechanical prin
ciples. Another example would be superconductors, materials that conduct electricity with zero resistance when co
oled below a certain temperature. This phenomenon has many applications in technology and science, such as magnet
ic resonance imaging (MRI) machines. Despite these examples, it's important to note that most of our everyday exp
eriences are well-described by classical physics rather than quantum mechanics.
(myenv) nehagopinath@Nehas-MacBook-Air src %
```

c. Experiment with Different Parameters

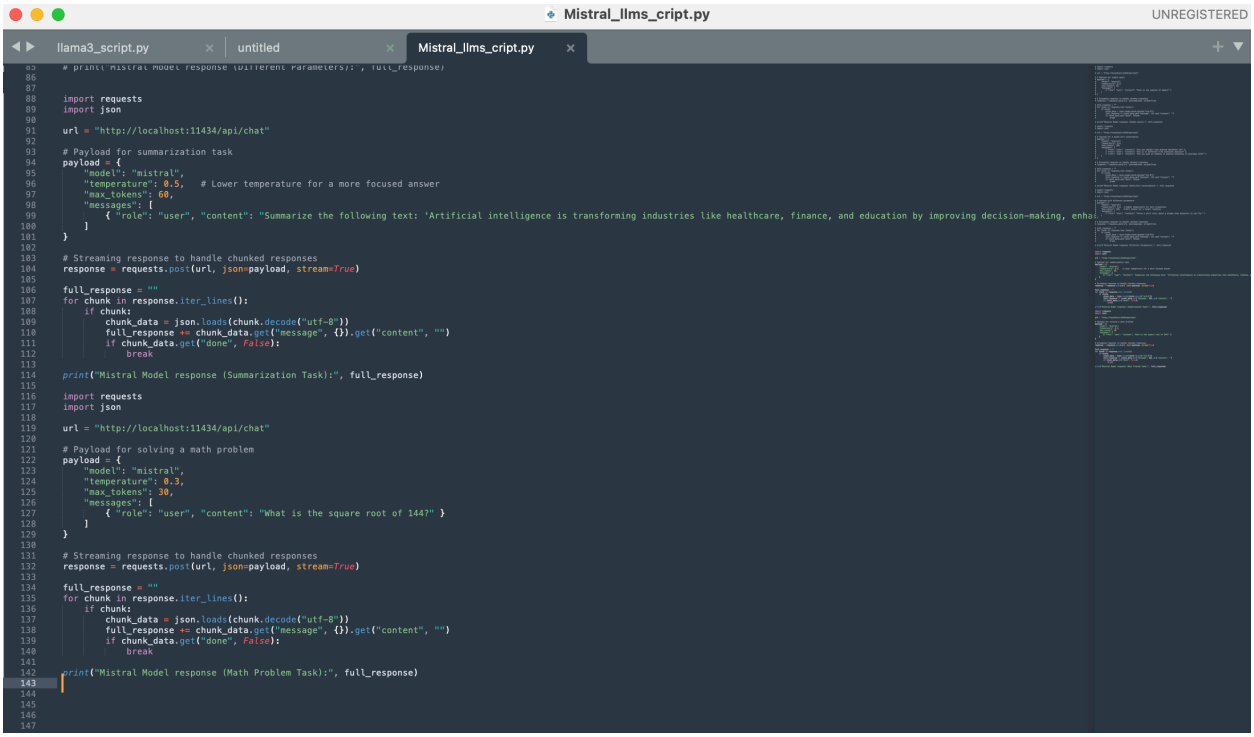
Mistral_llms_cript.py — UNREGISTERED

llama3_script.py | untitled | Mistral_llms_cript.py

```python
#         chunk_data = json.loads(chunk.decode("utf-8"))
#         full_response += chunk_data.get("message", {}).get("content", "")
#         if chunk_data.get("done", False):
#             break

# print("Mistral Model response (Multi-Turn Conversation):", full_response)

import requests
import json

url = "http://localhost:11434/api/chat"

# Payload with different parameters
payload = {
    "model": "mistral",
    "temperature": 0.9,  # Higher temperature for more creativity
    "max_tokens": 150,   # More tokens for a longer response
    "messages": [
        { "role": "user", "content": "Write a short story about a dragon that discovers it can fly." }
    ]
}

# Streaming response to handle chunked responses
response = requests.post(url, json=payload, stream=True)

full_response = ""
for chunk in response.iter_lines():
    if chunk:
        chunk_data = json.loads(chunk.decode("utf-8"))
        full_response += chunk_data.get("message", {}).get("content", "")
        if chunk_data.get("done", False):
            break

print("Mistral Model response (Different Parameters):", full_response)
```

```
(myenv) nehagopinath@Nehas-MacBook-Air src % python Mistral_llms_cript.py
/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2/myenv/lib/python3.9/site-packages/urllib3/__init__.py:3
5: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'Libre
SSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Mistral Model response (Different Parameters):  title: Ember's Ascent

In the verdant valley of Whisperwind, nestled among towering emerald peaks, lived a young dragon named Ember. Bor
n from the fiery heart of Mount Sentinel, Ember was unlike any other in her clan—a vibrant crimson hue that seeme
d to dance with an inner glow.

Ember spent her days exploring the valley's hidden caves and basking in the sun's warm embrace atop the peaks. Bu
t one peculiarity distinguished her from her kin: she had never flown.

The other dragons of Whisperwind soared effortlessly through the sky, leaving Ember to ponder their flight and wo
nder if perhaps there was something different about her. She yearned to join them, to feel the wind rush past her
 scales as they spiraled gracefully above the valley.

One fateful day, Ember came upon a hidden grove, shrouded in an ethereal mist. Intrigued by this mysterious place
, she ventured inside. As she breathed deeply of the thick, damp air, a gentle whisper caressed her ears.

"Dragon of crimson heart and soul ablaze, release your spirit, find your flight, embrace the wind and be at ease.
"

Ember's eyes widened as she realized that this was no ordinary grove but a sacred place of dragons past, where th
ey came to learn the ancient art of flight. As she pondered these words, she felt an unfamiliar tingling in her w
ings—tiny sparks igniting within their folds.

With a deep breath, Ember allowed herself to be consumed by the flame that lived within her. Her scales shimmered
, and the sparks in her wings grew brighter, until at last, they erupted into a blaze of crimson fire. The force
of the blast propelled her from the ground, sending her soaring high above Whisperwind.

Ember's heart raced as she marveled at the breathtaking panorama unfolding before her—the valley stretching out b
eneath her, the verdant peaks rising majestically in the distance. The wind howled around her, and she reveled in
 its caress, feeling more alive than ever before.

As days turned to weeks, Ember mastered the art of flight, soaring through the skies with grace and elegance. Her
 kin marveled at her newfound ability, and she shared the secret of the hidden grove, enabling them to ascend as
well. Together, they explored the world above, forming a bond stronger than any before it.

Ember had discovered that her difference was indeed a gift—one that allowed her to soar among her kin, to share i
n their joy, and to understand the true meaning of freedom. And in doing so, she learned that sometimes, being di
fferent is not a curse but rather a blessing, a beacon that illuminates the path to greatness.
(myenv) nehagopinath@Nehas-MacBook-Air src %
```

d. Test the Model's Performance on Specific Tasks




```
import requests
import json

url = "http://localhost:11434/api/chat"

# Payload for summarization task
payload = {
    "model": "mistral",
    "temperature": 0.5,   # Lower temperature for a more focused answer
    "max_tokens": 60,
    "messages": [
        { "role": "user", "content": "Summarize the following text: 'Artificial intelligence is transforming industries like healthcare, finance, and education by improving decision-making, enhancing efficiency, and enabling automation of tasks that were once too complex for computers.'" }
    ]
}

# Streaming response to handle chunked responses
response = requests.post(url, json=payload, stream=True)

full_response = ""
```

```
for chunk in response.iter_lines():
    if chunk:
        chunk_data = json.loads(chunk.decode("utf-8"))
        full_response += chunk_data.get("message", {}).get("content", "")
        if chunk_data.get("done", False):
            break

print("Mistral Model response (Summarization Task):", full_response)
```

```
(myenv) nehagopinath@Nehas-MacBook-Air src % python Mistral_llms_cript.py
/Users/nehagopinath/Documents/Courses /GenAI/Assignment 2/myenv/lib/python3.9/site-packages/urllib3/__init__.py:3
5: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'Libre
SSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Mistral Model response (Summarization Task):  Artificial Intelligence (AI) is revolutionizing sectors such as hea
lthcare, finance, and education. It helps in better decision-making processes, increases efficiency, and automate
s tasks previously considered too complex for computers.
Mistral Model response (Math Problem Task):  The square root of 144 is 12. This is because 12*12 equals 144.
(myenv) nehagopinath@Nehas-MacBook-Air src %
```

1.4 Comparative Analysis

- **Commands used for Ollama Llama3 –**

```
cd "/Users/nehagopinath/Documents/Courses/GenAI/Assignment 2"
source myenv/bin/activate
ollama pull llama3
ls
cd src
ls
python llama3_script.py
deactivate
```

Commands used for Mistral Model –

```
cd "/Users/nehagopinath/Documents/Courses/GenAI/Assignment 2"
source myenv/bin/activate
ollama pull mistral
ls
cd src
ls
python Mistral_llms_script.py
deactivate
```

- **Responses received & Screeshots – are added above**
- **Analysis of the interaction method**

  The interaction using curl commands with both Llama3 and Mistral models was straightforward and effective. Here are the key takeaways:

  a. Ease of Use
  Curl offers a simple and lightweight way to send queries to the models. Using it to interact with APIs required minimal setup, making it a practical choice for quickly testing different models.

  b. Real-time Response
  The use of streaming responses allowed for chunked data handling, improving response time perception, especially for longer queries. This feature is particularly useful in cases where incremental results are valuable.

  c. Performance
  Both models responded accurately to basic queries like "What is the capital of Japan?". The performance was similar between Llama3 and Mistral for simple tasks, but Mistral may offer advantages on lower-resource systems due to its lightweight nature.

  d. Customization
  Adjusting parameters like temperature and max_tokens enabled control over response characteristics, making the interaction flexible for various use cases, such as maintaining focus or limiting response length.

  e. Challenges
  Error handling could be improved to manage issues like malformed requests or incorrect payloads more gracefully.

  f. Overall Experience
  The curl method provided a quick and efficient way to interact with the models, ideal for initial prototyping and simple experiments.