Ollama + Kaggle + Cloudflare

Setup guide • What each code section does, in the correct order

Purpose

Run an Ollama model inside a Kaggle notebook and make its API reachable through a temporary Cloudflare Quick Tunnel so a compatible client, such as Hermes, can connect.

Useful links

Kaggle Notebooks: https://www.kaggle.com/code

Hugging Face model page: https://huggingface.co/theLittleStone/Qwen3.6-27B-AEON-Ultimate-Uncensored-MTP-GGUF

Cloudflare Quick Tunnels documentation: https://developers.cloudflare.com/cloudflare-one/connections/connect-networks/do-more-with-tunnels/trycloudflare/

Ollama: https://ollama.com/

Before you start

• Run these sections in order in the same Kaggle notebook session.

• Keep the notebook session alive. If Kaggle restarts, Ollama and the tunnel may stop, and the tunnel URL may change.

• The 27B Q4_K_M model is large. Downloading and loading it can take time and may exceed the runtime's practical memory limits.

• A Quick Tunnel URL is temporary. Do not treat it as a permanent production endpoint.

Security note: A Quick Tunnel exposes the API publicly while it is running. Avoid sending sensitive data through it, and stop the tunnel when finished.

1. Install the required software

Use this once at the beginning of a fresh Kaggle runtime. It installs Ollama and Cloudflare Tunnel. The `zstd` package is a utility and may not be needed in every runtime.

```
!apt-get update -qq
!apt-get install -y zstd
!curl -fsSL https://ollama.com/install.sh | sh
!wget -q https://github.com/cloudflare/cloudflared/releases/latest/download/cloudflared-linux-amd64.deb
!dpkg -i cloudflared-linux-amd64.deb
!/usr/local/bin/ollama --version
!cloudflared --version
```

What to check

Both version commands should print version information. If the package installation fails, review the cell output before continuing.

2. Start the Ollama API server

Starts Ollama in the background and writes its logs to a file instead of suppressing errors. Run this cell once per runtime.

```
import subprocess
import time
import requests

ollama_server = subprocess.Popen(
    ["/usr/local/bin/ollama", "serve"],
    stdout=open("/tmp/ollama.log", "w"),
    stderr=subprocess.STDOUT
)

time.sleep(5)

response = requests.get("http://127.0.0.1:11434/api/tags", timeout=10)
print("Local Ollama status:", response.status_code)
print(response.text)
```

Expected result: HTTP status `200`. If the request fails, inspect `/tmp/ollama.log` before proceeding.

3. Download the model

Use `ollama pull` to download the model without opening an interactive chat session. This can take a long time because the model is large. Run it once per runtime unless the model is already stored.

```
! /usr/local/bin/ollama pull hf.co/theLittleStone/Qwen3.6-27B-AEON-Ultimate-Uncensored-MTP-GGUF:Q4_K_M
! /usr/local/bin/ollama list
```

If the download appears stuck, check the notebook output and disk/memory availability. Avoid starting an interactive `ollama run` session if you need the notebook to continue to later cells.

4. Start a fresh Cloudflare Quick Tunnel

Creates a temporary public HTTPS URL that forwards requests to the local Ollama API. This cell reads the tunnel log and extracts the generated URL automatically.

```
import re
import subprocess
import time

with open("/tmp/cloudflared.log", "w") as tunnel_log:
    cloudflared = subprocess.Popen(
        [
            "cloudflared", "tunnel",
            "--url", "http://127.0.0.1:11434",
            "--http-host-header", "localhost:11434",
            "--protocol", "http2"
        ],
        stdout=tunnel_log,
        stderr=subprocess.STDOUT
    )

tunnel_url = None
for _ in range(30):
    time.sleep(1)
    with open("/tmp/cloudflared.log", "r") as f:
        log_text = f.read()
    match = re.search(r"https://[a-z0-9-]+\\.trycloudflare\\.com", log_text)
    if match:
        tunnel_url = match.group(0)
        break

print("Tunnel process status:", cloudflared.poll())
print(log_text)
print("Tunnel URL:", tunnel_url)
```

Copy the newly printed URL if you need to enter it manually. Do not reuse an old Quick Tunnel URL after a runtime or tunnel restart.

> **If the URL is None:** check the printed log text. The tunnel may still be starting, or the process may have failed.

5. Test both API endpoints

Checks the local API and the public tunnel. The tunnel must be running for the public tests.

```
import requests

local_response = requests.get(
    "http://127.0.0.1:11434/api/tags", timeout=15
)
print("Local /api/tags:", local_response.status_code, local_response.text[:1000])

if tunnel_url:
    public_tags = requests.get(f"{tunnel_url}/api/tags", timeout=20)
    print("Public /api/tags:", public_tags.status_code, public_tags.text[:1000])

    public_models = requests.get(f"{tunnel_url}/v1/models", timeout=20)
    print("Public /v1/models:", public_models.status_code, public_models.text[:2000])
else:
    print("No tunnel URL was detected. Check /tmp/cloudflared.log.")
```

Expected result

Expect HTTP 200 for `/api/tags` and, if supported by the installed Ollama version, `/v1/models`. Use the exact model ID returned by `/v1/models` in your client.

6. Connect Hermes or another compatible client

Use the current tunnel URL, not an old one.

Setting	Value
Base URL	Current tunnel URL followed by `/v1` Example: https://YOUR-URL.trycloudflare.com/v1
Model	The exact model ID returned by `/v1/models`
API key	Ollama normally does not require one locally. If the client requires a non-empty value, use a placeholder only if the client permits it; it is not real authentication.

Quick troubleshooting checklist

Local API is not responding: Check the Ollama process and read /tmp/ollama.log.

Model pull fails or runs out of resources: Confirm the model identifier, available disk space, and runtime memory.

Tunnel URL is missing: Read /tmp/cloudflared.log and verify that cloudflared is still running.

Public API test fails: Confirm the local endpoint works first, then confirm the tunnel process is alive and use the newest URL.

Client cannot find the model: Copy the exact model ID from /v1/models; do not assume it matches the Hugging Face repository name.

> **Reminder:** This setup is intended for temporary testing. A public Quick Tunnel URL is not an authenticated, permanent production API endpoint.