

# Integrasi Model Nusantara ke Dalam Server Lokal Ollama

Penyusun:

**Trigas Badmianto, S.T., M.Eng.,**  
Progli Teknik Elektronika, BBPPMPV Bidang Otomotif dan Elektronika  
(dengan Bantuan **GEMINI 2.5 PRO**)

## 1. Cara Memperoleh Model dari huggingface.co.

(PROMPT TO GEMINI)

Cara memperoleh model <https://huggingface.co/kalisai/Nusantara-0.8b-Indo-Chat> melalui command prompt cmd

Tentu, Anda bisa mengunduh model dari Hugging Face seperti kalisai/Nusantara-0.8b-Indo-Chat melalui Command Prompt (CMD) dengan beberapa cara. Cara yang paling umum dan direkomendasikan adalah menggunakan Git dan Git LFS (Large File Storage) karena banyak model, terutama yang berukuran besar, menggunakan LFS.

### 1.1. Mengunduh Model Menggunakan Git

Berikut adalah langkah-langkahnya:

#### 1.1.1. Instalasi Git dan Git LFS (Jika Belum Terpasang)

- **Instal Git:**
  - Kunjungi situs resmi Git: <https://git-scm.com/downloads>
  - Unduh installer yang sesuai untuk sistem operasi Windows Anda.
  - Jalankan installer dan ikuti petunjuk instalasi. Anda umumnya bisa menggunakan pengaturan default. Pastikan "Git Bash Here" dan "Git GUI Here" (opsional) serta "Git LFS (Large File Storage)" terpilih saat instalasi, atau instal LFS secara terpisah setelahnya.
- **Instal Git LFS (jika belum terinstal bersama Git):**
  - Kunjungi situs resmi Git LFS: <https://git-lfs.github.com/>
  - Unduh installer Windows dan jalankan.
  - Setelah terinstal, buka Command Prompt (CMD) atau Git Bash dan jalankan perintah berikut untuk menginisialisasi Git LFS untuk akun pengguna Anda (cukup lakukan sekali):

**Bash**

```
git lfs install
```

### 1.1.2. Mengunduh Model Menggunakan Git

Setelah Git dan Git LFS terpasang dan terkonfigurasi, Anda bisa mengunduh modelnya:

- **Buka Command Prompt (CMD).**
- **Pindah ke direktori tempat Anda ingin menyimpan model.** Misalnya, jika Anda ingin menyimpannya di D:\Models, ketik:

**Bash**

```
D:  
cd _Models LLM
```

(Buat folder Models terlebih dahulu jika belum ada: `mkdir _Models LLM`)

- **Gunakan perintah `git clone` untuk mengunduh repositori model.** Alamat repositori biasanya adalah URL Hugging Face dengan tambahan `.git` di akhir.

**Bash**

```
git clone https://huggingface.co/kalisai/Nusantara-0.8b-Indo-Chat.git
```

- **Proses Kloning dan Pengunduhan LFS:** Perintah `git clone` akan mulai mengunduh file-file repositori. Jika model menggunakan Git LFS (kemungkinan besar untuk model bahasa), Anda akan melihat pesan yang berkaitan dengan pengunduhan file LFS. Ini mungkin memakan waktu cukup lama tergantung ukuran model dan kecepatan internet Anda.

**Contoh Lengkap di CMD:**

**Bash**

```
# Pindah ke drive D (opsional, sesuaikan dengan drive Anda)  
D:  
  
# Membuat direktori jika belum ada (opsional)  
mkdir MyModels  
  
# Pindah ke direktori tersebut  
cd MyModels  
  
# Menginisialisasi Git LFS (hanya perlu sekali per instalasi Git LFS)  
git lfs install  
  
# Mengkloning repositori model  
git clone https://huggingface.co/kalisai/Nusantara-0.8b-Indo-Chat.git
```

Setelah perintah `git clone` selesai, Anda akan menemukan direktori baru bernama Nusantara-0.8b-Indo-Chat di dalam folder MyModels (atau folder mana pun yang Anda pilih) yang berisi semua file model.

## 1.2. Alternatif: Menggunakan huggingface-cli (Membutuhkan Python)

Jika Anda memiliki Python dan pip terinstal, Anda juga bisa menggunakan pustaka huggingface\_hub yang menyediakan antarmuka baris perintah.

- Instal huggingface\_hub (jika belum ada):

**Bash**

```
pip install huggingface_hub
```

- Unduh model menggunakan huggingface-cli:

**Bash**

```
huggingface-cli download kalisai/Nusantara-0.8b-Indo-Chat --local-dir Nusantara-0.8b-Indo-Chat --local-dir-use-symlinks False
```

Opsi `--local-dir` menentukan folder tujuan, dan `--local-dir-use-symlinks False` memastikan file benar-benar diunduh alih-alih menggunakan symlink (penting di Windows jika Anda ingin file fisik).

Pilih metode yang paling sesuai dengan kondisi sistem Anda. Menggunakan git clone adalah metode yang sangat umum untuk berinteraksi dengan repositori di Hugging Face.

## 2. Prompt to Gemini

**(PROMPT TO GEMINI)**

Saya pengembang teknologi kecerdasan artifisial (KA) ingin mengintegrasikan model "kalisai/Nusantara-0.8b-Indo-Chat" ke dalam server ollama sehingga bisa diakses secara lokal oleh chatbot sebagai front end nya, tidak harus pakai internet.

Model tersebut sudah saya download dari "https://huggingface.co/kalisai/Nusantara-0.8b-Indo-Chat" dalam folder hardisk saya di D:\Kegiatan\2025\_05\_19 RTL ToT KKA\Uji Coba LK42 ToT GTK\my\_PyThon\Integrasi Model Nusantara ke Ollama menjadi satu di folder Nusantara-0.8b-Indo-Chat. Struktur folder dan file adalah sebagai berikut:

```
.git/
.gitattributes
added_tokens.json
config.json
generation_config.json
merges.txt
model.safetensors
README.md
special_tokens_map.json
tokenizer.json
tokenizer_config.json
vocab.json
```

Isi dari config.json adalah

```
{
  "_name_or_path": "",
  "architectures": [
    "Qwen2ForCausalLM"
  ],
  "attention_dropout": 0.0,
  "bos_token_id": 151643,
  "eos_token_id": 151643,
  "hidden_act": "silu",
  "hidden_size": 1024,
  "initializer_range": 0.02,
  "intermediate_size": 2816,
  "max_position_embeddings": 32768,
  "max_window_layers": 21,
  "model_type": "qwen2",
  "num_attention_heads": 16,
  "num_hidden_layers": 40,
  "num_key_value_heads": 16,
  "rms_norm_eps": 1e-06,
  "rope_theta": 1000000.0,
  "sliding_window": 32768,
  "tie_word_embeddings": false,
  "torch_dtype": "bfloat16",
  "transformers_version": "4.38.2",
  "use_cache": true,
  "use_sliding_window": false,
  "vocab_size": 151936
}
```

Berikan urutan yang sistematis beserta penjelasannya hingga saya bisa mengintegrasikan model kalisai/Nusantara-0.8b-Indo-Chat ke dalam (server lokal) ollama!

### (Jawab GEMINI)

Tentu, mari kita integrasikan model kalisai/Nusantara-0.8b-Indo-Chat ke dalam Ollama agar bisa diakses secara lokal. Meskipun Ollama biasanya mengunduh model dari registry-nya, kita bisa mengimpor model lokal dengan beberapa langkah.

## 3. Persiapan Awal

1. **Pastikan Ollama Terinstal:** Jika Anda belum menginstal Ollama, unduh dan instal dari situs resmi mereka: <https://ollama.com/>. Pastikan Ollama server sudah berjalan. Anda bisa memeriksanya dengan membuka terminal dan menjalankan `ollama list` atau `ollama run llama2` (jika llama2 sudah diunduh).
2. **Pastikan Python dan huggingface\_hub Terinstal:** Meskipun Ollama akan bekerja secara mandiri, untuk beberapa langkah konversi (jika diperlukan) atau jika Anda ingin menggunakan script Python untuk berinteraksi, pastikan Anda memiliki Python dan library `huggingface_hub` terinstal.

**Bash**

```
pip install huggingface_hub
```

## 4. Langkah-langkah Integrasi Model ke Ollama

Ollama menggunakan format GGUF untuk model lokalnya. Model yang Anda unduh dari Hugging Face (model.safetensors) perlu dikonversi ke format GGUF. Beruntungnya, Hugging Face biasanya sudah memiliki konverter yang siap digunakan, atau model tersebut sudah tersedia dalam format GGUF di repositori Hugging Face (seringkali di tab "Files and versions").

**Penting:** Saat ini, Ollama tidak secara langsung mendukung impor model Safetensors ke GGUF. Anda perlu menggunakan alat pihak ketiga, yaitu llama.cpp atau ollama-import, untuk melakukan konversi ini.

## 4.1. Opsi 1: Menggunakan ollama-import (Paling Direkomendasikan jika Tersedia)

ollama-import adalah alat eksperimental yang dikembangkan untuk mempermudah impor model Hugging Face ke Ollama.

### 1. Instal ollama-import:

**Bash**

```
pip install ollama-import
```

**Status CMD**

```
ERROR: Could not find a version that satisfies the requirement ollama-import (from versions: none)
ERROR: No matching distribution found for ollama-import
```

### 2. Konversi Model ke GGUF:

Masuk ke direktori tempat model Nusantara-0.8b-Indo-Chat Anda berada (D:\\_Models LLM\Nusantara-0.8b-Indo-Chat).

**Bash**

```
cd "D:\_Models LLM\Nusantara-0.8b-Indo-Chat"
ollama-import
```

Perintah ini akan mencoba mengidentifikasi model dan mengkonversinya. Ikuti instruksi jika ada. Jika berhasil, akan ada file .gguf di direktori yang sama atau direktori keluaran yang ditentukan.

**Status CMD**

```
'ollama-import' is not recognized as an internal or external command,
operable program or batch file.
```

Hal ini terjadi karena perintah 'ollama-import' tidak dikenali.

## 4.2. Opsi 2: Menggunakan llama.cpp (Lebih Manual, Tapi Andal)

llama.cpp adalah proyek inti yang sering digunakan di balik layar oleh banyak proyek LLM, termasuk Ollama, untuk menjalankan model GGUF. Anda perlu membangun llama.cpp dan menggunakan skrip konversinya.

1. **Kloning dan Bangun llama.cpp:** Jika Anda belum memiliki llama.cpp, kloning repositorinya:

### Bash

```
git clone https://github.com/ggerganov/llama.cpp.git
cd llama.cpp
make
```

Pastikan cmake dan build-essential (untuk Linux/WSL) atau Visual Studio (untuk Windows) terinstal jika Anda mengalami masalah kompilasi.

2. **Pindahkan Model ke Direktori llama.cpp:** Untuk memudahkan, salin folder Nusantara-0.8b-Indo-Chat ke dalam direktori **llama.cpp/models**.

### Bash

```
xcopy /E /I "D:\_Models LLM\Nusantara-0.8b-Indo-Chat" "D:\_Kegiatan\2025_05_19 RTL ToT KKA\(\LK42 ToT GTK) Uji Coba\Integrasi Model Nusantara ke Ollama\llama.cpp\models\Nusantara-0.8b-Indo-Chat"
```

(Ganti path llama.cpp Anda jika berbeda).

Catatan: prompt ini tidak mengcopy folder .git dan seisiinya. Menurut Gemini untuk tujuan integrasi ke Ollama, file ini tidak diperlukan.

3. **Instal Dependensi Python untuk Konversi:** Masuk ke direktori llama.cpp/ dan instal dependensi Python yang dibutuhkan oleh skrip konversi.

### Bash

```
cd D:\_Kegiatan\2025_05_19 RTL ToT KKA\(\LK42 ToT GTK) Uji Coba\Integrasi Model Nusantara ke Ollama\llama.cpp
pip install -r requirements.txt
```

4. **Konversi Model ke GGUF:** Gunakan skrip convert.py yang ada di llama.cpp/ untuk mengkonversi model safetensors ke GGUF.

Gunakan sintaks berikut jika bobot model menjadi representasi 8-bit (**Hasilnya ~0.8 GB**):

### Bash

```
python convert_hf_to_gguf.py "D:\_Models LLM\Nusantara-0.8b-Indo-Chat" --outfile "D:\_Kegiatan\2025_05_19 RTL ToT KKA\(\LK42 ToT GTK) Uji Coba\Integrasi Model Nusantara ke Ollama\Nusantara-0.8b-Indo-Chat-q8_0.gguf" --outtype q8_0
```

Perintah tersebut akan mengonversi model "Nusantara-0.8b-Indo-Chat" dari direktori sumbernya ke format GGUF, sekaligus menerapkan kuantisasi 8-bit (q8\_0) dan menyimpannya sebagai Nusantara-0.8b-Indo-Chat-q8\_0.gguf. Hasilnya adalah file model yang ukurannya lebih kecil secara signifikan dibandingkan versi presisi penuh (f16), namun tetap mempertahankan kualitas yang sangat baik untuk dijalankan secara lokal melalui Ollama.

(Selanjutnya pelajari tentang metode kuantisasi q4\_K\_M, menjadi representasi 4-bit menggunakan teknik K-Quants)

Gunakan sintaks berikut jika bobot model menjadi representasi tetap 16 -bit (**Hasilnya ~1.6 GB**):

#### Bash

```
python convert_hf_to_gguf.py "D:\_Models LLM\Nusantara-0.8b-Indo-Chat" --outfile "D:\_Kegiatan\2025_05_19 RTL ToT KKA\ (LK42 ToT GTK) Uji Coba\Integrasi Model Nusantara ke Ollama\Nusantara-0.8b-Indo-Chat-f16.gguf" --outtype f16
```

Proses konversi ini bisa memakan waktu tergantung pada ukuran model dan spesifikasi hardware Anda.

## 4.3. Membuat Modelfile untuk Ollama

Setelah Anda memiliki file .gguf, langkah selanjutnya adalah membuat Modelfile yang akan memberitahu Ollama bagaimana cara menggunakan model tersebut.

### 4.3.1. Buat File Modelfile.

Buat file baru bernama Modelfile (tanpa ekstensi) di direktori yang mudah diakses (misalnya, di samping folder model GGUF Anda atau di folder proyek chatbot Anda).

### 4.3.2. Isi File Modelfile.

Berikut adalah sintaks atau kode komplit yang disarankan untuk Anda tuliskan ke dalam file Modelfile, dengan mempertimbangkan model Nusantara-0.8b-Indo-Chat (yang berbasis Qwen2) dan file GGUF q8\_0 yang telah Anda buat:

Code snippet

```
# Baris FROM menunjuk ke lokasi absolut file GGUF model Anda.
# Pastikan path ini benar-benar sesuai dengan lokasi dan nama file GGUF Anda.
# Gunakan tanda kutip jika path mengandung spasi atau karakter khusus.
FROM "D:\_Kegiatan\2025_05_19 RTL ToT KKA\ (LK42 ToT GTK) Uji Coba\Integrasi Model Nusantara ke Ollama\Nusantara-0.8b-Indo-Chat-q8_0.gguf"
```

```
# Baris TEMPLATE mendefinisikan bagaimana input akan diformat untuk model.
# Template ini spesifik untuk model berbasis Qwen2.
TEMPLATE """{{ if .System }}<|im_start|>system
```

```

{{ .System }}<|im_end|>
{{ end }}{{ if .Prompt }}<|im_start|>user
{{ .Prompt }}<|im_end|>
{{ end }}<|im_start|>assistant
""

```

# Baris SYSTEM menetapkan pesan sistem default untuk model.

# Ini membantu mengarahkan peran dan perilaku model.

SYSTEM "Anda adalah asisten AI yang ramah dan membantu, yang bisa berkomunikasi dalam Bahasa Indonesia."

# Baris PARAMETER mengatur berbagai parameter inferensi.

# Anda bisa menyesuaikan nilai-nilai ini sesuai kebutuhan.

PARAMETER temperature 0.7

# PARAMETER top\_k 50

# PARAMETER top\_p 0.9

# PARAMETER num\_ctx 4096 # Sesuaikan dengan kemampuan model dan memori Anda

# PARAMETER stop adalah token atau string yang akan menghentikan generasi teks.

# Untuk Qwen2, <|im\_end|> adalah token penting untuk mengakhiri giliran assistant.

PARAMETER stop "<|im\_start|>"

PARAMETER stop "<|im\_end|>"

PARAMETER stop "<|endoftext|>"

### Penjelasan Tambahan:

1. **FROM:** Pastikan sekali lagi bahwa path ke file .gguf Anda (Nusantara-0.8b-Indo-Chat-q8\_0.gguf) sudah 100% benar. Kesalahan kecil di sini akan menyebabkan Ollama tidak bisa menemukan modelnya.
2. **TEMPLATE:** Template ini krusial untuk model chat Qwen2 agar bisa membedakan antara pesan sistem, input pengguna, dan respons asisten.
3. **SYSTEM:** Pesan ini akan memberikan instruksi awal kepada model mengenai perannya.
4. **PARAMETER temperature 0.7:** Mengontrol "kreativitas" atau keacakan respons. Nilai umum antara 0.2 (lebih deterministik, fokus) hingga 1.0 (lebih kreatif, beragam). 0.7 adalah titik awal yang baik.
5. **PARAMETER top\_k 50** (dikomentari): Membatasi pilihan kata berikutnya dari 50 kata yang paling mungkin. Anda bisa mengaktifkannya jika mau.
6. **PARAMETER top\_p 0.9** (dikomentari): Metode sampling lain (nucleus sampling). Anda bisa mengaktifkannya, seringkali digunakan bersamaan atau sebagai alternatif top\_k.
7. **PARAMETER num\_ctx 4096** (dikomentari): Ukuran konteks maksimum yang bisa ditangani model dalam satuan token. Model Qwen2 0.8B biasanya mendukung konteks yang panjang (misalnya 32768 token dari config.json Anda: "max\_position\_embeddings": 32768). Namun, penggunaan konteks yang sangat besar akan membutuhkan lebih banyak memori. 4096 adalah nilai yang umum dan aman untuk banyak kasus. Anda bisa menyesuaikannya. Jika tidak disetel, Ollama akan menggunakan nilai default berdasarkan model.
8. **PARAMETER stop "<|...|>":** Ini sangat penting. stop memberitahu model kapan harus berhenti menghasilkan teks.
  - o <|im\_end|>: Token khusus Qwen2 yang menandakan akhir dari sebuah giliran (misalnya, akhir dari respons assistant).
  - o <|im\_start|>: Berguna untuk mencegah model menghasilkan giliran "user" atau "system" baru secara tidak sengaja setelah responsnya.



- `<endoftext>`: Token umum untuk akhir teks, yang juga digunakan oleh Qwen2 (eos\_token\_id: 151643).

Anda bisa menyalin seluruh blok kode di atas dan menempelkannya ke dalam file Modelfile Anda (pastikan tidak ada ekstensi tambahan pada nama file, jadi hanya Modelfile).

#### 4.3.2. Perintah Ollama Create

Perintah ini akan memberitahu Ollama untuk membuat sebuah model baru berdasarkan instruksi di Modelfile Anda. Sintaksnya adalah: `ollama create <nama_model_di_ollama> -f <path_ke_Modelfile_Andas>`

Contoh perintah yang bisa Anda gunakan, dengan asumsi Anda menamai Modelfile Anda `Modelfile_Nusantara_q8_0` dan menyimpannya di lokasi yang disebutkan di atas:

##### Bash

```
ollama create nusantara:0.8b-q8_0 -f "D:\_Kegiatan\2025_05_19 RTL ToT KKA\ (LK42 ToT GTK) Uji Coba\Integrasi Model Nusantara ke Ollama\Modelfile_Nusantara_q8_0"
```

##### Penjelasan Perintah:

- `ollama create`: Perintah dasar untuk membuat model di Ollama.
- `nusantara:0.8b-q8_0`: Ini adalah **nama yang akan Anda gunakan untuk memanggil model ini di Ollama**. Anda bebas memilih nama ini, tetapi sebaiknya deskriptif. Menggunakan format `nama:tag` adalah praktik yang baik (misalnya, `nusantara` adalah nama dasarnya, dan `0.8b-q8_0` adalah tag yang menjelaskan versinya).
- `-f`: Flag yang menandakan bahwa Anda akan menyediakan path ke Modelfile.
- `"D:\_Kegiatan\...\Modelfile_Nusantara_q8_0"`: Path absolut ke file Modelfile yang telah Anda buat. **Gunakan tanda kutip** karena path Anda mengandung spasi dan tanda kurung.

**Proses Pembuatan Model:** Setelah Anda menjalankan perintah di atas, Ollama akan mulai memproses Modelfile Anda. Ini melibatkan:

- Membaca path file GGUF dari baris `FROM`.
- Menyalin atau mereferensikan file GGUF tersebut ke dalam direktori penyimpanan internal Ollama.
- Menyimpan konfigurasi (template, parameter, pesan sistem) yang Anda definisikan. Proses ini mungkin memerlukan waktu beberapa saat tergantung ukuran file GGUF dan kecepatan disk Anda. Anda akan melihat output di terminal yang menunjukkan progresnya. Jika berhasil, biasanya akan ada pesan seperti `"success"`.

#### 4.4. Menguji Model

**Langkah Selanjutnya Setelah Model Berhasil Dibuat:** Jika perintah `ollama create` berhasil tanpa error, model Anda sekarang sudah terdaftar di Ollama secara lokal. Anda bisa:

- **Memverifikasi model:** Jalankan `ollama list` untuk melihat daftar model yang terinstal di Ollama Anda. Anda seharusnya melihat `nusantara:0.8b-q8_0` (atau nama apa pun yang Anda berikan) di daftar tersebut.

**Bash**

```
ollama list
```

- **Menjalankan model untuk pertama kali (testing):**

**Bash**

```
ollama run nusantara:0.8b-q8_0
```

Ini akan memuat model ke memori (mungkin perlu waktu saat pertama kali) dan Anda bisa langsung berinteraksi dengannya di terminal. Cobalah mengirimkan beberapa prompt dalam Bahasa Indonesia untuk menguji responsnya.

Ollama akan memuat model. Anda akan melihat prompt >>>. Ketik pertanyaan Anda dan tekan Enter.

1. **Contoh Percakapan:** >>> Halo, apa kabar?  
Model akan merespons.
2. **Keluar dari Mode Interaktif:** Ketik /bye atau tekan Ctrl+D (Linux/macOS) / Ctrl+Z lalu Enter (Windows) untuk keluar.

## 5. Integrasi ke Chatbot Mengakses Model dari Chatbot (Front-end)

Setelah model berjalan di server Ollama lokal, chatbot front-end Anda bisa berinteraksi dengannya melalui API REST Ollama. Ollama secara default berjalan di <http://localhost:11434>.

### 5.1. Contoh Dasar Chatbot

Berikut adalah contoh dasar bagaimana chatbot Python dapat berinteraksi dengan Ollama:

**Python**

```
import requests
import json

def chat_with_ollama(prompt, model_name="nusantara-0.8b-indo-chat"):
    url = "http://localhost:11434/api/generate"
    headers = {"Content-Type": "application/json"}
    data = {
        "model": model_name,
        "prompt": prompt,
        "stream": False # Set true jika ingin streaming respons
    }
    try:
        response = requests.post(url, headers=headers, data=json.dumps(data))
```

```

        response.raise_for_status() # Akan memunculkan HTTPException untuk respons status buruk (4xx
atau 5xx)
        result = response.json()
        return result["response"]
except requests.exceptions.RequestException as e:
    print(f"Error connecting to Ollama: {e}")
    return None

if __name__ == "__main__":
    print("Selamat datang di Chatbot Nusantara-0.8b-Indo-Chat! Ketik 'keluar' untuk mengakhiri.")
    while True:
        user_input = input("Anda: ")
        if user_input.lower() == "keluar":
            print("Sampai jumpa!")
            break

        response = chat_with_ollama(user_input)
        if response:
            print(f"Bot: {response}")
        else:
            print("Bot: Maaf, ada masalah saat berkomunikasi dengan model.")

```

### Penjelasan Kode Python:

- Menggunakan library requests untuk melakukan permintaan HTTP POST ke API Ollama.
- Endpoint yang digunakan adalah /api/generate untuk menghasilkan respons dari prompt.
- model: Nama model yang sudah Anda impor ke Ollama (nusantara-0.8b-indo-chat).
- prompt: Input dari pengguna.
- stream: False: Mengambil seluruh respons sekaligus. Jika Anda ingin respons yang dihasilkan secara bertahap (seperti mengetik), atur ke True dan tangani data yang diterima secara berurutan.

## 5.2. Chatbot Streaming Aktif

Berikut adalah contoh dasar bagaimana chatbot Python dapat berinteraksi dengan Ollama:

### Python

```

import requests
import json

#def chat_with_ollama(prompt, model_name = "llama3:latest"):
def chat_with_ollama(prompt, model_name = "nusantara:0.8b-q8_0"):

    url = "http://localhost:11434/api/generate"
    headers = {"Content-Type": "application/json"}
    data = {

```

```

    "model": model_name,
    "prompt": prompt,
    "stream": True # <--- Perubahan Kunci: Aktifkan streaming
}

full_response_content = [] # Untuk menyimpan seluruh respons jika diperlukan nanti
try:
    # Menggunakan stream=True pada requests.post juga
    response_stream = requests.post(url, headers=headers, data=json.dumps(data), stream=True)
    response_stream.raise_for_status() # Akan memunculkan HTTPException untuk respons status
    buruk (4xx atau 5xx)

    # Iterasi per baris dari stream
    for line in response_stream.iter_lines():
        if line:
            try:
                # Setiap baris adalah objek JSON terpisah
                json_chunk = json.loads(line.decode('utf-8'))
                token = json_chunk.get("response", "")
                print(token, end="", flush=True) # Cetak token segera, flush=True penting
                full_response_content.append(token)

                # Ollama mengirimkan 'done: true' di chunk terakhir
                if json_chunk.get("done"):
                    print() # Tambahkan baris baru setelah selesai
                    break
            except json.JSONDecodeError:
                # Kadang ada baris kosong atau non-JSON, abaikan jika perlu
                # atau tangani secara spesifik jika formatnya diketahui
                print(f"\n[Error decoding JSON chunk: {line.decode('utf-8')}]")
                continue # Lanjutkan ke baris berikutnya

    return "".join(full_response_content) # Mengembalikan seluruh respons yang telah digabungkan

except requests.exceptions.RequestException as e:
    print(f"Error connecting to Ollama: {e}")
    return None
except Exception as e: # Menangkap kesalahan lain yang mungkin terjadi selama streaming
    print(f"\nAn unexpected error occurred during streaming: {e}")
    return "".join(full_response_content) # Kembalikan apa yang sudah terkumpul sejauh ini

if __name__ == "__main__":
    print("Selamat datang di Chatbot ! Ketik 'keluar' untuk mengakhiri.")
    while True:
        user_input = input("Anda: ")
        if user_input.lower() == "keluar":

```

```

print("Sampai jumpa!")
break

print("Bot: ", end="") # Cetak "Bot: " sekali sebelum respons streaming
response_text = chat_with_ollama(user_input)

if response_text is None: # Ini akan terjadi jika ada RequestException
    # Pesan error sudah dicetak di dalam chat_with_ollama
    # Kita bisa tambahkan pesan umum di sini jika mau, tapi mungkin redundan
    print("Maaf, ada masalah saat berkomunikasi dengan model.")
elif not response_text and response_text is not None:
    # Jika fungsi mengembalikan string kosong (bukan None)
    # Ini berarti tidak ada token yang diterima, tapi tidak ada error koneksi
    # Fungsi chat_with_ollama sudah print() newline jika 'done'
    # jadi kita mungkin tidak perlu print apa-apa lagi, atau pesan spesifik
    print("Model tidak memberikan respons.")

```

### Penjelasan Kode Python:

- Fungsi inti `chat_with_ollama` mengirimkan prompt pengguna ke model Ollama lokal yang ditentukan (default "nusantara:0.8b-q8\_0") melalui API `http://localhost:11434/api/generate` untuk mendapatkan respons.
- Pengaturan "stream": True pada payload data yang dikirim ke Ollama dan parameter `stream=True` pada pemanggilan `requests.post` secara bersamaan mengaktifkan fitur streaming respons dari server.
- Skrip ini memproses respons streaming baris per baris, mem-parse setiap potongan JSON untuk mendapatkan token, lalu segera mencetaknya ke konsol hingga model menandakan selesai dengan `json_chunk.get("done")`.
- Selain menampilkan, skrip juga mengumpulkan seluruh token respons menjadi satu string dan menyertakan mekanisme penanganan error untuk masalah koneksi, parsing JSON, atau kesalahan streaming lainnya.
- Bagian utama program (`if __name__ == "__main__":`) menyediakan antarmuka command-line interaktif yang memungkinkan pengguna mengirim input, melihat respons streaming dari bot, dan keluar dari aplikasi.

## 6. Ringkasan Urutan Langkah

1. **Persiapan:** Instal Ollama, Python, dan `huggingface_hub`.
2. **Konversi Model ke GGUF:**
  - **Opsional 1 (Direkomendasikan):** Gunakan `ollama-import` dari direktori model Anda: `ollama-import`.
  - **Opsional 2 (Manual):** Kloning `llama.cpp`, bangun, instal dependensi Python, lalu konversi model: `python convert.py models/Nusantara-0.8b-Indo-Chat --outfile models/Nusantara-0.8b-Indo-Chat/nusantara-0.8b-indo-chat.gguf --model-type qwen2`.
3. **Buat Modelfile:** Buat file Modelfile dengan FROM yang menunjuk ke file `.gguf` dan sesuaikan SYSTEM serta TEMPLATE untuk Qwen2.

4. **Impor Model ke Ollama:** Jalankan `ollama create nusantara-0.8b-indo-chat -f Modelfile` dari direktori Modelfile Anda.
5. **Uji Model:** Jalankan `ollama run nusantara-0.8b-indo-chat` di terminal untuk berinteraksi langsung.
6. **Integrasi ke Chatbot:** Gunakan API REST Ollama (misalnya `http://localhost:11434/api/generate`) dari kode chatbot Anda untuk mengirim prompt dan menerima respons.

Dengan mengikuti langkah-langkah ini, Anda seharusnya berhasil mengintegrasikan model kalisai/Nusantara-0.8b-Indo-Chat ke dalam server Ollama lokal Anda dan membuatnya dapat diakses oleh chatbot. Hal ini sudah saya buktikan sendiri. Selamat mencoba!

## PUSTAKA

Z. A. Kusworo, "Nusantara: A Series of Versatile Open Weight Language Model of Bahasa Indonesia," Hugging Face, 2024. [Online]. Available: <https://huggingface.co/kalisai>.