```
import gradio as gr
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM
# -------------------------
# Load Model & Tokenizer
# -------------------------
model_name = "ibm-granite/granite-3.2-2b-instruct"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    torch_dtype=torch.float16 if torch.cuda.is_available() else torch.float32,
    device_map="auto" if torch.cuda.is_available() else None,
)

if tokenizer.pad_token is None:
    tokenizer.pad_token = tokenizer.eos_token


# -------------------------
# Generate AI Response
# -------------------------
def generate_response(prompt, max_length=1024):
    inputs = tokenizer(prompt, return_tensors="pt", truncation=True, max_length=512)

    if torch.cuda.is_available():
        inputs = {k: v.to(model.device) for k, v in inputs.items()}

    with torch.no_grad():
        outputs = model.generate(
            **inputs,
            max_length=max_length,
            temperature=0.7,
            do_sample=True,
            pad_token_id=tokenizer.eos_token_id,
        )

    response = tokenizer.decode(outputs[0], skip_special_tokens=True)
    response = response.replace(prompt, "").strip()
    return response


# -------------------------
# AI Functions
# -------------------------
def city_analysis(city_name):
    prompt = f"""Provide a detailed analysis of {city_name} including:
1. Crime index and safety statistics
2. Accident rates and traffic safety
3. Overall safety assessment
"""
    return generate_response(prompt, max_length=1000)


def citizen_interaction(query):
    prompt = f"""You are a government assistant. Provide accurate and helpful information about:
Citizen Query: {query}
"""
    return generate_response(prompt, max_length=1000)
```

```
# -------------------------
# Gradio Interface
# -------------------------
with gr.Blocks() as app:
    gr.Markdown("# 🏙️ City Analysis & Citizen Services AI")

    with gr.Tabs():
        # Tab 1: City Analysis
        with gr.TabItem("City Analysis"):
            with gr.Row():
                with gr.Column():
                    city_input = gr.Textbox(
                        label="Enter City Name",
                        placeholder="e.g., New York, London, Mumbai...",
                    )
                    analyze_btn = gr.Button("Analyze City")
                with gr.Column():
                    city_output = gr.Textbox(
                        label="City Analysis (Crime Index & Accidents)", lines=15
                    )

            analyze_btn.click(city_analysis, inputs=city_input, outputs=city_output)

        # Tab 2: Citizen Services
        with gr.TabItem("Citizen Services"):
            with gr.Row():
                with gr.Column():
                    citizen_query = gr.Textbox(
                        label="Your Query",
                        placeholder="Ask about public services, government policies, civic issues..
                        lines=4,
                    )
                    query_btn = gr.Button("Get Information")
                with gr.Column():
                    citizen_output = gr.Textbox(
                        label="Government Response", lines=15
                    )

            query_btn.click(
                citizen_interaction, inputs=citizen_query, outputs=citizen_output
            )

# -------------------------
# Launch App
# -------------------------
app.launch(share=True)
```

```
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggin
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or d
  warnings.warn(
tokenizer_config.json:       8.88k/? [00:00<00:00, 300kB/s]
vocab.json:       777k/? [00:00<00:00, 6.71MB/s]
merges.txt:       442k/? [00:00<00:00, 10.1MB/s]
tokenizer.json:       3.48M/? [00:00<00:00, 60.7MB/s]
added_tokens.json: 100%                                                  87.0/87.0 [00:00<00:00, 1.92kB/s]
special_tokens_map.json: 100%                                            701/701 [00:00<00:00, 31.5kB/s]
config.json: 100%                                                        786/786 [00:00<00:00, 21.6kB/s]
`torch_dtype` is deprecated! Use `dtype` instead!
model.safetensors.index.json:       29.8k/? [00:00<00:00, 2.98MB/s]
Fetching 2 files: 100%                                                   2/2 [03:09<00:00, 189.49s/it]
model-00001-of-00002.safetensors: 100%                                   5.00G/5.00G [03:09<00:00, 90.1MB/s]
model-00002-of-00002.safetensors: 100%                                   67.1M/67.1M [00:05<00:00, 10.4MB/s]
Loading checkpoint shards: 100%                                          2/2 [00:19<00:00,  7.94s/it]
generation_config.json: 100%                                             137/137 [00:00<00:00, 17.5kB/s]
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://a049d341910332e594.gradio.live

This share link expires in 1 week. For free permanent hosting and GPU upgrades, run `gradio de
```

City Analysis & Citizen Services AI

City Analysis | Citizen Services

Enter City Name

e.g., New York, London, Mumbai...

Analyze City

City Analysis (Crime Index & Accidents)