Citizen AI: Intelligent Citizen Intelligent Project Documentation

Project Documentation

1.Introduction

- ✓ Project title: Health AI: Intelligent Healthcare Assistant Project Documentation
- ✓ Team ID: NM2025TMID02137
- ✓ Team Leader: STEVE SHIBINA.S. S
- ✓ Team member: DEEPA.P
- ✓ Team member: SHANMUGA PRIYA.M
- ✓ Team member: BHUVANESHWARI.N

2.Project Overview

1.Purpose:

- ✓ The project provides AI-powered city analysis and citizen services support.
- ✓ It helps citizens and governments by generating insights on:
- ✓ Crime index and safety statistics.
- ✓ Accident and traffic safety data.
- ✓ General safety assessment of a city.
- ✓ Queries about public services, policies, or civic

2.Overview:

- Built using Hugging Face Transformers and Gradio.
- Uses IBM Granite model (granite-3.2-2b-instruct) for natural language understanding and response generation.
- Two main features:
- City Analysis Tab → Users input a city name to get safety and accident analysis.
- Citizen Services Tab → Users ask government-related queries and get AI-generated responses.

3.Architecture:

★Frontend: Gradio interface (tabs, textboxes, buttons).
★Backend: Python functions using Hugging Face Transformers for text generation.
★Model: ibm-granite/granite-3.2-2b-instruct.
★Execution: Runs on Google Colab with GPU acceleration (T4 GPU recommended).

4.Setup Instructions:

- Open Google Colab.
- Change Runtime → GPU → T4 GPU.
- Install dependencies:
- !pip install transformers torch gradio -q

- Run the provided script.
- The Gradio app will launch with a shareable link.

5.Folder Structure (suggested):

```
city_analysis_ai/
├── app.py  # Main Gradio app
├── requirements.txt  # Dependencies (transformers, torch, gradio)
├── README.md  # Documentation
├── /models  # (Optional) Custom models or configs
├── /notebooks  # Jupyter/Colab notebooks
├── /tests  # Testing scripts
├── /docs  # API documentation
```

6.Running the Application:

- Run app.py (or the Colab notebook).
- Open the Gradio app via generated share=True link.
- Use the City Analysis or Citizen Services tabs

7.API Documentation (functions in the project):

1. Takes a text prompt and generates AI-based response.
2. Used internally by other functions.

3. city_analysis(city_name)
4. Input: City name.
5. Output: Crime stats, accident info, safety assessment.
6. citizen_interaction(query)
7. Input: Citizen's query.
8. Output: AI response with policies, services, or civic guidance.

8.Authentication

- ❖No authentication required (public demo).
- ❖Can be extended with login (OAuth, JWT) if deployed in production.

9.User Interface:

- ★Built with Gradio Blocks.
- ★Tabs for switching between features.
- ★Textboxes for inputs and AI responses.
- ★Buttons to trigger analysis or query resolution.

10.Testing

Manual Testing:

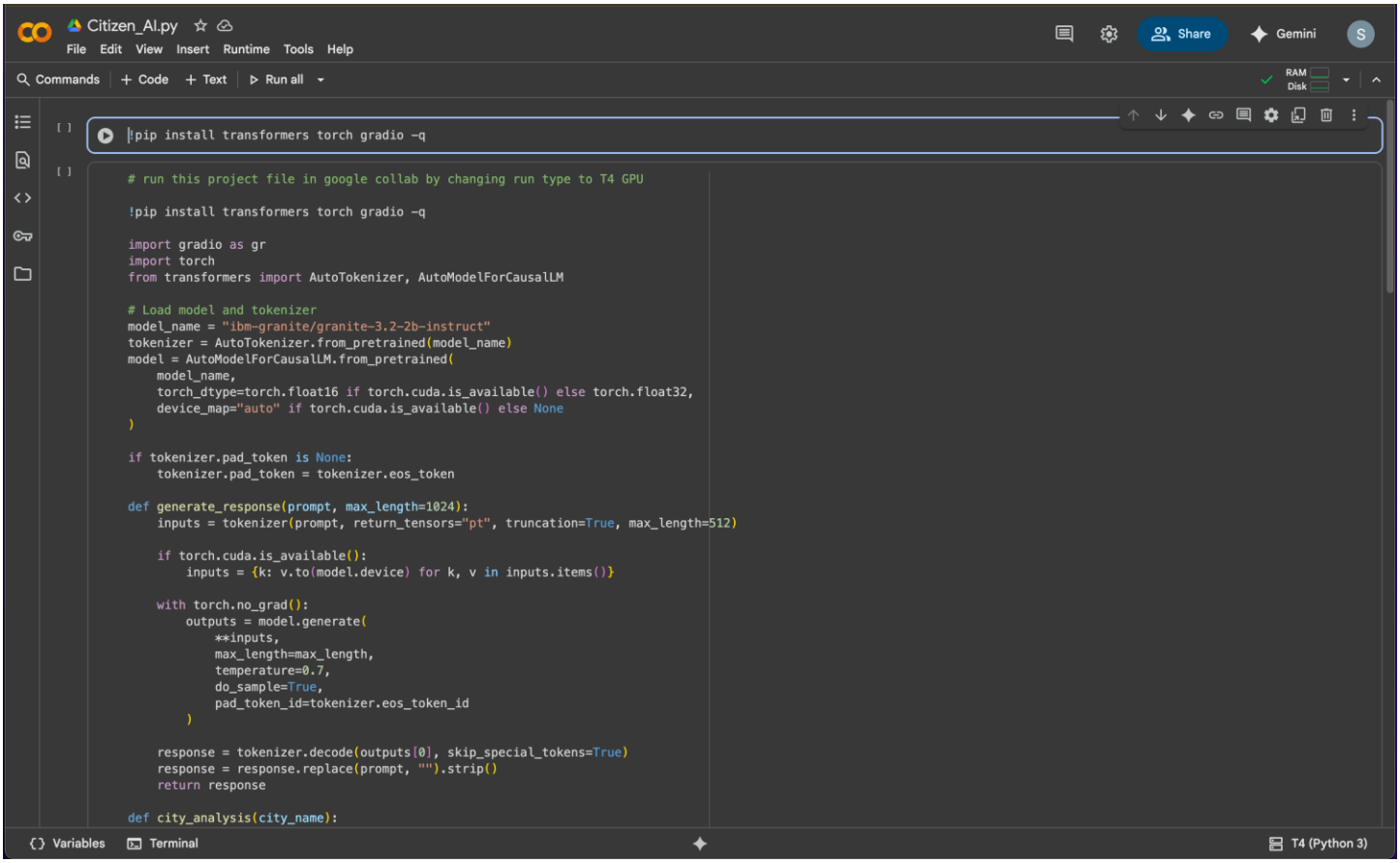
- ➢Enter a city name (e.g., Mumbai) → check AI output.

- Enter a citizen query (e.g., How to apply for voter ID?) → verify AI response.
- Automated Testing (optional):
- Unit tests for generate_response, city_analysis, citizen_interaction.
- Mock outputs to ensure consistency.

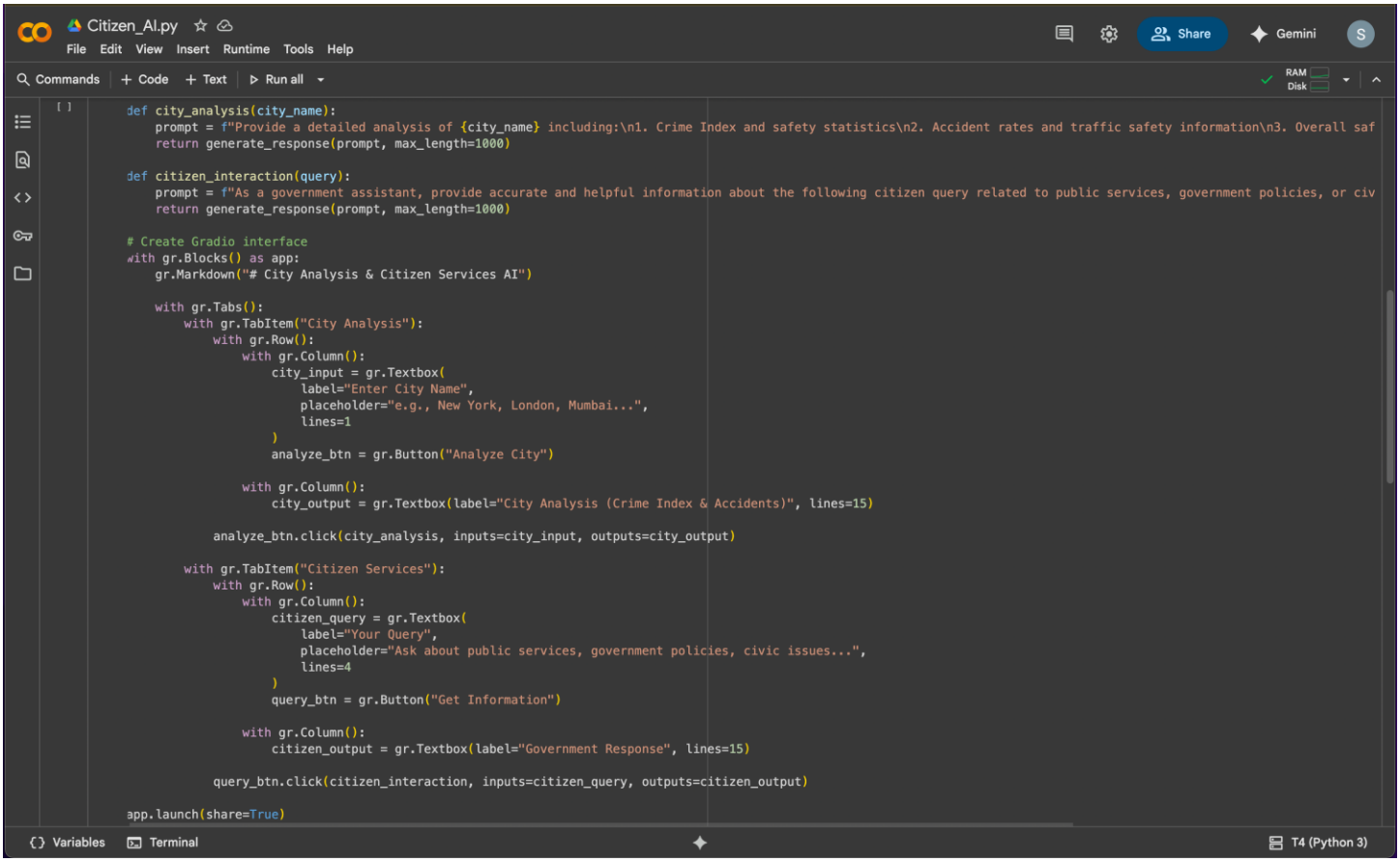
11.Screenshots:

- Input and Output:

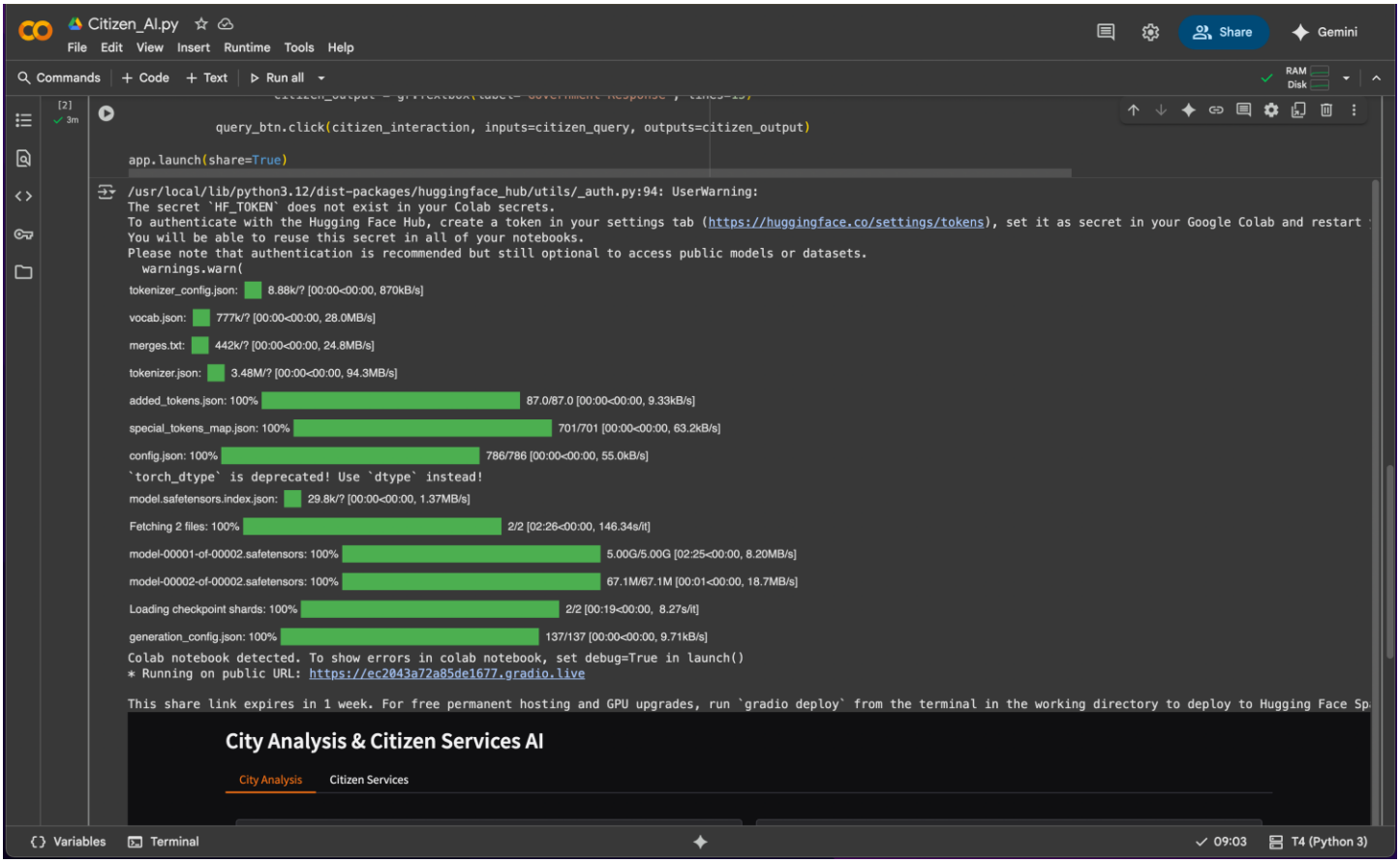
6

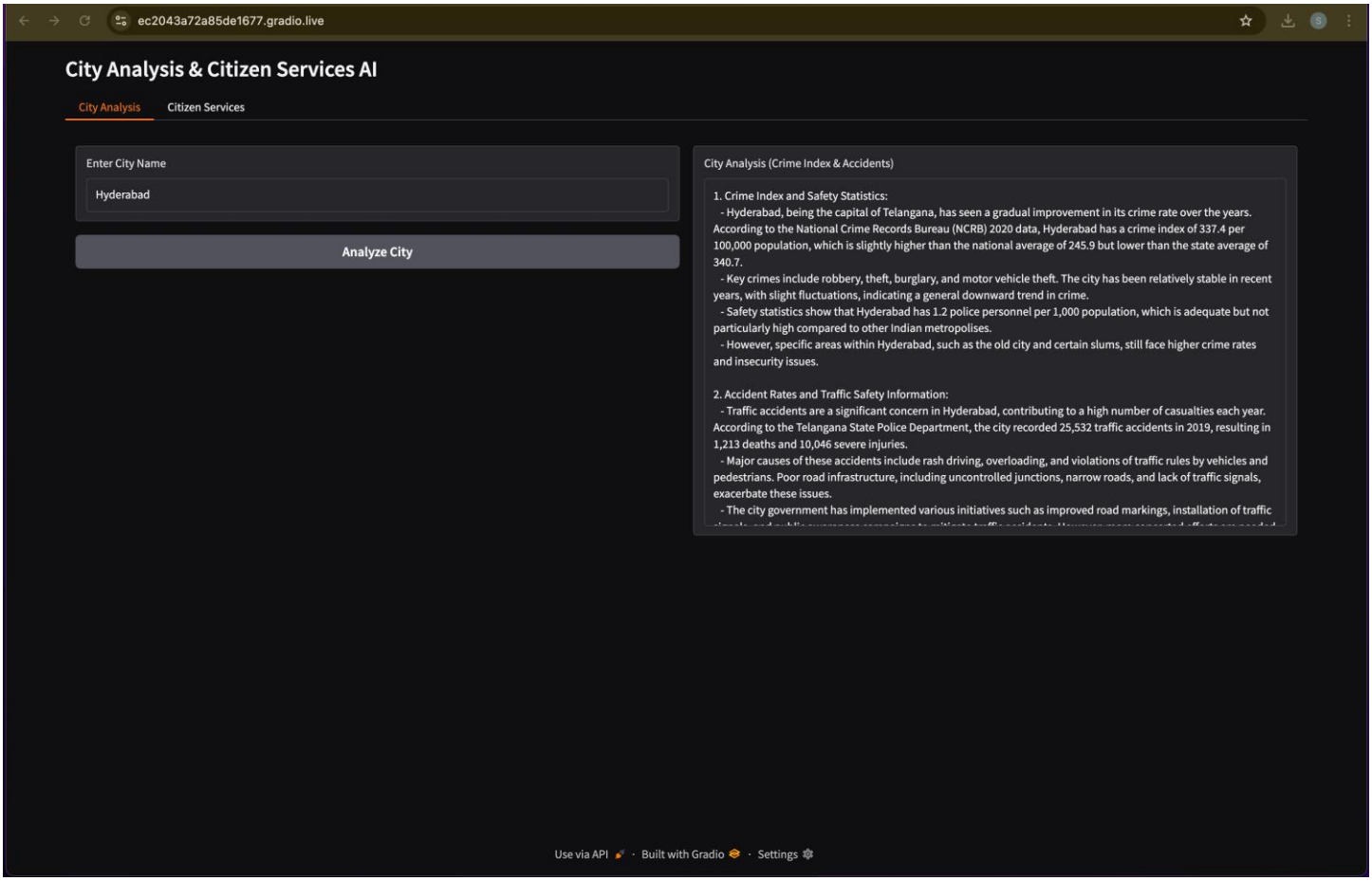

Citizen_AI.py
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code
+ Text
Run all
RAM
Disk
[ ]
!pip install transformers torch gradio -q
[ ]
run this project file in google collab by changing run type to T4 GPU
!pip install transformers torch gradio -q
import gradio as gr
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM
# Load model and 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
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
def city_analysis(city_name):
Variables
Terminal
T4 (Python 3)

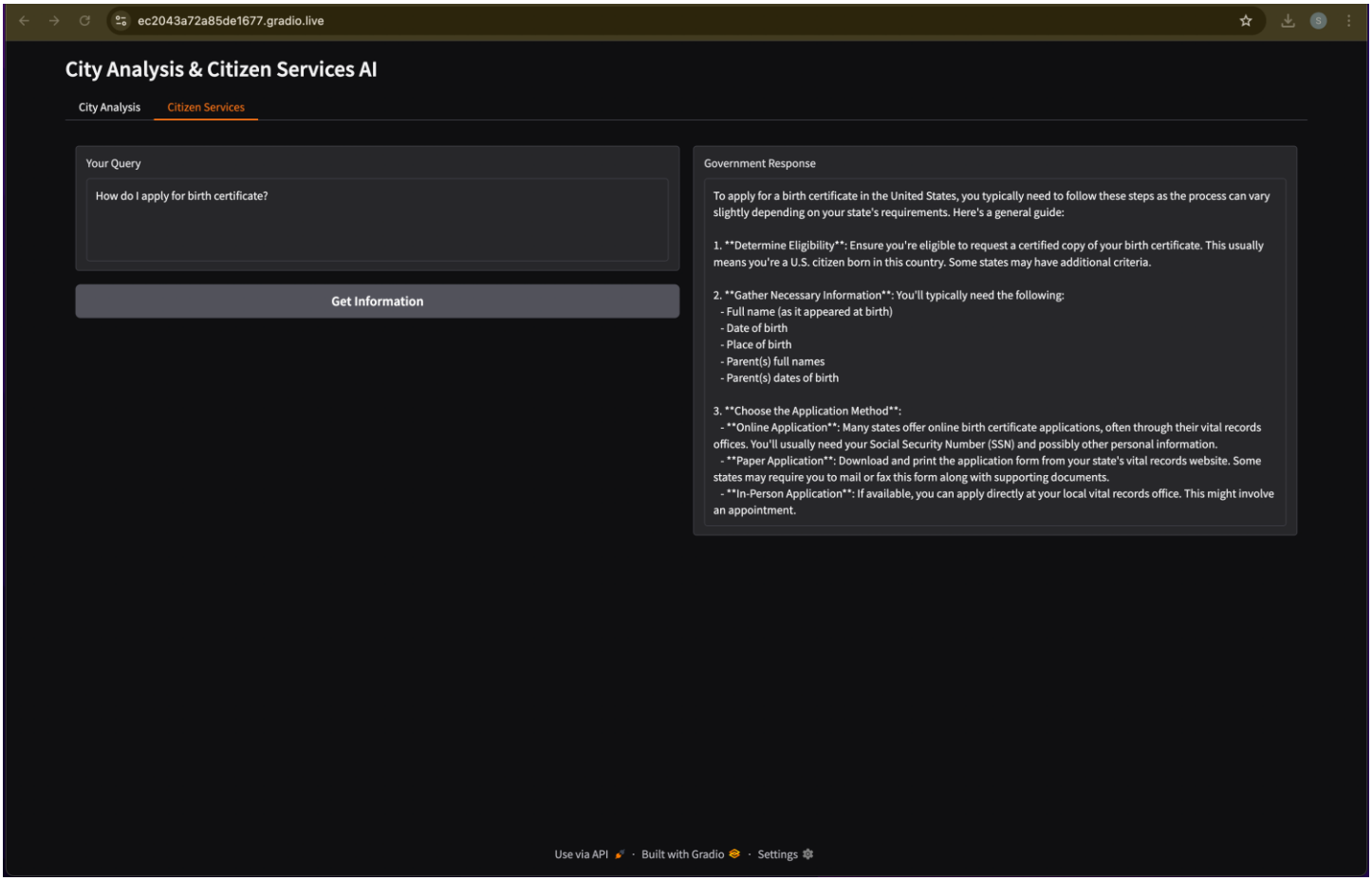
7


Citizen_AI.py
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code
+ Text
Run all
RAM
Disk
[ ]
def city_analysis(city_name):
prompt = f"Provide a detailed analysis of {city_name} including:\n1. Crime Index and safety statistics\n2. Accident rates and traffic safety information\n3. Overall saf
return generate_response(prompt, max_length=1000)
def citizen_interaction(query):
prompt = f"As a government assistant, provide accurate and helpful information about the following citizen query related to public services, government policies, or civ
return generate_response(prompt, max_length=1000)
Create Gradio interface
with gr.Blocks() as app:
gr.Markdown("# City Analysis & Citizen Services AI")
with gr.Tabs():
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...",
lines=1
)
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)
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)
app.launch(share=True)
Variables
Terminal
T4 (Python 3)

8


Citizen_AI.py
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code
+ Text
Run all
RAM
Disk
query_btn.click(citizen_interaction, inputs=citizen_query, outputs=citizen_output)
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://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart
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 datasets.
warnings.warn(
tokenizer_config.json: 8.88k/? [00:00<00:00, 870kB/s]
vocab.json: 777k/? [00:00<00:00, 28.0MB/s]
merges.txt: 442k/? [00:00<00:00, 24.8MB/s]
tokenizer.json: 3.48M/? [00:00<00:00, 94.3MB/s]
added_tokens.json: 100% 87.0/87.0 [00:00<00:00, 9.33kB/s]
special_tokens_map.json: 100% 701/701 [00:00<00:00, 63.2kB/s]
config.json: 100% 786/786 [00:00<00:00, 55.0kB/s]
`torch_dtype` is deprecated! Use `dtype` instead!
model.safetensors.index.json: 29.8k/? [00:00<00:00, 1.37MB/s]
Fetching 2 files: 100% 2/2 [02:26<00:00, 146.34s/it]
model-00001-of-00002.safetensors: 100% 5.00G/5.00G [02:25<00:00, 8.20MB/s]
model-00002-of-00002.safetensors: 100% 67.1M/67.1M [00:01<00:00, 18.7MB/s]
Loading checkpoint shards: 100% 2/2 [00:19<00:00, 8.27s/it]
generation_config.json: 100% 137/137 [00:00<00:00, 9.71kB/s]
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://ec2043a72a85de1677.gradio.live
This share link expires in 1 week. For free permanent hosting and GPU upgrades, run `gradio deploy` from the terminal in the working directory to deploy to Hugging Face Sp
City Analysis & Citizen Services AI
City Analysis
Citizen Services
Variables
Terminal
09:03
T4 (Python 3)


ec2043a72a85de1677.gradio.live
City Analysis & Citizen Services AI
City Analysis
Citizen Services
Enter City Name
Hyderabad
Analyze City
City Analysis (Crime Index & Accidents)
1. Crime Index and Safety Statistics:
- Hyderabad, being the capital of Telangana, has seen a gradual improvement in its crime rate over the years. According to the National Crime Records Bureau (NCRB) 2020 data, Hyderabad has a crime index of 337.4 per 100,000 population, which is slightly higher than the national average of 245.9 but lower than the state average of 340.7.
- Key crimes include robbery, theft, burglary, and motor vehicle theft. The city has been relatively stable in recent years, with slight fluctuations, indicating a general downward trend in crime.
- Safety statistics show that Hyderabad has 1.2 police personnel per 1,000 population, which is adequate but not particularly high compared to other Indian metropolises.
- However, specific areas within Hyderabad, such as the old city and certain slums, still face higher crime rates and insecurity issues.
2. Accident Rates and Traffic Safety Information:
- Traffic accidents are a significant concern in Hyderabad, contributing to a high number of casualties each year. According to the Telangana State Police Department, the city recorded 25,532 traffic accidents in 2019, resulting in 1,213 deaths and 10,046 severe injuries.
- Major causes of these accidents include rash driving, overloading, and violations of traffic rules by vehicles and pedestrians. Poor road infrastructure, including uncontrolled junctions, narrow roads, and lack of traffic signals, exacerbate these issues.
- The city government has implemented various initiatives such as improved road markings, installation of traffic
Use via API · Built with Gradio · Settings


ec2043a72a85de1677.gradio.live
City Analysis & Citizen Services AI
City Analysis
Citizen Services
Your Query
How do I apply for birth certificate?
Get Information
Government Response
To apply for a birth certificate in the United States, you typically need to follow these steps as the process can vary slightly depending on your state's requirements. Here's a general guide:
1. **Determine Eligibility**: Ensure you're eligible to request a certified copy of your birth certificate. This usually means you're a U.S. citizen born in this country. Some states may have additional criteria.
2. **Gather Necessary Information**: You'll typically need the following:
- Full name (as it appeared at birth)
- Date of birth
- Place of birth
- Parent(s) full names
- Parent(s) dates of birth
3. **Choose the Application Method**:
- **Online Application**: Many states offer online birth certificate applications, often through their vital records offices. You'll usually need your Social Security Number (SSN) and possibly other personal information.
- **Paper Application**: Download and print the application form from your state's vital records website. Some states may require you to mail or fax this form along with supporting documents.
- **In-Person Application**: If available, you can apply directly at your local vital records office. This might involve an appointment.
Use via API · Built with Gradio · Settings