**Citizen AI – Intelligent Citizen Engagement Platform**

|  |  |
| --- | --- |
| Date | 10 September 2025 |
| Team ID | NM2025TMID00649 |
| Project Name | Citizen AI – Intelligent Citizen Engagement Platform |
| Maximum Marks |  |

**Introduction:**

Citizen AI – Intelligent Citizen Engagement Platform is an AI-driven solution designed to assist governments, municipalities, and public organizations in providing real-time, accurate, and helpful information to citizens. By leveraging state-of-the-art large language models (LLMs) and a user-friendly interface, the platform offers insights on city safety (crime and accident statistics) and responds to citizen queries related to government services, policies, and civic issues.

The system aims to increase transparency, streamline citizen–government communication, and improve the decision-making experience for both officials and the public.

**Project Overview:**

This project integrates **Hugging Face Transformers**, **IBM Granite model**, and **Gradio** to create an interactive web-based platform. It has two major modules:

1. **City Analysis Module** – Generates detailed safety analysis, including crime index, accident rates, and overall safety assessment for any city.
2. **Citizen Services Module** – Responds to citizen queries about public services, policies, or civic issues using AI.

The application is designed with modular architecture, making it easy to add more models, APIs, or data sources in the future.

**Key Features:**

* **AI-Powered City Analysis:** Automatic generation of city safety reports based on user input.
* **Citizen Query Handling:** Real-time responses to civic and public service questions.
* **Scalable Architecture:** Backend built for easy integration with government datasets or APIs.
* **User-Friendly Interface:** Clean Gradio interface with separate tabs for City Analysis and Citizen Services.
* **Extensibility:** Future integration with dashboards, GIS maps, and real-time data feeds.

**Technology Stack:**

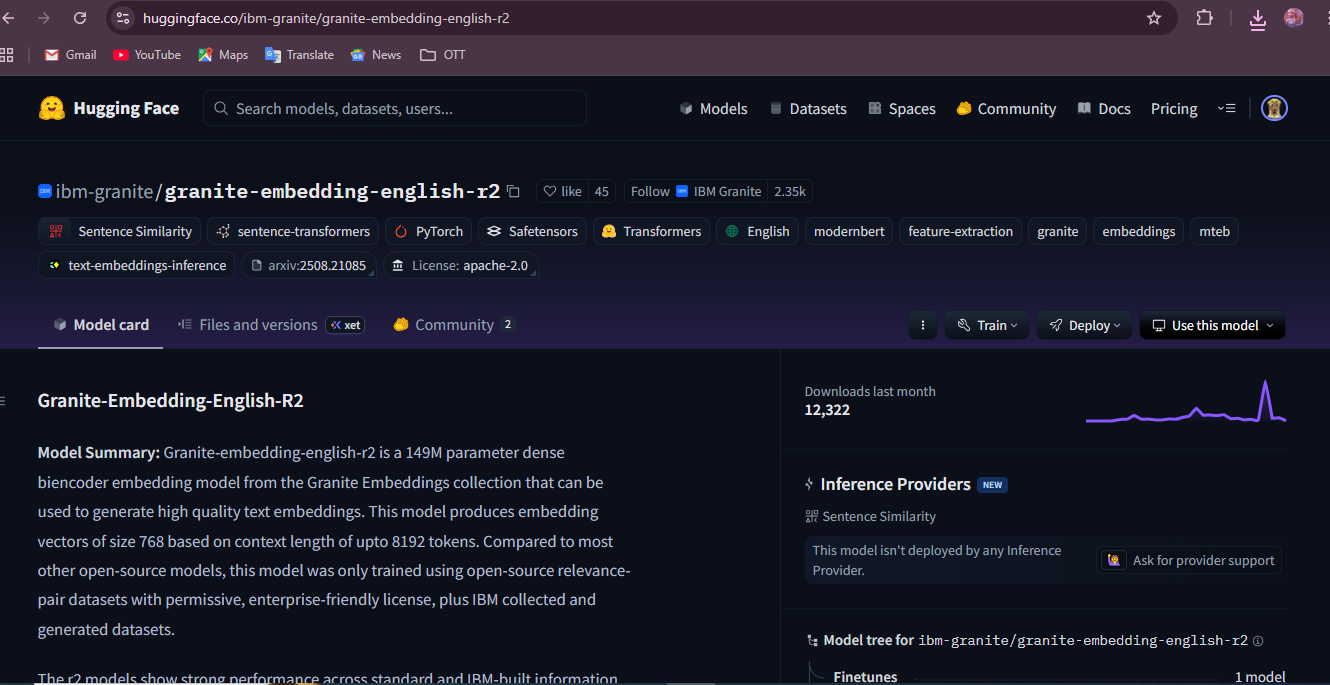
* **Programming Language:** Python 3.x
* **Frameworks & Libraries:**
  + Hugging Face Transformers
  + IBM Granite 3.2-2B Instruct Model
  + Torch (PyTorch)
  + Gradio (frontend UI)
* **Backend Framework (Planned):** Flask for routing and advanced processing
* **Deployment Environment:** Virtual environment (venv/conda), Cloud Hosting (Heroku/AWS)

**Architecture:**

The platform follows a modular architecture with three main layers:

1. **Presentation Layer (Frontend):**
   * Built using Gradio UI with tabs for different functionalities.
   * Handles user input and displays AI-generated responses.
2. **Application Layer (Business Logic):**
   * Python functions for generating prompts and passing them to the AI model.
   * Handles input preprocessing and output postprocessing.
3. **Data & Model Layer:**
   * Hugging Face model for natural language generation.
   * Future-ready for integration with live city safety datasets and government APIs.

A simplified flow:  
User Input → Gradio UI → Flask (Processing Layer) → AI Model → Response → Display on UI



**Installation & Local Setup:**

1. Clone the repository.
2. Create and activate a Python virtual environment.
3. Install dependencies:
4. pip install -r requirements.txt
5. Run the application:
6. python app.py
7. Open the generated link to access the Gradio interface.

**Application Code Overview:**

* **Model Loading:** Uses IBM Granite 3.2-2B Instruct model with dynamic GPU/CPU support.
* **Prompt Templates:**
  + city\_analysis(city\_name) – Generates safety analysis.
  + citizen\_interaction(query) – Handles citizen queries.
* **Frontend:** Gradio Blocks with Tabs for each module.
* **Future Backend:** Flask integration for handling routing, templates, and advanced logic.

**API Endpoints (Planned with Flask):**

* POST /analyze-city – Input city name, returns safety statistics.
* POST /citizen-query – Input query, returns AI-generated government response.
* GET /health – Check API health status.

**Data Handling & Security:**

* No sensitive citizen data is stored locally.
* Prompt inputs are processed in memory only.
* Future integration will support HTTPS for secure communication.
* Model responses are filtered for appropriateness where possible.

**Testing & Verification:**

* Unit tests for model prompt functions.
* UI tests for Gradio components.
* End-to-end testing planned once Flask backend is integrated.

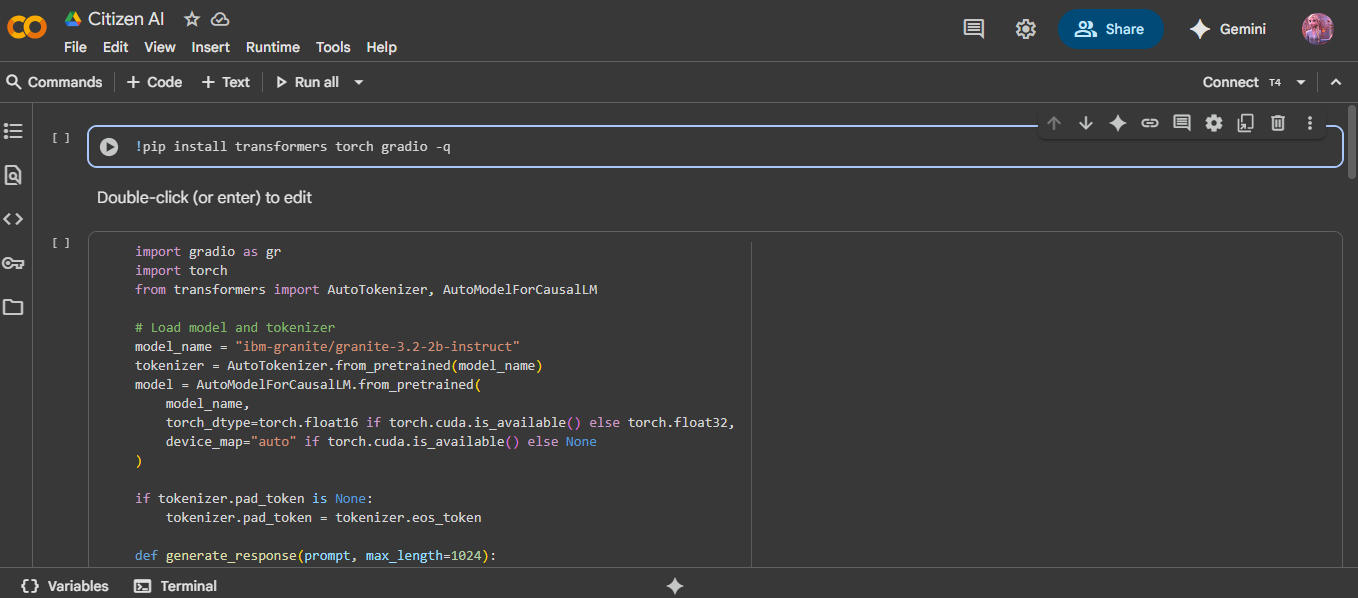
**Deployment Recommendations:**

* Host the backend on a secure server or cloud platform (AWS, Azure, GCP).
* Use containerization (Docker) for reproducible environments.
* Implement HTTPS for data privacy.
* Scale horizontally if traffic grows.

**Milestones & Activities:**

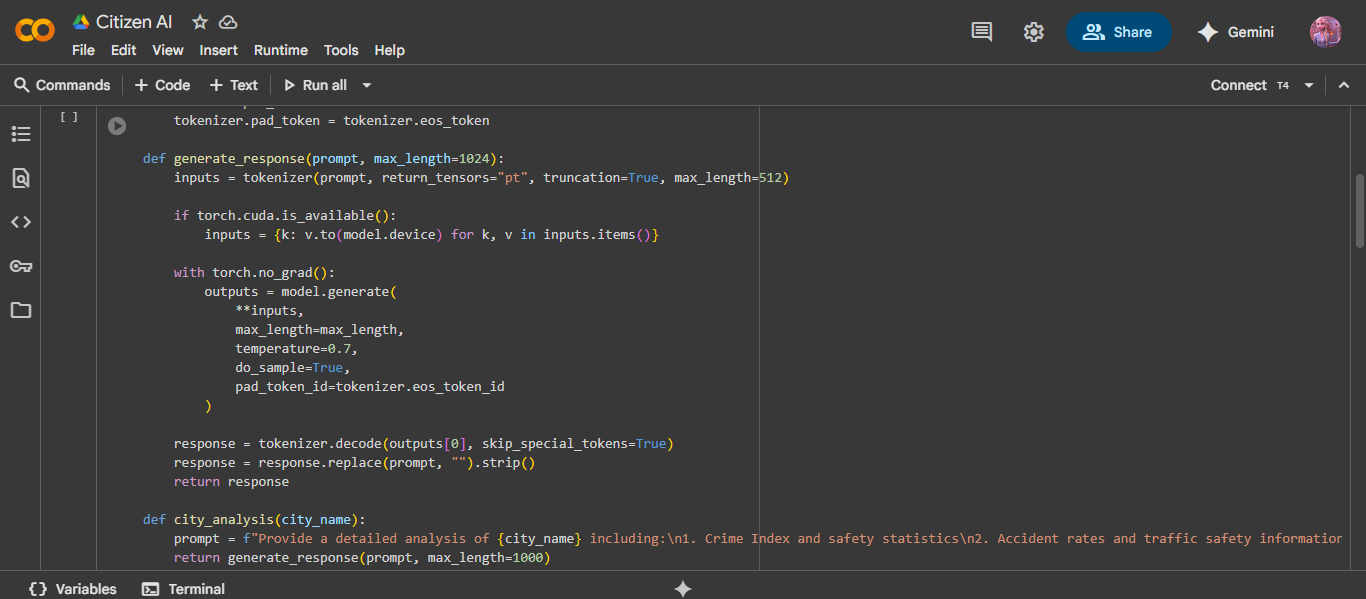
**Milestone 1: Project Setup and Architecture**

* **Activity 1.1: Select and Confirm AI Model**
  + Evaluate different models (IBM Granite, OpenAI, Falcon) and finalize the best-fit model.
* **Activity 1.2: Define the Architecture of the Application**
  + Design modular architecture for scalability.
* **Activity 1.3: Set Up the Development Environment**
  + Initialize Git repo, Python virtual environment, and install dependencies.



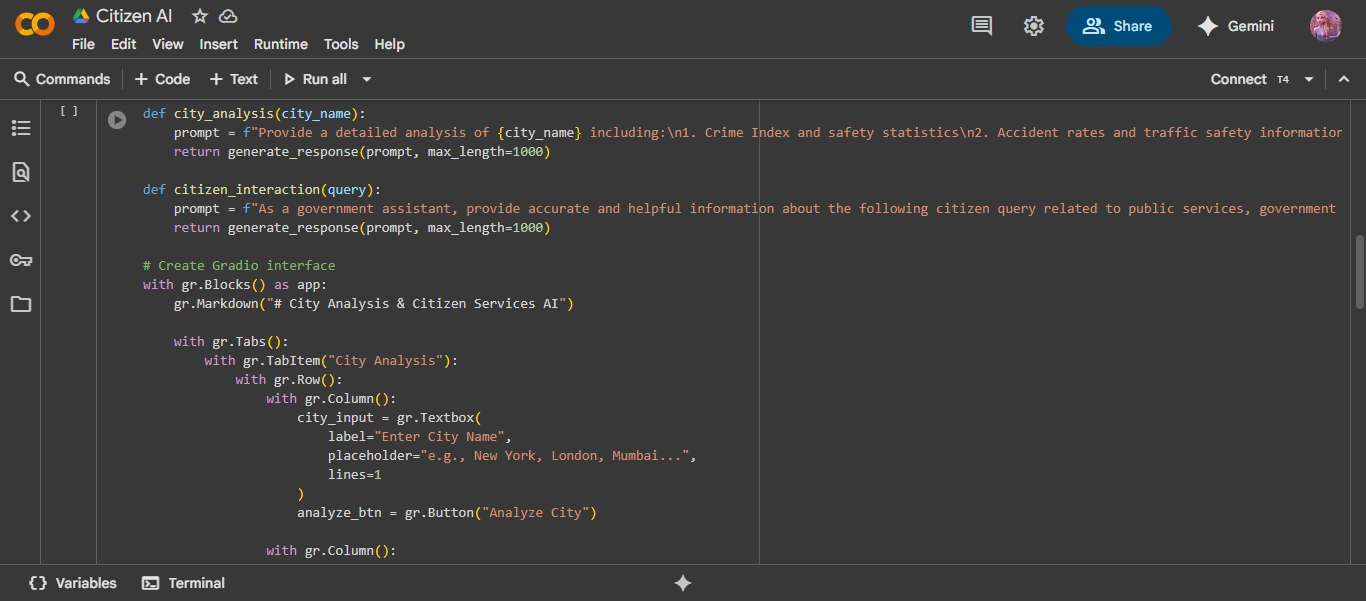
**Milestone 2: Core Functionalities**

* **Activity 2.1: Develop the Core Functionalities**
  + Implement AI prompt templates for city analysis and citizen query.
* **Activity 2.2: Implement Flask Backend for Managing Routing and User Input Processing**
  + Create REST API endpoints for both modules.
  + Handle JSON request/response.



**Milestone 3: Application Logic and Data Handling**

* **Activity 3.1: Writing the Main Application Logic in app.py**
  + Integrate model calls with backend logic.
  + Ensure error handling and response formatting.
  + Prepare data pipelines for future live datasets.



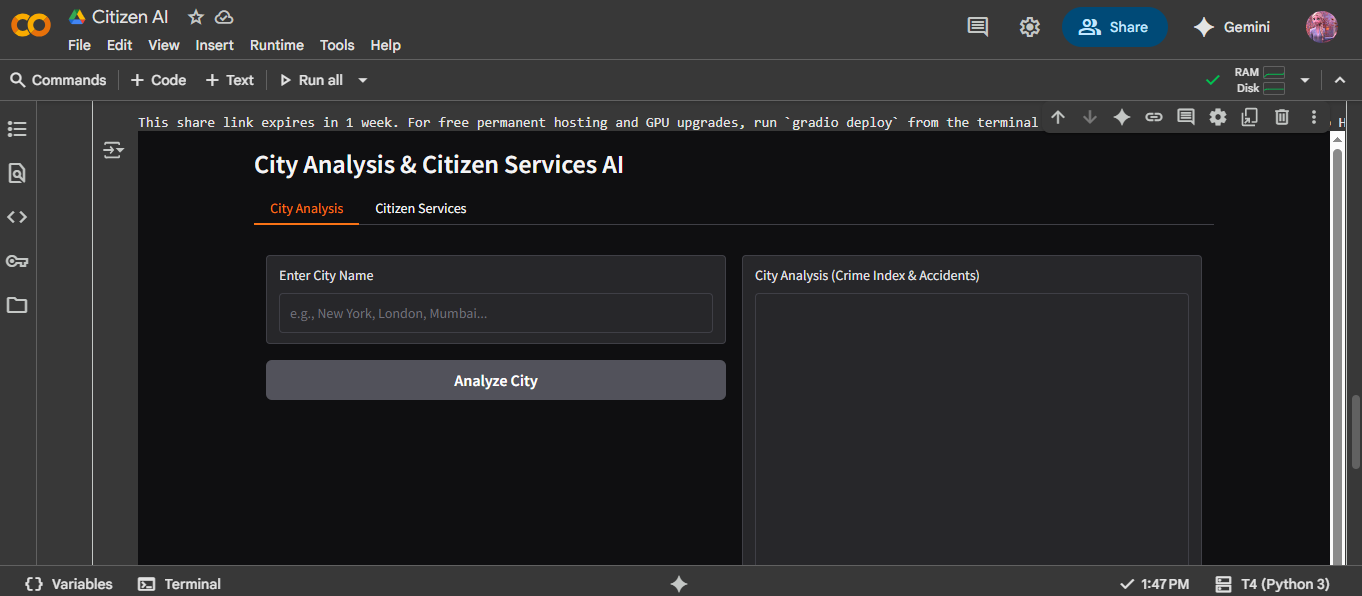
**Milestone 4: Frontend Development**

* **Activity 4.1: Designing and Developing the User Interface**
  + Use Gradio Blocks with clean layout and tabs.
* **Activity 4.2: Creating Dynamic Templates with Flask’s render\_template**
  + Build HTML templates for production UI.



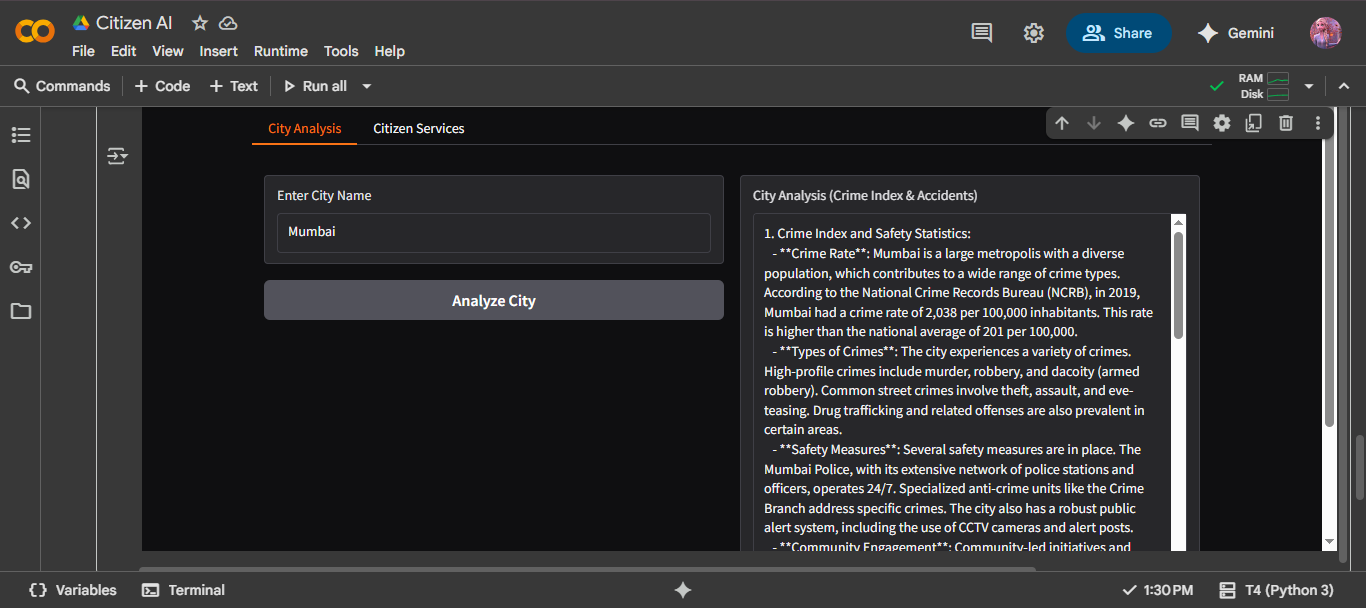
**Milestone 5: Deployment**

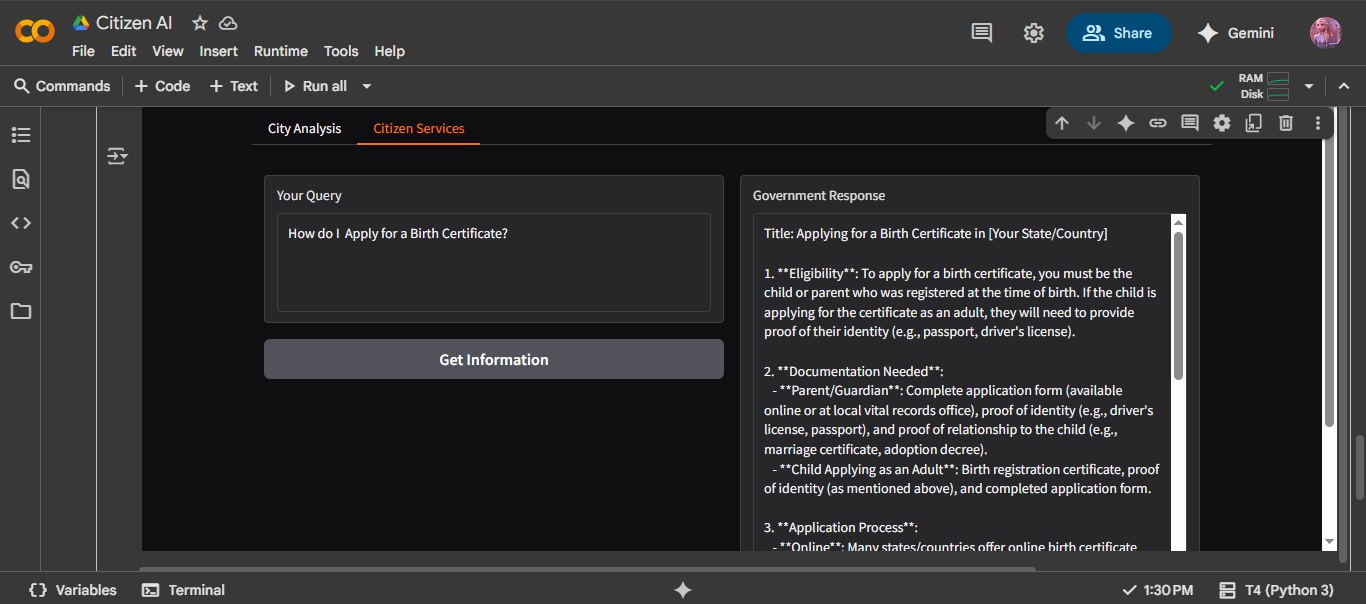
* **Activity 5.1: Set Up a Virtual Environment**
  + Ensure reproducible dependencies.
* **Activity 5.2: Testing and Verifying Local Deployment**
  + Deploy locally and verify all modules.



**Milestone 6: Functional Testing and Verification**

* Conduct functional testing for both modules (City Analysis and Citizen Services).
* Validate model outputs against sample city data and citizen queries.
* Record feedback and make improvements.





**Conclusion:**

Citizen AI – Intelligent Citizen Engagement Platform is a robust, AI-driven solution to enhance the way governments interact with citizens. By combining cutting-edge NLP models, a user-friendly interface, and modular architecture, the platform is well-positioned to scale into a comprehensive public service assistant capable of handling a wide variety of citizen needs.