**Citizen AI Project Documentation**

**1. Introduction**

* **Project Title:** Citizen AI – Intelligent Citizen Engagement Platform
* **Team Members:**
  + Janani K
  + Janani D
  + Deepika M

**2. Project Overview**

**Purpose**  
Citizen AI is an intelligent citizen engagement platform designed to revolutionize how governments interact with the public. By leveraging **Flask, IBM Granite LLM, and IBM Watson**, it provides real-time, AI-driven responses to citizen inquiries about services, policies, and civic issues. It integrates **natural language processing (NLP)** and **sentiment analysis** to assess public sentiment, track emerging issues, and generate actionable insights for government agencies.

The system improves **citizen satisfaction, government efficiency, and public trust** in digital governance by automating routine interactions and enabling data-driven policy decisions.

**Features**

* **Conversational Chat Assistant**
  + *Key Point:* Real-time citizen interaction
  + *Functionality:* Provides instant, human-like AI responses to questions on services, policies, and issues.
* **Citizen Sentiment Analysis**
  + *Key Point:* Public mood monitoring
  + *Functionality:* Classifies feedback as Positive, Neutral, or Negative to highlight satisfaction or concerns.
* **Dynamic Analytics Dashboard**
  + *Key Point:* Actionable insights
  + *Functionality:* Visualizes trends, sentiment, and reported issues for policymakers.
* **Concern Reporting**
  + *Key Point:* Transparent issue escalation
  + *Functionality:* Citizens can submit complaints/issues for tracking and resolution.

**Use Case Scenarios**

1. **Real-Time Conversational AI Assistant**
   * Citizens ask questions via chat; AI provides immediate, accurate responses.
2. **Citizen Sentiment Analysis**
   * Feedback is processed, classified by sentiment, and aggregated for decision-makers.
3. **Dynamic Dashboard**
   * Officials view sentiment trends, service ratings, and issue logs in real time.

**3. Architecture**

* **Frontend:** HTML/CSS templates (index, about, services, chat, dashboard, login).
* **Backend (Flask):** Manages routes, user authentication, and data processing.
* **LLM Integration (IBM Granite):** Provides conversational AI, text generation, and sentiment analysis.
* **Data Handling:** In-memory storage of chat history, sentiment results, and reported concerns (with plans for database integration).
* **Visualization:** Dynamic dashboard with charts and metrics for sentiment and issue tracking.

**4. Setup Instructions**

**Prerequisites**

* Python 3.7+
* Flask
* PyTorch (with CUDA for GPU acceleration)
* Hugging Face libraries: transformers, accelerate, bitsandbytes
* Hardware:
  + 16GB+ RAM
  + NVIDIA GPU with 8GB+ VRAM recommended
* Internet connection (for first-time model download)

**Installation Process**

1. Clone repository and set up project structure (app.py, templates/, static/).
2. Create and activate a virtual environment:
3. python -m venv env
4. source env/bin/activate  # Linux/Mac
5. env\Scripts\activate     # Windows
6. pip install -r requirements.txt
7. Install Flask, PyTorch, and Hugging Face dependencies.
8. Configure IBM Granite model path (ibm-granite/granite-3.3-8b-instruct).
9. Run the Flask backend with:
10. python app.py

**5. Folder Structure**

app.py                – Main Flask application

templates/            – HTML templates (index, about, services, chat, dashboard, login)

static/               – CSS, Images, Favicon

requirements.txt      – Python dependencies

**6. Running the Application**

* Launch Flask backend (python app.py).
* Open browser and navigate to http://localhost:5000.
* Use navigation menu for:
  + **Chat:** Interact with the AI assistant.
  + **Feedback:** Submit text for sentiment analysis.
  + **Dashboard:** View real-time citizen insights.
  + **Login:** Authenticate to access protected content.

**7. API Endpoints**

* POST /ask – Submit citizen questions for AI-generated responses.
* POST /feedback – Submit feedback for sentiment analysis.
* POST /concern – Report issues/concerns.
* GET /dashboard – View aggregated sentiment and issues.
* POST /login – User authentication.
* POST /logout – Session termination.

**8. Authentication**

* User login with session management.
* Planned: Role-based access (admin, citizen, government official).

**9. User Interface**

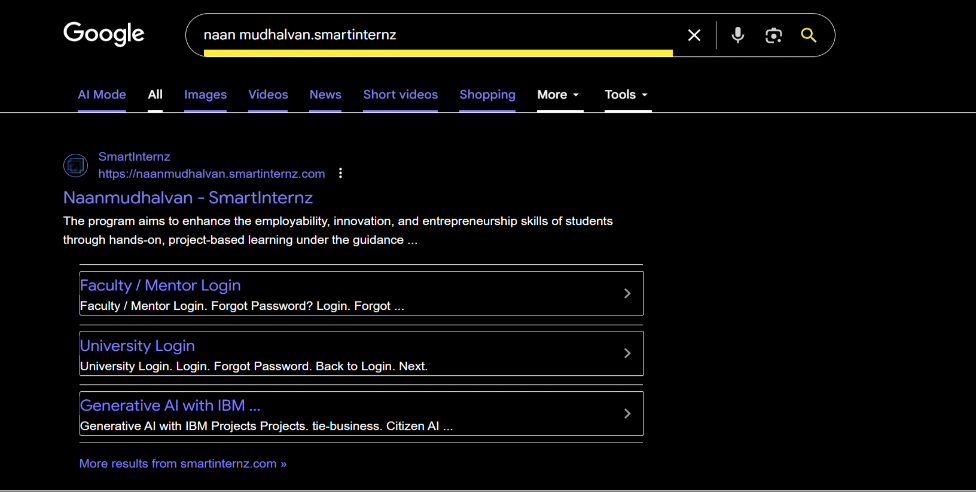
* **Index Page:** Welcome screen with intro and “Get Started” button.
* **Login Page:** Secure authentication form.
* **About Page:** Explains project mission, features, and benefits.
* **Chat Page:** AI-powered conversational assistant.
* **Dashboard:** Sentiment analysis (positive/neutral/negative) and recent citizen issues.

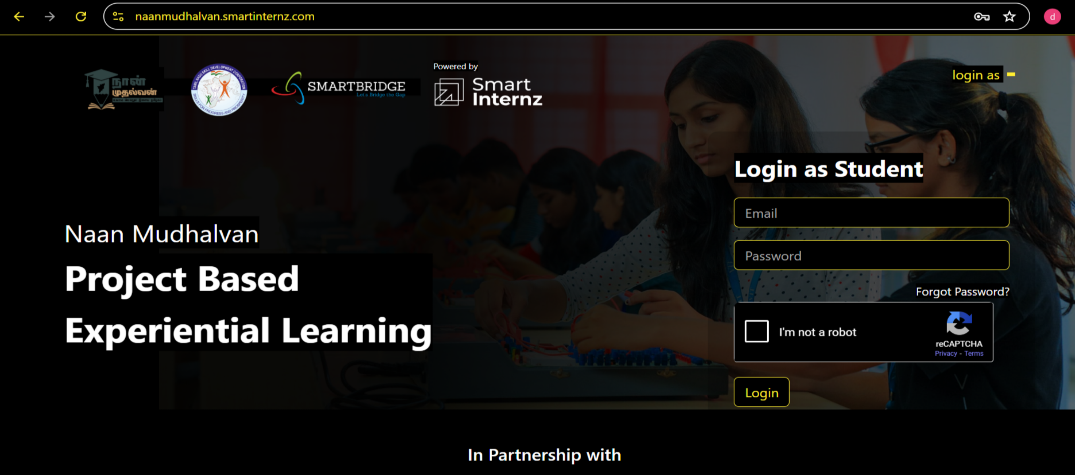
**10. Testing**

* **Unit Testing:** Flask routes and AI functions.
* **Integration Testing:** End-to-end flow for chat, feedback, and dashboard.
* **Manual Testing:** Form submission, sentiment results, and issue reporting.
* **Edge Cases:** Empty input, invalid login, malformed data.

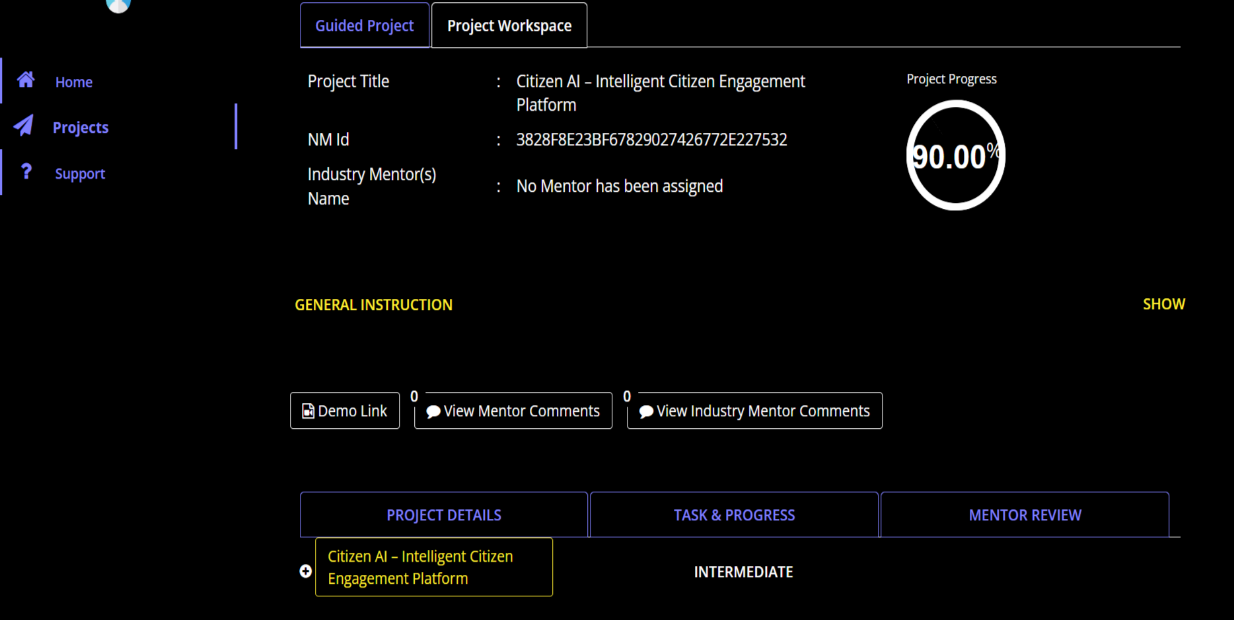
**11. Screenshots**

● Search for “Naan Mudhalavan Smart Interz” Portal in any Browser.



Smartinternz Login Page Smartinternz Login Page Smartinternz Login Page

Smartinternz Login Page



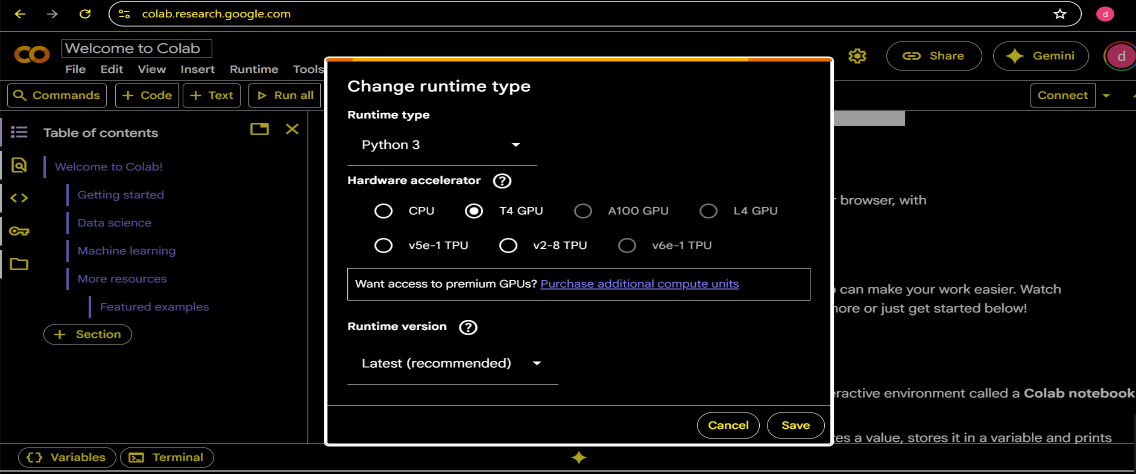
Smartinternz dashboard



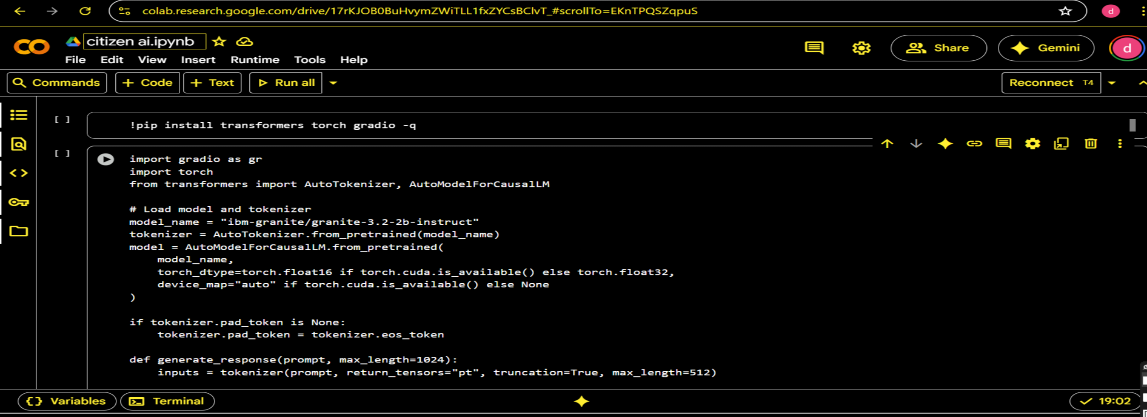
“granite-3.2-2b-instruct. In hugging face”

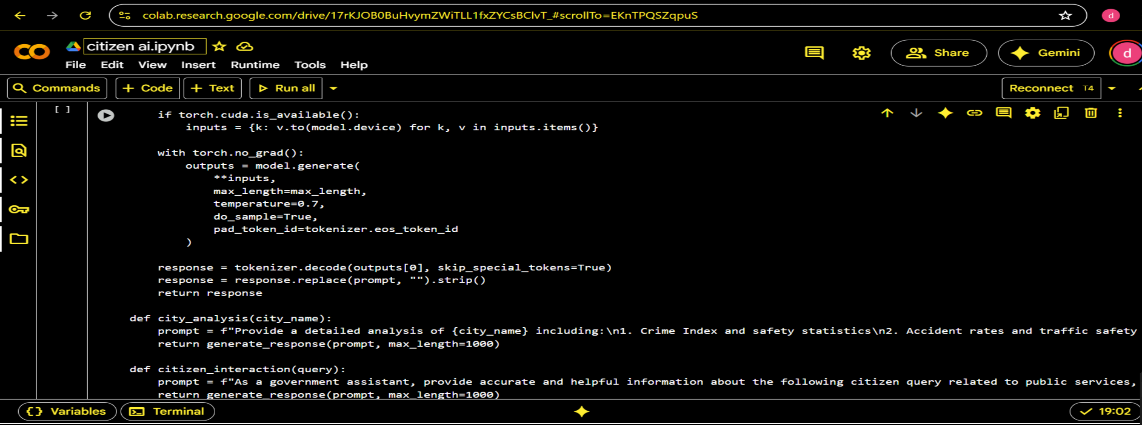


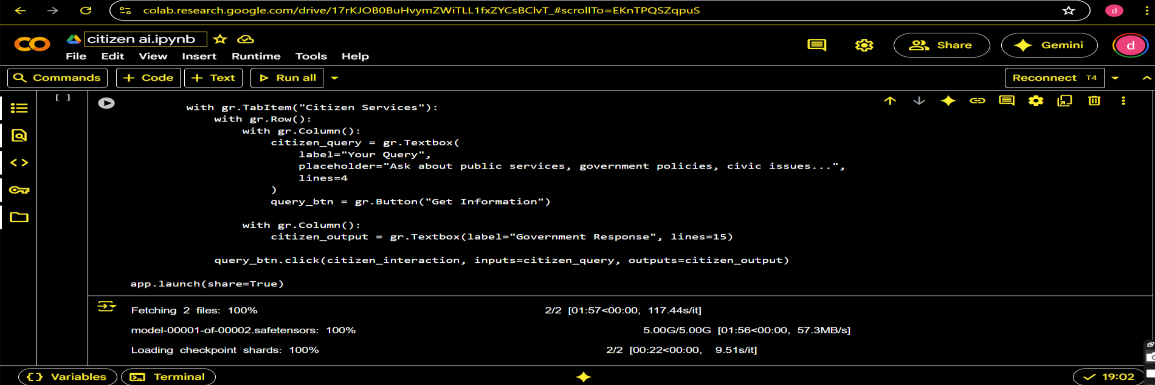
New Notebook notebook page in Google colab



Choose “T4 GPU” and click on “Save”



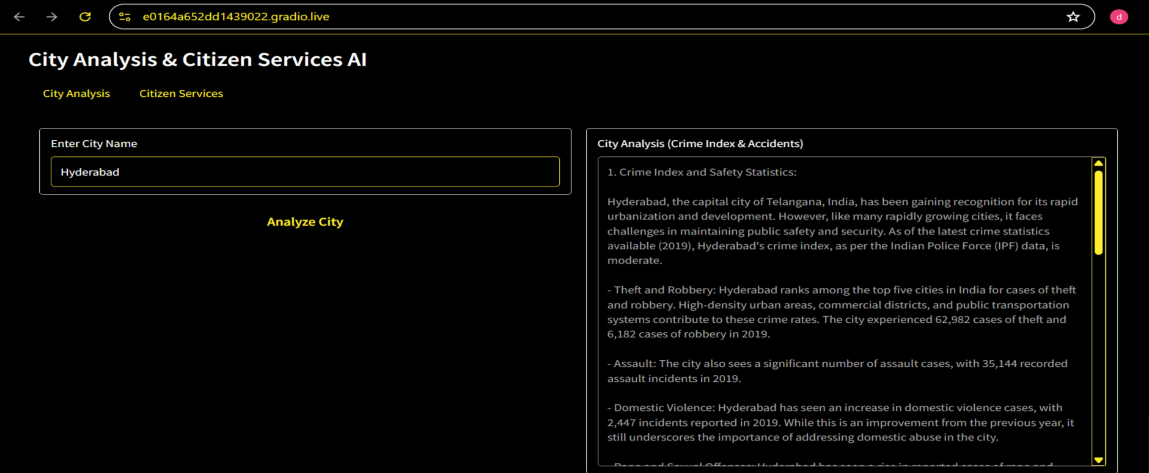


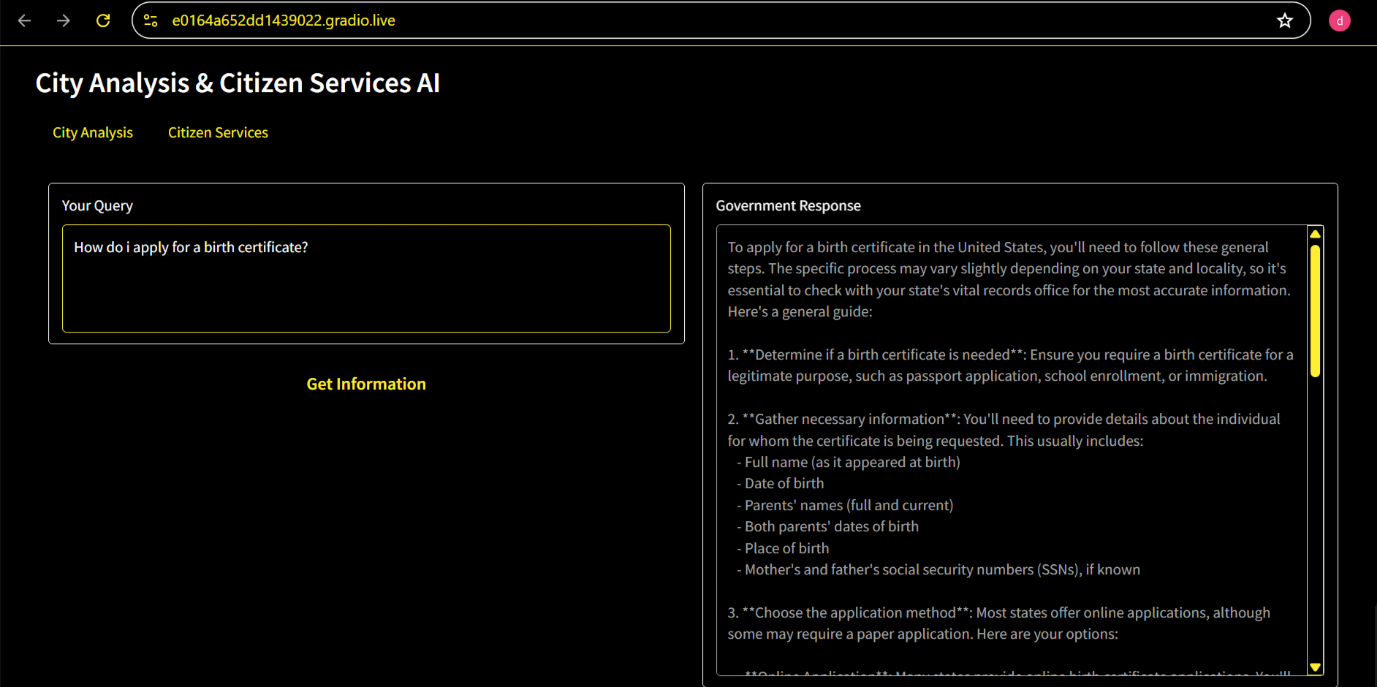


Citizen Ai Coding page



● Click on the URl to open the Gradio Application click on the link.





Citizen Ai Home Page

**12. Known Issues**

* Limited persistence (data stored in memory).
* CPU-only execution is very slow.Limited scalability until database integration.

**13. Future Enhancements**

* Database integration for persistent storage.
* Multi-language support.
* Mobile-friendly responsive design.
* Advanced NLP models for policy summarization.
* Integration with social media/public forums for sentiment mining.