CITIZEN AI - INTELLIGENT CITIZEN ENGAGEMENT PLATFORM

Project Documentation

1. Introduction

Project title: CITIZEN AI - INTELLIGENT CITIZEN ENGAGEMENT PLATFORM

Team members:

Srayaa G (Team lead)

Sujithra S

Swega KV

Uma Maheshwari B

2. Project Overview

Purpose:

The purpose of **Citizen AI – Intelligent Citizen Engagement Platform** is to enhance communication between citizens and the government through AI-driven services. It offers quick grievance redressal, personalized assistance, and multilingual support for inclusivity. By analyzing citizen feedback, it enables transparent and data-driven governance. The platform encourages active participation and easy access to public services. Overall, it makes governance smarter, efficient, and citizen-centric.

Features:

- 1) AI Chatbot
- Functionality: Provides instant replies to citizen queries, guides them to services, and reduces waiting time.
- 2) Grievance Redressal System
- Functionality: Citizens can submit complaints (e.g., water, roads, electricity) and track progress until resolved.
- 3) Multilingual & Voice Support
- Functionality: Enables people to interact in their local language or through voice commands for inclusivity.
- 4) Personalized Dashboard
- Functionality: Shows relevant schemes, notifications, and updates tailored to each citizen's needs.
- 5) Feedback & Polls
- Functionality: Collects opinions and suggestions from citizens, helping governments understand public sentiment.
- 6) Predictive Analytics
- Functionality: Analyzes patterns in complaints and feedback to predict and prevent recurring issues.
- 7) Secure & Scalable Platform
- Functionality: Protects citizen data with authentication and handles large-scale users without slowdown.
- 8) Mobile & Web Access
- Functionality: Provides convenient access anytime, anywhere through smartphones or computers.

3. Architecture

The architecture of CitizenAI is designed in layered form to ensure smooth citizen engagement and intelligent processing.

At the top, users interact through multiple channels like web portals, mobile apps, chatbots, and SMS. These inputs are handled by the Flask Application Layer, which connects to the AI & Processing Layer powered by IBM Granite Model for natural language understanding, and sentiment analysis for issue classification.

The Data Management Layer supports this with a knowledge base, citizen profiles, and an analytics database. Finally, a Flask-based reporting dashboard provides insights and analytics for better decision-making.

4. Setup Instructions

Prerequisites:

Python 3.7+ – Required to run the project.

Flask – Web framework for building the application.

PyTorch – To run the deep learning model (install CUDA version for GPU support).

Hugging Face Libraries – transformers, accelerate, bitsandbytes for IBM Granite model loading and quantization.

Hardware –

- RAM: 16GB+ recommended
- GPU: NVIDIA GPU with 8GB+ VRAM (for faster inference), CUDA drivers installed
- CPU-only is possible but very slow

Internet – Needed to download model files from Hugging Face Hub.

Project Structure –

- app.py (main app)
- templates/ (HTML files: index, about, services, chat, dashboard, login)
- static/ (CSS, images, favicon)
-

Installation:

- • Python 3.7+ – Programming environment.
- • Flask – Web framework for backend development.
- • PyTorch – Deep learning framework (with optional CUDA support for GPU).
- • Hugging Face Libraries –
- transformers
- accelerate
- bitsandbytes
- • CUDA Toolkit + NVIDIA Drivers (if using GPU) – For GPU acceleration with PyTorc
-

5. Folder Structure

project-root/ │── static # Main application script │── templates # Dependencies │── app.py # Documentation

6. Running the Application

- Create a virtual environment for the project.
- Activate the virtual environment.

- Install all required libraries from the requirements file.
- Run the Flask application using python app.py.

7. API Documentation

Currently only available through the Gradio UI. Future versions may expose FastAPI endpoints.

8. Authentication

Open app runs without authentication for demo purposes.

Future enhancements may include login, role-based access, and teacher/student accounts.

9. User Interface

Home Page (Index): Welcome and navigation links.

About Page: Information about the platform.

Services Page: List of available citizen services.

Chat Interface: AI chatbot for real-time interaction.

Dashboard: Analytics and summaries for authorities.

Login/Registration Page: User authentication and session management.

Static Elements: Navigation bar, footer, and input forms.

10. Testing

Unit Testing: Check individual functions like AI response generation and data handling.

Integration Testing: Verify that frontend (HTML/CSS) and backend (Flask + AI) work together.

User Flow Testing: Test login, chat, feedback submission, and dashboard navigation.

Performance Testing: Ensure AI responses are fast and system handles multiple users.

Debugging: Identify and fix errors in backend, frontend, or AI modules.

Final Validation: Confirm the app works smoothly before deployment.

11. Screenshots

(*To be added after running the app and capturing UI screenshots.*)

12. Known Issues

High resource usage – Large AI models require lots of RAM/VRAM.

Slow CPU performance – Responses are slower without GPU.

Compatibility errors – Mismatched library or CUDA versions can cause failures.

Chatbot limitations – AI may give incorrect or irrelevant answers.

Data handling & security – Loss of chat history or user data vulnerability.

Scalability & performance – May struggle with many concurrent users.

Language & accessibility issues – Multilingual or disability support may be limited.

13. Future Enhancements

Database Integration – Store chat history, feedback, and user data securely for persistence.

Multilingual & Accessibility Improvements – Support more languages, dialects, and accessibility features like screen readers and voice input.

Cloud Deployment & Scalability – Host on cloud platforms to handle large numbers of users efficiently.

Performance Optimization – Optimize AI inference for faster responses on CPU and GPU.

Analytics & Reporting – Add dashboards for authorities to monitor trends, complaints, and citizen sentiment.

Security Enhancements – Encrypt data, implement authentication, and safeguard user privacy.