Citizen AI – Digital Citizen Support System

Project Documentation

---

1. Introduction

Project Title: CitizenAI – Digital Citizen Support System

Team Members: Santhosh K, Sanjay Kumar, Shiva C, Shyam Akash

CitizenAI is developed to enhance the way people interact with government offices and public service departments. By using artificial intelligence, the platform creates a smarter way for resolving questions, collecting citizen feedback, and providing administrators with real-time insights. The system focuses on transparency, efficiency, and

strengthening the connection between authorities and citizens.

---

2. Project Overview

Purpose:

The aim of CitizenAI is to provide a modern, AI-assisted communication channel between the public and governing authorities. It allows citizens to raise issues, share feedback, and

ask queries, while enabling officials to analyze responses and address concerns quickly.

Key Features:

Virtual Assistant (Chatbot):

Highlights: Human-like conversation using AI.

Usage: Helps citizens get instant answers for queries anytime.

Opinion & Mood Analysis:

Highlights: Sentiment detection for collected feedback.

Usage: Sorts feedback into positive, neutral, or negative, making it easier for officials to judge overall public opinion.

Complaint Submission:

Highlights: Direct digital issue reporting.

Usage: Citizens can log problems, which are tracked until resolved.

Insight Dashboard:

Highlights: Graphs and statistics for decision-making.

Usage: Gives administrators a visual summary of concerns, queries, and sentiments.

Authentication System:

Highlights: Secure login and logout.

Usage: Ensures authorized access for both citizens and officials.

---

3. System Design & Architecture

Frontend (HTML/CSS/Bootstrap + Flask): Provides responsive and easy-to-use interfaces for chatting, reporting issues, and giving feedback.

Backend (Flask Framework): Handles data flow between the interface, AI components, and the database.

AI Engine (NLP with IBM Granite): Understands user input, detects intent, and performs sentiment classification.

Database (MySQL/SQLite): Stores citizen data, queries, complaints, and analysis records securely.

Analytics Layer: Generates visual reports and insights for administrators.

---

4. Setup Guidelines

Requirements:

Python 3.9 or higher

Virtual environment with pip

Flask and libraries from requirements.txt

IBM Granite API key for NLP features

Steps:

1. Download and extract project files.

2. Set up a virtual environment and activate it.

3. Install packages via pip install -r requirements.txt.

4. Configure .env with API credentials and DB details.

5. Start the project with python app.py or python app_demo.py.

6. Access from browser at: http://127.0.0.1:5000.

---

5. Folder Structure

```
CitizenAI/
  ├──── app.py            → Main program
  ├──── app_demo.py       → Demo version
```

├── requirements.txt → Dependencies

├── templates/ → User interfaces (HTML)

├── static/ → CSS, JS, and assets

├── screenshots/ → Output images

└── README.md → Documentation

---

6. Running the Project

Activate environment

Install required modules

Run app.py

Open http://127.0.0.1:5000 in browser

Use Chatbot, Feedback, and Dashboard sections

---

7. API Endpoints

POST /chat → Receives queries and responds with AI-generated answers

POST /submit-feedback → Saves feedback and detects sentiment

POST /report-concern → Logs problems for officials

GET /dashboard → Shows reports and analytics

---

8. Authentication

Basic session-based login/logout available.

Can be extended with:

JWT (JSON Web Tokens)

OAuth2 integration

Role-based permissions (admin/user)

Activity logs

---

9. User Interface

The UI is built for simplicity and mobile responsiveness.

Navigation menu for quick access

Chat window for AI queries

Feedback form for citizens

Dashboard with charts and reports

---

10. Testing Process

Unit Testing: Validating chatbot and utilities

API Testing: Checked endpoints with Postman

Manual Testing: Verified real user interactions

Edge Case Testing: Empty inputs, invalid entries, large text inputs

---

11. Current Limitations

High-end AI models may need better hardware

Demo version relies on temporary database storage

Voice and multi-language features not yet included

---

12. Future Improvements

Voice support for chatbot

Multi-language capabilities

Mobile app version for wider use

Advanced admin control panel

Deployment on cloud for scalability