# PROJECT TITLE

**CITIZEN AI PROJECT DOCUMENTATION**

# 1.Introduction

* **Project title : Citizen Ai**
* **Team member :**K. Priyadharshini
* **Team member :**K.S. Sangeetha
* **Team member :**S. Thamizharasi
* **Team member :**T. Terasa Helen Jyothi

# 2.project overview

* ***Purpose :***

The purpose of Citizen AI is to empower individuals with a digital assistant that improves civic engagement, personal well-being, and access to public services. It assists people in navigating government services, staying informed about policies and rights, participating in community discussions, and receiving personalized recommendations for their well-being, education, and career. For governments, it serves as a tool to understand public sentiment, improve communication, and co-create solutions with citizens, thereby bridging the gap between civic engagement, digital inclusion, and community resilience.

• **Features:**

***Conversational Interface***

Key Point: Human-like interaction

Functionality: Citizens can ask questions about services, policies, or community programs in natural language and get actionable responses.

***Service Navigator***

Key Point: Easy access to public services

Functionality: Guides users step by step through applications (healthcare, IDs, permits, benefits).

***Policy & Rights Simplifier***

Key Point: Plain-language explanations

Functionality: Converts legal and government documents into easy-to-understand summaries.

***Citizen Feedback & Polling***

Key Point: Civic voice amplification

Functionality: Collects public opinions, conducts micro-surveys, and channels aggregated insights to policymakers.

***Community Resource Finder***

Key Point: Localized guidance

Functionality: Recommends nearby services like clinics, schools, shelters, or training centers.

***Well-being & Education Coach***

Key Point: Personal growth support

Functionality: Provides personalized tips on health, financial literacy, and learning resources.

***Anomaly & Trend Detection***

Key Point: Data-driven monitoring

Functionality: Identifies emerging community issues (e.g., rising unemployment or health complaints).

***Multimodal Input Support***

Key Point: Flexible interactions

Functionality: Accepts text, voice, PDFs, and images for service navigation or policy explanation.

***Streamlit or Gradio UI***

Key Point: Accessible design

Functionality: Offers a simple dashboard for citizens with tabs for chat, resources, polls, and reports.

# 3. Architecture

***Frontend (Streamlit/Gradio):***

Interactive web interface with navigation sidebar, chat window, service guides, survey forms, and local resource maps.

***Backend (FastAPI):***

Serves as the core REST API, handling chat interactions, service queries, feedback collection, and report generation.

***LLM Integration (IBM Watsonx Granite or OpenAI GPT):***

Natural language processing for summaries, translations, and dialogue. Prompts are tuned for clarity, inclusivity, and neutrality.

***Vector Search (Pinecone):***

Stores embedded government documents, FAQs, and service guides. Enables semantic search for policy-related queries.

***ML Modules:***

Forecasting: Tracks trends in citizen engagement or service demand.

Anomaly Detection: Flags unusual patterns in feedback or usage data.

**4. Setup Instructions**

***Prerequisites:***

Python 3.9+

pip and venv

API keys (IBM Watsonx / OpenAI, Pinecone, optional geolocation services)

Internet access

Installation Process:

1. Clone the repository

2. Install dependencies from requirements.txt

3. Configure .env with credentials

4. Run FastAPI backend

5. Launch Streamlit/Gradio frontend

6. Interact with chat, polls, and service navigation modules

## 5. Folder Structure

app/ # FastAPI backend

└── api/ # Chat, feedback, services, resources

ui/ # Frontend pages

citizen\_dashboard.py # Streamlit/Gradio main entry

policy\_helper.py # Simplifies government docs

resource\_locator.py # Finds local services

trend\_forecaster.py # Forecasts community issues

anomaly\_checker.py # Detects irregular data patterns

feedback\_analyzer.py # Processes citizen input

## 6. Running the Application

* Start FastAPI server for backend endpoints
* Launch Streamlit/Gradio dashboard for UI
* Use sidebar navigation to:
* Chat with Citizen AI
* Access service navigator
* View summaries of policies
* Participate in polls/feedback
* Download reports

## 7. API Documentation

***Key Endpoints:***

* POST /chat/ask → Responds to citizen queries
* GET /get-services → Lists relevant government/public services
* POST /upload-doc → Summarizes or indexes a policy document
* GET /search-docs → Semantic search of policies
* POST /submit-feedback → Captures citizen feedback/poll responses
* GET /community-trends → Provides forecast insights

**8. Authentication**

## Demo Mode: Open access

## Secure Deployment:

## JWT tokens or API keys

## OAuth2 for government/NGO logins

## Role-based access (citizen, admin, researcher)

## Future: Personal accounts with history tracking

## 9. User Interface

* Sidebar navigation (chat, services, resources, feedback)
* Card-based service highlights (health, education, ID services)
* Real-time forms for polls and surveys
* Downloadable summaries/reports
* Accessibility-first design (large fonts, multilingual support, voice input)

## 10. Testing

* Unit Testing: NLP summarization, utility functions
* API Testing: Swagger UI, Postman
* Manual Testing: Service navigation, chat quality, feedback forms
* Edge Cases: Complex legal docs, invalid inputs, large uploads

**11.screen shots**

(To be added after UI prototype)

**12.Known Issues**

* Limited offline capability
* Some complex legal texts may lose nuance in summarization
* Local service locator dependent on available geodata

**13. Future Enhancements**

* Voice-based assistant (speech-to-text + TTS)
* Mobile app integration
* Multi-language support for all major regional languages
* AI-driven community dashboards for policymakers
* Integration with smart city IoT data for real-time citizen alerts