CITIZEN-AI

Project Documentation

1. Introduction

1.Project title: Citizen-AI
2.Team Leader: UMAMAHESWARI.S
3.Team Member: NAVEENA.M
4. Team Member: CHITHRA.S
5. Team Member: ATCHAYA.S
6. Team Member: KESIKA.K

1. project overview

- Purpose :

1. Enhance Citizen Engagement: Provide a platform for citizens to interact with government services efficiently.

2. Improve Public Services: Offer real-time responses to civic queries, improving the overall citizen experience.

3. Foster Transparency: Visualize engagement insights and sentiment data, promoting transparency in governance.

2. Feature

1. **Real-Time AI Chatbot: Answers civic queries efficiently.**
2. **Citizen Sentiment Analysis: Understands public mood from feedback.**
3. **Interactive Dashboard: Visualizes feedback trends and sentiment data.**
4. **Context-Aware Responses: Provides smarter, personalized replies.**
5. **Modular Flask Backend: Ensures clean architecture and maintainability.**

Conversational Interface

The conversational interface is a key component of Citizen-AI, enabling citizens to interact with government services through a natural language interface. The AI-powered chatbot provides:

1. Real-time responses: Citizens receive immediate answers to their queries.

2. Personalized interactions: The chatbot understands context and provides tailored responses.
3. Easy accessibility: Citizens can access government services through a user-friendly interface.

Policy Summarization

1. Concise Summaries: AI-powered tools generate concise summaries of policy documents, highlighting main themes and critical information.

2. Automated Generation: Advanced algorithms analyze text patterns, frequency of key terms, sentence structure, and contextual relationships to identify crucial information.

3. Customizable: Some tools allow users to adjust summary length and format, such as bullet points or paragraphs.

Resource Forecasting

1. Allocate resources efficiently: Predict the resources needed to deliver public services, ensuring optimal allocation.

2. Anticipate budget requirements: Estimate budget needs for policy implementation, enabling better financial planning.

3. Improve resource utilization: Identify areas of inefficiency and optimize resource use.

Anomaly Detection

1. Proactive issue resolution: Identify and address issues before they escalate.

2. Enhanced security: Detect potential security threats or breaches.

3. Data-driven decision-making: Inform policy decisions with insights from anomaly detection.

2. Architecture

The architecture of Citizen-AI is designed to support its core functionalities, including:

1. Data Ingestion: Collecting data from various sources, such as citizen interactions, public services, and feedback.

2. Data Processing: Analyzing and processing data using machine learning algorithms and natural language processing.

3. Anomaly Detection: Identifying unusual patterns or outliers in data.

4. Resource Forecasting: Predicting resource requirements for policy implementation.

5. Conversational Interface: Providing a user-friendly interface for citizens to interact with government services.

Key Components:

1. Backend: Built using Python and Flask, providing a robust and scalable architecture.

2. Frontend: Utilizes HTML, CSS, and JavaScript (Bootstrap) for a responsive and intuitive user interface.

3. AI Integration: Leverages IBM Granite models and IBM Watson for advanced AI capabilities.

Benefits:

1. Scalability: Designed to handle large volumes of data and user interactions.

2. Flexibility: Modular architecture allows for easy integration of new features and services.

3. Security: Robust security measures protect citizen data and ensure secure interactions.

3. Setup Instructions

1. Clone the repository: Clone the Citizen-AI repository from GitHub.

2. Install dependencies: Run pip install -r requirements.txt to install dependencies.

3. Configure environment variables: Set environment variables for IBM Watson and IBM Granite models.

4. Run the application: Run python app.py to start the Citizen-AI application.

5. Access the application: Access the application through the frontend interface.

4. Folder Structure

1. app: Application code, including backend and frontend.
2. config: Configuration files, including environment variables and settings.
3. models: IBM Granite models and machine learning algorithms.
4. templates: HTML templates for the frontend interface.
5. static: Static files, including CSS, JavaScript, and images.
6. requirements.txt: List of dependencies required to run the application.
7. (link unavailable): Main application file, responsible for running the application.
8. logs: Application logs, used for troubleshooting and debugging.

5. Running the Application

1. Navigate to the project directory: Open a terminal or command prompt and navigate to the Citizen-AI project directory.

2. Install dependencies: Run pip install -r requirements.txt to install the required dependencies.

3. Run the application: Run python app.py to start the Citizen-AI application.

4. Access the application: Open a web browser and access the application at http://localhost:5000 (or the specified port).

6. Authentication

Citizen-AI uses authentication to ensure secure access and protect citizen data. Key features include:

1. User registration: Citizens register for an account.

2. Login: Citizens log in using credentials.

3. Session management: Secure access to citizen data.

Benefits:

1. Secure access

2. Data protection

3. Personalized experience

7.User Interface

Citizen-AI's UI is designed for a seamless citizen experience, featuring:

1. User-friendly design

2. Conversational interface

3. Personalized experience

Benefits:

1. Easy access to services

2. Improved user experience

3. Increased engagement

Design Principles:

1. **Simple and intuitive**
2. **Responsive design**
3. **Accessible**

8.Testing

1. **Unit testing: Verifies individual components.**
2. **Integration testing: Ensures components work together.**
3. **UAT: Validates application meets user requirements.**

Benefits:

1. **Ensures functionality**
2. **Improves quality**
3. **Reduces risks**