Citizen AI

Project Documentation

1.Introduction

● Project Title: Citizen Ai

● Team Leader : Lokeshwaran V

● Team member : Akash P

● Team member : Balaji S

● Team member : Madheswaran V

● Team member : Manikandan P

**Purpose**

Citizen AI aims to leverage AI and technology to improve **public services**, **citizen engagement**, and **government transparency**. It could serve as an intelligent assistant to answer questions, help citizens access services, or analyze community needs. The goal is to create an ecosystem that allows governments and public institutions to connect with people more efficiently, while providing personalized and timely services.

* **For citizens**: It will help them access government services, resolve queries, and keep track of personal data and civic matters.
* **For governments and civic institutions**: Citizen AI helps in **data analysis**, understanding **public concerns**, and improving **policy-making decisions** based on real-time feedback.

**Features**

1. **Conversational Assistance for Public Services**
   * **Key Point**: AI-powered Q&A system for government-related queries
   * **Functionality**: Citizens can ask about government policies, documents, registration processes, deadlines for taxes, etc., and the system will provide clear, easy-to-understand explanations.
2. **Personalized Civic Notifications**
   * **Key Point**: Tailored alerts for citizens
   * **Functionality**: Based on location, age, profession, or personal data, Citizen AI can notify citizens about important dates (e.g., elections, tax deadlines) or public services available to them (e.g., subsidies, healthcare programs).
3. **Community Feedback Mechanism**
   * **Key Point**: Collecting citizen feedback
   * **Functionality**: Citizens can give feedback on local issues, infrastructure, or governmental services. AI analyzes this data to identify trends, generate reports, and suggest improvements to policymakers.
4. **Automated Public Service Requests**
   * **Key Point**: AI-driven service initiation
   * **Functionality**: Citizens can use the platform to initiate requests for services, like applying for a passport, filing complaints, or requesting waste removal. The AI helps streamline these processes by automatically filling out forms based on the citizen’s data.
5. **Policy Simulation and Analysis**
   * **Key Point**: Predictive policy outcomes
   * **Functionality**: AI can simulate the effects of various policy changes, helping governments forecast outcomes and make data-driven decisions. It can also explain the implications of existing policies to citizens in simpler terms.
6. **Real-Time Civic Data Insights**
   * **Key Point**: Data-driven civic analytics
   * **Functionality**: The AI analyzes and visualizes civic data (e.g., population trends, traffic patterns, environmental data) for both citizens and governments, helping make informed decisions about urban planning, health services, and more.
7. **Interactive, User-Friendly Platform (Streamlit/Gradio)**
   * **Key Point**: Simple interface for government-citizen interaction
   * **Functionality**: The platform provides a user-friendly interface with an easily navigable dashboard for both citizens and officials. Citizens can access services, feedback forms, and notifications, while officials can view reports, trends, and public data.

**Benefits**

* **Improved Citizen Engagement**: By offering a direct and simplified interface for interacting with government services and policies, Citizen AI ensures citizens are more informed and engaged.
* **Data-Driven Governance**: Governments can rely on real-time feedback and data analysis to shape more effective policies and services.
* **Efficiency**: Automated processes reduce bureaucratic delays and streamline public service requests.
* **Transparency**: Citizen AI enhances the transparency of government actions by offering easy-to-understand data insights and reports.

**Use Cases for Citizen AI**

* **Digital Government Assistant**: A chatbot for helping citizens navigate government portals, making appointments, or asking questions about civic services.
* **Civic Awareness**: Raising awareness about new government programs, policy changes, or urgent community matters.
* **Public Participation**: Citizens can use the platform to contribute feedback, vote on local issues, or propose new initiatives.

**Frontend (Gradio)**

* **Purpose**: The frontend is developed using **Gradio**, which provides an interactive and easy-to-use web interface. It will enable citizens to interact with various government services, submit feedback, and access information related to policies, regulations, or events.
* **Key Features**:
  + **Chat Interface**: For querying about government services, policies, and public events.
  + **Service Requests**: Citizens can request services like applying for IDs, voting registration, or file complaints.
  + **Feedback System**: Allows citizens to provide feedback on local issues or governmental actions.
  + **Notifications**: Alerts on important civic events or public services.

**Backend (Hugging Face + FastAPI Layer)**

* **Purpose**: The backend will process citizen queries, service requests, and feedback using **Hugging Face** models, and **FastAPI** for modular routing, scalability, and integration of various government data sources and service APIs.
* **Key Components**:
  + **IBM Granite 3.2.2-b Instruct Model**: Will process natural language queries and generate responses related to public services, policies, civic events, etc.
  + **FastAPI**: Handles routing for different functionalities like managing service requests, generating notifications, and managing citizen feedback.
  + **Service Integration**: It can integrate with various government data services (e.g., tax records, healthcare info, census data) via APIs.

**LLM Integration (IBM Granite 3.2.2-b Instruct)**

* **Purpose**: The IBM Granite LLM will power Citizen AI by processing natural language queries related to public services, generating relevant information on policies, and providing suggestions for civic engagement.
* **Functionality**:
  + **Policy Explainers**: Citizens can ask about specific laws, benefits, or government programs (e.g., "What are the eligibility criteria for free healthcare?").
  + **Civic Engagement Suggestions**: Based on the citizen's location or interests, the system can suggest local events, volunteering opportunities, or public meetings.
  + **Public Service Queries**: AI answers queries like "How to apply for a birth certificate?" or "What are the tax deadlines for this year?".

**Development Environment (Google Colab)**

* **Purpose**: The development and testing of the Citizen AI platform will take place in **Google Colab**. This platform offers GPU support for AI model inference and provides an easy-to-use environment for rapid prototyping and collaboration.
* **Key Features**:
  + Easy setup and testing for models in **Google Colab**.
  + Python 3.9+ environment with **pip** for managing dependencies.
  + Integrated with Gradio for frontend display and Hugging Face for model API access.

**Key Dependencies**

* **Transformers**: For loading and using pre-trained models (Granite from IBM).
* **Torch**: Core framework for deep learning model inference.
* **Gradio**: For building a web interface with chat-style interactions.
* **FastAPI**: For managing modular backend functionality (e.g., routing, API integration).
* **Requests**: For API integration (government services, real-time data).

**Folder Structure**

* **CitizenAI.py**:
  + A single script combining frontend (Gradio UI) and backend (model, processing logic).
  + **Frontend**:
    - Chat interface for government services queries.
    - Service request forms (birth certificate, ID application, etc.).
    - Feedback system for local government issues.
  + **Backend Logic**:
    - IBM Granite 3.2.2-b Instruct model integration for generating citizen-friendly explanations.
    - Functions for processing feedback, service requests, and government notifications.
    - API integrations for accessing public service data (tax records, healthcare).
  + **Configuration**:
    - API key setup for IBM and government service APIs.
    - Library imports for Transformers, Torch, FastAPI, and Gradio.
  + **Utilities**:
    - Handles input prompts for service queries, feedback, and notifications.
    - Manages real-time citizen interactions through Gradio.

**Running the Application**

1. **Open Google Colab** and create a new Python file (e.g., CitizenAI.py).
2. **Install dependencies**:
3. pip install transformers torch gradio fastapi requests
4. **Configure API keys** for IBM and any external government APIs in .env or directly in Colab.
5. **Import libraries**, load the Granite 3.2.2-b Instruct model, and connect to the Gradio interface.
6. **Run the Colab notebook cell** to launch the Citizen AI web interface.
7. The **Gradio interface link** will appear, click it to open the app and interact with the assistant.

**API Documentation**

Since this application is built using **Gradio** for the frontend, the backend endpoints are not exposed separately via FastAPI. All functionalities are provided through the **Gradio UI**, so there’s no need for Swagger documentation.

**Core Functions**:

* **Public Service Queries**:
  + **Input**: Any government service or policy (e.g., “How to apply for a passport?”).
  + **Output**: AI-generated instructions or information.
* **Civic Engagement**:
  + **Input**: Query about local events or volunteer opportunities (e.g., “What are the next local elections?”).
  + **Output**: Suggestions for engagement and public participation.
* **Feedback Submission**:
  + **Input**: Feedback about a local service (e.g., “Report road maintenance issue”).
  + **Output**: Confirmation of feedback submission and potential follow-up actions.

**Testing**:

* All core functionalities can be tested directly in the **Gradio UI**. As soon as a user interacts, the backend responds in real-time, without needing a separate API.

**User Interface (Gradio)**

The **Citizen AI UI** will feature a simple interface for interacting with the government and civic information.

**Main Modules:**

1. **Public Service Queries**:
   * **Input**: User asks questions about government services (e.g., "How to apply for a driver's license?").
   * **Output**: Simple, clear explanation generated by IBM Granite model.
2. **Service Request Forms**:
   * **Input**: Citizens can initiate public service requests (e.g., “Apply for a voter ID”).
   * **Output**: Direct links, step-by-step instructions, or auto-filled forms based on AI predictions.
3. **Feedback & Civic Engagement**:
   * **Input**: Citizen can submit feedback or ask about public events.
   * **Output**: AI provides next steps, event schedules, or confirms feedback submission.

**Conclusion**

The **Citizen AI** platform uses the same tech stack as EduTutor AI but focuses on **citizen-government interaction**. The main idea is to provide a simple, efficient, and accessible tool for citizens to access services, get information, and participate in local governance. The backend uses Hugging Face’s **Granite model** for natural language processing, while **Gradio** provides the interactive interface. Does this structure work for you, or would you like to fine-tune any of the features?