Citizen AI Project Documentation

1. Introduction

Project Title: Citizen AI – Intelligent Citizen Engagement Platform

Team Members:

SHAEERA BANU R

SYEDALI FATHIMA A

THASLIM T

2. Project Overview

Purpose

Citizen AI is an advanced citizen engagement platform designed to transform the way governments connect with the public. Built using Flask, IBM Granite LLM, and IBM Watson, it delivers real-time, AI-powered responses to citizen questions about government services, policies, and civic matters.

By incorporating natural language processing (NLP) and sentiment analysis, the platform evaluates public sentiment, identifies emerging concerns, and provides actionable insights for government agencies.

Citizen AI enhances citizen satisfaction, streamlines government operations, and fosters greater public trust in digital governance by automating routine communications and supporting data-driven policy making.

Features

Conversational Chat Assistant

Highlight: Real-time interaction with citizens

Description: Delivers immediate, human-like AI responses to inquiries about government services, policies, and civic concerns.

Citizen Sentiment Analysis

Highlight: Monitoring public sentiment

Description: Analyzes citizen feedback and categorizes it as Positive, Neutral, or Negative to identify satisfaction levels and areas of concern.

Dynamic Analytics Dashboard

Highlight: Insight-driven decision-making

Description: Presents visual data on trends, public sentiment, and reported issues to support informed policymaking.

Concern Reporting

Highlight: Transparent issue resolution

Description: Enables citizens to submit complaints or concerns, which are tracked for follow-up and resolution.

Use Case Scenarios

Interactive AI Chat Assistant

Citizens interact through a chat interface, receiving prompt and accurate responses from the AI regarding public services, government policies, and civic matters.

Sentiment Analysis of Citizen Feedback

The system processes citizen input, evaluates sentiment (Positive, Neutral, or Negative), and compiles the data to support informed decision-making.

Real-Time Analytics Dashboard

Government officials access a live dashboard that displays trends in public sentiment, service satisfaction, and reported issues, allowing for timely and data-driven responses.

3. Architecture

Frontend:
Built with HTML and CSS, the interface includes templates for the homepage, about section, services, chat interface, analytics dashboard, and user login.

Backend (Flask):
Responsible for handling application routes, user authentication, and backend data logic.

Large Language Model (IBM Granite):
Integrates advanced AI capabilities for natural conversation, text generation, and sentiment detection.

Data Management:
Currently uses in-memory storage to manage chat logs, sentiment data, and reported concerns, with future plans to implement a database solution.

Data Visualization:
Includes a dynamic dashboard that presents real-time charts and analytics for tracking sentiment patterns and issue reports.

4. Setup Instructions

Prerequisites

Python 3.7+

Flask

PyTorch (with CUDA for GPU acceleration)

Hugging Face libraries: transformers, accelerate, bitsandbytes

Hardware:

16GB+ RAM

NVIDIA GPU with 8GB+ VRAM recommended

Internet connection (for first-time model download)

Installation Process

1. Clone repository and set up project structure (app.py, templates/, static/).
2. Create and activate a virtual environment:
3. python -m venv env
4. source env/bin/activate # Linux/Mac
5. env\Scripts\activate # Windows
6. pip install -r requirements.txt
7. Install Flask, PyTorch, and Hugging Face dependencies.
8. Configure IBM Granite model path (ibm-granite/granite-3.3-8b-instruct).
9. Run the Flask backend with:
10. python app.py

5. Folder Structure

app.py – Main Flask application
templates/ – HTML templates (index, about, services, chat, dashboard, login)
static/ – CSS, Images, Favicon
requirements.txt – Python dependencies

6. Running the Application

Launch Flask backend (python app.py).

Open browser and navigate to http://localhost:5000.

Use navigation menu for:

Chat: Interact with the AI assistant.

Feedback: Submit text for sentiment analysis.

Dashboard: View real-time citizen insights.

Login: Authenticate to access protected content.

7. API Endpoints

POST /ask – Accepts citizen inquiries and returns AI-generated responses.
POST /feedback – Submits user feedback for sentiment analysis.
POST /concern – Allows users to report issues or concerns.
GET /dashboard – Retrieves aggregated sentiment data and reported issues.
POST /login – Handles user login and authentication.
POST /logout – Ends the current user session.

8. Authentication

Supports user login with session-based authentication.

Upcoming Feature: Role-based access control for different user types (citizens, administrators, government officials).

9. User Interface

Index Page: Introductory landing page with a welcome message and "Get Started" call to action.

Login Page: Secure login form for user authentication.

About Page: Describes the platform’s mission, key features, and benefits to users.

Chat Page: Interactive interface for AI-powered conversations with citizens.

Dashboard: Displays sentiment breakdown (positive, neutral, negative) and recent citizen-reported issues in real time.

10. Testing Strategy

Unit Testing: Covers Flask routes and core AI functionalities.

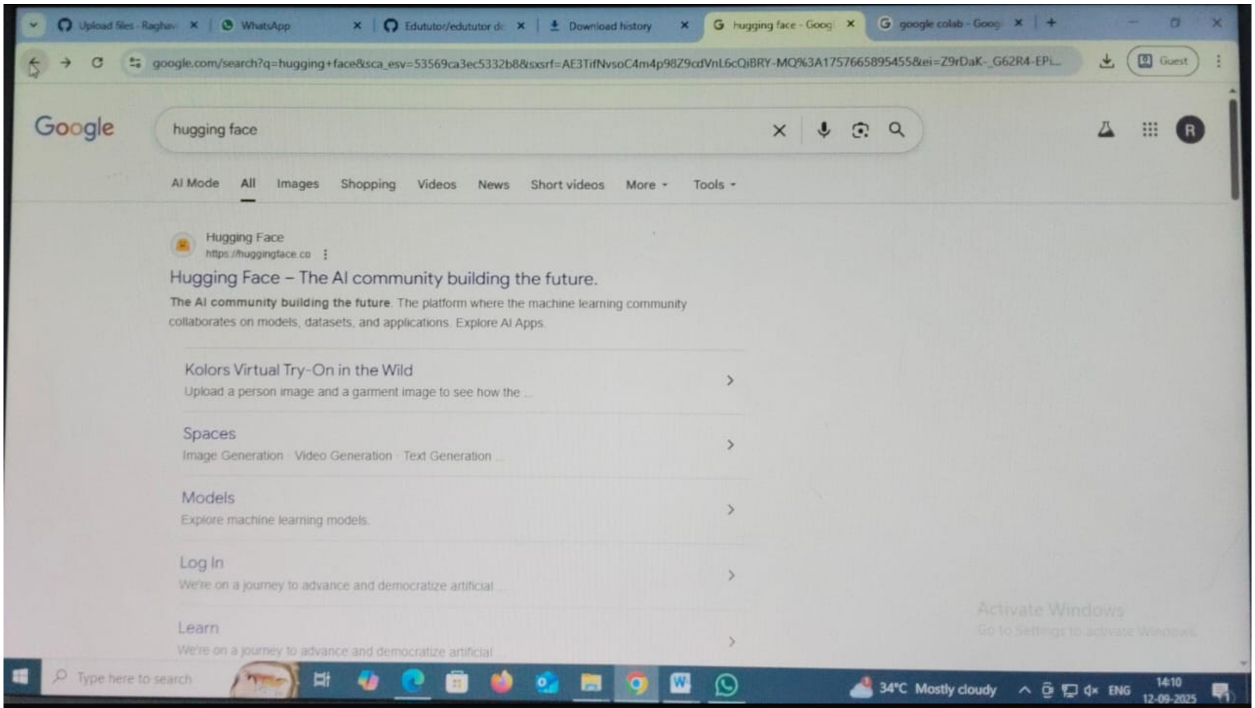
Integration Testing: Validates complete workflows across chat, feedback submission, and dashboard updates.

Manual Testing: Includes form submissions, sentiment classification accuracy, and issue reporting validation.

Edge Case Handling: Tests for scenarios like empty input fields, invalid login attempts, and improperly formatted data.

11. Screenshots

[Include UI mockups/screenshots for: Home/Index Page, Chat Interface, Dashboard View, Login Screen, and About Page]



google.com/search?q=hugging+face&sca_esv=53569ca3ec5332b8&sxsrf=AE3TifNvsoC4m4p98Z9cdVnL6cQiBRY-MQ%3A1757665895455&ei=Z9rDaK-_G62R4-EPi...

hugging face

Hugging Face
https://huggingface.co

Hugging Face – The AI community building the future.

The AI community building the future. The platform where the machine learning community collaborates on models, datasets, and applications. Explore AI Apps.

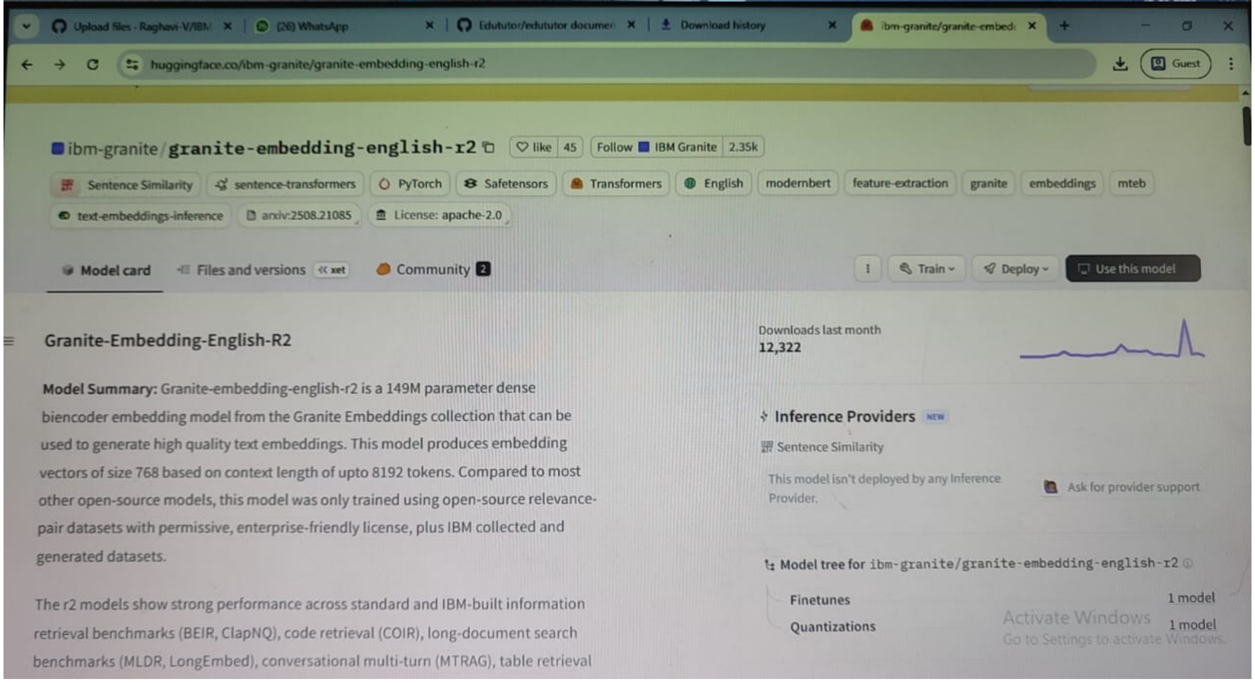
Kolors Virtual Try-On in the Wild
Upload a person image and a garment image to see how the ...

Spaces
Image Generation · Video Generation · Text Generation ...

Models
Explore machine learning models.

Log In
We're on a journey to advance and democratize artificial ...

Learn
We're on a journey to advance and democratize artificial ...



huggingface.co/ibm-granite/granite-embedding-english-r2

ibm-granite/**granite-embedding-english-r2**

Granite-Embedding-English-R2

Model Summary: Granite-embedding-english-r2 is a 149M parameter dense biencoder embedding model from the Granite Embeddings collection that can be used to generate high quality text embeddings. This model produces embedding vectors of size 768 based on context length of upto 8192 tokens. Compared to most other open-source models, this model was only trained using open-source relevance-pair datasets with permissive, enterprise-friendly license, plus IBM collected and generated datasets.

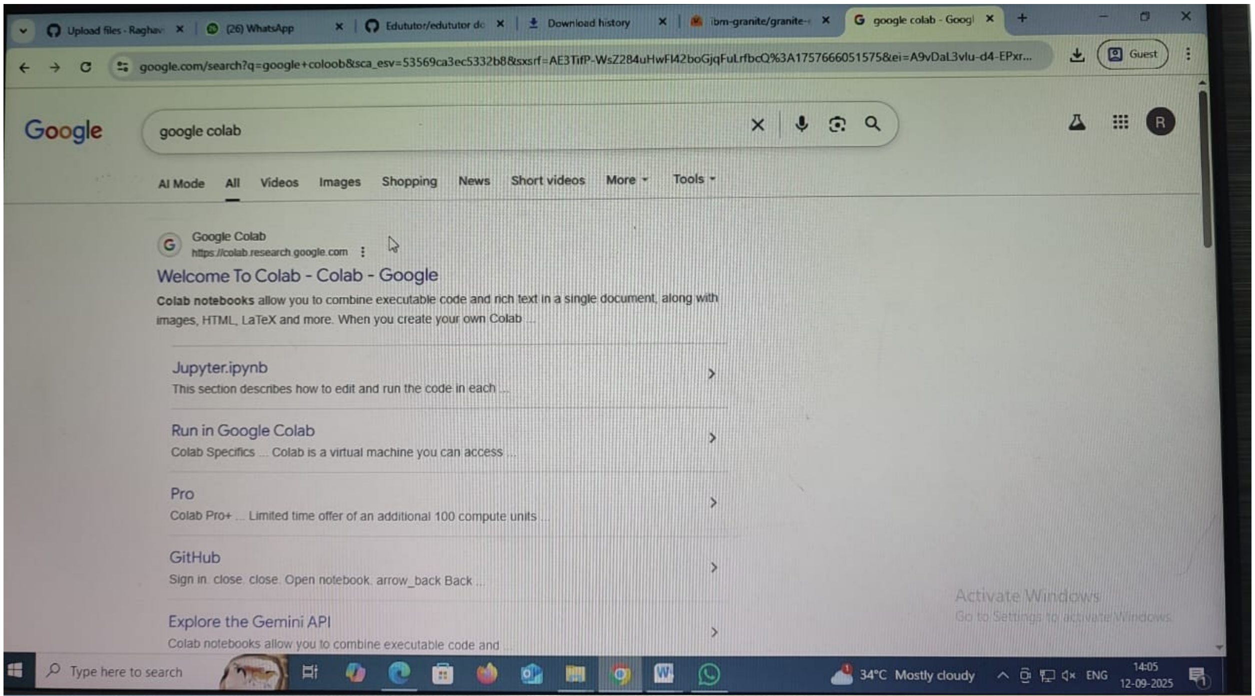
The r2 models show strong performance across standard and IBM-built information retrieval benchmarks (BEIR, ClapNQ), code retrieval (COIR), long-document search benchmarks (MLDR, LongEmbed), conversational multi-turn (MTRAG), table retrieval

Downloads last month
12,322

This model isn't deployed by any Inference Provider.

Model tree for ibm-granite/granite-embedding-english-r2

Finetunes	1 model
Quantizations	1 model



Google Colab
https://colab.research.google.com

Welcome To Colab - Colab - Google

Colab notebooks allow you to combine executable code and rich text in a single document, along with images, HTML, LaTeX and more. When you create your own Colab ...

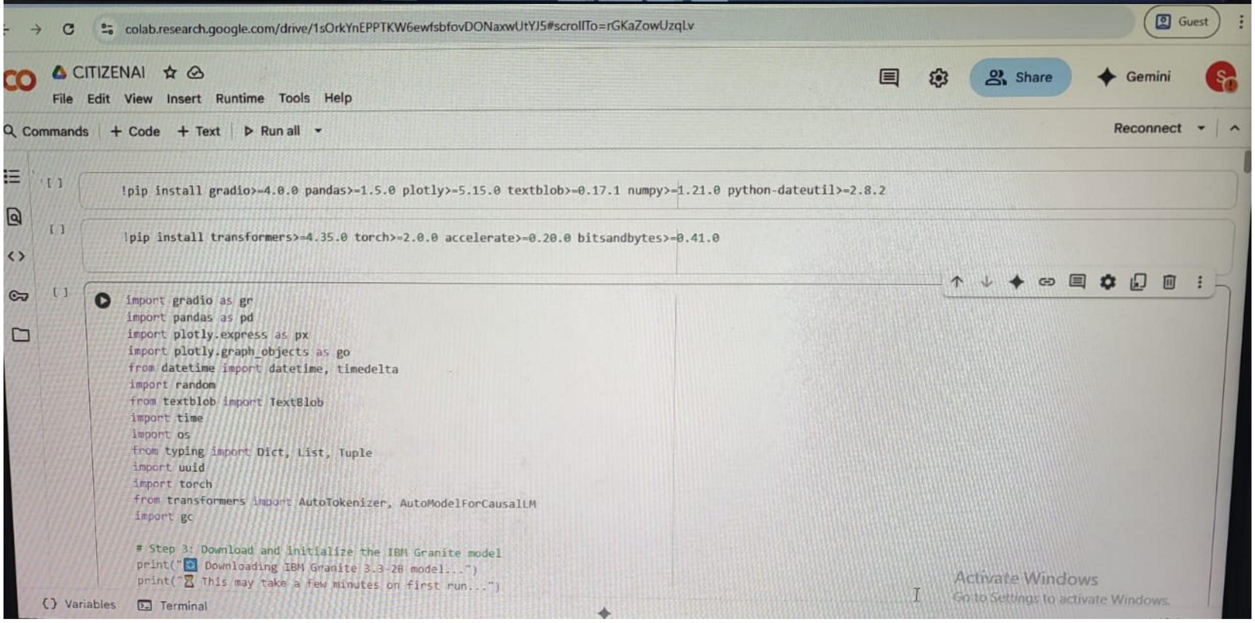
Jupyter.ipynb
This section describes how to edit and run the code in each ...

Run in Google Colab
Colab Specifics ... Colab is a virtual machine you can access ...

Pro
Colab Pro+ ... Limited time offer of an additional 100 compute units ...

GitHub
Sign in. close. close. Open notebook. arrow_back Back ...

Explore the Gemini API
Colab notebooks allow you to combine executable code and

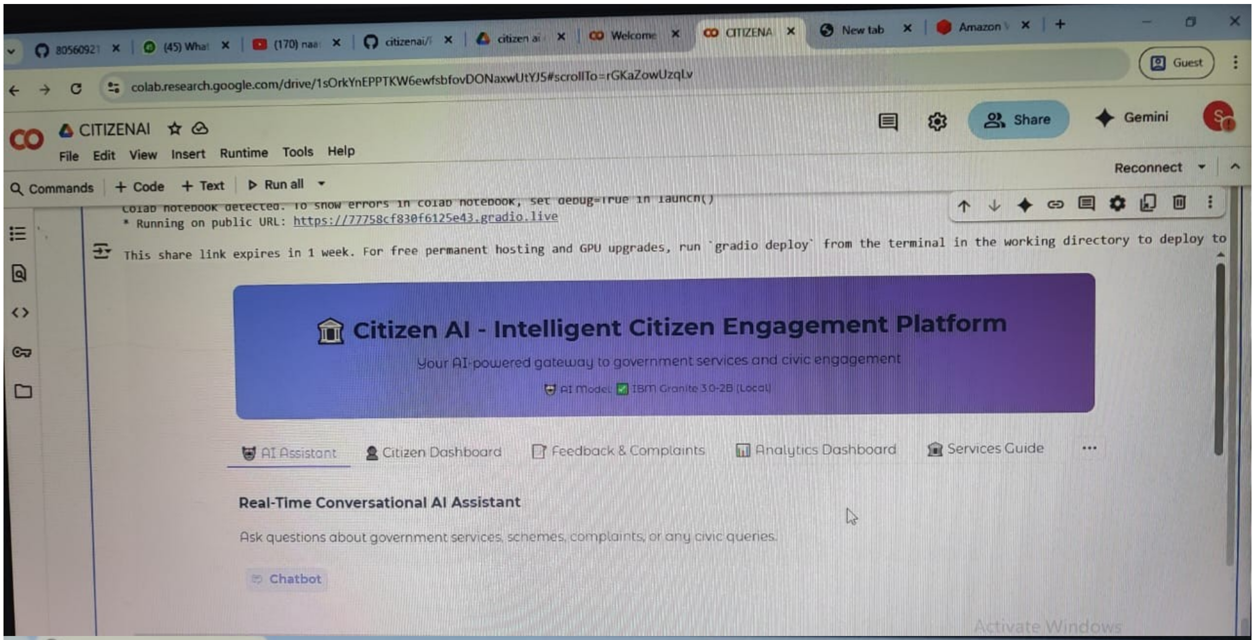


```
!pip install gradio>=4.0.0 pandas>=1.5.0 plotly>=5.15.0 textblob>=0.17.1 numpy>=1.21.0 python-dateutil>=2.8.2
```

```
!pip install transformers>=4.35.0 torch>=2.0.0 accelerate>=0.20.0 bitsandbytes>=0.41.0
```

```
import gradio as gr
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
from datetime import datetime, timedelta
import random
from textblob import TextBlob
import time
import os
from typing import Dict, List, Tuple
import uuid
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM
import gc

# Step 3: Download and initialize the IBM Granite model
print("🔄 Downloading IBM Granite 3.3-2B model...")
print("⏳ This may take a few minutes on first run...")
```



CITIZENAI
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share
Gemini
Reconnect

```
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://77758cf830f6125e43.gradio.live

This share link expires in 1 week. For free permanent hosting and GPU upgrades, run `gradio deploy` from the terminal in the working directory to deploy to
```

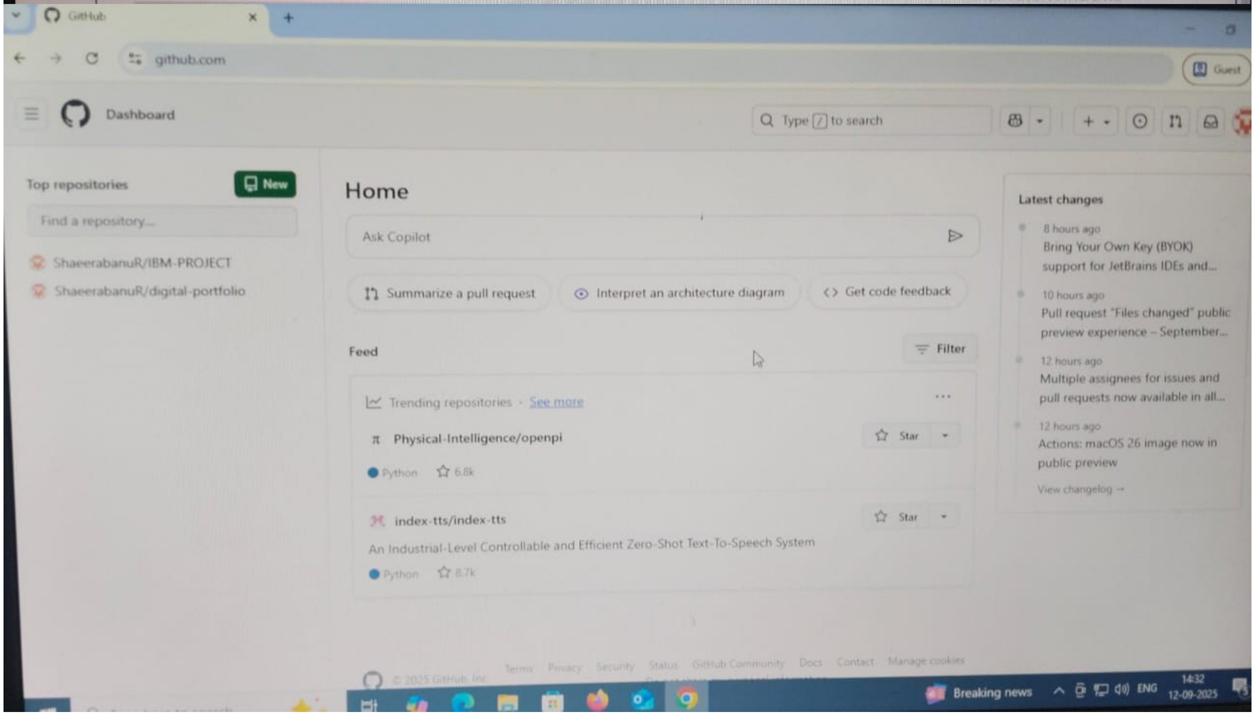
🏛 Citizen AI - Intelligent Citizen Engagement Platform

Your AI-powered gateway to government services and civic engagement

AI Model: ✅ IBM Granite 3.0-2B (Local)

Real-Time Conversational AI Assistant

Ask questions about government services, schemes, complaints, or any civic queries.



GitHub
github.com
Dashboard
Type / to search
Top repositories
New
Find a repository...
ShaeerabanuR/IBM-PROJECT
ShaeerabanuR/digital-portfolio
Home
Ask Copilot
Summarize a pull request
Interpret an architecture diagram
Get code feedback
Feed
Filter
Trending repositories · See more
Physical-Intelligence/openpi
Python 6.8k
index-tts/index-tts
An Industrial-Level Controllable and Efficient Zero-Shot Text-To-Speech System
Python 8.7k
Star
Latest changes
8 hours ago
Bring Your Own Key (BYOK) support for JetBrains IDEs and...
10 hours ago
Pull request "Files changed" public preview experience – September...
12 hours ago
Multiple assignees for issues and pull requests now available in all...
12 hours ago
Actions: macOS 26 image now in public preview
View changelog →
Breaking news
14:32
12-09-2025

12. Current Limitations

Data is temporarily stored in memory (no long-term persistence).
Performance is slow due to CPU-only execution.
Scalability is restricted until database support is added.

13. Planned Enhancements

Integration with a database for reliable data storage.
Support for multiple languages.
Fully responsive, mobile-friendly design.
Adoption of advanced NLP models for improved policy summarization.
Connectivity with social media and public forums for sentiment analysis.