ProjectTitle

CitizenAI:Intelligent Citizen Engagement Platform

ProjectDocumentation

1. INTRODUCTION

Project Title: Citizen AI

Team Leader: SARITHA R

TeamMember:RAMJANBE GAMM

TeamMember:SANGEETHA M

TeamMember:SANGEETHA D

2. PROJECTOVERVIEW

- Purpose:

Citizen AIis designed to improve governance and community engagement by providing citizens with an accessible platform to interact with government policies, services, and updates. It leveragesAItodeliveraccurateinformation,summarizedocuments,collectcitizenfeedback,and assist officials in decision-making. The system empowers both citizens and authorities by ensuring transparency, inclusivity, and real-time communication.

- Features:**Conversational**

Interface

KeyPoint:NaturallanguageQ&A

Functionality:CitizenscanaskquestionsaboutpoliciesschemesorservicesandreceiveAI-powered responses.

PolicySummarization

KeyPoint:Simplified understanding

Functionality:Convertslengthydocumentsintoconcise summaries.

CitizenFeedbackCollection

KeyPoint:Communityengagement

Functionality:Collectsopinionsandsuggestionsfromcitizensforgovernance improvements.

Service Recommendation

KeyPoint: Personalized assistance

Functionality:Suggests governmentschemesandservicesbasedonuserprofile.

DataAnalyticsforOfficials

KeyPoint:Informeddecisions

Functionality:Providesinsightsfromcitizenfeedbackandqueries.

MultimodalSupport

KeyPoint:Flexibleinputs

Functionality:Acceptstext,PDFs,andvoicequeries.

User-FriendlyDashboard

KeyPoint:Accessibility

Functionality:Intuitivedesignforbothcitizensandgovernment officials.

3. Architecture

Frontend(Streamlit/React): Interactivedashboardwithchat,documentupload,and visualization panels.

Backend (FastAPI): API-driven backend for chat, feedback collection, and report generation.

LLMIntegration:AImodelfornaturallanguageprocessing,summarization,andrecommendations. **Vector**

Database (Pinecone/FAISS): Stores embedded documents for semantic search.

AnalyticsModule:Processescitizenfeedbackandgeneratesvisualreportsfor policymakers.

4. Setupinstructions

5. Prerequisites:

- Python3.9+
- Virtualenvironmentsetup
- APIkeysforAI/Vector DBservices
- Internetconnection

InstallationSteps:

1. Clonerepository

2. Installdependenciesfrom requirements.txt

3. Configure.envwithcredentials

4. Runbackendserver (uvicorn)

5. Launchfrontend(streamlitrunapp.py)

6. Interactwithchatbot, uploaddocs,andview analytics

6. FolderStructure

```
citizen_ai/
├──app/                #BackendAPI(FastAPI)
│  ├── routes/          #APIroutes(chat, feedback, reports)
│  ├── models/           #Datamodels
│  └── services/         #Business logic
│
├──ui/                 #Frontend(Streamlit)
│  ├── pages/           # Dashboardpages
│  └── components/       # UIcomponents
│
├── embeddings/        #Document embeddingsstorage
├──analytics/          #Dataprocessing& reports
├── main.py            #Entrypoint forbackend
└── dashboard.py       # Entrypoint forfrontend
```

7. Runningthe Application

1. StartFastAPIbackend.

2. RunStreamlit dashboard.

3. Uploaddocuments or submit queries.

4. ViewAI-generatedsummaries,reports,and citizenfeedbackanalytics.

8. APIDocumentation

POST/chat/ask → Querythe AIassistant

POST/upload-doc→Uploadpolicyorschemedocument

GET /search-docs → Search relevant documents

GET/recommend-service→Getsuggestedschemes POST

/submit-feedback → Store citizen feedback

9. Authentication

Token-basedauthentication(JWT/APIkeys)
Role-based access: Citizen, Official, Admin

Futurescope:OAuth2integrationwithgovernmentdigitalIDs

10. User

Interface Sidebar

navigation

ChatwindowforQ&A

Policy/documentsummarizationtab

Feedback form

Analyticsdashboardwithvisualcharts

Downloadable reports

11. Testing

UnitTesting:CoreAIfunctions&utilities API

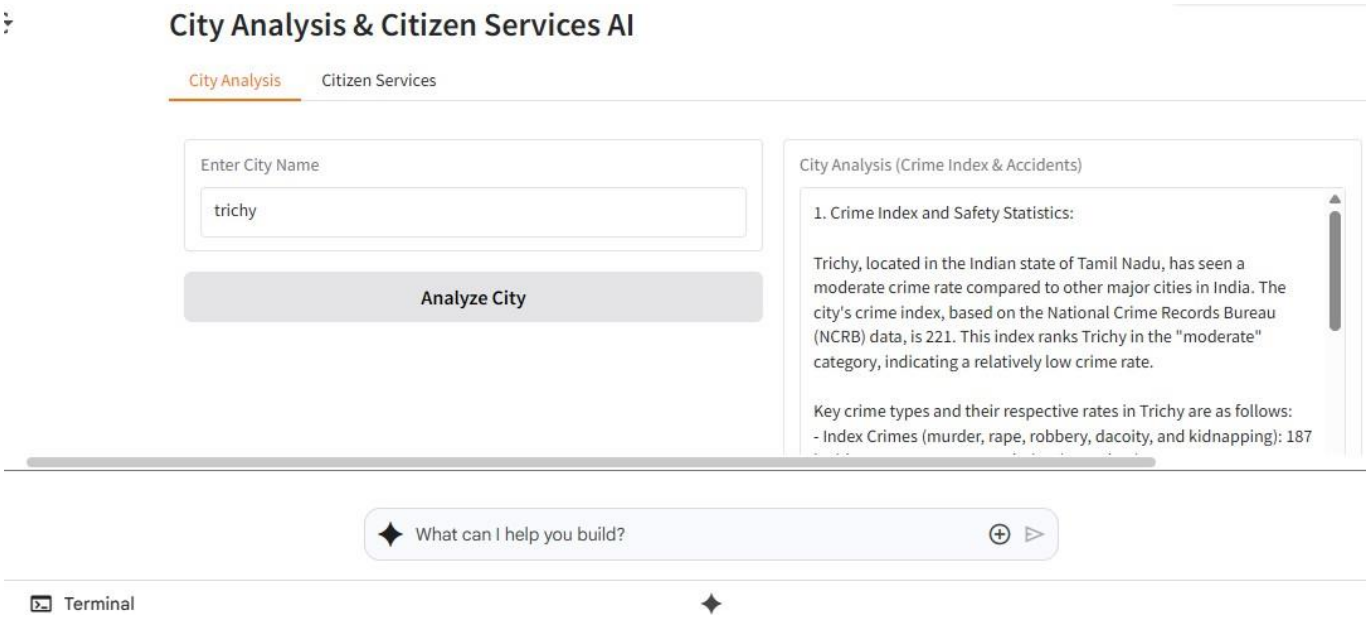
Testing: Postman & Swagger UI

Manual Testing: User interactions & document handling

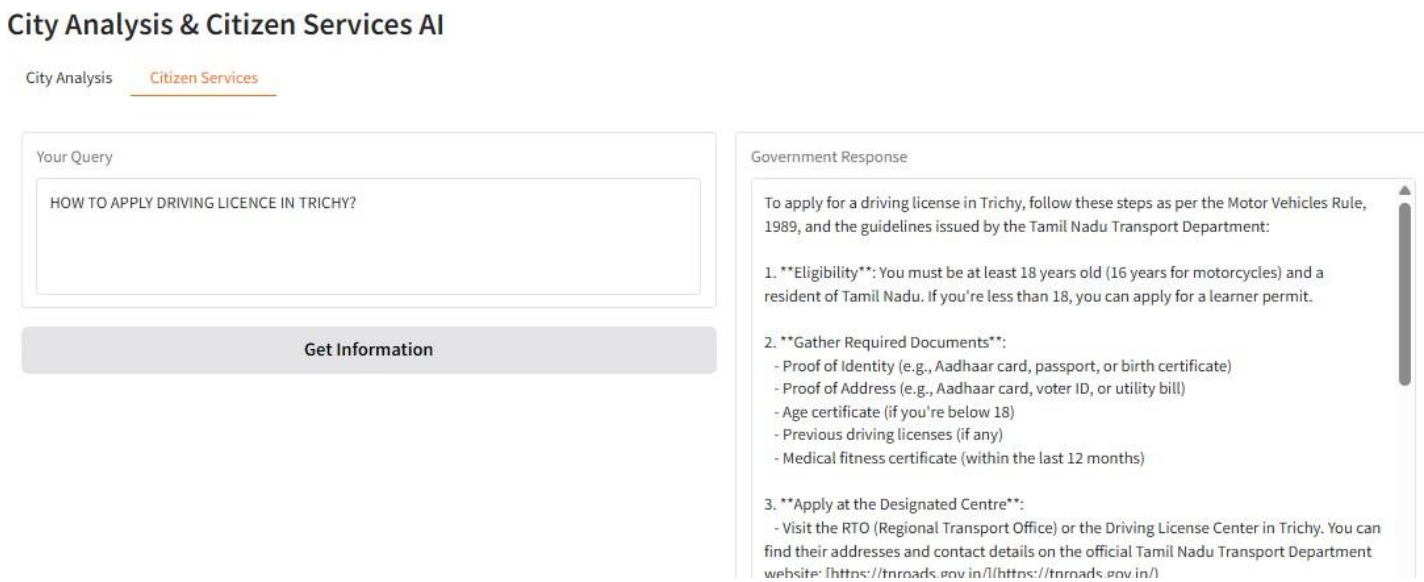
EdgeCases:Malformedinput,largedocuments,invalidkeys

12. Screenshots:

SCREENSHOT1:



SCREENSHOT2:



13. KnownIssues

Limitedsupportforregionallanguages(plannedenhancement) Requires

stable internet connection

Largedocument embeddingmayslowresponse

14. FutureEnhancements

Multilingual support

Mobile app integration

Voice-basedinteraction

Blockchain-enabledfeedbacktransparency

Predictive analytics for policy planning