**PROJECT REPORT**

Title: **Citizen AI - Intelligent Citizen Engagement Platform**

### **1. INTRODUCTION**

#### **1.1 Project Overview**

CitizenAI is an intelligent citizen services assistant powered by IBM Watson Machine Learning and Generative AI. It aims to enhance accessibility to government-related insights by offering users accurate, personalized, and data-driven guidance.

#### **1.2 Purpose**

The purpose of CitizenAI is to serve as a virtual citizen services companion, helping users understand concerns, receive AI-generated responses, and access evidence-based service suggestions through a user-friendly interface

### **2. IDEATION PHASE**

#### **2.1 Problem Statement**

In today’s world, people often turn to the internet for government-related queries but struggle with reliability and accuracy. CitizenAI addresses this by providing trusted, AI-powered government-related insights.

**2.2 Empathy Map Canvas**

#### The empathy map helps understand the target users' mindset and expectations when interacting with a virtual health assistant like CitizenAI.

|  |  |
| --- | --- |
| Category | Description |
| Says | "I want quick answers to my citizen questions.""Is this concern serious?" |
| Thinks | "Can I trust this information?""Will this help me avoid a government office visit?" |
| Does | Searches concerns onlineAsks friends or family for government-related advice |
| Feels | Anxious about concernsUncertain about next steps |
| Pains | Conflicting online informationLack of access to immediate citizen services |
| Gains | Reliable AI suggestionsQuick advice at homecitizen engagement insights |

#### **2.3 Brainstorming**

The team explored various ideas such as public grievance bots, COVID concern trackers, and AI policy assistants before settling on an all-in-one intelligent assistant with chat, prediction, and analytics powered by IBM Watson.

### **3. REQUIREMENT ANALYSIS**

#### **3.1 Customer Journey Map**

#### The customer journey in the CitizenAI application follows a streamlined, user-friendly process:

#### Start Application – The user opens the CitizenAI web app (built using Flask).

#### Ask Civic Question – The user either types a civic-related question or fills out a concern form.

#### Receive Response – The AI model (IBM Granite) responds with likely conditions and guidance.

#### Review Service Guidance Plan – The app displays a structured, AI-generated service recommendation.

#### View Citizen Analytics – Users can visualize citizen reports, concern categories, public feedback, and receive trend-based insights.

#### End/Next Action – The user can reset the session, consult a official, or continue exploring the app.

#### **3.2 Session Requirements**

* Real-time concern input via chat
* Prediction based on user profile
* Personalized service plans

Visualization of civic etrics

**3.3 Data Flow Diagram**



#### **3.4 Technology Stack**

* Frontend: Flask
* Backend: Python
* AI Service: IBM Watson ML (Granite 3-2 8B)
* Visualization: Chart.js
* **Environment Management:** virtualenv

### **4. PROJECT DESIGN**

#### **4.1 Problem-Solution Fit**

People need quick, understandable, and trustworthy government information. CitizenAI fulfills this by using government-related LLMs for better accuracy.

#### **4.2 Proposed Solution**

A layered app with UI (Flask), core logic (Python functions), and AI service (IBM Granite). It guides users from concern input to personalized plans.

#### **4.3 Solution Architecture**

* **UI Layer:** Chat, About,Dashboard,Index,Layout,login (Flask)
* **Application Logic:** app.py handles flow and calls AI
* **Helper Logic:** utils.py includes model setup and data
* **AI Layer:** IBM Granite 3-2 8B Instruct v2 via secure API

### **5. PROJECT PLANNING & SCHEDULING**

#### **5.1 Project Planning**

|  |  |  |
| --- | --- | --- |
| **Week Duration** | **Dates** | **Activities** |
| Week 1 | June 12 – June 19 | Idea finalization, architecture planning, frontend UI with Flask |
| Week 2 | June 20 – June 26 | Backend AI integration, testing, debugging, and documentation |

This two-week schedule allowed the team to focus on clear milestones and complete the CitizenAI project within the planned timeline.

### **6. FUNCTIONAL AND PERFORMANCE TESTING**

#### **6.1 Performance Testing**

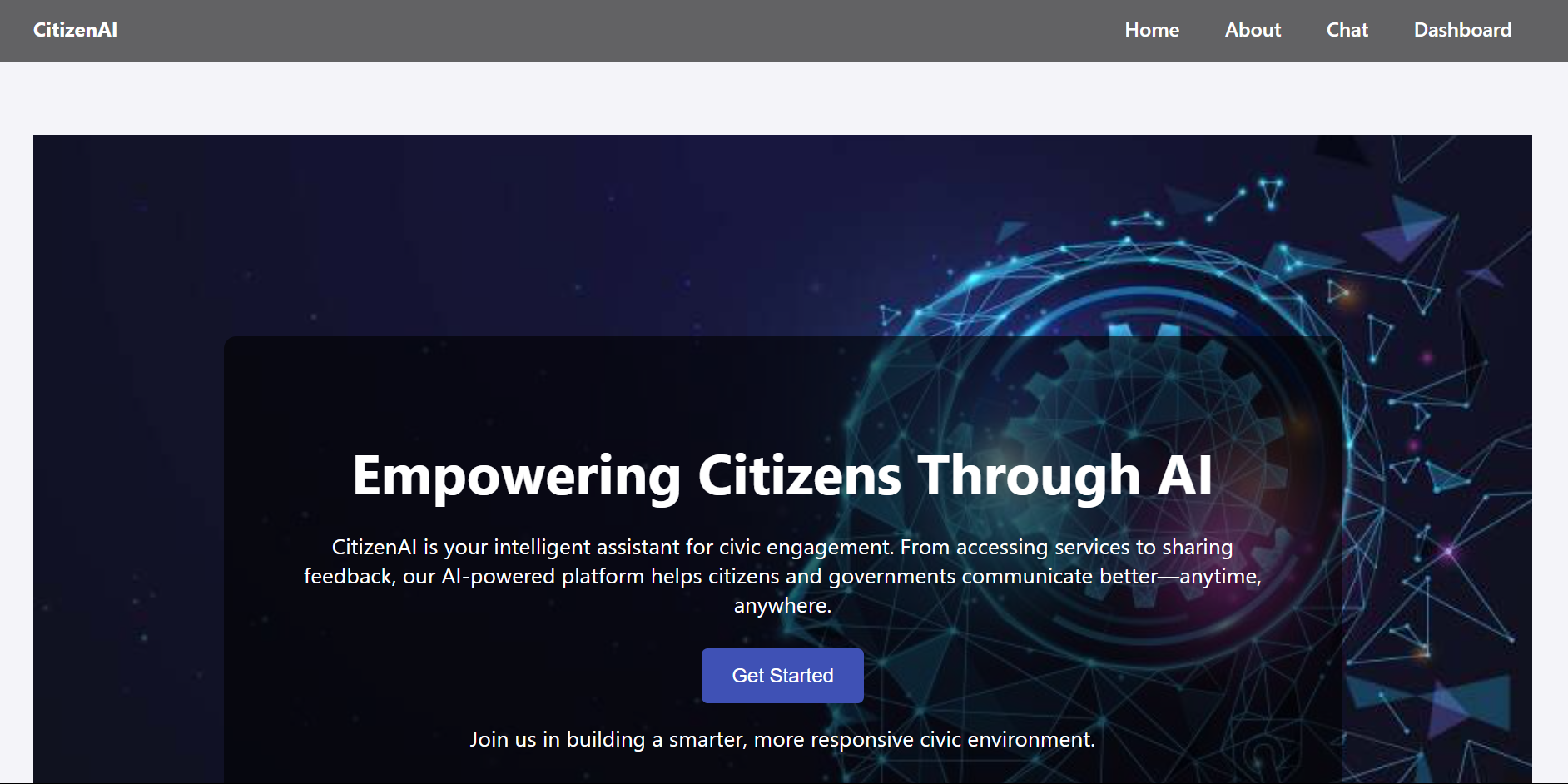
* Unit Testing: Model init, dummy citizen generation
* Integration Testing: Chat to AI flow
* Manual Testing: Verifying each feature with sample users
* Error Handling: Invalid API or missing input cases handled

### 

### **7. RESULTS**

#### **7.1 Output Screenshots**

* Screenshot of chat interface



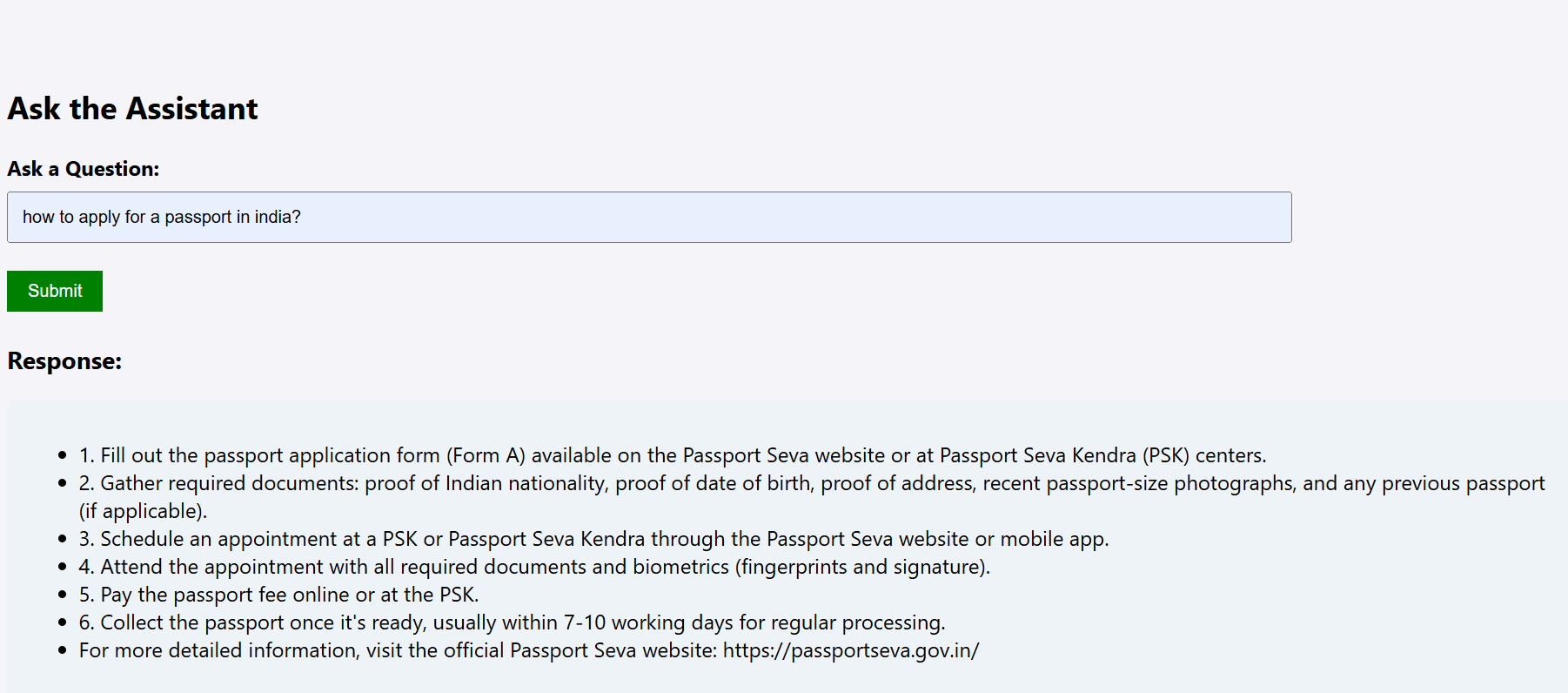
* Screenshot of Login



* Screenshot of Response



* Response 2:



### **8. ADVANTAGES & DISADVANTAGES**

Advantages: - Easy to use - AI-powered recommendations - Visualization of citizen feedback data

**Disadvantages:** - No real-time data integration - No authentication or user profiles - General-purpose AI model

### **9. CONCLUSION**

CitizenAI successfully demonstrates the application of AI in citizen services by combining user interface simplicity with powerful backend intelligence. While currently a prototype, it holds potential for real-world deployment with enhancements.

### **10. FUTURE SCOPE**

* Add secure user login
* Use real citizen databases
* Integrate with IoT civic data integrations
* Fine-tune AI on government-related data
* Add alerts, appointment booking

### **11. APPENDIX**

* **GitHub Link:** https://github.com/geethikamamidala
* **Source Code Files:** app.py, ai\_utils.py, .env.