## ****Citizen AI - Intelligent Citizen Engagement Platform****

## Project Documentation

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

**Team Members:**

**Team Leader :** Meer Muzamil Hussain

**Team member :** Mamidala Geethika Navya

**Team member :** M Rampavan

**Team member :** Malireddy Priyanka Reddy

**1. INTRODUCTION:**

1.1 Project Overview:

In the evolving landscape of smart governance, citizen engagement is a pivotal factor in ensuring transparency, accountability, and responsiveness from public institutions. Citizen AI - Intelligent Citizen Engagement Platform is an AI-driven web application designed to empower citizens by enabling seamless interaction with governmental services. Built using Flask for the backend and integrated with IBM's Granite language model, the platform provides intelligent, real-time responses to citizen queries across domains such as public services, policy information, grievance redressal, and civic awareness.

The system leverages advanced natural language processing to interpret user questions and generate accurate, context-aware responses. The user interface is designed for ease of use, enabling citizens of all backgrounds to interact effectively with the platform. Additionally, the platform includes authentication mechanisms for user security, and it can be scaled or customized for different departments or municipalities.

### **1.2 Purpose:**

The primary purpose of the Citizen AI project is to:

**Facilitate proactive governance** by offering an AI assistant that provides instant, reliable information to citizens.

**Reduce manual workload** of government helpdesk systems by automating frequently asked queries.

**Bridge the information gap** between government services and the public by acting as a 24/7 virtual assistant.

**Enhance citizen satisfaction** by offering timely support, thus fostering trust and cooperation between citizens and authorities.

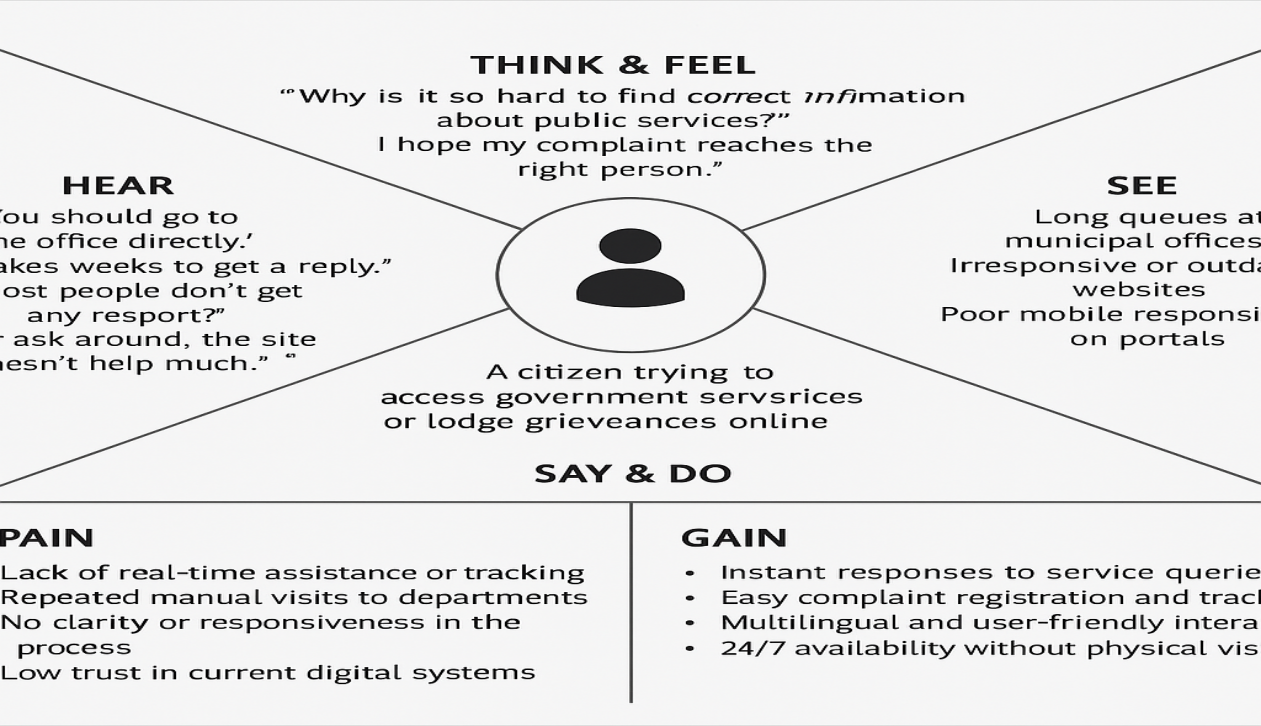
**Demonstrate the application of AI** (particularly IBM’s Granite models) in public service domains.

1. **IDEATION PHASE:**

| **Problem Statement (PS)** | **I am (Customer)** | **I’m trying to** | **But** | **Because** | **Which makes me feel** |
| --- | --- | --- | --- | --- | --- |
| **PS-1** | a citizen needing local government service information | get quick, accurate answers to questions about public services (like water supply, waste management, etc.) | I have to wait in long queues or browse unclear websites | current systems are slow, outdated, and lack proper AI support | frustrated, helpless, and ignored |
| **PS-2** | a concerned citizen with a complaint or issue | raise a complaint or grievance and get updates | there is no clear or fast way to track it | most systems don't provide real-time status or acknowledgement | anxious, uncertain, and undervalued |

* 1. Problem statement:
  2. Empathy Map Canvas:

The **Empathy Map Canvas** is a collaborative tool that helps teams understand the thoughts, feelings, needs, and behaviors of users. It allows you to build empathy by visualizing what users say, do, think, and feel when interacting with a product or service.



* 1. Brainstorming:

During the brainstorming phase, our goal was to generate a wide range of ideas to solve the problems citizens face while interacting with government services. The focus was on designing a solution that is accessible, intelligent, and user-friendly. Below are some of the key ideas that emerged during brainstorming:

### ****Idea 1: AI-Powered Chatbot for Government Services****

Provide instant, 24/7 answers to FAQs related to services like water supply, roads, electricity, sanitation, etc.

Use natural language understanding to handle queries in English and regional languages.

### 🔹 ****Idea 2: Grievance Registration and Tracking System****

Allow citizens to raise complaints easily using a conversational interface.

Provide real-time tracking and status updates to build trust and transparency.

### 🔹 ****Idea 3: Personalized Citizen Dashboard****

Users can log in to view local notices, services relevant to their area, and history of previous interactions.

Authenticated access through secure login.

### ****Idea 4: Sentiment and Feedback Analyzer****

Analyze citizen feedback and emotion behind queries to help authorities prioritize issues.

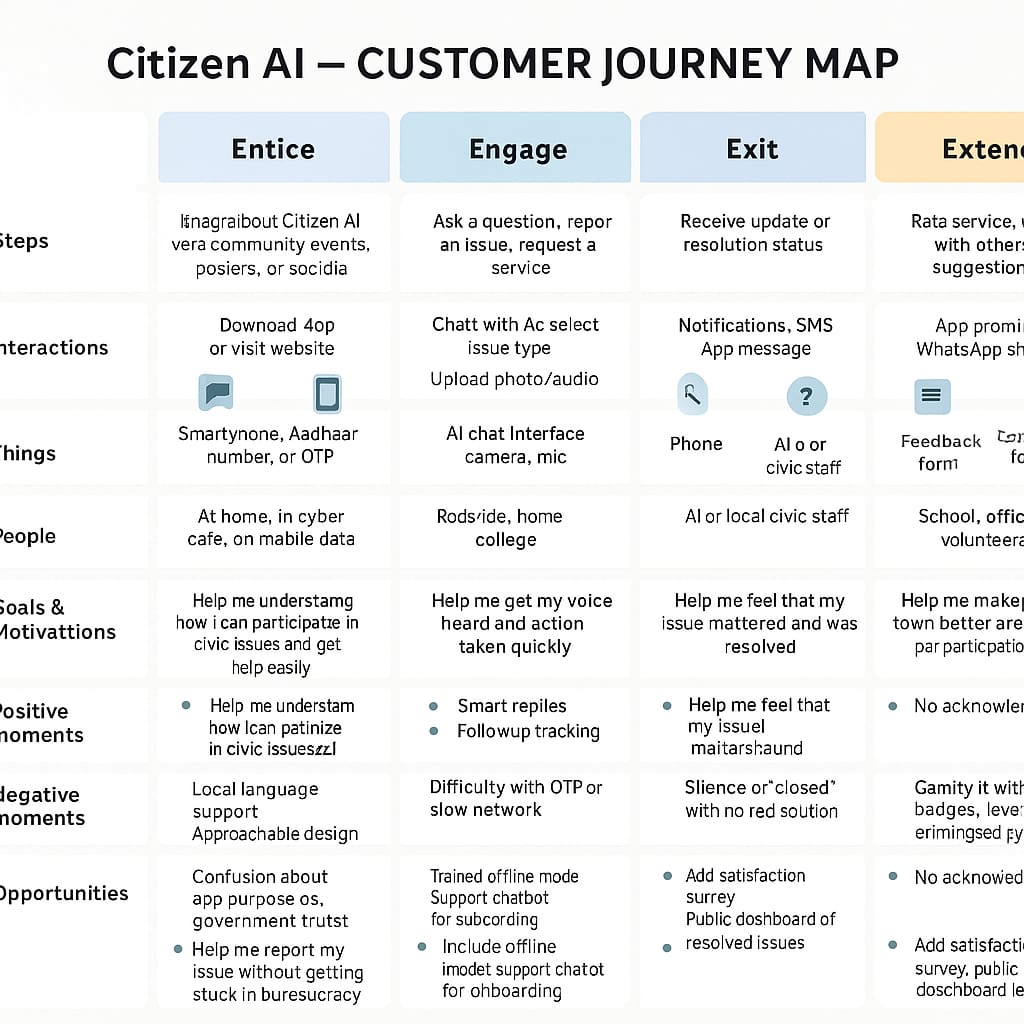
Visual dashboard for government officials to monitor citizen satisfaction.

### 🔹 ****Idea 5: Voice-Enabled Access****

Add voice input capability to help senior citizens or less tech-savvy users interact easily.

Especially useful in rural or low-literacy areas.

1. **REQUIREMENT PHASE:**
   1. Customer Journey Map:



* 1. Solution Requirements:

**Functional Requirements**

| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| --- | --- | --- |
| **FR-1** | User Registration | Registration via form, Gmail, LinkedIn |
| **FR-2** | User Confirmation | Confirmation via email or OTP |
| **FR-3** | Query Handling via AI Assistant | Natural language input, multilingual support, intent recognition |
| **FR-4** | Grievance Registration & Tracking | Form-based complaint entry, AI-based tracking updates, user dashboard |

**Non-Functional Requirements:**

| **FR No.** | **Non-Functional Requirement** | **Description** |
| --- | --- | --- |
| **NFR-1** | Usability | Interface should be simple, responsive, and accessible to citizens of all literacy levels |
| **NFR-2** | Security | Ensure secure login, data encryption, and protection of user details |
| **NFR-3** | Reliability | The system should provide consistent responses and handle failures gracefully |
| **NFR-4** | Performance | The assistant should respond to queries within 2 seconds under normal load |
| **NFR-5** | Availability | 24/7 availability with minimal downtime (<1%) |
| **NFR-6** | Scalability | System must support growing number of users and departments without performance drop |

* 1. Architecture Diagram:



Table-1: System Components and Technologies

| **S.No** | **Component** | **Description** | **Technology Used** |
| --- | --- | --- | --- |
| 1 | User Interface | Interface for user interaction via chat and web | HTML, CSS, React JS, Flask, Gradio |
| 2 | Application Logic-1 | Authentication and user login/registration | Python, Flask |
| 3 | Application Logic-2 | Speech-to-text input processing | IBM Watson STT API |
| 4 | Application Logic-3 | Chatbot response generation and routing | IBM Granite Model via IBM Watsonx (Granite Model) |
| 5 | Database | Storage for user queries, responses, and logs | MongoDB Atlas / PostgreSQL |
| 6 | Cloud Database | If hosted on the cloud, stores persistent app data | IBM Cloudant / DB2 |
| 7 | File Storage | Temporary storage for uploaded audio/image files | IBM Cloud Object Storage / Local Filesystem |
| 8 | External API-1 | Weather and location-based civic data APIs | IBM Weather API, Map APIs |
| 9 | External API-2 | Aadhaar-based identity verification (if needed) | UIDAI or similar gov API (placeholder) |
| 10 | Machine Learning Model | Granular language understanding and response generation | ibm-granite/granite-3b-3.3-instruct (IBM Watsonx (Granite Model) model) |
| 11 | Infrastructure (Cloud/Local) | Local development or cloud deployment using containerized services | Docker, Kubernetes, IBM Cloud, Google Colab |

Table-2: Application Characteristics

| **S.No** | **Characteristics** | **Description** | **Technology Used** |
| --- | --- | --- | --- |
| 1 | Open-Source Frameworks | Utilizes modern, open-source tools and frameworks for rapid deployment | Flask, React, Gradio, MongoDB Atlas |
| 2 | Security Implementation | Secured API endpoints, token-based auth, role-based access, HTTPS enforcement | OAuth2, JWT, SSL, IAM (if extended) |
| 3 | Scalable Architecture | Modular backend (API, logic, storage) with containerized deployment to scale across user demand | Docker, Kubernetes |
| 4 | Availability | Local and cloud deployments supported with minimal downtime via container orchestration | IBM Cloud, Google Cloud, Kubernetes |

1. **PROJECT DESIGN:**
   1. Problem Solution Fit

| **Section** | **Content** |
| --- | --- |
| **1. CUSTOMER SEGMENT(S)** (CS) | Students, citizens, urban youth, and local community members who want to report civic issues, get help, or engage in governance. |
| **2. JOBS-TO-BE-DONE / PROBLEMS** (J&P) | - Reporting civic issues like garbage, potholes, water problems - Seeking help from local authorities - Tracking issue status - Raising awareness or complaints anonymously |
| **3. TRIGGERS** (TR) | - Seeing unattended problems in their area - Delays or lack of response from local authorities - Frustration over repeated complaints being ignored - Hearing from peers that they got a response using Citizen AI |
| **4. EMOTIONS: BEFORE / AFTER** (EM) | **Before**: Helpless, ignored, doubtful if action will be taken **After**: Heard, empowered, hopeful, and community-driven |
| **5. AVAILABLE SOLUTIONS** (AS) | - Traditional complaints via paper or phone - Grievance redressal apps (limited or region-specific) - Direct visits to municipal offices ❌ Drawbacks: Time-consuming, bureaucratic, not trackable |
| **6. CUSTOMER CONSTRAINTS** (CC) | - No access to proper grievance system - Language barriers - Lack of digital awareness - Network issues, outdated devices |
| **7. BEHAVIOUR** (BE) | - Users try to call, tweet, or ask ward officers - Many give up after initial complaints are ignored - Some raise issues in community WhatsApp groups or social media |
| **8. CHANNELS OF BEHAVIOUR** (CH) | **8.1 ONLINE**: Citizen AI app/website, social media, voice/chat interface **8.2 OFFLINE**: Word-of-mouth, community centers, schools/colleges awareness campaigns |
| **9. PROBLEM ROOT CAUSE** | - Poor civic complaint handling systems - No transparency or tracking - Government portals are hard to use - Low trust in civic systems |
| **10. YOUR SOLUTION** (SL) | Citizen AI – a chatbot and web platform (powered by IBM Granite) ✅ Accepts voice/text/photo complaints ✅ Offers multilingual support ✅ Sends updates & tracks status ✅ Built for simplicity, reach & civic trust |

* 1. Proposed Solution

| **S.No.** | **Parameter** | **Description** |
| --- | --- | --- |
| 1. | **Problem Statement (Problem to be solved)** | Citizens often struggle to access accurate, timely, and understandable information related to government schemes, civic issues, or services due to technical language, poor interfaces, or lack of awareness. |
| 2. | **Idea / Solution Description** | Citizen AI is a chatbot-based platform built using Flask and integrated with open-source language models (like IBM Granite) that provides conversational support for common civic queries in a user-friendly and accessible manner. |
| 3. | **Novelty / Uniqueness** | Unlike typical civic portals, Citizen AI uses lightweight open-source AI models to offer real-time, human-like responses without relying on large APIs or cloud costs—ideal for low-resource public deployments. |
| 4. | **Social Impact / Customer Satisfaction** | The platform empowers citizens with clear, AI-driven access to civic knowledge, improving transparency, reducing misinformation, and enhancing participation in governance. |
| 5. | **Business Model (Revenue Model)** | Initially offered as a free civic tool. Potential revenue streams include offering AI solutions to municipalities, NGOs, or civic tech providers as white-label solutions or support contracts. |
| 6. | **Scalability of the Solution** | The system is modular and deployable on low-cost hardware. It supports language model swaps, multilingual support, and can be integrated into local civic websites or mobile apps. |

* 1. Solution Architecture

[Citizen User]

|

| User inputs query via web UI (chat.html)

v

[Flask Backend Server]

|

| Receives query

| (Handles routing, API logic)

v

[IBM Watsonx (Granite Model) Pipeline (IBM Granite or IBM Granite)]

|

| Generates AI response

v

[Response Returned]

|

v

[chat.html via Jinja2 Templates]

|

| Displays response

v

[User Interface]

| **Component** | **Description** |
| --- | --- |
| **Citizen User** | End-user accessing the system through browser interface |
| **Frontend (chat.html)** | HTML interface rendered via Jinja2 template for chat interactions |
| Flask Server | Handles routing, endpoints (/chat, /feedback, etc.), processes inputs, and calls the model |
| IBM Watsonx (Granite Model) Model | Language generation model (IBM Granite or IBM Granite) using transformers.pipeline |
| **Response Delivery** | Generated text is rendered back into the template and shown in real-time |
| **Static Assets** | CSS/JS served via Flask static routes |

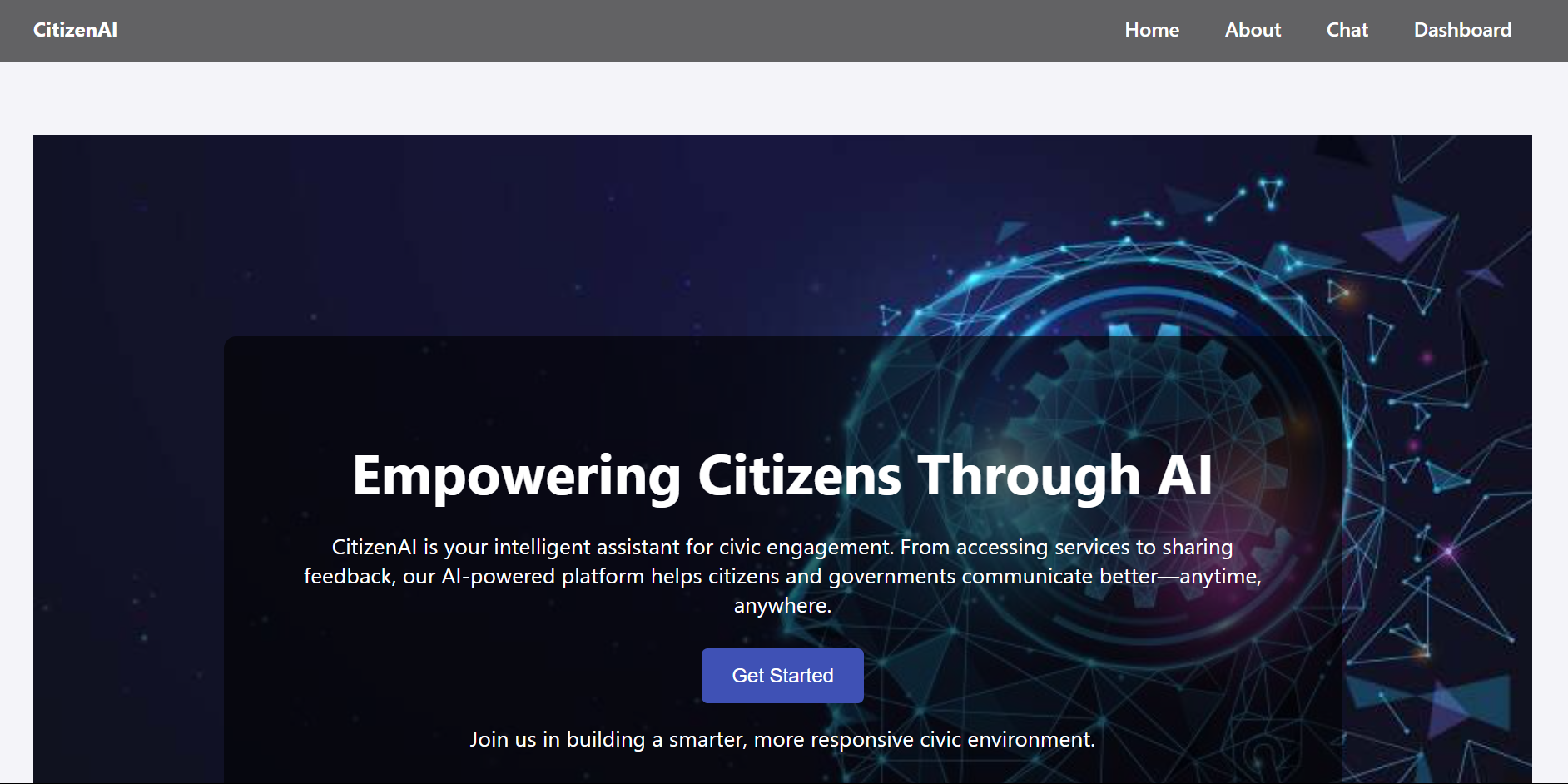
1. **PROJECT PLANNING & SCHEDULING**

5.1 Project Planning

| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | Chat Interface | USN-1 | As a user, I can ask civic-related questions and get answers from the AI model. | 3 | High | 2 |
| Sprint-1 | Sentiment Analysis | USN-2 | As a user, my query is analyzed for sentiment to personalize the response. | 2 | Medium | 1 |
| Sprint-2 | Feedback Collection | USN-3 | As a user, I can submit feedback on the AI’s response. | 1 | Low | 1 |
| Sprint-2 | MongoDB Integration | USN-4 | As a developer, I can store chat history and sentiment data in MongoDB. | 3 | High | 2 |
| Sprint-1 | Homepage + About Page | USN-5 | As a user, I can view a simple homepage and learn more on the about page. | 1 | Low | 1 |
| Sprint-2 | IBM Granite Model Integration | USN-6 | As a developer, I can integrate IBM’s Granite model via WatsonX API. | 3 | High | 3 |

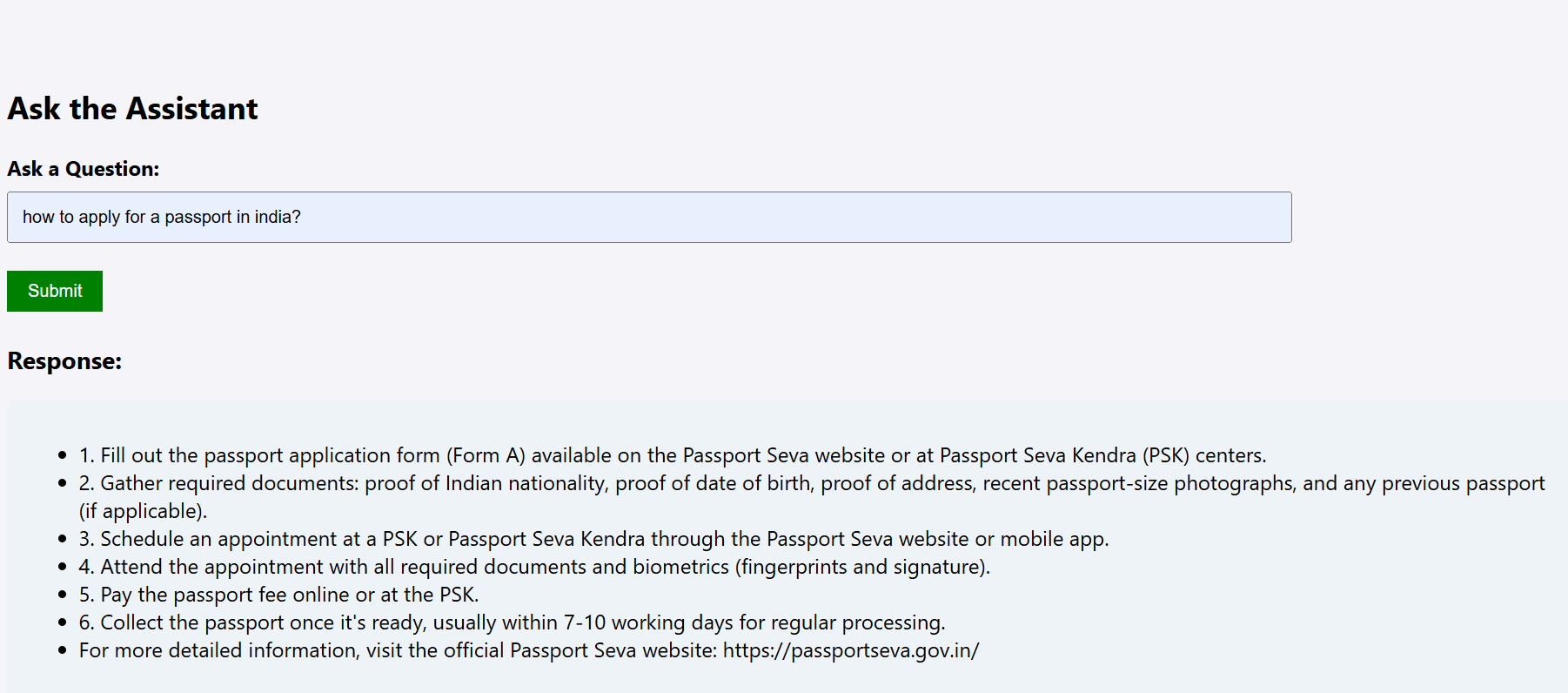
**6. RESULTS**

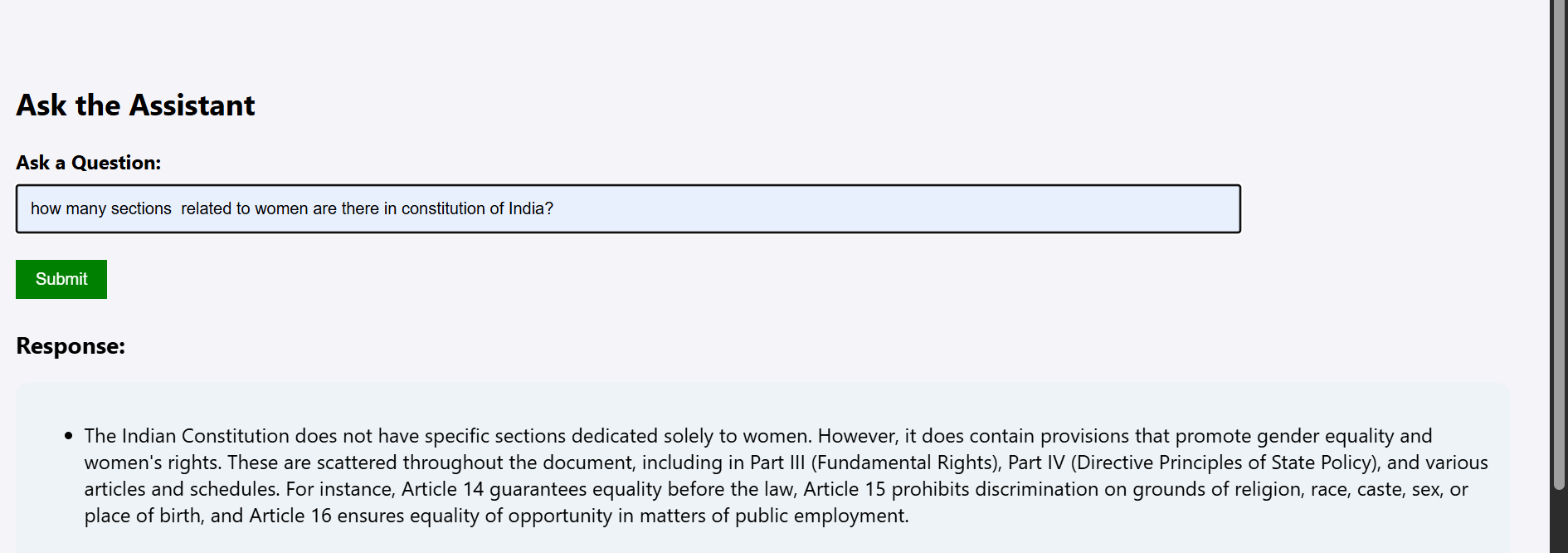
* 1. Output Screenshots

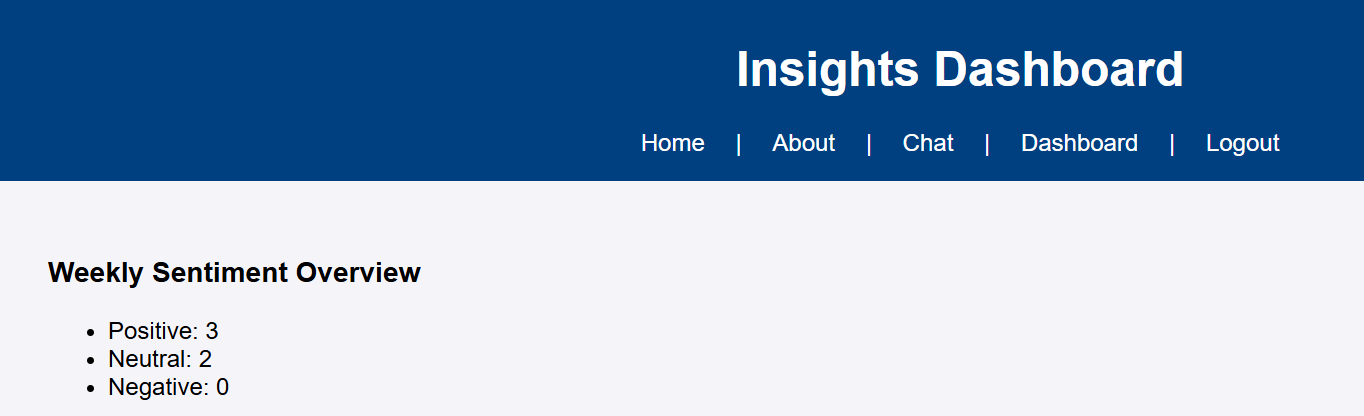














1. **ADVANTAGES AND DISADVANTAGES**

### ****Advantages****

**Civic Engagement Empowerment**

Helps users get quick, AI-powered answers to government or civic-related queries, promoting digital literacy and awareness.

**Open Source & Extensible**

The use of open technologies (Flask, MongoDB, IBM Watsonx (Granite Model), IBM Granite, etc.) makes it easy to customize and expand (e.g., adding more models, APIs, dashboards).

**Sentiment Analysis Feature**

Analyzing user emotions helps in personalizing responses and flagging critical or sensitive concerns.

**Data Storage & Logging**

Uses MongoDB to store chat history, feedback, and sentiment—useful for analytics, audit, or future training data.

**Cloud-Model Integration**

Integration with IBM WatsonX (Granite) allows access to enterprise-grade AI capabilities.

**Disadvantages**

**Privacy & Data Concerns**

Without proper user consent and encryption, storing queries and sentiments could raise data privacy issues.

**UI/UX Simplicity**

The current front-end (based on HTML templates) is basic and may not be suitable for a wide public deployment without enhancement.

**Dependency on External APIs**

Reliance on IBM WatsonX or other hosted models introduces latency, cost, and availability issues.

**Lack of Multilingual Support**

May not support regional languages or dialects, limiting accessibility for diverse user bases in India or globally.

1. **CONCLUSION:**

The CITIZEN AI project aims to bridge the gap between citizens and civic information by leveraging the power of artificial intelligence. By integrating a conversational chatbot with civic knowledge, the system provides users with instant responses to queries related to public services, government schemes, and general civic concerns. The use of models like IBM Granite and IBM Granite enables natural language understanding and generation, creating a more intuitive and accessible experience for users, especially those unfamiliar with complex governmental websites or processes.

In addition to its core chat functionality, the system incorporates features such as sentiment analysis and feedback collection. These additions not only help personalize responses but also enable the system to monitor user satisfaction and identify emotionally charged or urgent concerns. The use of MongoDB for storing chats, feedback, and sentiments ensures the project is data-driven and offers potential for continuous learning and improvement. The modular design, built using Flask and Python, also makes it easy to expand and adapt for future enhancements.

While the project successfully meets its initial goals, there are opportunities for further development. Enhancing the user interface, supporting multiple languages, and integrating with official government APIs could significantly increase the system's effectiveness and reach. Additionally, addressing data privacy and ethical considerations will be crucial as the system scales. Overall, CITIZEN AI serves as a strong proof of concept for AI-assisted civic engagement and has the potential to become a valuable tool in promoting digital governance and public participation.