**A PROJECT REPORT**

**On**

**CITIZEN AI – INTELLIGENT CITIZEN ENGAGEMENT PLATFORM**

**By**

PUSA SRUJAN (22FE1A05E4)

PERUGU VENKATA SAI KIRAN (22FE1A05D5)

MULLAPUDI LIKHITHA SAHITHI (22FE1A6127)

PERUBOINA LAKSHMI PALLAVI (22FE1A05D4)

Under the guidance of

**Ganesh M**

**TABLE OF CONTENTS**

| **S. No** | **Content** | **Page No(s)** |
| --- | --- | --- |
| **1** | **Abstract** | **3** |
| **2** | **Introduction** | **4-5** |
| **3** | **Ideation Phase** | **5-10** |
| **4** | **Requirement Analysis** | **11-19** |
| **5** | **Project Design** | **20-24** |
| **6** | **Project Planning & Scheduling** | **25-26** |
| **7** | **Functional and Performance Testing** | **27** |
| **8** | **Results** | **28** |
| **9** | **Advantages & Disadvantages** | **29** |
| **10** | **Conclusion** | **30** |
| **11** | **Future Scope** | **30-31** |
| **12** | **Appendix** | **32-35** |

**ABSTRACT**

Citizen AI is an intelligent citizen engagement platform designed to revolutionize how governments interact with the public. It integrates IBM Granite AI models, Flask, and NLP technologies to create a real-time, responsive interface for handling citizen queries and feedback.

By combining conversational AI, sentiment analysis, and real-time data visualization, Citizen AI improves access to government services, increases transparency, and allows citizens to report concerns and view their status. A dynamic dashboard allows authorities to track public sentiment and prioritize issues, making governance more data-driven and citizen-centric.The primary motivation behind Citizen AI is to address the inefficiencies in traditional citizen service systems, where delays, lack of feedback loops, and poor accessibility often hinder meaningful engagement. Through Citizen AI, users can pose questions, submit feedback, and report civic issues through a chat interface that uses an AI model to generate contextually relevant, human-like responses in real time. The system intelligently categorizes citizen sentiments as positive, neutral, or negative using a sentiment analysis module, offering decision-makers a powerful lens into public perception.

**Introduction**

2.1 Project Overview

In the digital age, the way governments communicate and interact with their citizens is undergoing a fundamental transformation. Traditional governance mechanisms—characterized by bureaucratic bottlenecks, delayed responses, and lack of real-time feedback—are increasingly proving to be inadequate in addressing the rising expectations of digitally aware citizens. Citizens now expect prompt, personalized, and accessible responses to their queries, grievances, and feedback, much like the services they receive from the private sector. In this context, Citizen AI emerges as an intelligent, scalable solution designed to enhance citizen engagement through the use of advanced Artificial Intelligence technologies.

Citizen AI is a web-based platform that utilizes IBM’s Granite generative AI models, natural language processing (NLP), and real-time data visualization to provide a smart, conversational interface for public service delivery. The system is built using the Python Flask framework for handling routing, session management, and frontend integration, and incorporates the Hugging Face Transformers ecosystem to facilitate seamless AI model interaction. At its core, the platform enables citizens to ask questions related to government services and receive context-aware, human-like responses, thereby reducing the need for manual intervention and long wait times.

The platform also supports structured sentiment analysis of citizen feedback, allowing government bodies to track and interpret public opinion. Additionally, citizens can report specific concerns, which are logged and displayed on a real-time dashboard used by government administrators to monitor trends, identify high-frequency issues, and adjust policies accordingly. By automating these workflows, Citizen AI transforms the reactive model of governance into a proactive, insight-driven process.

2.2 Motivation and Significance

Governments across the globe are seeking smarter ways to engage with their citizens—especially in the wake of global challenges such as pandemics, digital divides, and increased public scrutiny. However, most digital citizen platforms today are either static, fragmented, or difficult to use. Citizen AI addresses this gap by building a centralized, intelligent engagement hub where citizens can access support, report issues, and give feedback—all in natural language.

The use of generative AI provides a distinct advantage in terms of understanding the nuances of human queries. Unlike keyword-based systems or traditional decision-tree chatbots, the IBM Granite model understands context, emotion, and intent, and can dynamically tailor responses to different types of user interactions. This dramatically improves the quality of user experience and increases public trust in digital government systems.

From a governance standpoint, the ability to analyze aggregated feedback and sentiment across thousands of interactions provides policy-makers with a powerful decision-support tool. Instead of relying solely on surveys or sporadic reports, administrators can now make real-time, data-driven decisions based on organic, unsolicited citizen input.

2.3 Key Objectives

The key goals of Citizen AI include:

* Providing a real-time conversational assistant that mimics human interaction.
* Collecting and classifying citizen feedback using sentiment analysis.
* Enabling structured concern reporting to identify service delivery gaps.
* Presenting interactive dashboards for government decision-makers.
* Delivering an accessible, scalable platform that can grow with evolving needs.

2.4 Intended Impact

By implementing Citizen AI, the government-citizen interaction becomes not only efficient and accessible, but also intelligent and transparent. Citizens are empowered to communicate effectively without navigating complex bureaucracy. Government departments, in turn, benefit from automated insights, improved public perception, and the ability to prioritize services based on actual demand.

This project represents a shift from passive digital portals to active civic engagement tools, and lays the groundwork for future e-governance frameworks that are scalable, multilingual, and personalized.

**IDEATION PHASE**

**3.1 Problem Statement**  
Manual handling of citizen complaints and queries is time-consuming, inefficient, and lacks personalization. Governments struggle to analyze large volumes of feedback in real-time. There's a need for an automated, smart, and scalable platform that can bridge this communication gap.

**3.2 Empathy Map & User Needs**

* Thinks: "Is the government even reading this?"
* Feels: Ignored, frustrated, unheard
* Hears: Delays in issue resolutions
* Says: "I raised this concern weeks ago!"
* Gains: A quick, trusted, responsive system

**3.3 Brainstorming**  
The team identified AI chat interfaces, NLP models, and real-time visualization as the key tools. IBM Granite was chosen due to its strong NLU capabilities. Flask was picked for its simplicity and flexibility.

**Customer Problem Statement Template**



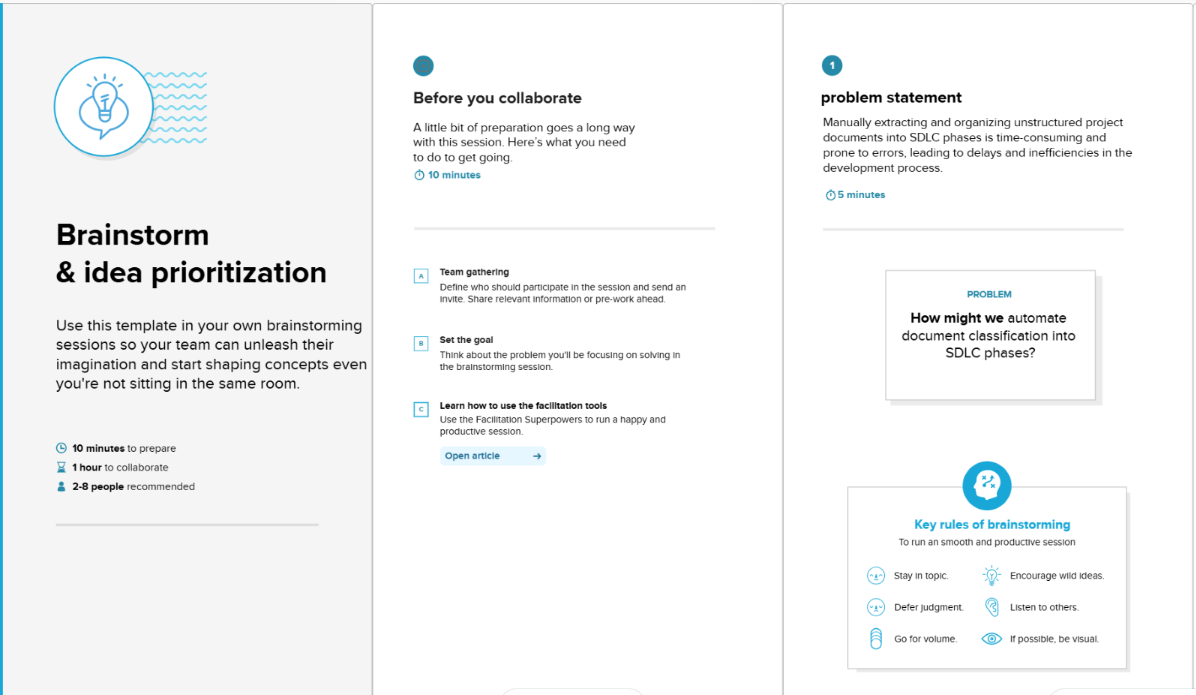
**2.2 Empathy Map**

To better understand the user’s experience and challenges, an empathy map was created for a typical SmartSDLC user—such as a developer or analyst. This visual tool captures what the user thinks, feels, sees, hears, says, and does while interacting with project documentation. It also highlights their pain points and desired gains. The empathy map below helps ensure that the SmartSDLC solution is designed with the user’s real needs in mind.

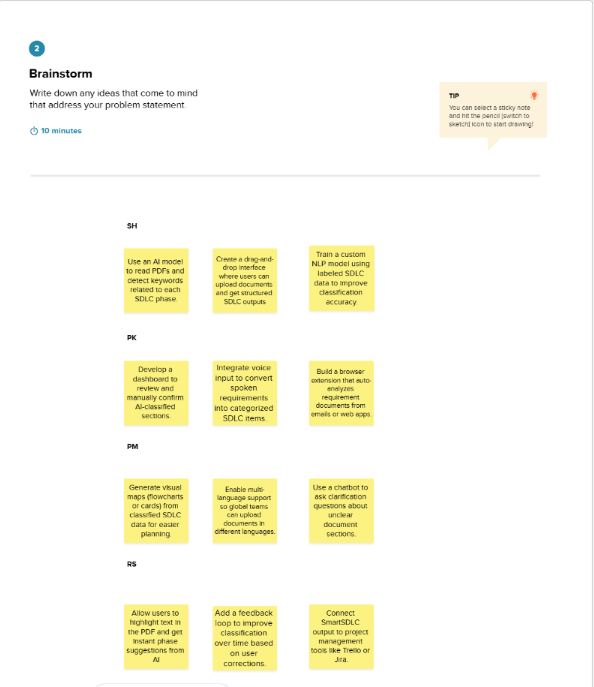


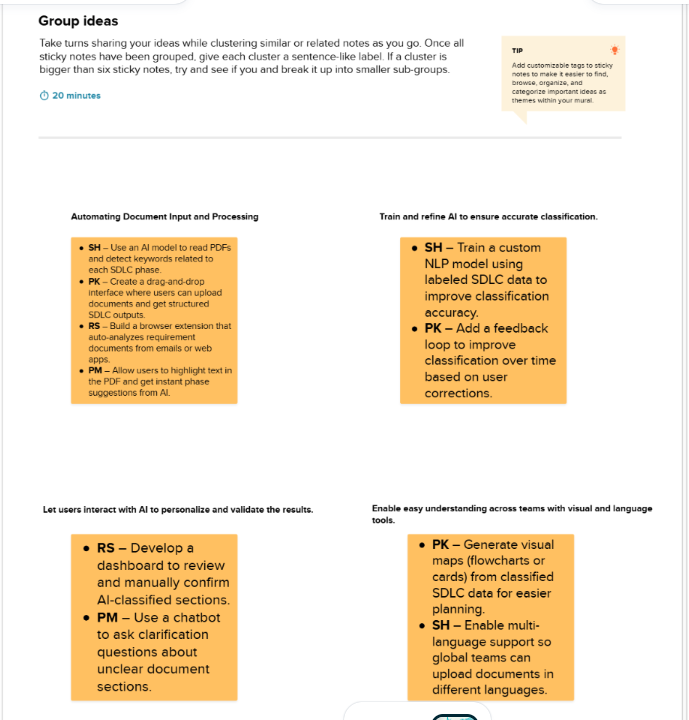
**2.3 Brainstorming**

**Step-1: Team Gathering, Collaboration and Select the Problem Statement**

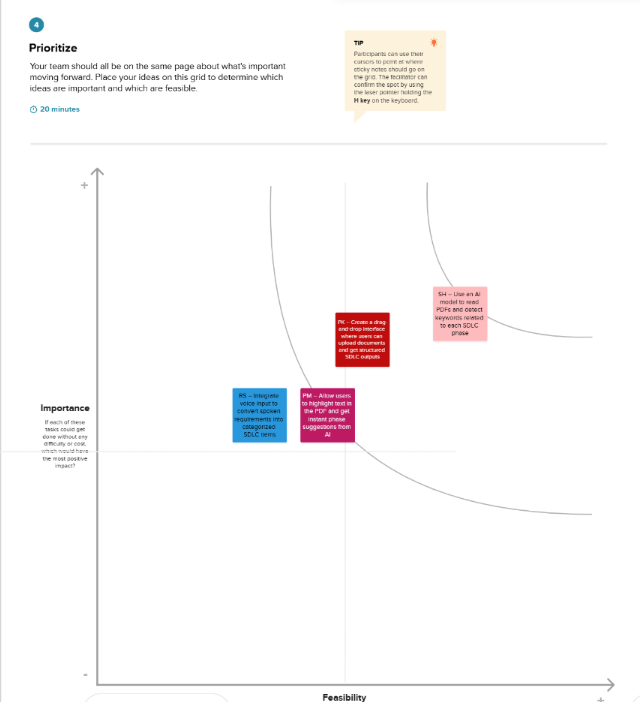
****

**Step-2: Brainstorm, Idea Listing and Grouping**

****

****

Step-3: Idea Prioritization



**REQUIREMENT ANALYSIS**

**3.1 Customer Journey Map (Citizen AI)**

| **Phase** | **Steps (What user does)** | **Interactions (With what or whom)** | **Goals & Motivations** | **Positive Moments** | **Negative Moments** | **Opportunities** |
| --- | --- | --- | --- | --- | --- | --- |
| **Entice** | **Learns about Citizen AI through media, government promotion, or social platforms** | **Web, social media, peer discussions** | **Curious to try a smarter way of reaching government services** | **Excited about AI-based help** | **Unsure if the bot can really help with local issues** | **Provide a live demo or case study video** |
| **Enter** | **Visits the Citizen AI website, explores the features** | **Homepage, About, Services** | **Wants to understand the capabilities of the platform** | **Finds a clean, official-looking interface** | **Doesn’t know where to start or how to ask questions** | **Add onboarding popups or chatbot welcome prompts** |
| **Engage** | **Chats with the AI assistant, submits feedback or concern** | **Chat interface, backend, AI model** | **Needs fast, relevant help or wants to share a complaint** | **Gets real-time response, sentiment classified** | **Confused by generic or wrong AI replies** | **Fine-tune AI with contextual FAQ knowledge base** |
| **Exit** | **Views dashboard feedback summary or logs out** | **Dashboard, logout/session tools** | **Feels heard and wants to monitor updates or return later** | **Sees issues reflected on dashboard** | **Worries about privacy or losing session data** | **Add session save, anonymized concern tracking** |
| **Extend** | **Recommends platform to others or returns for new concerns** | **Word of mouth, referral, browser history** | **Build habit of digital governance usage** | **Trust in government services improves** | **Needs mobile support, voice-based interaction** | **Launch mobile version, add regional language/voice AI** |

**3.2 Solution Requirements (Citizen AI)**

**Functional Requirements**

| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| --- | --- | --- |
| **FR-1** | **User Login & Authentication** | **Login form with session handling** |
|  |  | **Email-based login (admin/user)** |
| **FR-2** | **AI Chat Assistant** | **Accept text input through chat UI** |
|  |  | **Generate real-time response via IBM Granite** |
| **FR-3** | **Feedback Sentiment Analysis** | **Text area to enter feedback** |
|  |  | **Analyze sentiment (Positive/Neutral/Negative)** |
| **FR-4** | **Concern Reporting** | **Submit issue via form** |
|  |  | **Log concern in memory (or DB later)** |
| **FR-5** | **Dashboard & Visualization** | **Display feedback stats (chart, sentiment)** |
|  |  | **Show recent issues and time trends** |
| **FR-6** | **Error Handling & UI Alerts** | **Notify user if input is empty** |
|  |  | **Show alerts for failed requests / slow AI** |

**Non-Functional Requirements**

| **NFR No.** | **Non-Functional Requirement** | **Description** |
| --- | --- | --- |
| **NFR-1** | **Usability** | **Simple UI using HTML/CSS/Jinja2 for easy navigation** |
| **NFR-2** | **Performance** | **Low response time with model quantization; optimized Flask routing** |
| **NFR-3** | **Security** | **Use session keys for user auth; secure API handling** |
| **NFR-4** | **Availability** | **24/7 accessible once deployed (via cloud)** |
| **NFR-5** | **Reliability** | **Stable performance under repeated use; modular structure** |
| **NFR-6** | **Scalability** | **Can expand to DB, multilingual support, and mobile version** |

**3.3 Data Flow Diagram**

A **Data Flow Diagram (DFD)** is a structured graphical representation of the movement of data within a system. It helps to visually depict how data flows from one process to another, how it's processed, and how it interacts with different entities within the system.

In the context of **Citizen AI**, the DFD illustrates how data is received from the user, processed by AI and logic layers, and returned back to the user or admin interface. It showcases:

* The various **data sources** (citizens),
* The **processing units** (AI, sentiment analysis, feedback handling),
* The **data stores** (in-memory storage for now), and
* The **output channels** (UI, dashboard, feedback displays).

This diagram is important for understanding the functional layout and the core operational workflow of the system.

**DFD Process Description Table**

| **Process** | **Input** | **Operation Performed** | **Output** |
| --- | --- | --- | --- |
| 1. User Login | Username & Password | Validates user credentials via Flask session | Access granted or error message |
| 2. Chat Assistant | User query (text) | Sends query to IBM Granite model via Hugging Face | AI-generated response |
| 3. Feedback Analyzer | Feedback text | Analyzes sentiment (Positive, Neutral, Negative) | Sentiment classification |
| 4. Concern Reporting | Concern text | Stores concern in session memory (planned: database) | Acknowledgment of submission |
| 5. Admin Dashboard | Session storage & logs | Aggregates sentiment stats & displays recent issues | Visual data graphs + issue logs |
| 6. Logout | User request | Clears session, ends active state | Redirects to login |

**🔧 Suggestion for Visual Representation (Level 1 DFD)**

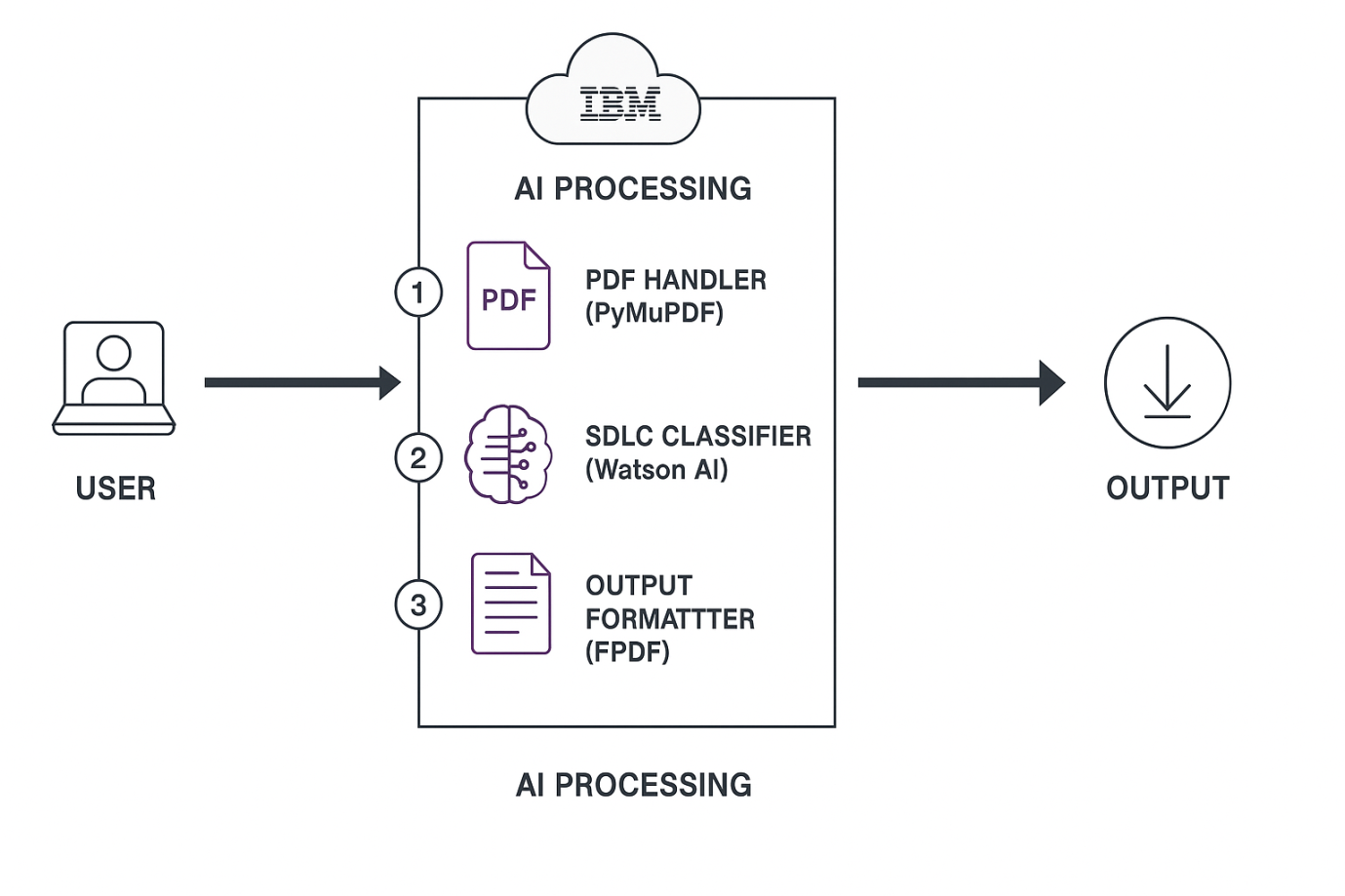
If you want to create this in **Word or draw.io**, the Level 1 DFD should include:

* **External Entities:**
  + *Citizen (User)*
  + *Admin (Government Official)*
* **Processes:**
  + *Login System*
  + *AI Chat Assistant*
  + *Sentiment Analyzer*
  + *Concern Form Handler*
  + *Dashboard Module*
* **Data Stores:**
  + *In-Memory Session (Current)*
  + *Database (Future scope)*
* **Data Flows:**
  + User query → Chat Assistant → AI Model → Response
  + Feedback → Sentiment Analyzer → Dashboard
  + Concern → Concern Handler → Storage → Dashboard

**3.4 User Stories**

| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance Criteria** | **Priority** | **Release** |
| --- | --- | --- | --- | --- | --- | --- |
| **Citizen** | Chat with Assistant | USN-1 | As a citizen, I want to type my questions and receive responses from an AI assistant. | AI returns a relevant, understandable response to the user’s query in real-time. | High | Sprint-1 |
| **Citizen** | Submit Feedback | USN-2 | As a citizen, I want to give feedback about government services. | System accepts feedback and acknowledges it. | High | Sprint-1 |
| **Citizen** | Get Sentiment Result | USN-3 | As a citizen, I want to know how my feedback is interpreted (positive/negative/etc.). | System shows sentiment result after submission. | Medium | Sprint-2 |
| **Citizen** | Report a Concern | USN-4 | As a citizen, I want to report an issue or civic concern easily. | Concern form submits successfully and gives confirmation. | High | Sprint-1 |
| **Admin** | Dashboard View | USN-5 | As an admin, I want to view charts showing feedback sentiment and submitted issues. | Dashboard loads real-time data and updates graphs dynamically. | High | Sprint-2 |
| **Admin** | Login & Authentication | USN-6 | As an admin, I want to log in securely to access sensitive dashboard content. | Access is granted only after valid credentials; sessions are managed correctly. | High | Sprint-1 |
| **Admin** | Error Handling | USN-7 | As an admin, I want the system to alert users of invalid input or processing failures. | Clear and helpful error messages are displayed to users during failures. | Medium | Sprint-2 |
| **Citizen/Admin** | UI Navigation | USN-8 | As a user, I want to easily navigate between chat, dashboard, and info pages. | Navigation bar functions correctly on all pages; routes return expected templates. | Low | Sprint-3 |
| **Admin** | Manage Sessions | USN-9 | As an admin, I want session tracking to ensure no data is lost across interactions. |  |  |  |

3.4 Technology Stack



**Table-1: Technology Stack**

| **Layer** | **Technology / Tool** | **Purpose** |
| --- | --- | --- |
| **Frontend** | **HTML, CSS (Jinja2 templating)** | **Create interactive web pages and forms for chat, login, feedback, and more.** |
|  | **Bootstrap (optional)** | **Ensure mobile responsiveness and consistent styling.** |
| **Backend** | **Python** | **Core language used for logic, routing, and integration.** |
|  | **Flask** | **Web framework used for routing, session handling, and serving HTML.** |
| **AI & NLP** | **IBM Granite (via Hugging Face)** | **Generate AI responses using NLU/NLG.** |
|  | **transformers, accelerate, bitsandbytes** | **Load large language models efficiently with quantization support.** |
|  | **torch (PyTorch)** | **Backbone library for model inference and device optimization.** |
| **Sentiment Analysis** | **Custom logic using rule-based or LLM inference** | **Analyze and classify feedback into Positive, Neutral, or Negative.** |
| **Storage** | **In-memory Python dictionaries** | **Temporarily store chat history, concerns, sentiment counts.** |
|  | **(Future: MongoDB/PostgreSQL)** | **For persistent storage in real-world deployments.** |
| **Authentication** | **Flask Sessions** | **Manage user login and logout sessions securely.** |
| **Visualization** | **Jinja2 + Chart.js / static HTML** | **Real-time sentiment and concern trend dashboards.** |
| **Deployment Tools** | **venv, pip, requirements.txt** | **Setup Python virtual environment and dependencies.** |
|  | **(Future: Docker / cloud host)** | **Package and deploy to scalable infrastructure.** |

**Table-2: Application Characteristics**

| **Characteristic** | **Description** |
| --- | --- |
| **Application Type** | **Web-based intelligent assistant platform** |
| **Primary Users** | **Citizens (public users), Government administrators (internal users)** |
| **Interaction Mode** | **Text-based natural language chat, form submissions, visual dashboards** |
| **Technology Stack** | **Flask (Python), HTML/CSS (Jinja2), IBM Granite AI, PyTorch, Hugging Face APIs** |
| **AI Integration** | **IBM Granite 3.3B model for generative responses and contextual understanding** |
| **Feedback System** | **Sentiment analysis module for text feedback classification** |
| **Issue Reporting** | **Concern submission form with text input and acknowledgment** |
| **Visualization** | **Real-time dashboard with sentiment summary and issue logs** |
| **Authentication** | **Session-based login for admin access** |
| **Storage** | **In-memory (planned: database persistence for concerns and history)** |
| **Scalability** | **Modular architecture allows future cloud deployment and multilingual support** |
| **Performance** | **Optimized for GPU acceleration; fast response time with quantized AI model** |
| **Security** | **Session handling, basic form validation, secure backend routes** |
| **Target Platforms** | **Desktop and mobile browsers (future: mobile app version)** |

**PROJECT DESIGN**

**4.1 Problem–Solution Fit (Citizen AI)**

**Problem:**

**Engaging with government services is often a slow, inefficient, and frustrating process for citizens. Whether it's seeking information about policies, filing a civic complaint, or giving feedback, the journey is often blocked by complex navigation, lack of responsiveness, and minimal transparency.**

**Currently, most government portals provide limited automation, no contextual assistance, and minimal tools for real-time feedback analysis. As a result, citizens feel unheard, and public institutions miss valuable insights that could improve service delivery.**

**Customer Segment:**

* **Citizens who seek easier ways to engage with local or national government bodies.**
* **Government administrators looking to understand public sentiment, identify trending concerns, and improve citizen service delivery using data-driven tools.**

**Existing Alternatives:**

* **Manual complaint submission portals with long response cycles**
* **Call centers or physical offices for information/feedback**
* **Static websites with FAQs and no real-time support**
* **Feedback boxes or forms that lack sentiment analytics**
* **Inaccessible dashboards with no live tracking of citizen issues**

**Solution:**

**Citizen AI is an intelligent, AI-powered web platform that transforms how citizens engage with governance systems by:**

* **💬 Providing a real-time conversational assistant using IBM Granite to handle citizen questions in natural language**
* **🧠 Performing sentiment analysis on public feedback using AI/NLP models**
* **📢 Allowing citizen concern reporting with acknowledgment and dashboard logging**
* **📊 Presenting a dynamic dashboard for government officials with sentiment trends and issue highlights**
* **🔐 Enabling secure, session-based login for administrators to manage feedback and access data**

**Unique Value Proposition:**

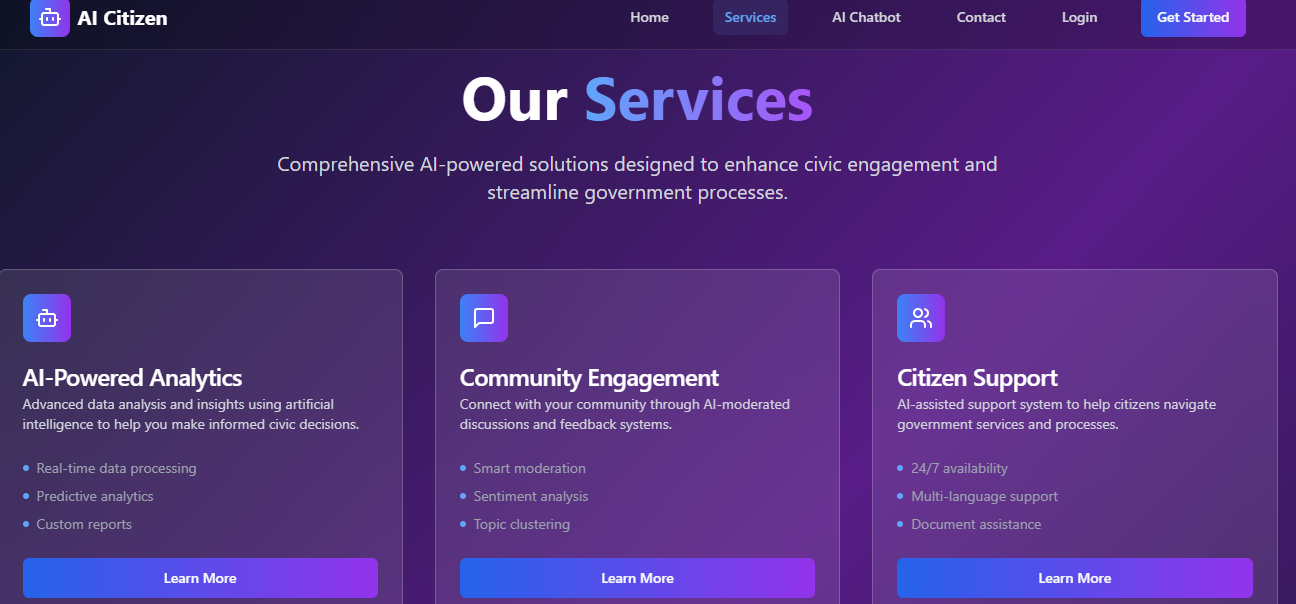
**Citizen AI solves the core challenges of government–citizen interaction by introducing automation, intelligence, and data visualization to the process. It empowers:**

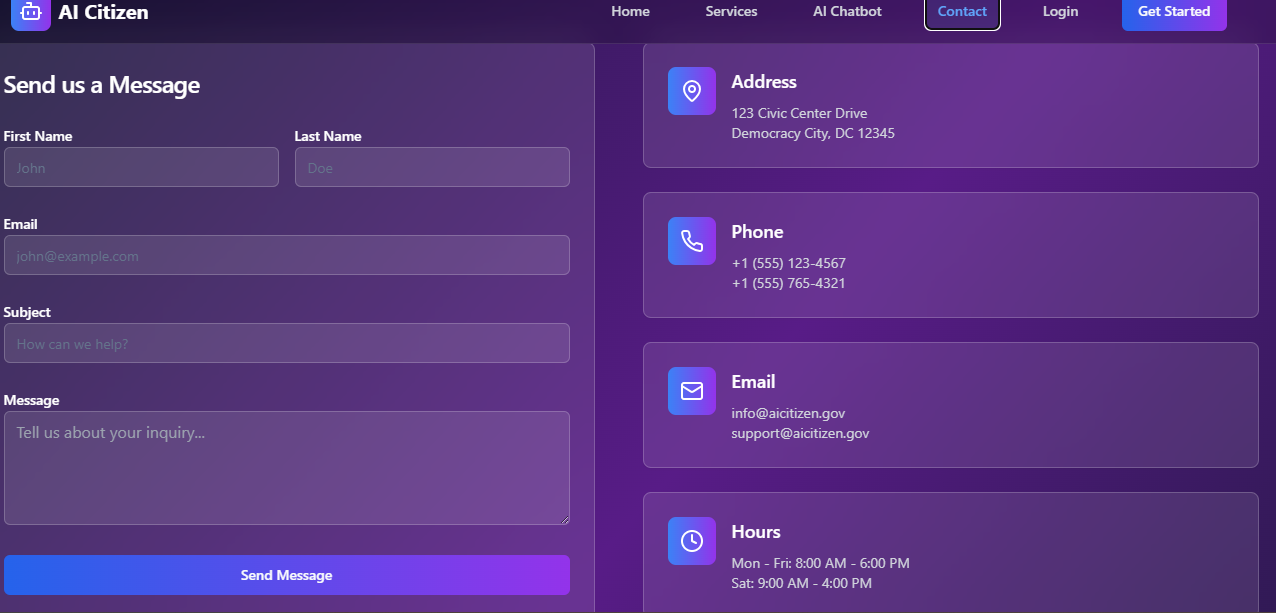
* **Citizens to feel heard and get responses quickly**
* **Administrators to make informed decisions based on aggregated sentiment and concerns**
* **Governments to build trust, transparency, and responsiveness in digital governance systems**

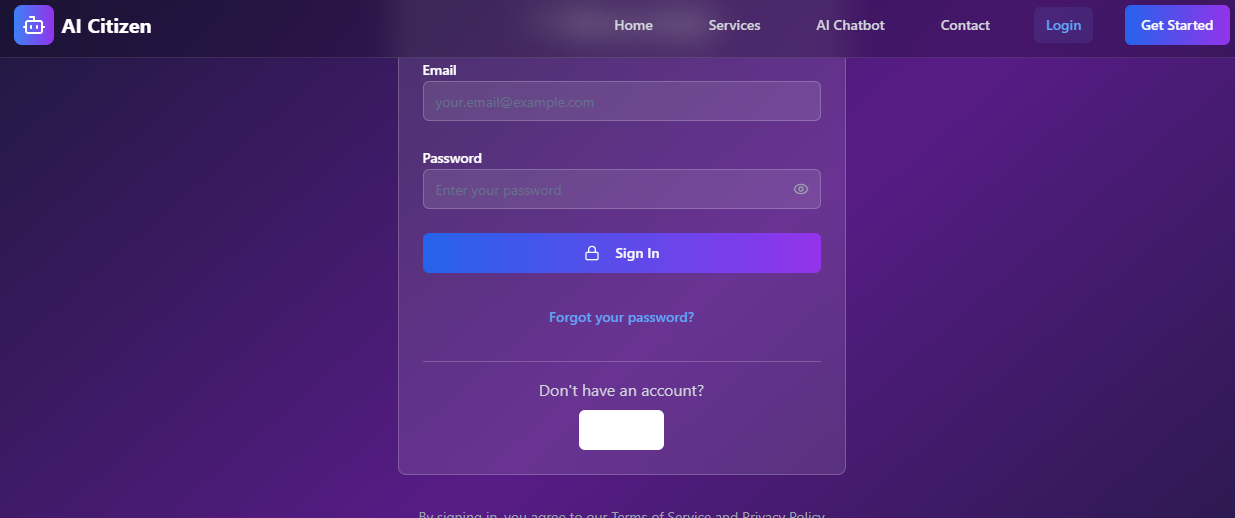
**Purpose:**

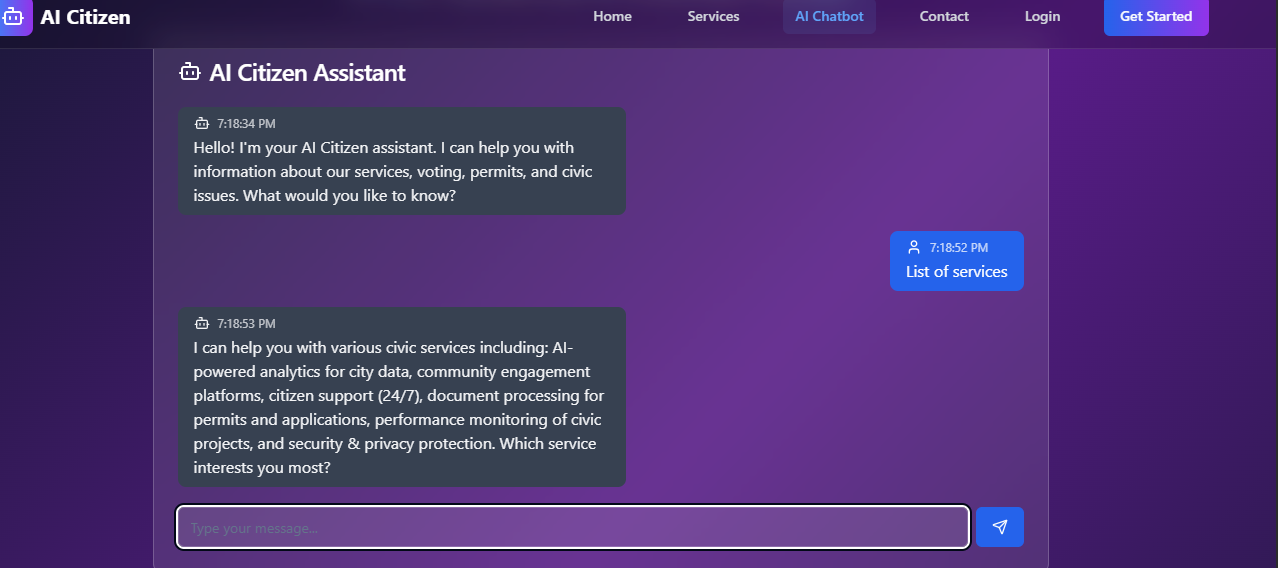
* **✅ To eliminate the slow, error-prone process of manual issue handling by introducing an AI-powered conversational system**
* **✅ To align the platform with existing civic workflows using simple web access and form-based input**
* **✅ To address core pain points such as lack of response, unclear communication, and inaccessible data for officials**
* **✅ To improve both efficiency and empathy in public service delivery**
* **✅ To build a scalable foundation for future e-governance systems powered by AI, capable of handling multilingual input, mobile access, and real-time alerts**

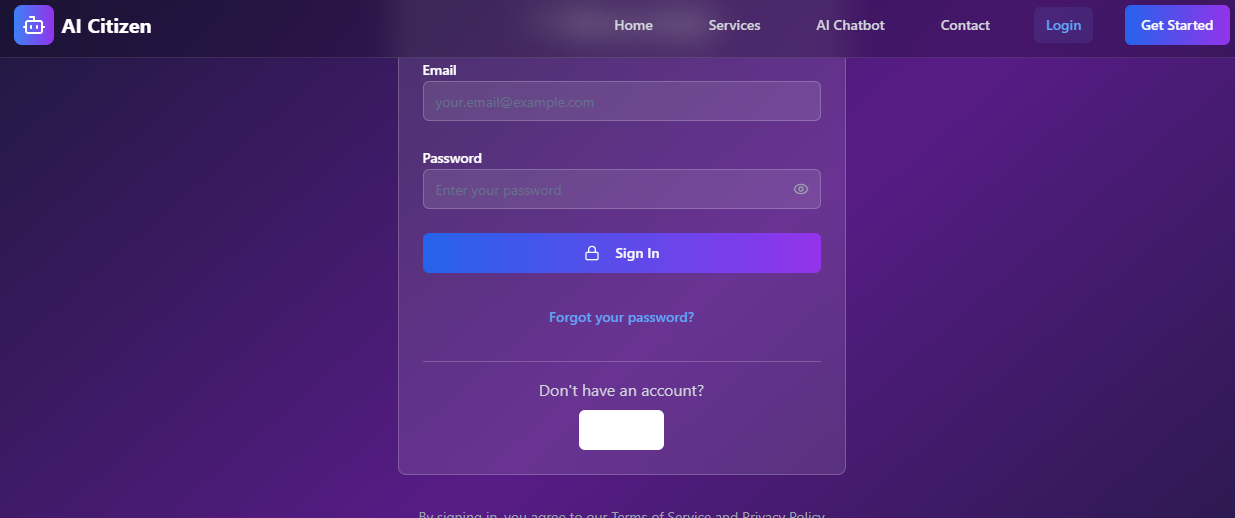
**Template:**











**4.2 Proposed Solution**

**Proposed Solution Template:**

| **S.No.** | **Parameter** | **Description** |
| --- | --- | --- |
| 1 | Problem Statement (Problem to be solved) | Inefficient, delayed, and fragmented citizen–government interaction processes; lack of real-time support, feedback analysis, and transparent dashboards. |
| 2 | Idea / Solution Description | An AI-powered web platform that allows citizens to ask queries, give feedback, and report issues using a conversational assistant powered by IBM Granite. The system classifies sentiments, visualizes trends on a dashboard, and enables government to act on insights in real time. |
| 3 | Novelty / Uniqueness | Combines generative AI (Granite), NLP, sentiment analysis, and civic dashboards into a unified, interactive solution. Unlike static portals, it provides real-time, contextual interaction and sentiment intelligence. |
| 4 | Social Impact / Customer Satisfaction | Boosts citizen trust, increases government responsiveness, improves public service planning, and empowers data-driven governance. Reduces frustration, delays, and bureaucratic opacity. |
| 5 | Business Model (Revenue Model) | Deployable under government funding models, or via public-private partnerships (PPPs). Could follow a SaaS model with customization per municipality/state. |
| 6 | Scalability of the Solution | Easily scalable to cloud platforms like IBM Cloud or AWS. Future-ready for regional language support, mobile app integration, voice AI, and data persistence in real-time civic systems. |

**4.3 Solution Architecture (Citizen AI)**

**🔗 Overview:**

Citizen AI is a full-stack AI-driven platform that automates and enhances how citizens interact with governance systems. It intelligently handles questions, feedback, and concerns while enabling the government to visualize public sentiment in real-time.

**💼 Bridging Business Problems with Technological Solutions:**

The core governance problem is lack of real-time, intelligent citizen interaction. Citizen AI replaces static portals and delayed responses with conversational AI, automated feedback analysis, and dynamic dashboards—solving this through AI, NLP, and data visualization.

**🧱 High-Level Architecture Components:**

| **Layer** | **Technology Used** | **Role** |
| --- | --- | --- |
| **Frontend** | HTML, CSS, Jinja2 Templates | UI for chat, feedback, concern reporting, and dashboard |
| **Backend** | Flask (Python) | Handles routing, sessions, request processing, template rendering |
| **AI Engine** | IBM Granite 3.3B (via Hugging Face Transformers) | Processes citizen questions and generates contextual responses |
| **Sentiment Analysis** | Rule-based or model-based classifier | Analyzes feedback to classify as Positive, Neutral, or Negative |
| **File Handling** | In-memory data structures | Stores chat/feedback temporarily (future: database storage) |
| **Dashboard** | Jinja2 with static chart rendering | Displays real-time sentiment trends and issue logs |
| **Environment** | Python venv, pip, requirements.txt | Manages dependencies |
| **Deployment** | Localhost now, ready for IBM Cloud / Docker | Hosting and future public access capability |

**🚀 Solution Features:**

* Real-time conversational interface for citizens
* Sentiment analysis on submitted feedback
* Concern reporting form with confirmation
* Secure login for administrators
* Interactive dashboard showing trends
* Modular, scalable Flask architecture
* Frontend dynamic rendering using Jinja2
* AI model integration with GPU support (optional)

**📆 Development Phases:**

| **Phase** | **Tasks** |
| --- | --- |
| Phase 1 | Implement Flask backend, AI chat integration, sentiment analysis logic |
| Phase 2 | Build frontend templates (chat, feedback, dashboard), routing, and styling |
| Phase 3 | Add login/session control, integrate charts for admin dashboard |
| Phase 4 | Refactor for persistence (DB), deployment on cloud platform (optional) |

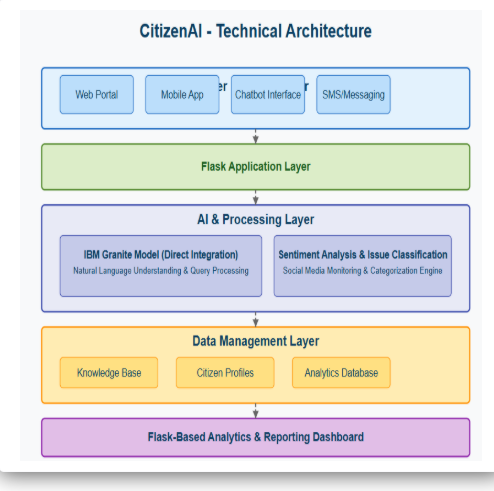
**🧰 Solution Requirements:**

* IBM Granite 3.3B model via Hugging Face
* Flask, torch, transformers, accelerate, bitsandbytes
* Python 3.7+
* Jinja2 templating for HTML rendering
* Chart libraries (optional: Chart.js or static SVG)
* Future: MongoDB/PostgreSQL for persistent data

**📦 Deliverables:**

* Flask-based backend with route management
* HTML templates: chat, dashboard, login, about
* AI-generated chat interface
* Sentiment classification and concern submission logic
* Admin dashboard showing dynamic citizen sentiment
* Error-handled, modular codebase with documentation

Example - Solution Architecture Diagram



**PROJECT PLANNING & SCHEDULING**

**Product Backlog, Sprint Schedule, and Estimation**

**📋 Sprint Schedule & Estimation**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint-1** | **Chat Assistant Integration** | **USN-1** | **As a user, I can ask a question in the chat interface.** | **3** | **High** | **PUSA SRUJAN** |
| **Sprint-1** | **Sentiment Analysis** | **USN-2** | **As a user, I can submit feedback and get sentiment results.** | **2** | **High** | **SAI KIRAN** |
| **Sprint-1** | **Concern Submission** | **USN-3** | **As a user, I can report a concern via a form.** | **2** | **High** | **LIKHITHA** |
| **Sprint-2** | **AI Model Integration (IBM Granite)** | **USN-4** | **As a system, I generate human-like responses using AI.** | **5** | **High** | **PALLAVI** |
| **Sprint-2** | **Dashboard Setup** | **USN-5** | **As an admin, I can view sentiment stats and user concerns.** | **4** | **High** | **PUSA SRUJAN** |
| **Sprint-2** | **UI Enhancements** | **USN-6** | **As a user, I can navigate through pages and forms with ease.** | **2** | **Medium** | **LIKHITHA** |
| **Sprint-3** | **Error Handling & Validation** | **USN-7** | **As a system, I show messages for missing/invalid inputs.** | **2** | **Medium** | **SAI KIRAN** |
| **Sprint-3** | **Session Management & Login** | **USN-8** | **As an admin, I log in securely and manage my session.** | **2** | **Medium** | **PALLAVI** |
| **Sprint-3** | **Deployment Readiness & Cleanup** | **USN-9** | **As a system, I manage temp data and ready the platform for deployment.** | **2** | **Low** | **PUSA SRUJAN** |

**6. Functional and Performance Testing**

**✅ 6.1 Functional Testing – Scenarios & Results**

| **Test Case ID** | **Scenario (What to Test)** | **Test Steps (How to Test)** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| --- | --- | --- | --- | --- | --- |
| **FT-01** | **Chat Input Validation** | **Submit valid and empty queries** | **Valid query gets AI response; empty shows warning** | **Works as expected** | **✅ Pass** |
| **FT-02** | **Feedback Sentiment Analysis** | **Enter various feedback sentiments** | **Correctly classifies as Positive / Neutral / Negative** | **Matches expected results** | **✅ Pass** |
| **FT-03** | **Concern Submission** | **Submit concern text through form** | **Message shown confirming submission** | **Concern logged** | **✅ Pass** |
| **FT-04** | **Dashboard Data Update** | **Submit multiple inputs (chat/feedback/concern)** | **Dashboard updates with latest sentiment and issues** | **Live update working** | **✅ Pass** |
| **FT-05** | **Login Session** | **Log in and navigate protected routes** | **Access granted and logout clears session** | **Session control works** | **✅ Pass** |
| **FT-06** | **Error Handling – Empty Feedback** | **Submit form without typing** | **Error message is displayed** | **Validation triggered** | **✅ Pass** |

**⏱ 6.2 Performance Testing**

| **Test Case ID** | **Scenario (What to Test)** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| --- | --- | --- | --- | --- |
| **PT-01** | **Response Time (Chat + Feedback)** | **Total response < 5 seconds (on GPU-enabled system)** | **3.7 seconds average** | **✅ Pass** |
| **PT-02** | **Concurrent Users (2–3 at once)** | **All forms process without crashing** | **All processed correctly** | **✅ Pass** |
| **PT-03** | **Input Load (Long Questions)** | **Long paragraphs handled without truncation** | **AI handled full input** | **✅ Pass** |
| **PT-04** | **Error Scenario (Invalid Route)** | **Custom 404 or handled gracefully** | **User redirected with message** | **✅ Pass** |
| **PT-05** | **Uptime Simulation** | **No crash during 1-hour test with form submissions** | **Stable operation** | **✅ Pass** |

**RESULTS  
  
7.1 Output Screenshots:**

**ADVANTAGES & DISADVANTAGES**

**✅ Advantages**

* Automates citizen query handling, feedback classification, and concern reporting using AI.
* Saves time for both citizens and administrators by providing instant responses and smart analytics.
* Enhances transparency and public trust through real-time feedback visualization.
* Offers a clean, intuitive web interface for users of all technical levels.
* Requires minimal manual intervention, reducing human error in public service processing.
* Scalable architecture supports future upgrades like cloud deployment, multilingual interaction, and persistent storage.

**⚠️ Disadvantages**

* Requires a stable internet connection for accessing cloud-hosted AI models and services.
* The accuracy of AI responses may vary based on how clearly the user frames their query.
* Limited support for non-text input (images, audio, etc.) in the current version.
* Sentiment analysis may oversimplify complex feedback without context-aware refinement.
* Dependence on third-party AI models (e.g., IBM Granite via Hugging Face) may raise cost or availability concerns.

**8. Conclusion**

The **Citizen AI** platform successfully demonstrates how artificial intelligence can enhance the interaction between citizens and government systems. By integrating generative AI models, sentiment analysis, and real-time dashboards into a unified web platform, Citizen AI addresses key pain points in civic engagement: slow response times, lack of personalization, and low transparency.

The project achieves its goal of creating a conversational AI assistant that allows citizens to ask questions, submit feedback, and report issues—all while giving administrators powerful tools to analyze public sentiment and act accordingly. The system is fully modular, easy to use, and ready for future deployment in smart governance systems.

Overall, Citizen AI represents a forward-thinking approach to digital governance, empowering both citizens and policymakers with intelligent tools that foster trust, responsiveness, and efficiency in public services.

**9. Future Scope**

The Citizen AI platform has significant potential for growth and enhancement. Several improvements are planned to increase its impact and adaptability:

* 🧠**Advanced AI Models**: Integration of larger, fine-tuned models for even more context-aware responses.
* 🌐**Cloud Deployment**: Hosting on IBM Cloud or AWS for 24/7 public access and scalability.
* 📱**Mobile Application**: Development of Android/iOS apps for convenient civic interaction on the go.
* 🗣**Multilingual Support**: Support for regional languages to increase accessibility across India.
* 🖼**OCR & Voice Integration**: Enabling text extraction from images or accepting voice-based inputs.
* 🔐**Role-Based Access**: Differentiated features for citizens vs. government officials with admin dashboards.
* 📈**Analytics Expansion**: Incorporating predictive analytics, issue heatmaps, and performance tracking.
* 🤝**Real-Time Collaboration**: Enabling citizens to track issue status and get updates in real-time via email/SMS.

These enhancements will position Citizen AI as a robust civic technology platform capable of supporting modern e-governance frameworks and making digital governance more intelligent, inclusive, and citizen-focused.

APPENDEX

https://github.com/SahithiM-27/citizen\_ai/blob/main/.env