**Project Design Phase-II**

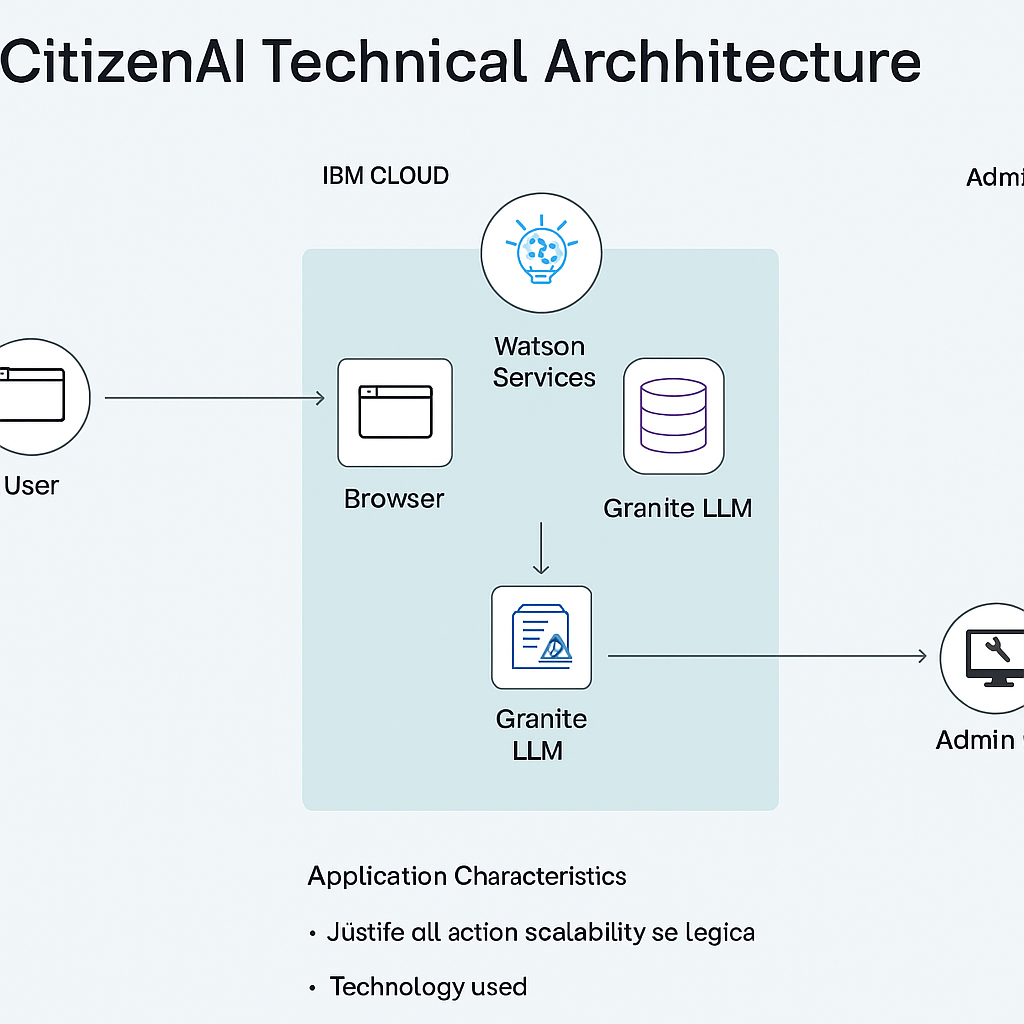
**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 31 January 3035 |
| Team ID | LTVIP2025TMID35138 |
| Project Name | Citizen AI – Intelligent Citizen Engagement Platform |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

The CitizenAI platform follows a modular, scalable architecture:

* Frontend: Web UI using React.js or HTML/CSS for citizen interaction
* Middleware: Python Flask server to manage business logic, route queries to AI
* AI Layer: IBM Watson Assistant / Granite for NLP and intelligent Q&A
* Database Layer: MongoDB and IBM Cloudant for data persistence
* Cloud Deployment: Hosted on IBM Cloud using either Cloud Foundry or Kubernetes
* Optional APIs: Aadhaar, Weather, Civic scheme APIs
* Admin Panel: For scheme management, user monitoring, and analyticsv



**Table-1 : Components & Technologies:**

| **S. No** | **Component** | **Description** | **Technology** |
| --- | --- | --- | --- |
| **1** | **User Interface** | **Frontend interface used by citizens to interact with chatbot and dashboard** | **HTML5, CSS3, JavaScript, Bootstrap, React JS** |
| **2** | **Application Logic-1** | **Handles user session, routing, chatbot input & output** | **Python (Flask Framework)** |
| **3** | **Application Logic-2** | **Converts speech to text if voice input is enabled** | **IBM Watson Speech-to-Text** |
| **4** | **Application Logic-3** | **Processes natural language queries, generates AI responses** | **IBM Watson Assistant / IBM Granite LLM** |
| **5** | **Database** | **Stores scheme data, user queries, and preferences** | **MongoDB (NoSQL)** |
| **6** | **Cloud Database** | **Cloud-based persistent document storage** | **IBM Cloudant** |
| **7** | **File Storage** | **Stores uploaded PDFs / images and related extracted content** | **IBM Cloud Object Storage or Local Filesystem** |
| **8** | **External API-1** | **Fetches real-time civic data or environmental info** | **IBM Weather API (optional civic integration)** |
| **9** | **External API-2** | **For future authentication with Aadhaar or regional ID** | **UIDAI Aadhaar API (planned integration)** |
| **10** | **Machine Learning Model** | **NLP-based query understanding, document summarization** | **IBM Granite / Custom Prompt-tuned LLM** |
| **11** | **Infrastructure** | **Cloud-based deployment** | **IBM Cloud (Cloud Foundry or Kubernetes) or Render.com** |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristic** | **Description** | **Technology / Approach** |
| --- | --- | --- | --- |
| **1** | **Open-Source Frameworks** | **Reusable, modular open-source tech stack for frontend/backend** | **React.js, Flask (Python), Bootstrap, Node.js (optional)** |
| **2** | **Security Implementations** | **Secure user login, session control, encrypted communications** | **OAuth 2.0, HTTPS, JWT Tokens, SHA-256 Password Hashing** |
| **3** | **Scalable Architecture** | **Modular services with independent components for API, ML, UI (support microservices if needed)** | **Flask + MongoDB + IBM Watson = loosely coupled system** |
| **4** | **Availability** | **High uptime via cloud deployment, potential for load balancing & autoscaling** | **IBM Cloud Kubernetes / Cloud Foundry with auto-scaling** |
| **5** | **Performance** | **Fast AI replies, optimized DB queries, potential caching for scheme data & UI component** | **CDN (optional), MongoDB indexing, Async Flask handlers** |