**Project Design Phase**

**Citizen AI -Intelligent citizen engagement platform**

|  |  |
| --- | --- |
| Date | 28,June 2025 |
| Team ID | LTVIP2025TMID60842 |
| Project Name | Citizen Ai -intelligent citizen engagement platform |
| Maximum Marks | 4 Marks |

**Technical Architecture – Citizen AI: Intelligent Citizen Engagement Platform**

**Use Case:  
A robust platform enabling efficient citizen services interaction via multiple channels (voice, chat, mobile/web). Integrates AI for smart response, processes identity verification (Aadhar), weather alerts, document uploads, and public service requests.**

**Architectural Diagram**

**(⚠️ I can generate a visual diagram if you'd like – just let me know.)**

**High-Level Layers (C4 Inspired):**

1. **Presentation Layer – Web/Mobile/Chatbot UI**
2. **Application Logic Layer – Multi-service AI-enabled processing**
3. **Data & ML Layer – Local/Cloud DB + AI model integration**
4. **Infrastructure Layer – Scalable, resilient cloud services**

**Table 1: Components & Technologies**

| **S.No** | **Component** | **Description** | **Technology Used** |
| --- | --- | --- | --- |
| **1** | **User Interface** | **Multi-channel access: Web, Mobile App, Voice Bot, Chatbot** | **HTML, CSS, JavaScript, ReactJS, Flutter, Watson Assistant UI** |
| **2** | **Application Logic-1** | **Core citizen request handling logic (complaints, queries, feedback)** | **Java / Python (Flask, Spring Boot)** |
| **3** | **Application Logic-2** | **Voice-based interaction and transcriptions** | **IBM Watson Speech-to-Text (STT)** |
| **4** | **Application Logic-3** | **Conversational interface for services (chatbots)** | **IBM Watson Assistant** |
| **5** | **Database** | **Structured and unstructured citizen request data** | **MySQL / MongoDB** |
| **6** | **Cloud Database** | **Cloud-hosted citizen services DB** | **IBM Cloudant / IBM Db2 on Cloud** |
| **7** | **File Storage** | **Uploads (documents, images, complaints with attachments)** | **IBM Cloud Object Storage / IBM Block Storage** |
| **8** | **External API-1** | **Weather alerts for civic planning or emergency notification** | **IBM Weather API** |
| **9** | **External API-2** | **Identity validation and eKYC integration** | **Aadhar API, DigiLocker API** |
| **10** | **Machine Learning Model** | **Sentiment analysis, topic classification on citizen feedback** | **IBM Watson NLP / Custom ML model in Python** |
| **11** | **Infrastructure (Server/Cloud)** | **Hosting and deployment of services** | **IBM Cloud Kubernetes Service / Cloud Foundry / CI/CD via Tekton** |

**Table 2: Application Characteristics**

| **S.No** | **Characteristics** | **Description** | **Technology Used** |
| --- | --- | --- | --- |
| **1** | **Open-Source Frameworks** | **Backend & frontend built using community-supported libraries** | **ReactJS, Flask, Spring Boot, Kubernetes, Docker** |
| **2** | **Security Implementations** | **Secure APIs, encrypted data, role-based access, identity management** | **OAuth 2.0, SHA-256, JWT, HTTPS, IBM IAM, OWASP top 10** |
| **3** | **Scalable Architecture** | **Modular microservices for scalable deployment & independent maintenance** | **Kubernetes, REST APIs, Cloud-native architecture** |
| **4** | **Availability** | **Highly available through distributed architecture, failover, and auto-scaling** | **IBM Load Balancer, Multi-Zone Regions, Replica Sets** |
| **5** | **Performance** | **Optimized with CDN, request throttling, Redis-based caching, asynchronous processing** | **IBM Cloud CDN, Redis, RabbitMQ, API Gateway** |