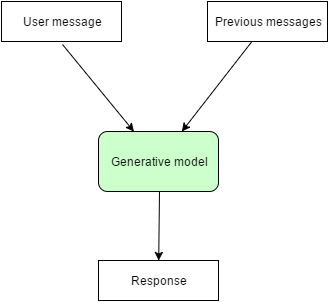
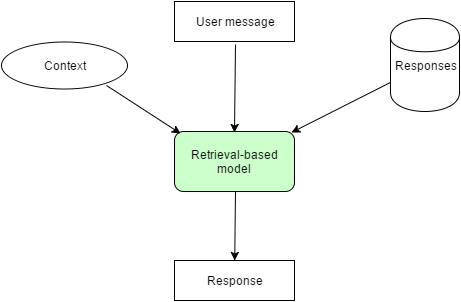
**Project Design Phase-II**

**3.4Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 31 Janaury 2025 |
| Team ID | LTVIP2025TMID32100 |
| Project Name | Citizen AI |
| Maximum Marks | 4 Marks |





**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | Chatbot interface for user interaction (issue reporting, Q&A). | **Gradio**, HTML, CSS. |
|  | Application Logic-1 | Handles routing, query preprocessing, response formatting | **Python**, FastAPI |
|  | Application Logic-2 | AI model interaction and natural language understanding | **IBM Granite 3.3**, Hugging Face Transformers |
|  | Application Logic-3 | Query classification and response generation logic | Python-based logic & custom intents |
|  | Database | Optional storage for FAQs, schemes, issue categories | JSON files / SQLite (optional) |
|  | Cloud Database | For storing user queries, feedback, issue logs (future scope) | IBM Cloudant, Firebase (optional) |
|  | File Storage | Storing logs, screenshots (if uploaded), or static JSON files | Local Filesystem / IBM Object Storage (future) |
|  | External API-1 | To fetch area-specific government service information (future) | e-Seva / RTI APIs (future integration) |
|  | External API-2 | (Optional) To verify user or connect to citizen services | Aadhar API, etc. |
|  | Machine Learning Model | Understand and respond to user queries in natural language | IBM Granite Model / Fine-tuned Transformers |
|  | Infrastructure (Server / Cloud) | Deployed via Google Colab; can migrate to IBM Cloud or local host | **Google Colab**, IBM Cloud, Local/VM, Docker |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | Frameworks and tools used to build the platform | Gradio (Python), FastAPI, Hugging Face Transformers |
|  | Security Implementations | Application-level security (basic for now, expandable) | Token-based access, HTTPS (when deployed), basic auth Future: OAuth2, IAM, OWASP practices |
|  | Scalable Architecture | Modular backend, pluggable AI, potential for microservices and API gateways | Microservices-friendly: FastAPI + AI Models separated |
|  | Availability | Can be deployed to cloud, supports scaling through containerization and serverless platforms | IBM Cloud, Docker, Cloud Foundry (optional) |
|  | Performance | Optimized for fast inference using lightweight models, Gradio sessions cached | GPU-enabled Colab, Caching in FastAPI (future), Preloaded responses |