**Project Design Phase**

**Solution Architecture**

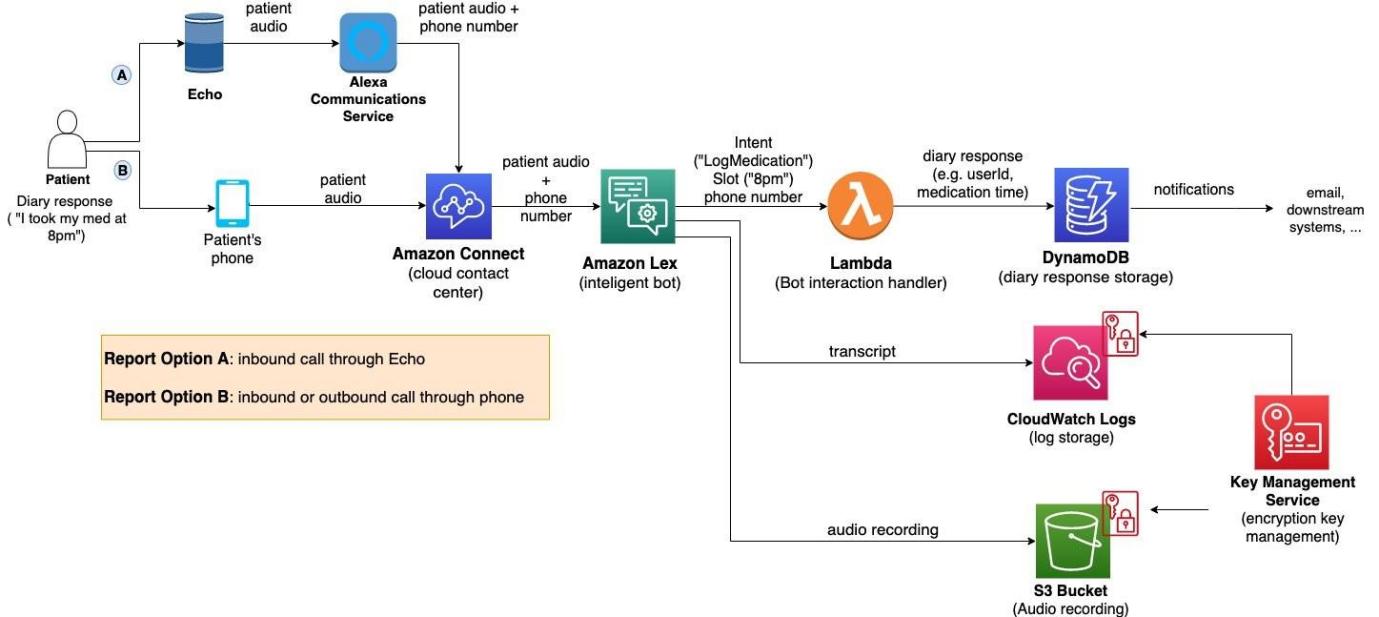
|  |  |
| --- | --- |
| Date | 24 June 2025 |
| Team ID | LTVIP2025TMID59935 |
| Project Name | HealthAI: Intelligent Healthcare Assistant Using IBM Granite |
| Maximum Marks | 4 Marks |

**Solution Architecture:**

HealthAI bridges the gap between users and accessible healthcare through a sophisticated, AI-powered platform. It uses IBM Watson Machine Learning and Granite-13b-instruct-v2, integrated via Streamlit, to create a seamless, secure, and intelligent health assistant. HealthAI assists users in symptom analysis, personalized treatment, trend tracking, and medical Q&A, improving patient autonomy and decision-making.  
  
Goals of the Solution Architecture:  
- To deliver intelligent health support using natural language understanding and medical knowledge modeling.  
- To define the data flow between user input, AI processing (Granite model), and output generation.  
- To describe the core components and how they interact: UI (Streamlit), AI Engine (IBM Granite & Watson ML), APIs, Analytics Dashboard.  
- To ensure data privacy and secure API key management.  
- To define a scalable architecture capable of handling multiple health scenarios in real-time.  
  
Solution Components & Flow:  
  
1. Frontend (User Interface):  
- Streamlit-based interface for accessibility.  
- Modules:  
 - Patient Chat  
 - Disease Prediction  
 - Treatment Plans  
 - Health Analytics Dashboard  
  
2. AI & ML Engine:  
- IBM Granite-13b-instruct-v2 model for natural language processing.  
- Watson Machine Learning for model training and deployment.  
- Input: Symptom descriptions, medical questions, diagnosis, health data.  
- Output: Diagnoses, treatment suggestions, health insights, Q&A responses.  
  
3. Backend Logic:  
- Flask/FastAPI (optional) to route and process user input.  
- Requests sent to IBM Watson API endpoints.  
- Use of structured health data formats (e.g., HL7/FHIR).  
  
4. Database Layer:  
- Stores:  
 - Patient profile (with consent)  
 - Historical symptom logs  
 - Analytical insights  
 - Chat history (for continuity)  
  
5. Security & API Management:  
- Role-based authentication (user-level access to personal data).  
- API key usage secured via environment variables.  
- Encryption for data at rest and in transit.  
  
Architecture Diagram (Suggested):  
  
[User]   
 ↓ (Symptoms / Questions / Data)  
[Streamlit UI]  
 ↓  
[Backend Service Layer]  
 ↓ ↘  
[IBM Watson & Granite ML] [Database Layer]  
 ↓  
[Output to User: Diagnosis / Chat / Insights]

* Find the best tech solution to solve existing business problems.
* Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
* Define features, development phases, and solution requirements.
* Provide specifications according to which the solution is defined, managed, and delivered.

**Example - Solution Architecture Diagram:**



*Figure 1: Architecture and data flow of the voice patient diary sample application*

**Reference: <https://aws.amazon.com/blogs/industries/voice-applications-in-clinical-research-powered-by-ai-on-aws-part-1-architecture-and-design-considerations/>**