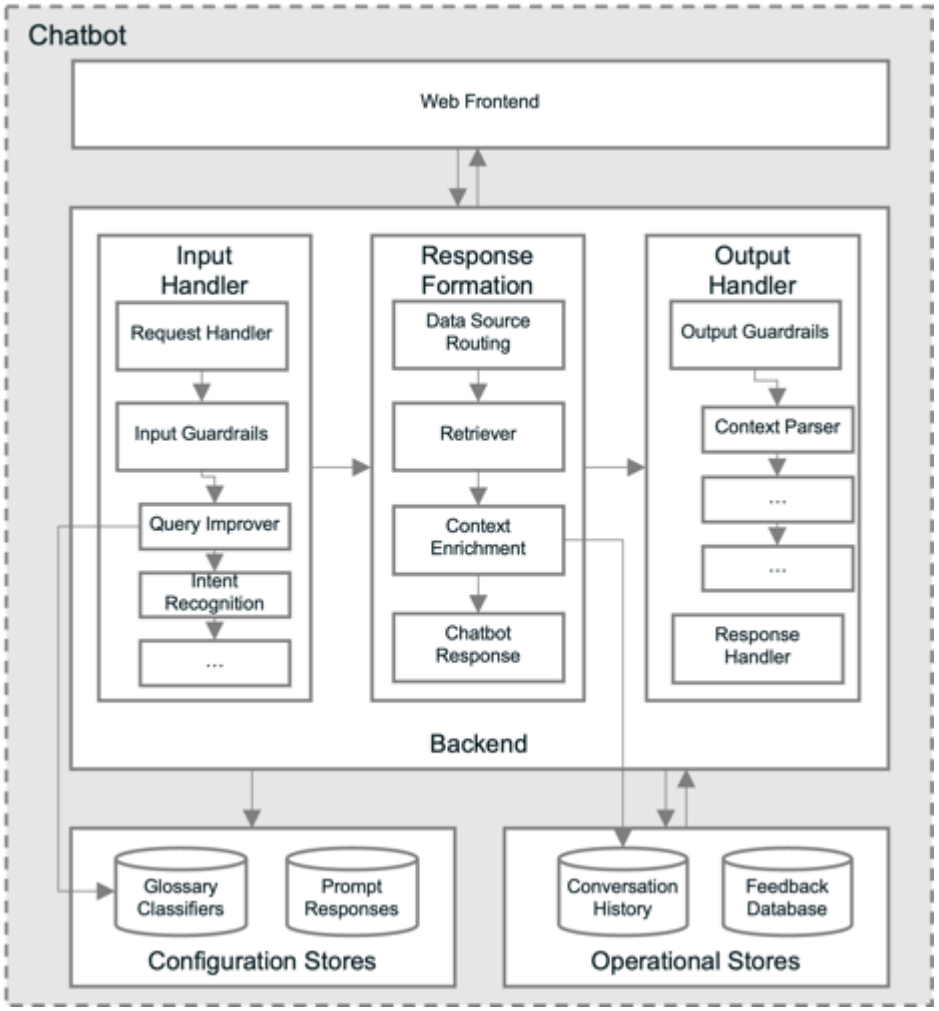
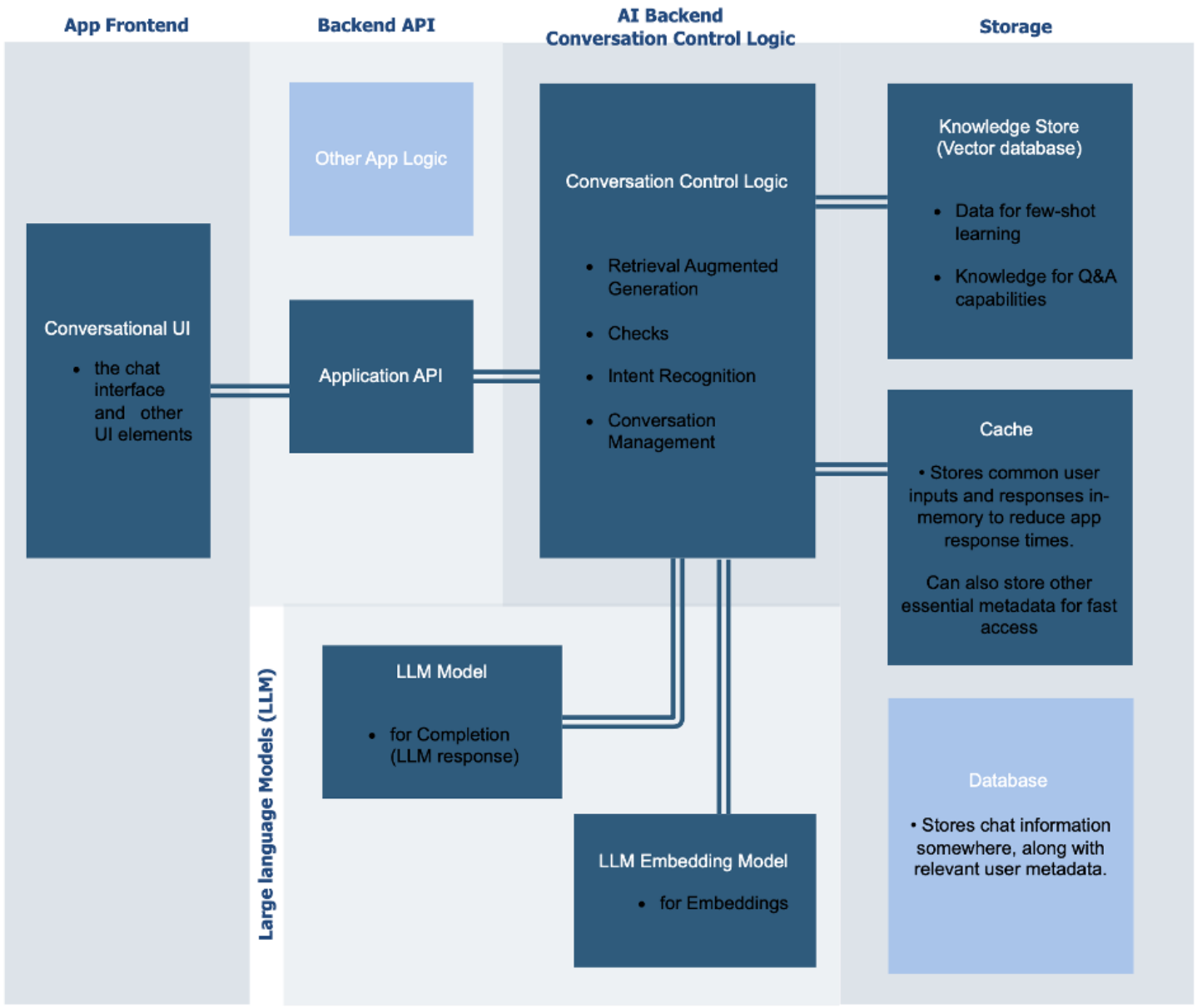

Chatbot
Web Frontend
Input Handler
Request Handler
Input Guardrails
Query Improver
Intent Recognition
...
Response Formation
Data Source Routing
Retriever
Context Enrichment
Chatbot Response
Output Handler
Output Guardrails
Context Parser
...
...
Response Handler
Backend
Glossary Classifiers
Prompt Responses
Configuration Stores
Conversation History
Feedback Database
Operational Stores


App Frontend
Backend API
AI Backend
Conversation Control Logic
Storage
Other App Logic
Conversation Control Logic
Retrieval Augmented Generation
Checks
Intent Recognition
Conversation Management
Knowledge Store (Vector database)
Data for few-shot learning
Knowledge for Q&A capabilities
Conversational UI
the chat interface and other UI elements
Application API
Cache
• Stores common user inputs and responses in-memory to reduce app response times.
Can also store other essential metadata for fast access
Large language Models (LLM)
LLM Model
for Completion (LLM response)
LLM Embedding Model
for Embeddings
Database
• Stores chat information somewhere, along with relevant user metadata.

Below is the architecture and detailed component breakdown for the Sustainable Smart City Assistant using IBM Granite LLM, inspired by IBM's AI-powered backend pattern.

🏛️ Architecture Overview

This is a high-level C4 Context + Container Diagram:

- **Users interact through a Web or Mobile UI (chat interface, dashboard).**
- **The Front-end communicates with a Flask Backend, hosted on IBM Cloud.**
- **The backend handles:**
  - **User Auth & Registration**
  - **Query Routing**
  - **City Data Retrieval from internal or external APIs.**
  - **LLM Interaction with IBM Granite LLM and custom ML pipelines.**
  - **Response Generation**
- **Data is stored in:**
  - **User DB (SQL/NoSQL) for auth and preferences.**
  - **City Services DB for environmental, transport, utilities data.**

- Feedback / Logs DB for continuous retraining.

- Deployed with Cloud infrastructure (Cloud Foundry/Kubernetes), load balancers, monitoring, and security layers.

Table-1: Components & Technologies

S.No	Component	Description	Technology
1	User Interface	Web & mobile chat + dashboards	React.js, HTML/CSS, JS, React Native
2	Application Logic-1	Core backend application logic	Python (Flask)
3	Application Logic-2	Natural Language Processing integration	IBM Granite LLM API
4	Application Logic-3	Data processing & query orchestration	Python (custom ML pipelines)
5	Database	Structured user data	PostgreSQL / MongoDB
6	Cloud Database	Scalable cloud DB for city data & logging	IBM DB2 / Cloudant
7	File Storage	Storing logs, reports, and static assets	IBM Cloud Object Storage
8	External API-1	Real-time weather, traffic & utility data	IBM Weather API, City Data APIs
9	External API-2	Identity verification & geolocation	Municipal/Aadhaar API / GeoCoding API
10	Machine Learning Model	Custom sustainability model (e.g. energy prediction)	Python, scikit-learn, TensorFlow/PyTorch
11	Infrastructure (Server / Cloud)	App deployment platform	IBM Cloud Foundry / Kubernetes + LB

Table-2: Application Characteristics

S.No	Characteristic	Description	Technology
1	Open-Source Frameworks	Frontend, backend, ML pipelines	React.js, Flask, scikit-learn, TensorFlow, Kubernetes
2	Security Implementations	Encrypt data at rest/in transit, access control, OWASP hardening	TLS/HTTPS, JWT/OAuth, IAM, OWASP best practices
3	Scalable Architecture	3-tier microservices, containerized deployment, autoscaling	Kubernetes, Cloud Foundry, Docker, Helm
4	Availability	Geo-redundant deployment, HTTPS LB, passive failover	IBM Cloud Load Balancer, Pod redundancy, CI/CD pipelines
5	Performance	High request volume, caching, CDN, async processing	Redis Cache, CDN, Celery, RabbitMQ for asynchronous jobs

---

✅ **Summary**

- **Component Architecture closely mirrors modern microservices with clear separation of UI, backend logic, LLM integration, and data layers.**
- **Technologies are IBM-focused with Granite LLM, Cloudant/DB2, and OSS components.**
- **Non-functional design ensures security, availability, scalability, and performance.**
