B. Why Microservices is the Best Choice for AssistX

1. Scalability

- AI processing service may require:
    - High CPU/GPU resources
- Ticket service may need:
    - High database throughput
- Microservices allow:
    - Scaling only required services
    - Horizontal scaling of AI module separately

✓ Efficient resource utilization

---

2. Maintainability

- Codebase is divided into smaller services
- Easier debugging and testing
- Changes in AI logic do not affect:
    - Authentication module
    - Ticket management module

✓ Improves modularity and maintainability

---

3. Performance

- Independent services reduce system bottlenecks
- Load balancing possible per service
- AI model can be optimized without affecting other services

✓ Better performance under heavy traffic

---

4. Fault Isolation

- If AI service crashes:
    - Ticket management still works
- If Notification service fails:
    - Core system remains operational

✓ Improved system reliability

---

5. Future Extensibility

AssistX can later include:

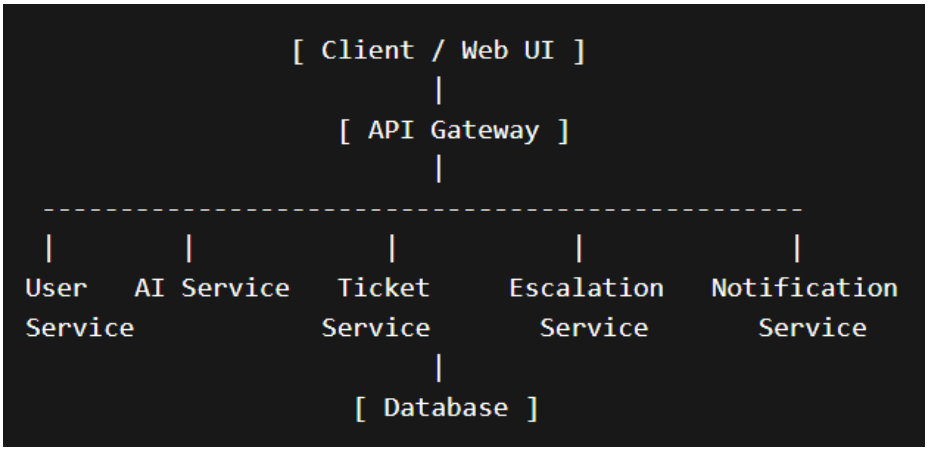
- Chatbot integration
- Voice-based support
- Third-party CRM integration
- Analytics dashboard

Microservices allow:

- Adding new services without modifying the entire system

✓ Highly extensible architecture

Simple Architectural Diagram



AssistX follows a **Microservices Architecture** because:

- It consists of **loosely coupled, independently deployable services**
- Each service represents a **specific business capability**
- It provides:
    - High scalability
    - Better maintainability
    - Fault isolation

Therefore, Microservices Architecture is the most suitable choice for AssistX.