EchoNest AI - Contract Compliance Summary

Overview

This document provides a summary of compliance requirements for the EchoNest AI system based on the service contracts. It highlights critical requirements that must be strictly followed during development to ensure proper integration between components.

Critical Compliance Requirements by Component

1. Frontend (Web App)

Authentication & User Management

- **Must** implement JWT token storage, refresh, and expiry handling
- **Must** support multi-role UI (parent/teacher/admin) with appropriate access controls
- **Must** implement password reset and email verification flows

Content Management

- **Must** support multiple file formats (PDF, DOCX, audio, video) for upload
- **Must** implement SSE consumers for real-time upload progress
- **Must** allow content assignment to specific children or groups

RAG Chat Interface

- **Must** implement token streaming via SSE for responsive chat experience
- **Must** display source attribution for RAG responses
- **Must** provide feedback and flagging mechanisms for content control

Metrics & Diagnostics

- **Must** display device status and sync information
- **Must** visualize usage analytics by child or group
- **Must** provide notification preferences management

2. Backend Server (Online Version)

Authentication & User API

- **Must** implement JWT-based authentication with proper expiry and refresh
- **Must** support multi-role authorization (parent/teacher/admin)
- **Must** provide email verification and password reset functionality

Content Processing

- **Must** handle file uploads with progress reporting via SSE
- **Must** process audio transcription using Whisper API
- **Must** generate embeddings for RAG using SentenceTransformers/Qdrant

RAG & Chat Services

- **Must** implement token streaming via SSE
- **Must** provide source attribution for responses
- **Must** handle feedback and flagging mechanisms

Device Synchronization

- **Must** generate content manifests based on child/group assignments
- **Must** enforce resource constraints (max 10 documents or 100MB)
- **Must** support chunked or streamed content delivery
- **Must** track device status and sync information

OTA Updates

- **Must** provide version checking and update manifests
- **Must** support download resumption for interrupted transfers
- **Must** include checksum validation for security

3. Backend Server (Offline Capable)

Local HTTP Server

- **Must** expose local endpoints on localhost:8080
- **Must** provide health, LLM query, RAG query, and session endpoints
- **Must** handle voice input/output routing

Local LLM Engine

- **Must** implement GGUF-based model loading
- **Must** enforce token limits and content filtering

- **Must** provide fallback when cloud LLM is unavailable

Local RAG System

- **Must** utilize locally cached documents
- **Must** implement vector search for relevant content
- **Must** inject document context into LLM prompts

Session Management

- **Must** maintain short-term memory for conversation continuity
- **Must** reset sessions on boot unless persistent memory is enabled

4. Device Client (EchoNest-Edge)

Voice Manager

- **Must** implement STT with maximum 500ms latency
- **Must** filter and route voice input to appropriate handlers
- **Must** rate-limit voice interactions to avoid overload

Sync Manager

- **Must** fetch manifests at defined intervals (default: 15 mins)
- **Must** download and cache content within resource constraints
- **Must** report sync status to backend
- **Must** validate available storage before download

OTA Update Client

- **Must** check for updates at regular intervals
- **Must** validate checksums after download
- **Must** implement safe update process with fallback mechanism

Emotion Controller

- **Must** display emotions via LED matrix with <100ms latency
- **Must** respond to emotion triggers from conversation

Cross-Component Compliance Requirements

Authentication & Security

- All components **must** use HTTPS for communication

- Device authentication **must** use secure device tokens
- All user data **must** be properly secured and not exposed in logs

Offline Functionality

- System **must** function with limited capabilities when offline
- Transition between online and offline modes **must** be seamless
- User experience **must** remain consistent regardless of connectivity

Performance Requirements

- STT processing **must** complete within 500ms
- LLM responses **must** generate within 2 seconds
- Emotion feedback **must** display within 100ms
- Frontend **must** remain responsive during streaming operations

Error Handling

- All components **must** implement appropriate retry mechanisms
- Device Client **must** handle network failures gracefully
- Backend Offline **must** provide fallback functionality when online services fail

Contract Validation Checklist

During development and testing, the following should be validated:

1. **API Conformance**
2. All endpoints match contract specifications (paths, methods, parameters)
3. Response formats and status codes match contract specifications
4. Authentication mechanisms are properly implemented
5. **Performance Validation**
6. Response times meet specified requirements
7. Resource usage stays within constraints
8. Streaming functionality works as expected
9. **Fallback Testing**
10. System degrades gracefully when components fail
11. Offline functionality works as expected

12. Recovery from failures is automatic when possible

13. **Security Validation**

14. Authentication is properly implemented
15. Sensitive data is properly protected
16. Input validation prevents injection attacks