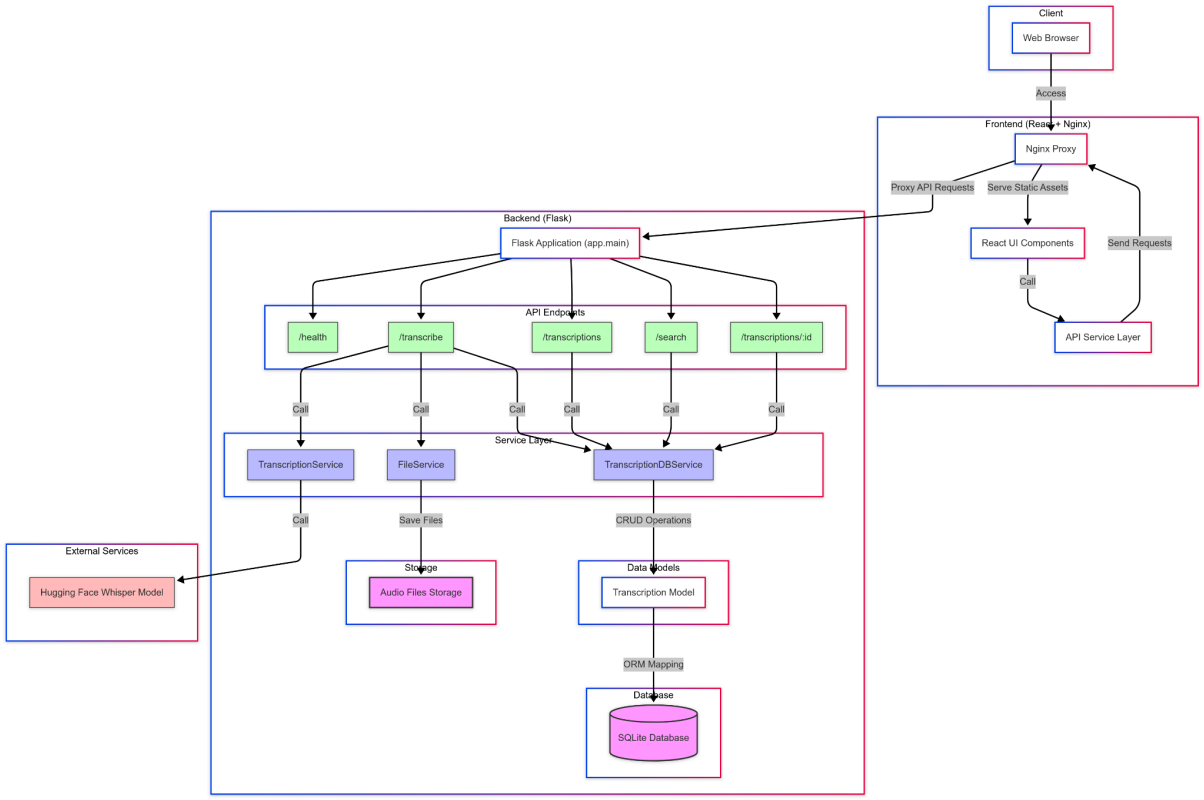
Architecture



1. Client Layer

- **Web Browser**: Users access the application through a web browser

2. Frontend Layer

- **React UI Components**: User interface components
- **API Service Layer**: JavaScript services that encapsulate API calls
- **Nginx Proxy**:
    - Serves static assets
    - Proxies API requests to the backend

3. Backend Layer

- **Flask Application**: Core application entry point
- **API Endpoints**:
    - /health: Health check endpoint
    - /transcribe: Process audio transcription
    - /transcriptions: Retrieve all transcriptions
    - /search: Search transcriptions
    - /transcriptions/:id: Get specific transcription

- **Service Layer**:
  - TranscriptionService:
    - Uses Whisper model for audio transcription
    - Implemented as a singleton
  - FileService:
    - Handles file uploads and storage
    - Generates unique filenames
  - TranscriptionDBService:
    - Encapsulates all database operations
    - Provides CRUD functionality
- **Data Models**:
  - Transcription: Transcription record model

4. Storage Layer

- **SQLite Database**:
  - Persists transcription records
  - Preserved through Docker volumes
- **Audio Files Storage**:
  - Stores uploaded audio files
  - Preserved through Docker volumes

5. External Services

- **Whisper Model**:
  - Provided by Hugging Face Transformers
  - Performs speech recognition

6. Data Flow

1. User uploads audio file
2. Request passes through Nginx proxy to Flask backend
3. FileService saves the audio file
4. TranscriptionService calls Whisper model for transcription
5. TranscriptionDBService saves results to database
6. Results are returned to frontend and displayed to user

7. Storage Volumes

- **sqlite-data**: Persists database
- **uploads-data**: Persists uploaded audio files