Simple Knowledge Base (simple-kb) 技術および機能アーキテクチャ

1. プロジェクト概要

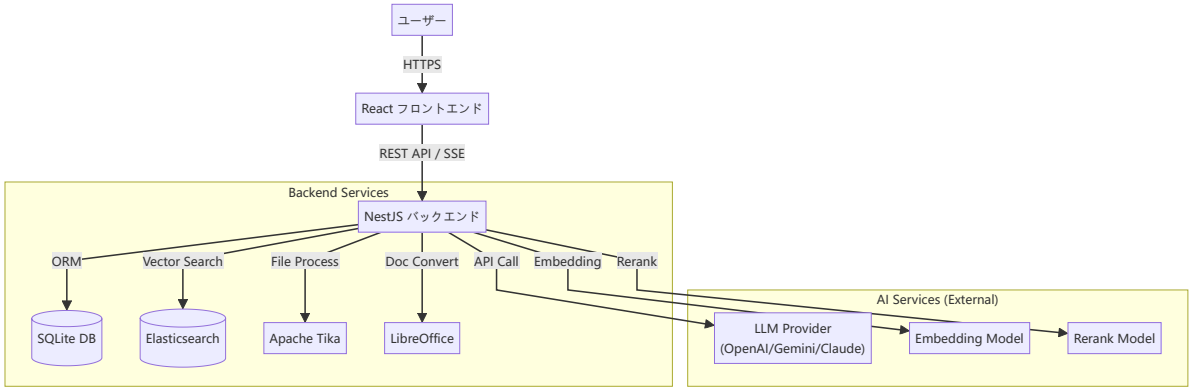
Simple Knowledge Base (simple-kb) は、React と NestJS をベースにしたフルスタックのRAG（検索拡張生成）システムです。ユーザーは多様な形式のドキュメントをアップロードし、カスタム設定でインデックス化を行い、大規模言語モデル（LLM）を用いてナレッジベースに基づいた高度な問答を行うことができます。

最近のアップデートでは、Google NotebookLM に触発された「ナレッジグループ（Notebooks）」機能や「ポッドキャスト生成」機能が追加され、単なる検索システムを超えた学習・分析プラットフォームへと進化しています。

---

2. 技術アーキテクチャ

以下は、システムの全体的な技術アーキテクチャを示す図です。



2.1 フロントエンド (Frontend)

モダンなReactエコシステムを採用し、高速でインタラクティブなUIを実現しています。

- **フレームワーク**: React 19 + Vite
  - 最新のReact機能（Hooks, Context API）を活用。
  - Viteによる高速な開発サーバーとビルド。
- **言語**: TypeScript
  - 型安全性による堅牢なコードベース。
- **スタイリング**: Tailwind CSS
  - ユーティリティファーストCSSによる迅速なUI構築。
- **UIコンポーネント**: Lucide React (アイコン)
- **状態管理**: React Context + Hooks
  - `AuthContext`, `LanguageContext` などでグローバル状態を管理。
- **通信**: Axios + Server-Sent Events (SSE)
  - RESTful APIとの通信およびAI生成テキストの流式表示（ストリーミング）。

2.2 バックエンド (Backend)

スケーラブルでモジュール化されたNode.jsアプリケーションです。

- **フレームワーク**: NestJS
  - Angularに影響を受けたモジュラーアーキテクチャ。
  - TypeScriptによる完全な型サポート。
- **AIオーケストレーション**: LangChain.js
  - LLM、エンベディング、ベクターストアの統合管理。
- **データベース**:
  - **SQLite**: ユーザー情報、設定、ファイルメタデータなどのリレーショナルデータ。
  - **Elasticsearch**: ドキュメントのベクトル埋め込み（Embedding）と全文検索インデックス。ベクトル次元数の自動検出とインデックス再構築に対応。
- **認証**: Passport.js + JWT
  - セキュアなステートレス認証。

2.3 インフラ・ファイル処理

- **ファイル解析**:
  - **Apache Tika**: 「高速モード」でのテキスト抽出。
  - **Vision Pipeline**: 「精密モード」での画像・レイアウト解析（PDF -> 画像 -> Vision Model）。
- **ドキュメント変換**: LibreOffice
  - Office文書（Word, PPTなど）をPDFに変換して処理。
- **音声生成**: Edge-TTS (または類似サービス)
  - ポッドキャスト生成機能における音声合成。

---

3. 機能アーキテクチャ

システムは以下の主要な機能モジュールで構成されています。

3.1 ユーザー管理とセキュリティ

- **認証**: ユーザー登録、ログイン、JWTによるセッション管理。
- **データ隔離**: 各ユーザーは独自のナレッジベース、設定、チャット履歴を持ち、他ユーザーからはアクセスできません。
- **多言語UI**: 英語、中国語、日本語のインターフェース切り替えに対応。

3.2 知識管理 (Knowledge Management)

- **ファイルアップロード**:
  - ドラッグ＆ドロップによる複数ファイルアップロード。
  - 対応フォーマット: PDF, Word, Excel, PPT, TXT, MD, 画像など。
- **処理モード**:
  - **高速モード (Fast Mode)**: テキストのみを高速に抽出。コスト効率が良い。

    - **精密モード (Precise Mode)**: ページを画像化し、Visionモデルでレイアウトや図表を含めて解析。
- **インデックス設定**:
    - チャンクサイズ（Chunk Size）、オーバーラップ（Overlap）のカスタマイズ。
    - エンベディングモデルの選択（OpenAI, Geminiなど）。

3.3 ナレッジグループ (Knowledge Groups)

- **概念**: ファイルを論理的なグループ（ノートブック）にまとめる機能。
- **目的**: 特定の研究テーマやプロジェクトごとに資料を整理し、そのグループに限定したチャットが可能。
- **機能**: グループの作成、編集、削除、ファイルとの関連付け。

3.4 RAGチャットシステム

- **ハイブリッド検索**:
    - ベクトル検索（意味的類似性）とキーワード検索（完全一致）を組み合わせ、リランク（Rerank）モデルで精度を向上。
- **コンテキスト認識**: ユーザーの質問履歴や現在選択されているナレッジグループを考慮。
- **流式回答 (Streaming Generation)**: AIの思考過程と回答をリアルタイムで表示。
- **引用表示**: 回答の根拠となったドキュメントのソースと該当箇所を提示。

3.5 ポッドキャスト生成 (Podcasts)

- **概要**: ナレッジグループ内の資料に基づき、AIホストとゲストによる音声対話を生成。
- **グローバル生成**: 全てのナレッジ、または特定のグループを指定してポッドキャストを作成。
- **機能**: トピック指定、スクリプト生成（トランスクリプト）、音声再生。

3.6 ビジョンモデルによる高度なドキュメント処理 (Advanced Visual Processing)

- **PPT/PDFの視覚的解析**: 従来のテキスト抽出では失われがちなPowerPointやPDFのレイアウト情報、図表、グラフを保持。
- **Vision Pipeline**:
    - **ページ画像化**: ドキュメントの各ページを高解像度画像に変換。
    - **マルチモーダル解析**: GPT-4oやClaude 3.5 Sonnetなどのビジョン対応モデルを使用し、画像内のテキストと視覚要素を統合して理解・説明。
    - **構造化データ化**: 複雑なスライドや帳票も、人間が見たままの文脈でインデックス化。

3.7 インタラクティブなノート作成 (Screenshot & Notes)

- **領域選択とOCR**: PDFプレビュー画面で任意の領域をマウスで矩形選択。
- **自動テキスト抽出**: 選択範囲の画像からOCR（光学文字認識）でテキストを即座に抽出。
- **ノート保存**: 抽出したテキストとキャプチャ画像をセットで「ノート」として保存し、後から参照や引用が可能。

3.8 システム全体設定 (System Configuration)

- **一元管理ドロワー**: 画面右上の設定アイコンから、システム全体の動作を一括設定。
- **柔軟なモデル切り替え**:
  - **LLM**: チャットや推論に使用するメインモデル。
  - **Embedding**: 検索精度を左右するベクトル化モデル。
  - **Vision**: 精密モードで使用する画像解析モデル。
- **即時反映**: 設定変更はシステム全体に即座に適用され、再起動なしで異なるモデルの挙動をテスト可能。

3.9 モデル管理 (Model Management)

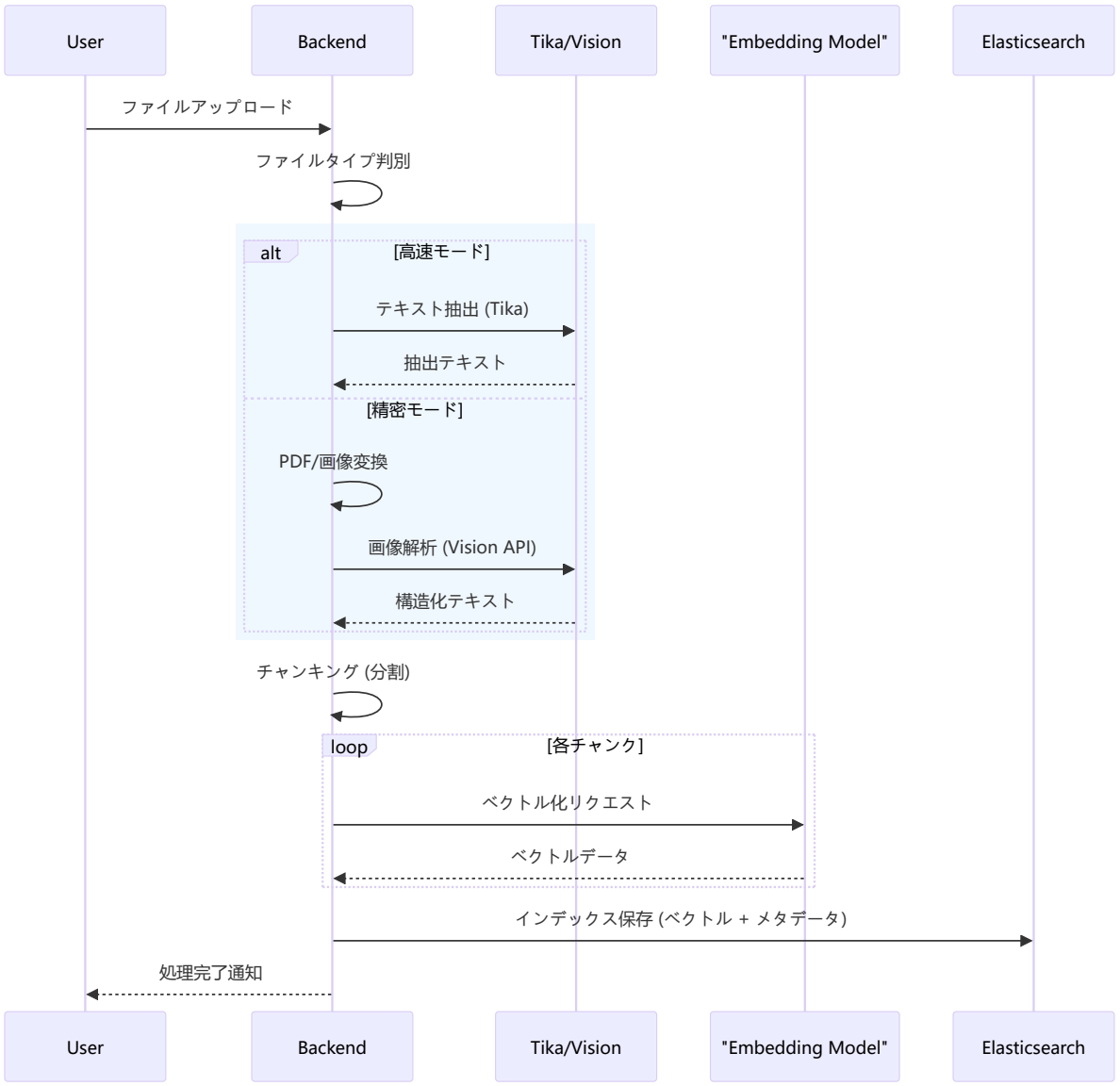
- **BYOK (Bring Your Own Key)**: ユーザー自身のAPIキーを設定可能。
- **一元管理**:「システム構成（System Settings）」ドロワーから、グローバルなモデル設定（LLM, Embedding, Rerank, Vision）を一括管理。
- **マルチプロバイダー**: OpenAI, Google Gemini, Anthropic (Claude), Ollama などのモデル設定をサポート。
- **カスタム設定**: Temperature, Top-K, Max Tokens などの推論パラメータを調整可能。

---

4. データフロー

4.1 ドキュメント取り込みフロー

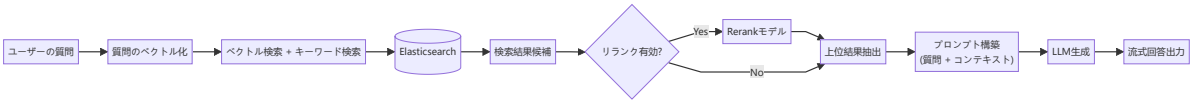
ドキュメントがアップロードされてから検索可能になるまでの処理フローです。



1. **アップロード**: ユーザーがファイルを送信。
2. **前処理**: ファイルタイプに応じた変換（例: docx -> pdf）。
3. **解析**:
    - (高速モード) Tikaでテキスト抽出。
    - (精密モード) PDFを画像化 -> Vision APIで解析。
4. **チャンキング**: 設定されたルールでテキストを分割。
5. **埋め込み**: Embedding APIでベクトル化。
6. **保存**: ベクトルとメタデータをElasticsearchに保存。

4.2 RAG検索・生成フロー

ユーザーの質問から回答生成までのRAGプロセスフローです。



1. **クエリ受信**: ユーザーの質問を受け取る。
2. **検索**: 質問をベクトル化し、Elasticsearchで類似チャンクを検索（+キーワード検索）。

3. **リランク (Optional)**: 検索結果をRerankモデルで再評価し、関連度順に並べ替え。
4. **プロンプト構築**: 上位のチャンクをコンテキストとしてシステムプロンプトに組み込む。
5. **生成**: LLMにプロンプトを送信し、回答を生成。
6. **レスポンス**: 回答と参照ソースをフロントエンドにストリーミング送信。