

Web アプリケーション仕様書

千葉工業大学 情報工学科 1 年

学籍番号 25G1006

氏名 安部 孔明

2025 年 12 月 28 日

目次

1	ソースコードの管理場所	2
2	はじめに	2
3	利用者向けマニュアル	3
3.1	システム概要とトップページ	3
3.2	各機能の操作	3
4	管理者向けマニュアル	5
4.1	システム要件	5
4.2	起動手順	5
4.3	データの仕様（注意点）	5
5	開発者向け仕様書	6
5.1	共通仕様	6
5.2	ディレクトリ構成	6
5.3	機能別 API および画面遷移仕様	7
5.4	実装の工夫点（テンプレートの共通化）	11

1 ソースコードの管理場所

本課題にて開発したアプリケーションのソースコードは、以下の GitHub リポジトリにて公開・管理している。

https://github.com/Abekomei/webpro_06

2 はじめに

本レポートは、Web プログラミングの最終課題として開発した「統合データ管理システム」の仕様書である。

本システムは、日常生活における異なる属性のデータ（ゲームのキャラ評価、カラオケの楽曲、サブスクリプション契約）を、統一された Web インターフェース上で一元管理することを目的としている。

本ドキュメントは、以下の 3 部構成で記述する。

1. 利用者向けマニュアル
2. 管理者向けマニュアル
3. 開発者向け仕様書

3 利用者向けマニュアル

3.1 システム概要とトップページ

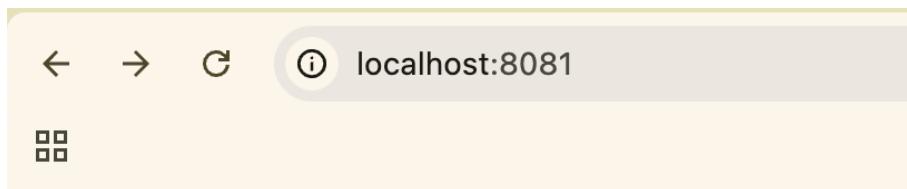
本システムは、トップページ（メニュー画面）から各アプリケーションへアクセスし、データの閲覧・追加・削除を行うことができる。

3.1.1 アクセス方法

Web ブラウザ（Google Chrome 等）より以下の URL にアクセスする。

`http://localhost:8081/`

アクセスすると以下のメニュー画面が表示され、目的の管理アプリを選択できる。



統合データ管理システム

管理したいデータを選択してください：

- [APEX キャラTier表](#)
- [カラオケ管理リスト](#)
- [サブスク管理リスト](#)

図1 トップページ（メニュー画面）

3.2 各機能の操作

ここでは「Apex Legends キャラ管理」を例に操作方法を説明する（他アプリも操作は共通である）。

3.2.1 一覧画面

各アプリのメイン画面では、登録データが表形式で表示される。

- **詳細:** データの全項目を確認できる詳細画面へ移動する。
- **編集:** 登録済みのデータを修正する画面へ移動する。

- **削除:** データをリストから削除する（確認画面なしで即時実行される）。
- **+新規追加:** 新しいデータを登録するフォームへ移動する。

The screenshot shows a web browser window with the URL `localhost:8081/apex`. The title of the page is "APEX キャラTier表" (APEX Character Tier List). Below the title is a button labeled "+ 新規追加" (New Addition). A table displays two characters: Raze (スカーミッシャー) and Gibraltar (サポート), both in Tier S. Each row has a "操作" (Operation) column with three links: 詳細 (Details), 編集 (Edit), and 削除 (Delete).

ID	キャラ名	クラス	Tier	操作
1	レイス	スカーミッシャー	S	詳細 編集 削除
2	ジブラルタル	サポート	A	詳細 編集 削除

図 2 一覧画面の例 (Apex Legends)

3.2.2 詳細画面

一覧画面で「詳細」をクリックすると、そのデータに関するすべての情報（メモなど）が表示される。

The screenshot shows a web browser window with the URL `localhost:8081/apex/detail/1`. The title of the page is "詳細データ" (Detailed Data). Below the title, there is a bulleted list of character details: ID: 1, キャラ名: レイス, クラス: スカーミッシャー, Tier: S, and 評価メモ: ポータルが強い. At the bottom of the page is a link labeled "一覧に戻る" (Return to List).

- ID: 1
- キャラ名: レイス
- クラス: スカーミッシャー
- Tier: S
- 評価メモ: ポータルが強い

[一覧に戻る](#)

図 3 詳細画面の例 (Apex Legends)

4 管理者向けマニュアル

4.1 システム要件

- **OS:** Windows, macOS, Linux
- **実行環境:** Node.js (v14.x 以上推奨)
- **依存ライブラリ:** Express, EJS

4.2 起動手順

ターミナルでプロジェクトディレクトリに移動し、以下のコマンドを実行する。

Listing 1 サーバー起動コマンド

```
1 $ node apph.js
```

Server started on port 8081! と表示されれば起動完了である。

4.3 データの仕様（注意点）

本システムは学習用プロトタイプのため、データベースを使用せずメモリ上（変数）でデータを管理している。サーバーを再起動すると、追加・削除されたデータは初期状態に戻る仕様となっている。

5 開発者向け仕様書

本章では、システムの内部構造、API 仕様、および画面遷移について記述する。

5.1 共通仕様

全システム共通で、以下の技術スタックを採用している。

- フレームワーク: Express
- テンプレートエンジン: EJS
- データ管理: 配列変数（インメモリ）

5.2 ディレクトリ構成

```
webpro_06/
├── app.js          (コントローラ兼モデル)
├── views/
│   ├── top.ejs      (トップページ)
│   ├── common_list.ejs (共通一覧テンプレート)
│   └── common_detail.ejs (共通詳細テンプレート)
└── public/
    ├── apex_new.html
    ├── karaoke_new.html
    └── subscription_new.html
└── package.json
```

5.3 機能別 API および画面遷移仕様

各アプリケーションは共通の画面遷移ロジックに基づいて設計されている。詳細な遷移フローを以下の図に示す。

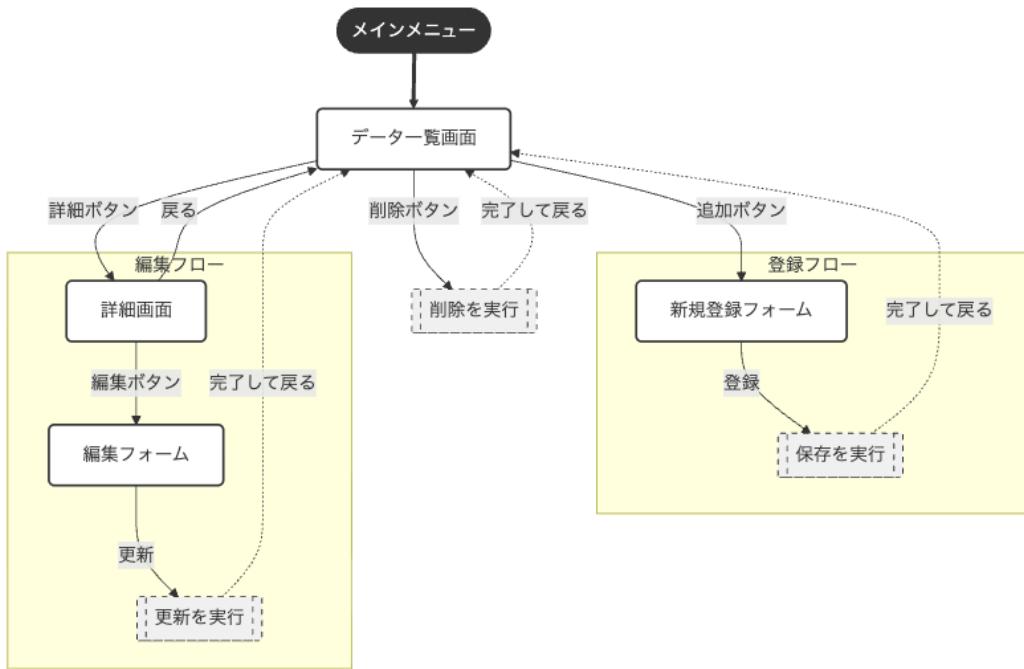


図 4 システム共通の画面遷移図

以下の表に、各アプリケーションのデータ構造および API 仕様を示す。

5.3.1 トップページ (メニュー)

システム全体の入り口となるランディングページである。

表 1 トップページ API 仕様

機能	メソッド	パス	処理内容
メニュー表示	GET	/	top.ejs をレンダリングし、リンクを表示

5.3.2 Apex Legends キャラ管理

ゲームキャラクターの Tier (強さランク) を管理する機能。

データ構造 (変数名: apexData)

表 2 Apex データ構造

プロパティ名	型	説明
id	Number	データ識別用 ID
name	String	キャラクター名
tier	String	強さのランク (S, A など)
memo	String	詳細メモ

API 仕様

表 3 Apex API 仕様

機能	メソッド	パス	処理内容
一覧表示	GET	/apex	データを一覧表示 (common_list.ejs)
詳細表示	GET	/apex/detail/:id	指定 ID の詳細を表示 (common_detail.ejs)
新規登録画面	GET	/apex/create	入力フォーム (apex_new.html) ヘリダイレクト
データ追加	POST	/apex/add	フォーム値を受け取り配列に追加し一覧へ遷移
編集画面	GET	/apex/edit/:id	編集フォームを表示
データ更新	POST	/apex/update/:id	データを更新し一覧へ遷移
データ削除	GET	/apex/delete/:id	指定 ID を配列から削除し一覧へ遷移

5.3.3 カラオケ楽曲管理

練習したい曲やキー設定を管理する機能。

The screenshot shows a browser window with the URL `localhost:8081/karaoke`. The title bar says "カラオケ管理リスト". Below the title, there is a button labeled "+ 新規追加". The main content is a table with the following data:

ID	曲名	アーティスト	キー	操作
1	怪獣の花唄	Vaundy	原曲	詳細 編集 削除
2	マリーゴールド	あいみょん	+2	詳細 編集 削除

図 5 カラオケ 一覧

The screenshot shows a browser window with the URL `localhost:8081/karaoke/detail/1`. The title bar says "詳細データ". The main content lists the following details for the song:

- ID: 1
- 曲名: 怪獣の花唄
- アーティスト: Vaundy
- キー: 原曲
- メモ: サビが高い

Below the list is a link "[一覧に戻る](#)".

図 6 カラオケ 詳細

データ構造 (変数名: karaokeData)

表4 カラオケ データ構造

プロパティ名	型	説明
id	Number	データ識別用 ID
song	String	曲名
artist	String	アーティスト名
key	String	キー設定 (原曲, +2 など)
memo	String	メモ (サビが高いなど)

API 仕様

表5 カラオケ API 仕様

機能	メソッド	パス	処理内容
一覧表示	GET	/karaoke	データを一覧表示 (common_list.ejs)
詳細表示	GET	/karaoke/detail/:id	指定 ID の詳細を表示 (common_detail.ejs)
新規登録画面	GET	/karaoke/create	入力フォーム (karaoke_new.html) ヘリダイレクト
データ追加	POST	/karaoke/add	フォーム値を受け取り配列に追加し一覧へ遷移
編集画面	GET	/karaoke/edit/:id	編集フォームを表示
データ更新	POST	/karaoke/update/:id	データを更新し一覧へ遷移
データ削除	GET	/karaoke/delete/:id	指定 ID を配列から削除し一覧へ遷移

5.3.4 サブスクリプション管理

契約中のサービスと月額料金を管理する機能。一覧表示時に合計金額を計算するロジックを含む。

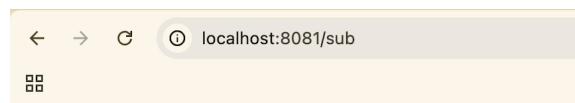


図7 サブスク 一覧

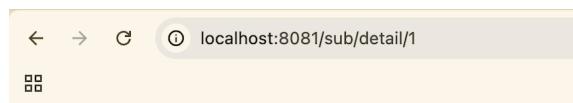


図8 サブスク 詳細

+ 新規追加

ID	サービス名	月額(円)	更新日	操作
1	Netflix	1980	毎月25日	詳細 編集 削除
2	Amazon Prime	600	毎月20日	詳細 編集 削除

[一覧に戻る](#)

図7 サブスク 一覧

図8 サブスク 詳細

データ構造 (変数名: subData)

表6 サブスク データ構造

プロパティ名	型	説明
id	Number	データ識別用 ID
service	String	サービス名
price	Number	月額料金 (円)
renewal	String	更新日
memo	String	メモ

API 仕様

表7 サブスク API 仕様

機能	メソッド	パス	処理内容
一覧表示	GET	/sub	合計金額を計算し一覧表示 (common_list.ejs)
詳細表示	GET	/sub/detail/:id	指定 ID の詳細を表示 (common_detail.ejs)
新規登録画面	GET	/sub/create	入力フォーム (subscription_new.html) へリダイレクト
データ追加	POST	/sub/add	フォーム値を受け取り配列に追加し一覧へ遷移
編集画面	GET	/sub/edit/:id	編集フォームを表示
データ更新	POST	/sub/update/:id	データを更新し一覧へ遷移
データ削除	GET	/sub/delete/:id	指定 ID を配列から削除し一覧へ遷移

5.4 実装の工夫点（テンプレートの共通化）

本システムでは、開発効率と保守性を高めるため、3つの異なるアプリに対し**共通の EJS テンプレート**を使用している。

5.4.1 Config オブジェクトによる制御

apph.js 側で、各アプリの表示項目（カラム名）をオブジェクトとして定義している。

Listing 2 設定オブジェクトの例（Apex 用）

```
1 const APEX_CONFIG = {  
2     title: "APEX_キャラ表 Tier",  
3     baseUrl: "/apex",  
4     listColumns: [  
5         { label: "ID", key: "id" },  
6         { label: "キャラ名", key: "name" },  
7         { label: "Tier", key: "tier" }  
8     ],  
9     // ...  
10};
```

5.4.2 共通テンプレートの利用

定義した Config をテンプレートに渡すことで、単一のファイルで多様なデータを表示可能とした。

Listing 3 common.list.ejs (抜粋)

```
1 <% for (let col of columns) { %>  
2     <th><%= col.label %></th>  
3 <% } %>
```

これにより、新しいアプリを追加する際も HTML を作成する必要がなく、設定の追加のみで実装が可能となっている。