2025 年度後期 Web プログラミング期末課題仕様書

25G1064 佐土駿

2025 年 12 月 27 日

1　はじめに

本稿は，作成した 3 つの web アプリケーションに関連する仕様書を，「開発者向け仕様書」，「管理者向け仕様書」，「利用者向け仕様書」の 3 つに分けて記述する．ソースコードは GitHub 上に公開しており，以下の URL からアクセスできる．

`https://github.com/nazonomikan/webpro_06.git`

2　開発者向け仕様書

本節では，開発者向けに各 web アプリケーションのシステム構成，使用技術，ディレクトリ構成，主要なソースコードについて説明する．

2.1　app1:kimatu_saki

2.1.1　システム構成

本アプリケーションは，Node.js，Express を用いたサーバアプリケーションであり，テンプレートに EJS，静的資産に'public/' を使用する．サーバ実装は'app5.js' にまとめており，ビューは'views/'，静的ファイルは'public/' に配置している．サーバ起動は 'npm install'で依存をインストール後，'node app5.js'で行うことでできる．

2.1.2　目的

本システムの目的は，「プロジェクトセカイ カラフルステージ！ feat. 初音ミク」に登場するキャラクターのうちの一人である「天馬咲希」の所持カードを閲覧・管理し，さらに，実際のゲーム内での編成の参考にすることができるようにすることである．

2.1.3　機能詳細

本システムの機能の詳細について説明する．本システムでは，「プロジェクトセカイ カラフルステージ！ feat. 初音ミク」に登場するキャラクターのうちの一人である「天馬咲希」の所持カードの一覧表示，詳細表示，追加，編集，削除機能を使用できる．さらに，編成画面にて，所持しているカードの中から最大 5 枚を選択し，編成の総合力を計算することができる．

2.1.4　データ構造

app1 では，カードデータを’app5.js’ 内のメモリ上の配列 ‘saki‘で管理している．代表的なフィールドは以下の表 1 の通りである．

表 1: ‘saki‘配列の代表フィールド

項目名	詳細説明
id	カードの識別子，内部管理をするために使用している（数値）
name	カード名（文字列）
type	カードのタイプ（キュート/クール/ミステリアス/ハッピー/ピュア）
rarity	レアリティ（数値）（1-4）
skill_name	スキル名（文字列）
skill_level	スキルレベル（数値）（1-4）
rank	カードのマスターランク（数値）（0-5）
total_status	カードの初期総合力（数値）

2.1.5　ページ構造と HTTP メソッドとリソース名

本データのページ構造，また，対応するメソッドについて図 1 を用いて説明する．

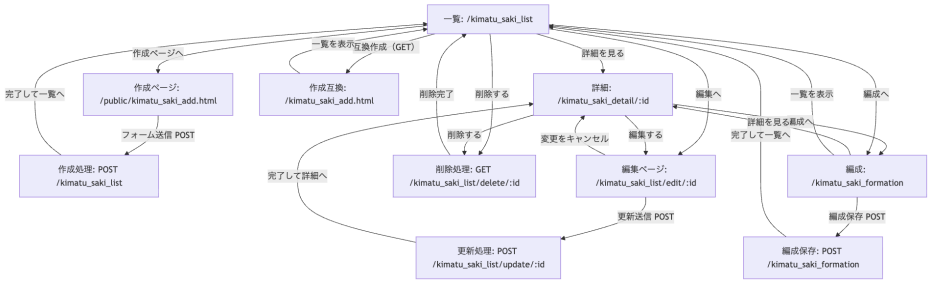

図 1: app1:kimatu_saki のページ構造

本システムの起点となるホームページは`/kimatu_saki_list` であり，ここから各機能へ遷移する．一覧表示からは，詳細表示，編集，削除，追加フォーム，編成画面へ遷移できる．詳細表示からは，編集，削除，編成画面へ遷移できる．編集ページからは，更新処理を経て，編集内容を確認するために詳細表示へ遷移する．削除，追加フォームからは，それぞれ削除処理，追加処理を経て

一覧表示へ遷移する．編成画面からは，選択したカードの編成を保存し，一覧表示または詳細表示へ遷移できる．また，使用しているファイル名と詳細について表 2 に示す．

表 2: app1:kimatu_saki の主要ファイル

ファイル名	説明
app5.js	サーバアプリケーション本体．saki 配列を管理し，ルーティングとロジックを実装している．
views/kimatu_saki_list.ejs	一覧表示ページのテンプレート．カード名，タイプが表示され，データ編集・削除ページ，編成ページへ行くことができる．
views/kimatu_saki_detail.ejs	詳細表示ページのテンプレート．一覧表示から選択されたカードのレアリティなどの詳細を表示する．また，データ編集，削除ページへ行くことができる．
views/kimatu_saki_edit.ejs	編集ページのテンプレート．id に対応するカードの編集フォームを表示する．
views/kimatu_saki_formation.ejs	編成画面のテンプレート．編成中のカードを選択し，総合力を計算する機能を提供する．
public/kimatu_saki_add.html	追加フォームの静的 HTML．カード追加用のフォームを提供する．
public/kimatu_saki.css	カスタム CSS ファイル．本システムにおける見た目の改善のために独自のスタイルを作成し，レスポンシブ対応の簡易実装.

また，主要なエンドポイントについて表 3 に示す．

表 3: app1:kimatu_saki の主要エンドポイント

HTTP メソッド	リソース名	説明
GET	/kimatu_saki_list	一覧表示
GET	/kimatu_saki_detail/:id	詳細表示
POST	/kimatu_saki_list	追加処理
GET	/kimatu_saki_list/edit/:id	編集ページ表示
POST	/kimatu_saki_list/update/:id	更新処理
GET	/kimatu_saki_list/delete/:id	削除処理
GET/POST	/kimatu_saki_formation	編成画面表示/保存

2.1.6　リソース名ごとの機能の詳細

各エンドポイントの機能の詳細について説明する．主なエンドポイントの機能は，表 4 の通りである．

表 4: app1:kimatu_saki のリソース名ごとの機能詳細

リソース名	機能詳細
/kimatu_saki_list	saki 配列を EJS に渡して一覧を描画する．
/kimatu_saki_detail/:id	指定 id のオブジェクトを検索して詳細を表示.
/kimatu_saki_list (POST)	フォームから送られた値を受け取り，メモリ配列を更新する（DB は未使用）.
/kimatu_saki_list/edit/:id	指定 id のオブジェクトを検索して編集フォームを表示.
/kimatu_saki_list/update/:id	フォームから送られた値を受け取り，メモリ配列を更新する（DB は未使用）.
/kimatu_saki_list/delete/:id	指定 id のオブジェクトを検索してメモリ配列から削除する（DB は未使用）.
/kimatu_saki_formation	編成中のキャラの total_status に対してランク補正（rank * 600）とタイプボーナス，および定数の増幅（コード中に定義）を適用して計算し，合計値を表示する．

2.1.7　授業外使用技術について

本節では，授業で扱わなかったが本システムで採用した技術とその採用理由について説明する．

総合力計算方法　編成画面で選択された各カードの `total_status` に対して，カードの `rank` に基づくランク補正（コード中では実際の増加量を参考に `rank * 600` としている），タイプごとのボーナス，および定数による増幅を順次適用して合計値を算出する手法を採用した．理由は，実機に近い補正ルールを再現することでユーザーが現実的な編成評価を行えるようにするためである．計算は主にサーバ側で一貫して行い，表示はクライアント側で受け取ってレンダリングすることで，サーバの整合性とクライアントの応答性を両立させている．

編成画面　編成画面は最大 5 枚までのカードをチェックボックスで選択し，選択状態をクライアント側で一時保持してからサーバへ送信・保存する仕組みとしている．編成の保存，総合力計算のトリガーはフォーム送信／非同期通信いずれにも対応させており，ユーザーが一覧画面を離れずに操作を完了できるようにした．採用理由は，詳細画面に遷移して戻る従来のフローだとスクロール位置や選択状態が失われる可能性があるためであり，選択の即時反映と一覧の継続的な参照を可能にすることで操作性を向上させるためである．

CSS(見た目・UI)　レイアウトには’display:flex’ を多用し，カード表示はカード型の横並び（フレックスコンテナ）で実装した．また，カードのホバー時に’transition’ を用いた軽い浮き上がりアニメーションの付与を行い，選択中のカードや編成の入力欄は’position:fixed’ と’z-index’ による

オーバーレイ表示で編成用 UI を重ねて表示する実装を行っている．理由は，カードをカードらしく視覚的に整理することで目的のカードが見つけやすくなること，固定オーバーレイにより編成中でも一覧の位置や詳細データを保持できること，および小さなアニメーションで操作のフィードバックを与えることで視認性と利便性が向上すると判断したためである．

2.2 app2:kimatu_gakumasu

2.2.1 システム構成

構成は app1 と同様に Node.js / Express / EJS を採用している。カード（学マス）管理用の CRUD をメモリ配列で実装している。

2.2.2 目的

学マスカードの一覧管理（作成・編集・削除・詳細表示）を行うための簡易管理ツール。

2.2.3 機能詳細

主な機能:

- 一覧 `/kimatu_gakumasu_list`（`views/kimatu_gakumasu_list.ejs`）
- 詳細 `/kimatu_gakumasu_detail/:id`（`views/kimatu_gakumasu_detail.ejs`）
- 追加フォーム `public/kimatu_gakumasu_add.html` と POST `/kimatu_gakumasu_list`
- 編集 `/kimatu_gakumasu_list/edit/:id` と更新 POST `/kimatu_gakumasu_list/update/:id`
- 削除 `/kimatu_gakumasu_list/delete/:id`

2.2.4 データ構造

メモリ上の配列 ‘gakumasu‘。代表フィールド:

- ‘id‘, ‘name‘, ‘cost‘, ‘cost_type‘, ‘rarity‘, ‘unlock_level‘, ‘effect‘

2.2.5 ページ構造

代表ファイル:

- `views/kimatu_gakumasu_list.ejs`, `views/kimatu_gakumasu_detail.ejs`, `views/kimatu_gakumasu_edit.ejs`
- 追加フォーム: `public/kimatu_gakumasu_add.html`

2.2.6 HTTP メソッドとリソース名

主要エンドポイントは app1 と同様な REST 風の CRUD（GET 一覧/詳細，POST 作成/更新，GET 削除）を採用している。

2.2.7 リソース名ごとの機能の詳細

各エンドポイントはメモリ配列操作を行い，変更後は一覧表示にリダイレクトまたはレンダリングする簡易実装である。

2.2.8 ページ遷移

2.2.9 授業外使用技術について

2.3 app3:kimatu_compass

2.3.1 システム構成

構成は app1/app2 と同様。コンパスカード（スキルカード）を管理する CRUD 機能を提供する。

2.3.2 目的

コンパスカード（攻撃/防御/HP/CT 等を持つカード）の登録・編集・表示を行う。

2.3.3 機能詳細

主な機能:

- 一覧 `/kimatu_compass_list`（`views/kimatu_compass_list.ejs`）
- 詳細 `/kimatu_compass_detail/:id`（`views/kimatu_compass_detail.ejs`）
- 追加フォーム `public/kimatu_compass_add.html` と POST `/kimatu_compass_list`
- 編集 `/kimatu_compass_list/edit/:id` と更新 POST `/kimatu_compass_list/update/:id`
- 削除 `/compass_list/delete/:id`（実装上のパス注意：一部ルート名が統一されていない）

2.3.4 データ構造

配列 ‘compass‘。代表フィールド:

- ‘id‘, ‘name‘, ‘type‘, ‘attack‘, ‘defence‘, ‘hp‘, ‘cool_time‘, ‘damage_type‘, ‘damage‘, ‘damage_count‘, ‘effect‘

2.3.5 ページ構造

代表ファイル: `views/kimatu_compass_list.ejs`, `views/kimatu_compass_detail.ejs`, `views/kimatu_compass_edit.ejs`, `public/kimatu_compass_add.html`。

2.3.6　HTTP メソッドとリソース名

2.3.7　リソース名ごとの機能の詳細

2.3.8　ページ遷移

2.3.9　授業外使用技術について

3　管理者向け仕様書

本節では，管理者向けに各 web アプリケーションのインストールから起動までの手順，不具合情報について説明する．また，ここでは，代表である app1:kimatu_saki について説明する．app2,app3 についても同様の手順でインストール，起動が可能であるが，一部機能は実施していないため，注意が必要である．しかし，app2 と app3 は双方ともにシステム構成や使用技術は同じである．

3.1　インストールから起動までの手順

3.1.1　インストール方法

前提: Node.js がインストールされていること。リポジトリのルートで依存をインストールする:

```
npm install
```

（‘package.json‘ が存在する場合は ‘npm install‘ で必要パッケージが入る。なければ ‘npm install express ejs‘ を手動で行う）

3.1.2　起動方法

サーバ起動コマンド:

```
PORT=8080 node app5.js
```

または単に `node app5.js`（`PORT` 環境変数が未指定の場合は 8080 が使われる）。ブラウザで `http://localhost:8080/kimatu_home` を開く。

3.1.3　起動できない場合

よくある原因:

- Node.js が未インストール
- ポートが既に使用中（別プロセスが 8080 を使用）
- 必要モジュールが未インストール（‘express‘ など）

対処法: ‘node -v‘ を確認し，‘npm install‘ を再実行。ポート競合は別ポートを指定して起動するか既存プロセスを停止する。

3.2 終了方法

実行中のターミナルで ‘Ctrl+C‘ を押す。

3.3 分かっている不具合

既知の欠点／注意点:

- データはメモリ上に保持しており，サーバ再起動で消える（永続化なし）。
- 入力値のバリデーションが最小限であり，不正データの送信を想定していない。
- ルート名に若干の不整合（例: コンパス削除のパス名が `/compass_list/delete/:id` で，その他が `kimatu_compass_list` ベースになっている箇所がある）。
- 削除処理が GET で行われているため，意図しないアクセスでデータが消える可能性がある。

4 利用者向け仕様書

本節では，利用者向けに各 web アプリケーションの概要，使用できる機能について説明する．また，ここでは，代表である app1:kimatu_saki について説明する．app2,app3 についても同様の手順で利用が可能であるが，一部機能は実施していないため，注意が必要である．

4.1 システム概要

本システムはローカルで起動する Node/Express サーバ上で動作する Web アプリ群で，ブラウザから一覧・詳細・追加・編集・削除操作を行える。静的ファイルは ‘public/‘、テンプレートは ‘views/‘ に格納される。

4.2 目的

学習目的のための CRUD 操作・画面設計練習。複数のデータ型（キャラクター／カード）を管理し，簡易的な計算（編成の総合力）を行うことを想定している。

4.3 使用できる機能

利用者がブラウザから行える操作:

- 各種一覧表示（キャラ，学マス，コンパス）
- 詳細表示と編集
- 新規追加（フォーム送信）
- 削除（一覧から）
- 編成画面での選択・保存・合計値計算（`kimatu_saki`）

4.4 操作方法

基本的な操作フロー:

- サーバを起動して `http://localhost:8080/kimatu_home` を開く。
- 目的の一覧ページ（"1 つ目一覧"=キャラ，"2 つ目一覧"=学マス，"3 つ目一覧"=コンパス）をクリックする。
- 一覧から名前をクリックして詳細を確認，編集や削除は一覧上のリンクから行う。
- 追加は各一覧ページの「追加」ボタンからフォームを開き，必要項目を入力して送信する。
- 編成画面では複数キャラをチェックして「保存」すると編成がサーバ側に保持され，補正値を反映した合計値が確認できる。

注意: データはサーバのメモリ内にあるためサーバ停止で消失します。永続化が必要な場合はデータベース導入が必要です。