web プログラミング仕様書

25G1088　津村 大樹

2025 年 12 月 28 日

1　Github 上のリポジトリの URL

`https://github.com/BIG-POG/webpro_06`

2　開発者向け仕様書

2.1　にじさんじ SEEd's メンバー一覧

2.1.1　概要

本システムでは VTuder 事務所であるにじさんじの元 SEEd's 一期生の情報を管理し，REST API の原則に従って，データの一覧，詳細表示，追加，変更，削除の 5 つの操作を提供するシステムである．

2.1.2　データ構造

データ構造は以下の表 1のようになる．

表 1　にじさんじ SEEd's メンバーのデータ構造

項目	データ型	説明
id	整数	メンバーの ID
name	文字列	メンバーの名前
year	文字列	メンバーの活動期間
species	文字列	メンバーの種族
birthday	文字列	メンバーの誕生日
fanmark	文字列	メンバーのファンマーク
fanname	文字列	メンバーのファンネーム

表 1に示すデータは `nijisanji_seeds` に保存し，メンバーの ID，名前，活動期間，種族，誕生日，ファンマーク，ファンネームを管理する．id は各メンバーを一意に識別するためのものであり，データの追加を行った場合には自動で設定される．

2.1.3　ページ遷移

本システムのページ遷移は以下の図 1のようになる．

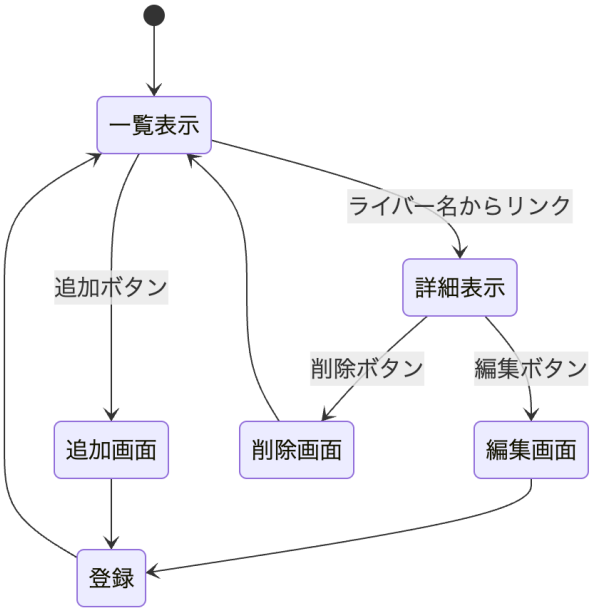

図 1　にじさんじ SEEd's メンバーページ遷移図

図 1に示すように，トップページからにじさんじ SEEd's メンバー一覧ページへ遷移し，そこから各メンバーの詳細ページへ遷移できる．詳細ページからは編集ページへ遷移でき，編集ページでデータを更新した後，一覧ページへ戻ることができる．また，詳細ページからメンバーの削除も可能であり，削除後は一覧ページへ戻る．そして，メンバー一覧ページから新規メンバー登録ページへ遷移できる．

2.1.4　リソースの説明

本システムで使用する各ページのリソースの HTTP メソッドとリソース名は以下の表 3に示す．

表 2　にじさんじ SEEd's メンバーリソース一覧

HTTP メソッド	リソース名
get	/nijisanji
get	/nijisanji/create
get	/nijisanji/:number
get	/nijisanji/delete/:number
post	/nijisanji_new
get	/nijisanji/edit/:number
post	/nijisanji/update/:number

表 3　にじさんじ SEEd's メンバーリソース一覧

表 3に示すように，各リソースは HTTP メソッドとリソース名で構成されている．get メソッドはデータの取得に使用され，post メソッドはデータの作成や更新に使用される．リソース名は各ページの URL パスを表している．/nijisanji はメンバーの一覧を表示するページであり，サーバーから nijisanji_seeds データを取得し，nijisanji.ejs テンプレートに渡して表示する．

/nijisanji/create は新規メンバー登録ページであり，登録用のフォームである nijisanji_create.html を表示し，フォームから入力されたデータを post メソッドで/nijisanji_new リソースに送信し，新しいメンバーを追加する．

/nijisanji/:number は各メンバーの詳細ページであり，指定されたメンバー ID に対応するデータを取得し，詳細表示用のテンプレートである nijisanji_detail.ejs を表示する．

/nijisanji/delete/:number は各メンバーの削除処理を行うリソースであり，指定されたメンバー ID に対応するデータを削除し，一覧ページに遷移する．

/nijisanji_new は新規メンバーの登録処理を行うリソースであり，登録ページから送信されたデータを受け取り，新しいメンバーを nijisanji_seeds に追加し，一覧ページに遷移する．

/nijisanji/edit/:number は各メンバーの編集ページであり，指定されたメンバー ID に対応するデータを取得し，編集用のテンプレートである nijisanji_edit.ejs を表示する．

/nijisanji/update/:number は各メンバーの更新処理を行うリソースであり，編集ページから送信されたデータを受け取り，対応するメンバーのデータを更新し，一覧ページに遷移する．

2.2 歴代ミスターオリンピアチャンピオン一覧

2.2.1 概要

本システムでは世界最高峰のボディビル大会であるミスターオリンピアの歴代チャンピオンの情報を管理し，REST API の原則に従って，データの一覧，詳細表示，追加，変更，削除の 5 つの操作を提供するシステムである．

2.2.2 データ構造

データ構造は以下の表 4のようになる．

表 4　歴代ミスターオリンピアチャンピオンのデータ構造

項目	データ型	説明
id	整数	チャンピオンの ID
year	文字列	チャンピオンの獲得年
name	文字列	チャンピオンの名前
from	文字列	チャンピオンの出身地
height	文字列	チャンピオンの身長
strengths	文字列	チャンピオンの長所

表 4に示すデータは OlympiaChampion に保存し，チャンピオンの ID，獲得年，名前，出身地，身長，長所を管理する．id は各チャンピオンを一意に識別するためのものであり，データの追加を行った場合には自動で設定される．

2.2.3 ページ遷移

本システムのページ遷移は以下の図 2のようになる．

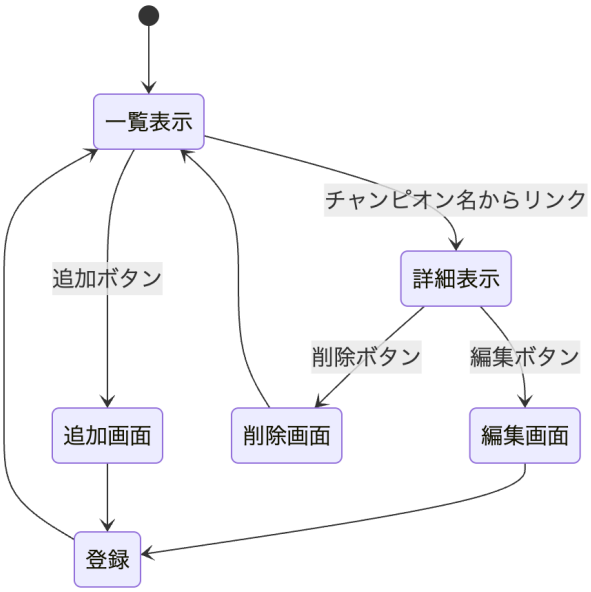

図 2　歴代ミスターオリンピアチャンピオンページ遷移図

図 2に示すように，トップページから歴代チャンピオン一覧ページへ遷移し，そこから各チャンピオンの詳細ページへ遷移できる．詳細ページからは編集ページへ遷移でき，編集ページでデータを更新した後，一覧ページへ戻ることができる．また，詳細ページからチャンピオンの削除も可能であり，削除後は一覧ページへ戻る．そして，歴代チャンピオン一覧ページから新規のチャンピオン登録ページへ遷移できる．

2.2.4　リソースの説明

本システムで使用する各ページのリソースの HTTP メソッドとリソース名は以下の表 5に示す．

表 5　歴代ミスターオリンピアチャンピオンリソース一覧

HTTP メソッド	リソース名
get	/Olympia
get	/Olympia/create
get	/Olympia/:number
get	/Olympia/delete/:number
post	/Olympia_new
get	/Olympia/edit/:number
post	/Olympia/update/:number

表 5に示すように，各リソースは HTTP メソッドとリソース名で構成されている．get メソッドはデータの取得に使用され，post メソッドはデータの作成や更新に使用される．リソース名は各ページの URL パスを表している．/Olympia は歴代チャンピオンの一覧を表示するページであり，サーバーから OlympiaChampion データを取得し，Olympia.ejs テンプレートに渡して表示する．/Olympia/create は新規のチャンピオン登録ページであり，登録用のフォームである Olympia_create.html を表示し，フォームから入力されたデータを post メソッドで/Olympia_new リソースに送信し，新しいチャンピオンを追加する．/Olympia/:number は各チャンピオンの詳細ページであり，指定された ID に対応するデータを取得し，詳細表示用のテンプレートである Olympia_detail.ejs を表示する．/Olympia/delete/:number は各チャンピオンの削除処理を行うリソースであり，指定された ID に対応するデータを削除し，一覧ページに遷移する．/Olympia_new は

新規のチャンピオンの登録処理を行うリソースであり，登録ページから送信されたデータを受け取り，新しいチャンピオンを OlympiaChampion に追加し，一覧ページに遷移する．/Olympia/edit/:number は各チャンピオンの編集ページであり，指定された ID に対応するデータを取得し，編集用のテンプレートである Olympia_edit.ejs を表示する．/Olympia/update/:number は各チャンピオンの更新処理を行うリソースであり，編集ページから送信されたデータを受け取り，対応するチャンピオンのデータを更新し，一覧ページに遷移する．

2.3 ARMORED CORE6 登場 AC 一覧

2.3.1 概要

本システムでは 2023 年にフロム・ソフトウェアから発売された ARMORED CORE6 に登場する AC の情報を管理し，REST API の原則に従って，データの一覧，詳細表示，追加，変更，削除の 5 つの操作を提供するシステムである．

2.3.2 データ構造

データ構造は以下の表 6のようになる．

項目	データ型	説明
id	整数	AC の ID
ac_name	文字列	AC の名前
name	文字列	搭乗者
rank	文字列	ランク
affiliate	文字列	所属

表 6　ARMORED CORE6 登場 AC のデータ構造

表 6に示すデータは AC に保存し，AC の ID，名前，搭乗者，ランク，所属を管理する．id は各 AC を一意に識別するためのものであり，データの追加を行った場合には自動で設定される．

2.3.3 ページ遷移

本システムのページ遷移は以下の図 3のようになる．

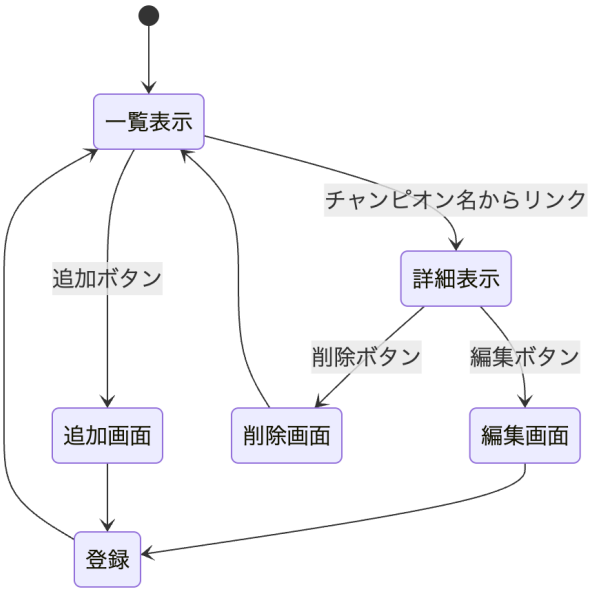

図 3　ARMORED CORE6 登場 AC ページ遷移図

図 3に示すように，トップページから登場 AC 一覧ページへ遷移し，そこから各 AC の詳細ページへ遷移できる．詳細ページからは編集ページへ遷移でき，編集ページでデータを更新した後，一覧ページへ戻ることができる．また，詳細ページから AC の削除も可能であり，削除後は一覧ページへ戻る．そして，登場 AC 一覧ページから新規の AC 登録ページへ遷移できる．

2.3.4　リソースの説明

本システムで使用する各ページのリソースの HTTP メソッドとリソース名は以下の表 7に示す．

表 7　ARMORED CORE6 登場 AC リソース一覧

HTTP メソッド	リソース名
get	/ac6
get	/ac6
get	/ac6/:number
get	/ac6/delete/:number
post	/ac6_new
get	/ac6/edit/:number
post	/ac6/update/:number

表 7に示すように，各リソースは HTTP メソッドとリソース名で構成されている．get メソッドはデータの取得に使用され，post メソッドはデータの作成や更新に使用される．リソース名は各ページの URL パスを表している．/ac6 は AC の一覧を表示するページであり，サーバーから AC データを取得し，ac6.ejs テンプレートに渡して表示する．/ac6/create は新規の AC 登録ページであり，登録用のフォームである ac6_create.html を表示し，フォームから入力されたデータを post メソッドで/ac6_new リソースに送信し，新しい AC を追加する．/ac6/:number は各 AC の詳細ページであり，指定された ID に対応するデータを取得し，詳細表示用のテンプレートである Olympia_detail.ejs を表示する．/ac6/delete/:number は各 AC の削除処理を行うリソースであり，指定された ID に対応するデータを削除し，一覧ページに遷移する．/ac6_new は新規の AC の登録処理を行うリソースであり，登録ページから送信されたデータを受け

取り，新しい AC を AC に追加し，一覧ページに遷移する．/ac6/edit/:number は各 AC の編集ページであり，指定された ID に対応するデータを取得し，編集用のテンプレートである ac6_edit.ejs を表示する．/ac6/update/:number は各 AC の更新処理を行うリソースであり，編集ページから送信されたデータを受け取り，対応する AC のデータを更新し，一覧ページに遷移する．

3　管理者向け仕様書

3.1　インストール方法

本システムを運用するために必要なソフトウェアとそのインストール手順について説明する．まず初めに，MacOS にパッケージ管理システムである Homebrew をインストールする．ターミナルで以下のコマンドを実行する．

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/
    install/HEAD/install.sh)"
```

パスワードの入力を求められた場合は，MacOS のログインパスワードを入力しプロンプトが表示されるまで待つ．次に以下の 2 つのコマンドを順に実行する．

```
( echo; echo 'eval "$(/opt/homebrew/bin/brew shellenv)"') >> ~/.zprofile
eval "$(/opt/homebrew/bin/brew shellenv)"
```

次に，JavaScript の実行環境である Node.js を管理するために必要な nodebrew をインストールする．ターミナルで以下の 4 つのコマンドを順に実行する．

```
brew install nodebrew
nodebrew setup
echo 'export PATH=$HOME/.nodebrew/current/bin:$PATH' >> ~/.zshrc
source ~/.zshrc
```

次に，Node.js の最新版をインストールする．ターミナルで以下の 2 つのコマンドを順に実行する．

```
nodebrew install stable
nodebrew ls
```

ここで表示された最新版のバージョン番号を確認し，以下のコマンドを実行してインストールした最新版を使用するように設定する．仕様書ではバージョン番号を v24.3.0 と仮定する．最後に以下の 2 つのコマンドを順に実行する．

```
nodebrew use v24.3.0
npm install -g npm
```

3.2　起動方法

3.3　起動できない場合

3.4　分かっている不具合