Web プログラミング仕様書

25G1086 土谷陸

2025 年 12 月 28 日

1 Github の URL

`https://github.com/kei428-maker/webpro_06.git`

2 開発者向け仕様書

2.1 ポーカー役管理システム

2.1.1 概要

本システムは，ポーカーの役に関するデータを管理するための Web アプリケーション機能です． 利用者は，ポーカーの役について以下の情報を閲覧，登録，編集，および削除することができます．

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　ポーカー役データの構造

項目	データ型	説明
id	整数	役の ID
name	文字列	役の名前
power	文字列/整数	役の強さ
probability	文字列	出現確率
noujiru	整数	脳汁度

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．

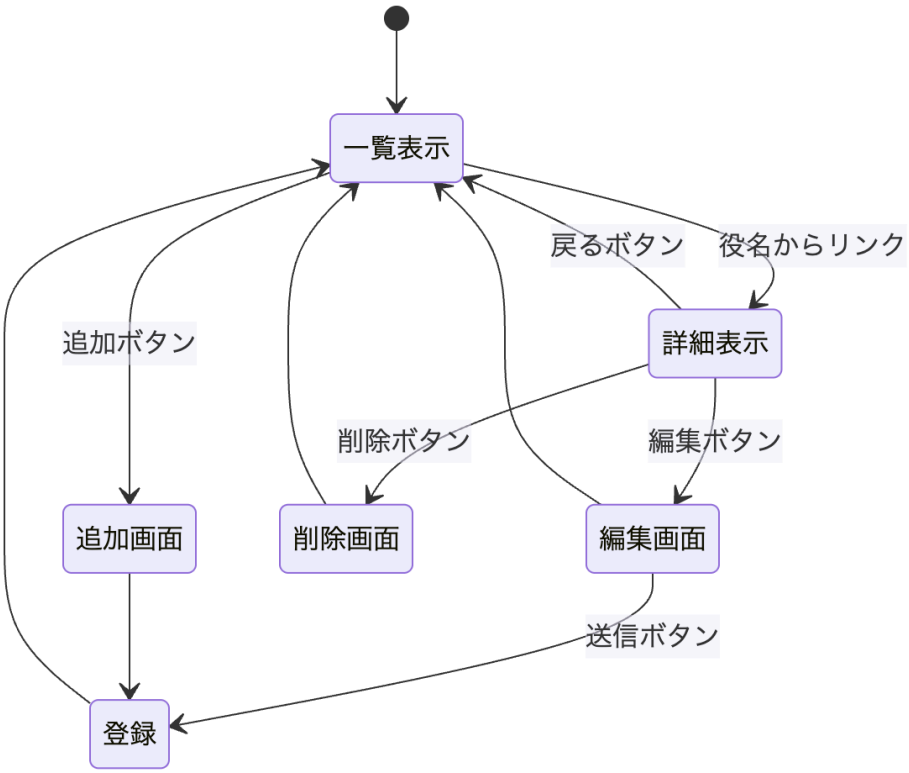

図 1　ポーカー役管理システムページ遷移図

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 2 に示す．

表 2　リソース一覧 (ポーカー)

HTTP メソッド	リソース名	説明
GET	`/poker2`	一覧表示．サーバーからデータを取得し一覧を表示する．
GET	`/poker2/create`	新規登録画面．静的な登録フォームへリダイレクトする．
POST	`/poker2`	新規登録処理．フォームから送信されたデータを追加する．
GET	`/poker2/:number`	詳細表示．指定された ID の役の詳細を表示する．
GET	`/poker2/edit/:number`	編集画面．指定された ID のデータの編集フォームを表示する．
POST	`/poker2/update/:number`	更新処理．編集されたデータで既存データを更新する．
GET	`/poker2/delete/:number`	削除処理．指定された ID のデータを削除し一覧へ戻る．

2.1.5 リソースの説明

本システムは，HTTP メソッドと URL パスを組み合わせた RESTful な設計に基づき，ポーカーの役データに対する各種操作を実現している．

まず，データの閲覧についてであるが，Web ブラウザから `/poker2` へ GET メソッドでアクセスすることで，サーバー内に保持されているすべての役データを取得し，一覧画面として表示する仕組みとなっている．個別の役の詳細情報を確認したい場合は，`/poker2/:id` (GET) へアクセスすることで，指定された ID に対応する役の詳細画面が表示される．

データの更新・管理機能については以下の通りである．新しい役を登録する際は，まず `/poker2/create` (GET) にアクセスして新規登録用の入力フォームを表示させる．ユーザーがフォームに必要な情報を入力し送信すると，`/poker2` に対して POST リクエストが送られ，サーバー側でデータの追加処理が実行される．既存の役情報を変更する場合は，`/poker2/edit/:id` (GET) にアクセスして編集画面を表示し，変更内容を送信することで `/poker2/update/:id` へ POST リクエストが行われ，データが更新される．なお，データの削除に関しては，`/poker2/delete/:id` への GET リクエストを受け取った時点で即座に該当データを削除し，一覧画面へリダイレクトする簡易的な実装を採用している．

2.2 サッカー日本代表管理システム

2.2.1 概要

本システムは，サッカー日本代表選手の情報を管理する Web アプリケーションである．選手の名前，ポジション，所属チーム，背番号を管理対象とし，日本代表メンバーの構成を一覧で確認したり，メンバーの入れ替えを行ったりする機能を提供する．

2.2.2 データ構造

データ構造は以下の表 3 のようになる．

表 3　サッカー日本代表選手データの構造

項目	データ型	説明
id	整数	選手の ID
name	文字列	選手の名前
position	文字列	ポジション
team	文字列	所属チーム
backnumber	整数	背番号

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．

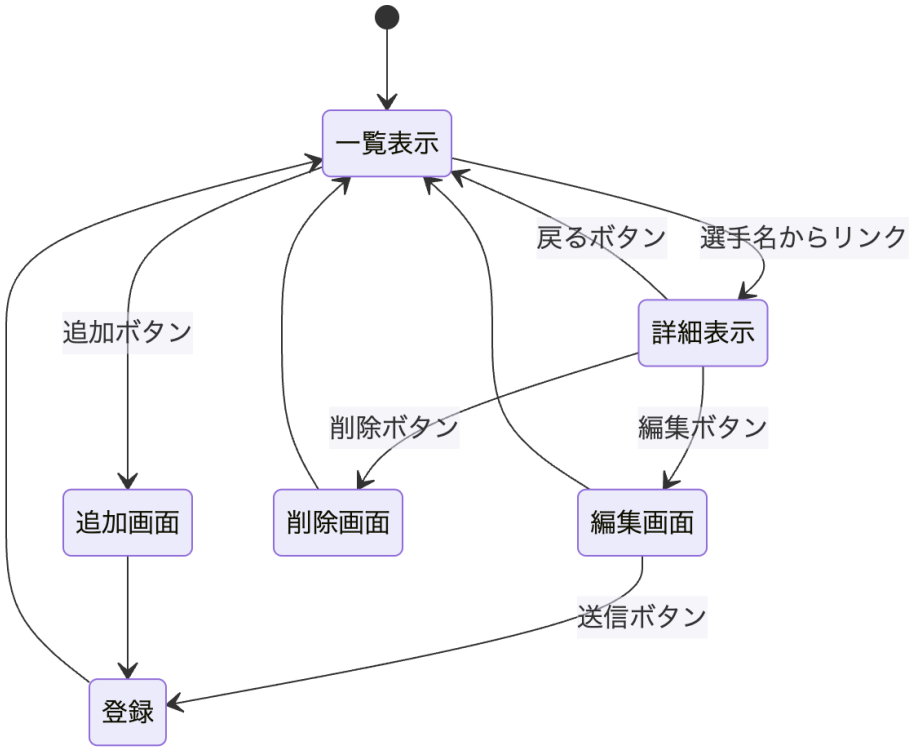

図 2　サッカー日本代表管理システム ページ遷移図

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 4 に示す.

表 4　リソース一覧 (サッカー)

HTTP メソッド	リソース名	説明
GET	`/soccer2`	一覧表示. サーバーからデータを取得し一覧を表示する.
GET	`/soccer2/create`	新規登録画面. 静的な登録フォームへリダイレクトする.
POST	`/soccer2`	新規登録処理. フォームから送信されたデータを追加する.
GET	`/soccer2/:number`	詳細表示. 指定された ID の選手の詳細を表示する.
GET	`/soccer2/edit/:number`	編集画面. 指定された ID のデータの編集フォームを表示する.
POST	`/soccer2/update/:number`	更新処理. 編集されたデータで既存データを更新する.
GET	`/soccer2/delete/:number`	削除処理. 指定された ID のデータを削除し一覧へ戻る.

2.2.5 リソースの説明

本システムは RESTful な設計指針に基づき，URL と HTTP メソッドの組み合わせによって各操作を実現している．

まず，データの閲覧に関する機能について説明する．ブラウザから `/soccer2` に GET メソッドでアクセスすると，サーバーはメモリ上に保存されている全サッカー選手のデータを取得し，一覧画面として表示する．また，特定の選手の詳細情報を確認したい場合は，`/soccer2/:id` (GET) にアクセスすることで，指定された ID を持つ選手の詳細画面が表示される．

次に，データの変更に関する機能についてである．新規データの追加は，まず `/soccer2/create` (GET) にアクセスして登録用フォームを表示させ，ユーザーが情報を入力して送信ボタンを押すことで，`/soccer2` に対して POST リクエストが送信され，データが追加される．既存データの編集も同様に，`/soccer2/edit/:id` (GET) で現在の値が入ったフォームを表示し，`/soccer2/update/:id` へ POST リクエストを送ることで更新処理が行われる．なお，削除機能については簡易的な実装を採用しており，`/soccer2/delete/:id` への GET リクエストをトリガーとして該当データの削除を実行し，処理完了後は自動的に一覧画面へリダイレクトする仕様となっている．

2.3 コムドットメンバー管理システム

2.3.1 概要

本システムは，グループ「コムドット」のメンバー情報を管理する Web アプリケーションである．メンバーの名前，年齢，血液型，身長をデータとして保持し，ファンのためのメンバーリストとして情報の閲覧や更新を行う機能を提供する．

2.3.2 データ構造

データ構造は以下の表 5 のようになる．

表 5　コムドットメンバーデータの構造

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
old	整数	年齢
bloodtype	文字列	血液型
tall	整数	身長

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．

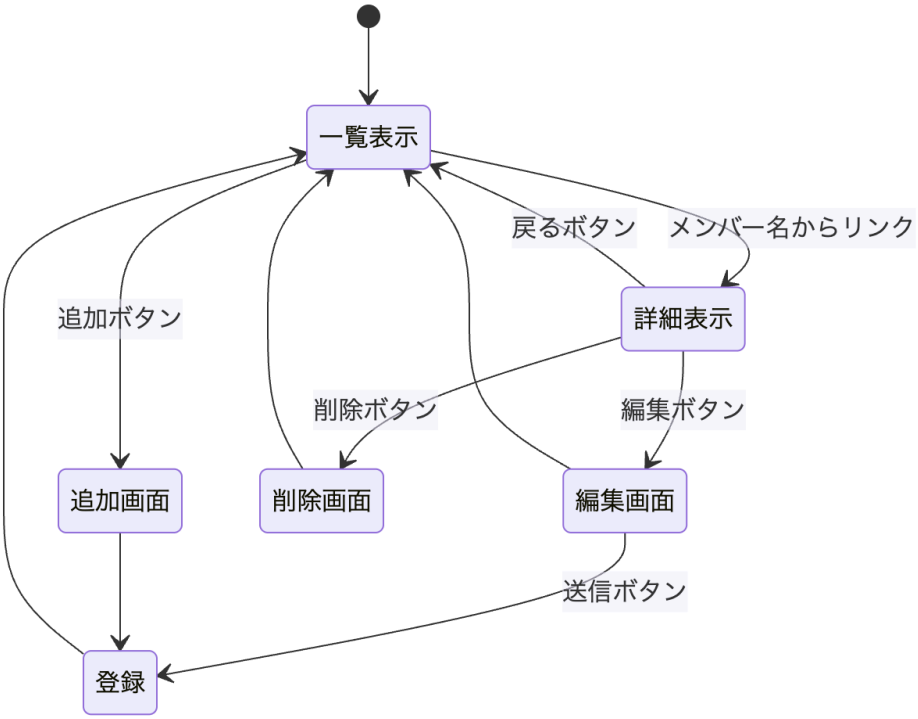

図 3　コムドットメンバー管理システム ページ遷移図

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 6 に示す.

表 6　リソース一覧 (コムドット)

HTTP メソッド	リソース名	説明
GET	`/com2`	一覧表示．サーバーからデータを取得し一覧を表示する.
GET	`/com2/create`	新規登録画面．静的な登録フォームへリダイレクトする.
POST	`/com2`	新規登録処理．フォームから送信されたデータを追加する.
GET	`/com2/:number`	詳細表示．指定された ID のメンバーの詳細を表示する.
GET	`/com2/edit/:number`	編集画面．指定された ID のデータの編集フォームを表示する.
POST	`/com2/update/:number`	更新処理．編集されたデータで既存データを更新する.
GET	`/com2/delete/:number`	削除処理．指定された ID のデータを削除し一覧へ戻る.

2.3.5 リソースの説明

コムドットメンバー管理システムにおいても，前述のシステムと同様のリソース設計を採用している．

メンバー情報の一覧表示は `/com2` への GET リクエストによって行われ，個別のメンバー詳細プロフィールの閲覧は `/com2/:id` (GET) によって提供される．データの管理機能に関しては，`/com2/create` で新規登録フォームを取得し，`/com2` への POST 送信によって新規メンバーを追加する．既存メンバーの情報の更新は，`/com2/edit/:id` で編集画面を表示し，フォーム送信により `/com2/update/:id` (POST) へデータを送信することで完了する．削除処理については `/com2/delete/:id` へのアクセスによって，該当するメンバーのデータを削除し，一覧画面へ戻る挙動となる．

3 管理者向け仕様書

3.1 インストール方法

本システムの動作環境を構築するための手順を以下に示す．なお，本システムは Node.js 環境で動作するため，事前に Node.js がインストールされていることを前提とする．

1. **ソースコードの取得**
   ターミナルを開き，任意のディレクトリで以下のコマンドを実行して GitHub リポジトリからソースコードを取得する．

   ```
   git clone https://github.com/kei428-maker/webpro_06.git
   ```

2. **ディレクトリの移動**
   クローンによって作成されたプロジェクトディレクトリに移動する．

   ```
   cd webpro_06
   ```

3. **依存パッケージのインストール**
   本システムが必要とするライブラリを一括でインストールする．以下のコマンドを実行する．

   ```
   npm install
   ```

   コマンド実行後，エラーが表示されず，ディレクトリ内に `node_modules` フォルダが生成されていればインストールは完了である．

3.2 起動方法

```
node app_kadai.js
```

ターミナルに

`Example app listening on port 8080!`

と表示されたら起動できている状態である．

3.3 起動できない場合

起動コマンドを実行してもサーバーが正常に立ち上がらない場合，主な原因としてポート番号の競合やモジュールの未インストールが考えられる．

まず，ターミナルに `Error: listen EADDRINUSE: address already in use :::8080` というエラーメッセージが表示された場合は，ポート 8080 番が他のプロセスによって既に使用されている状態である．この問題を解決するには，競合している他のプロセスを終了させるか，あるいは `app_kadai.js` 内で定義されているポート番号を別の値に変更して再度起動を試みる必要がある．

また，`Error: Cannot find module 'express'` のようなモジュールが見つからない旨のエラーが表示された場合は，実行に必要なライブラリがインストールされていないことが原因である．この場合は，プロジェクトのルートディレクトリで `npm install` コマンドを実行し，依存関係にあるパッケージをインストールすることで解決できる．

3.4 終了方法

起動中のサーバーを停止させるには，サーバーを実行しているターミナルウィンドウにおいて，キーボードの `Ctrl` キーを押しながら `C` キーを押下する．これにより，実行中の Node.js プロセスに割り込み信号が送信され，サーバーが停止してコマンドプロンプトが入力を受け付ける状態に戻る．

3.5 分かっている不具合

本システムの現行バージョンにおいては，いくつかの機能制限および不具合が確認されているため，運用にあたっては注意が必要である．

最も重要な制限事項として，データの永続性が確保されていない点が挙げられる．本システムはデータベースを使用せず，メモリ上の変数配列でデータを管理している仕様のため，サーバーを再起動または終了すると，それまでに追加・編集・削除されたデータはすべて失われ，ソースコードに記述された初期状態にリセットされる．

また，入力機能に関してはバリデーションが実装されていない．そのため，入力フォームを空欄のまま送信したり，数値が必要な項目に文字列を入力したりした場合でも処理が実行され，不正なデータが登録される可能性がある．さらに，データの削除機能については確認ダイアログが表示されず，削除ボタンを押下した瞬間に処理が実行される仕様となっているため，操作ミスによるデータ消失のリスクがある．

4 利用者向け仕様書

4.1 概要

本システムは，Web ブラウザ上で動作する情報管理システムである．利用者は，登録されている「サッカー選手」「コムドットメンバー」「ポーカーの役」の 3 種類のデータについて，情報の閲覧，新規追加，編集，削

除を行うことができる．直感的な画面操作により，データベースに関する専門知識がなくとも容易にデータを管理することが可能である．

4.2 使用できる機能

本システムで利用者が使用できる主な機能は以下の通りである．

- **一覧表示機能**: 登録されているデータの一覧を表示する．
- **詳細表示機能**: 各データの詳細な情報を表示する．
- **新規追加機能**: 新しいデータをシステムに登録する．
- **編集機能**: 既存のデータを修正・更新する．
- **削除機能**: 不要なデータをシステムから削除する．

4.3 起動画面と一覧表示

本システムには統合されたトップページは存在しないため，利用したい機能の URL に直接アクセスすることで利用を開始する．ここでは，「サッカー選手管理機能」を例に操作手順を説明する．

Web ブラウザのアドレスバーに `http://localhost:8080/soccer2` と入力しアクセスすると，図 4 のような一覧画面が表示される．

日本代表スターティングメンバーの詳細

ID　　名前

1　上田綺世

2　南野拓実

3　久保建英

4　三笘薫

5　堂安律

6　守田英正

7　遠藤航

8　伊藤洋輝

9　板倉滉

10 瀬古歩夢

11 鈴木ザイオン

追加

図 4　サッカー選手一覧画面

図 4 に示すように，現在登録されているサッカー選手の ID，名前などの主要な情報がリスト形式で表示される．この画面が本機能の基本画面となる．

4.4　詳細情報の閲覧

一覧画面において，詳細を確認したい選手の「ID」または「名前」のリンクをクリックすると，詳細表示画面へ遷移する．詳細表示画面では図 5 のような画面が表示される．

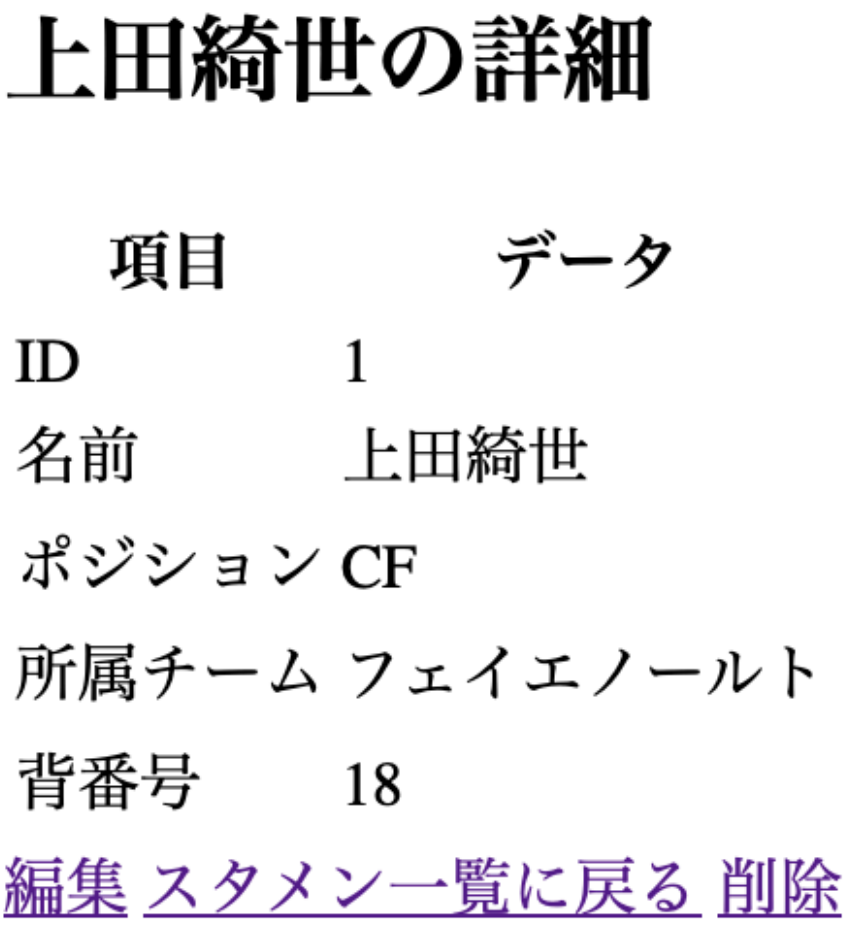

図 5　サッカー選手詳細画面

図 5 に示すように，その選手のポジション，所属チーム，背番号といった詳細な情報が表示される．この画面から，情報の編集や削除を行うことが可能である．

4.5　情報の追加

新しい選手データを追加する場合，一覧画面にある「Create」や「新規作成」といったリンクをクリックする．すると，図 6 のような新規登録画面が表示される．

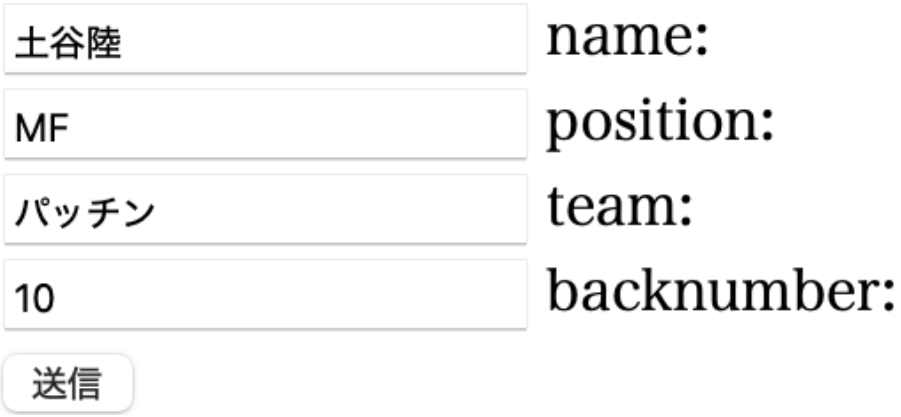

図 6　サッカー選手新規登録画面

図 6 に示す入力フォームに，追加したい選手の「名前」「ポジション」「チーム」「背番号」を入力し，登録ボタンをクリックする．入力内容が送信され，データが追加されると，自動的に一覧画面へ戻り，新しいデータが末尾に追加されていることが確認できる．

4.6　情報の編集

既存のデータを修正したい場合，詳細表示画面（図 5）にある「Edit」リンクをクリックする．すると，図 7 のような編集画面が表示される．

日本代表スターティングメンバーの詳細

ID	名前
1	上田綺世
2	南野拓実
3	久保建英
4	三笘薫
5	堂安律
6	守田英正
7	遠藤航
8	伊藤洋輝
9	板倉滉
10	瀬古歩夢
11	鈴木ザイオン
12	土谷陸

追加

図 7　サッカー選手編集画面

図 7 に示すように，現在の登録情報がフォームに入力された状態で表示される．修正したい項目を書き換え，更新ボタンをクリックすることで変更が保存される．更新が完了すると，自動的に一覧画面へ戻り，変更内容が反映される．

4.7　情報の削除

データを削除したい場合，詳細表示画面（図 5）などにある「Delete」リンクをクリックする．本システムでは削除の確認画面は表示されず，リンクをクリックした時点で即座にデータが削除されるため，操作には注意が必要である．削除処理が完了すると一覧画面へ戻り，該当のデータがリストから消去されていることが確認できる．