

Web プログラミング及び演習  
レポート課題  
(スケジュール管理用 Web アプリの作成)

愛知工業大学情報科学部情報科学科  
コンピュータシステム専攻 3 年  
講義名 Web プログラミング及び演習  
学籍番号 K21071  
氏名 須田倫代

## 1. 課題内容

スケジュール管理用 Web アプリを作成する

要求仕様:

- (1) 登録ユーザでログイン後、スケジュールの表示、登録、変更、削除が出来る。
- (2) 表示は、リスト形式とする。
- (3) スケジュールはデータベースにテーブルを作成して登録する。
- (4) スケジュールは開始日時、終了日時、場所、内容を必ず含むものとする。

追加仕様:(必須ではない、加点対象)

- (1) 週間のスケジュールを表示できるようにする。
- (2) 月間のスケジュールを表示できるようにする

※作成に当たって CakePHP などのフレームワークを使用しても良い。

## 2. 作成したスケジュール管理用 Web アプリケーション

今回の課題で作成したスケジュール管理用 Web アプリケーションは PHP のフレームワークである Laravel を使用して作成している。

### 2.1 画面の状態遷移

作成したスケジュール管理用 Web アプリケーションの画面遷移は以下のようになっている。

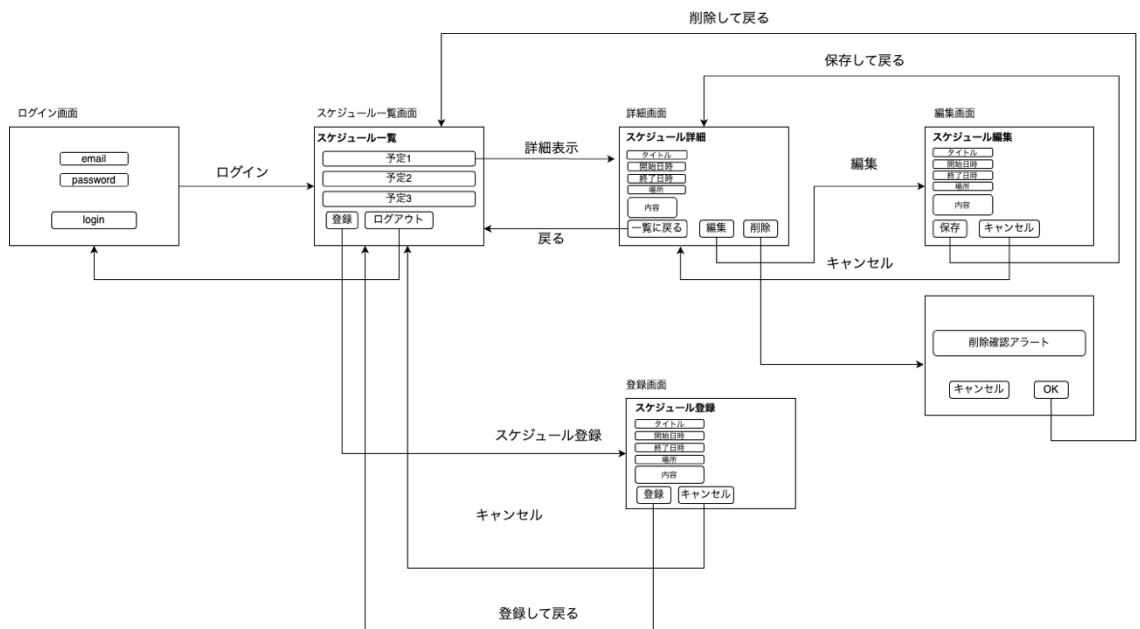


図 1 画面の状態遷移図

php artisan serve コマンドを実行してサーバーを起動し、http://localhost:8000 にアクセスするとログイン画面が表示されるようになっている。

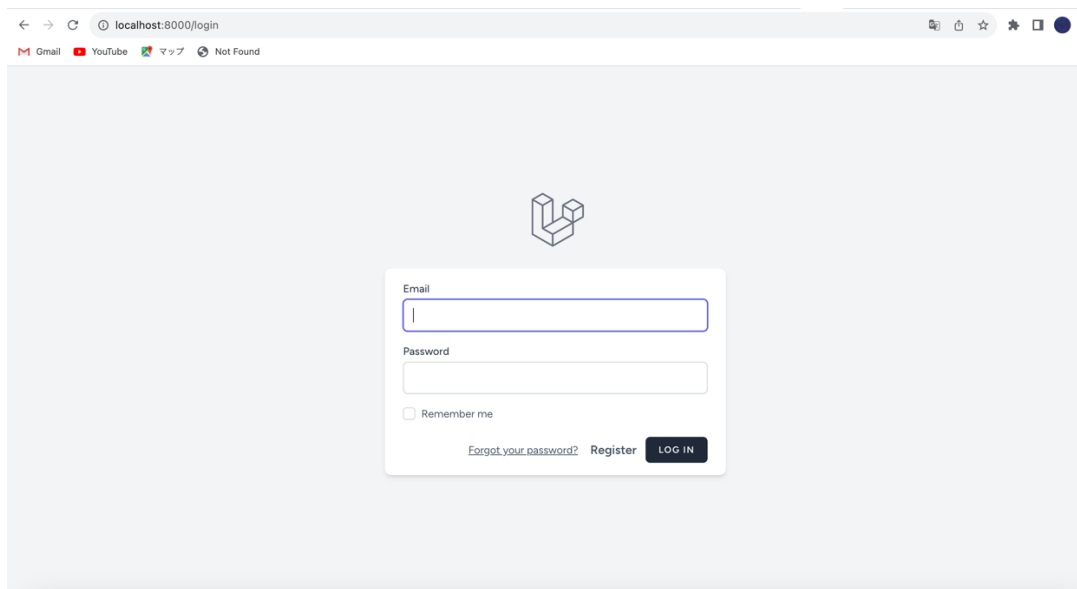


図2 ログイン画面

予め登録したユーザのメールアドレスとパスワードでログインすると、スケジュールの一覧画面が表示される。ユーザの新規登録は Register をクリックすると行うことができる。

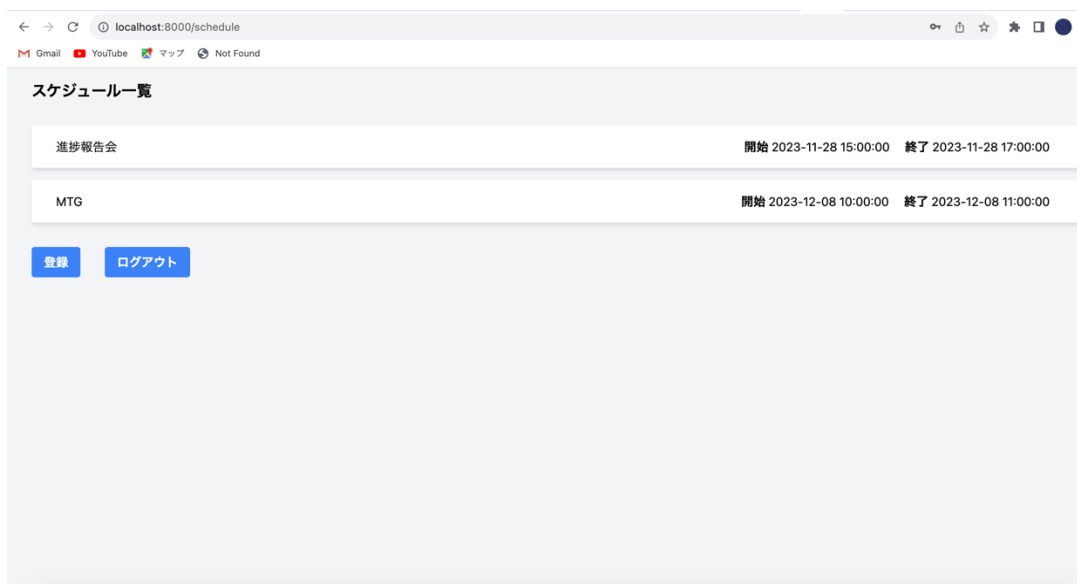


図3 スケジュール一覧画面

スケジュール一覧画面は、ログインユーザが登録したスケジュール、登録ボタン、ログアウトボタンで構成されている。スケジュールのカードをクリックすると、そのスケジュールの詳細画面が表示される。

スケジュールの詳細画面は、スケジュールのタイトル、開始日時、終了日時、場所、内容を含むスケジュールの詳細と、一覧に戻るボタン、編集ボタン、削除ボタンで構成されている。一覧に戻るボタンをクリックするとスケジュール一覧画面に戻ることができる。

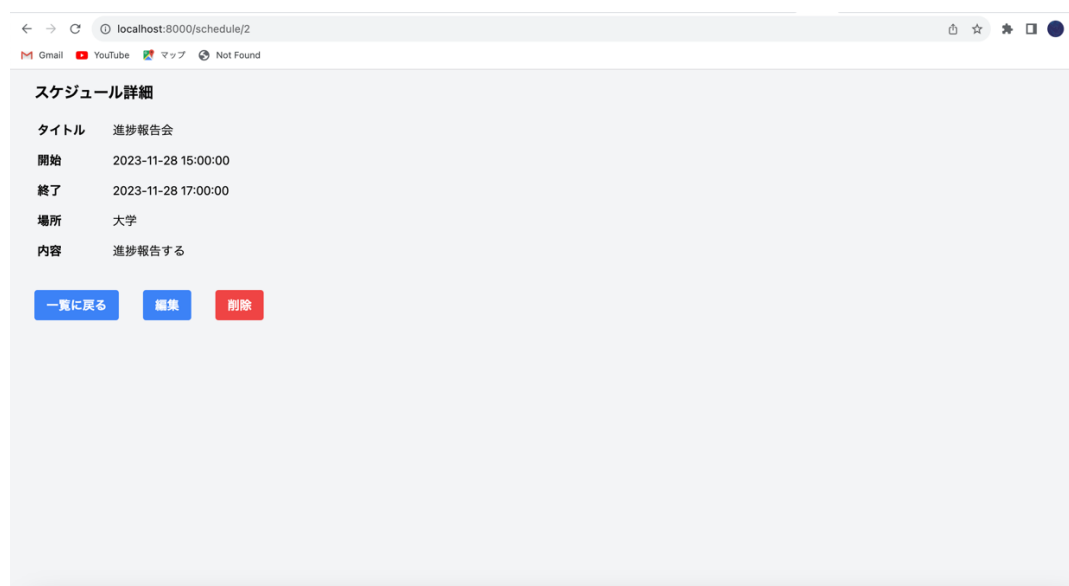


図 4 スケジュール詳細画面

編集ボタンをクリックすると編集画面に遷移し、スケジュールの詳細を編集することが可能である。編集画面の保存ボタンをクリックすれば編集内容がデータベースに保存され、スケジュールの詳細画面に戻るようになっている。スケジュールの詳細画面に戻るため、変更内容をすぐに確認することができる。

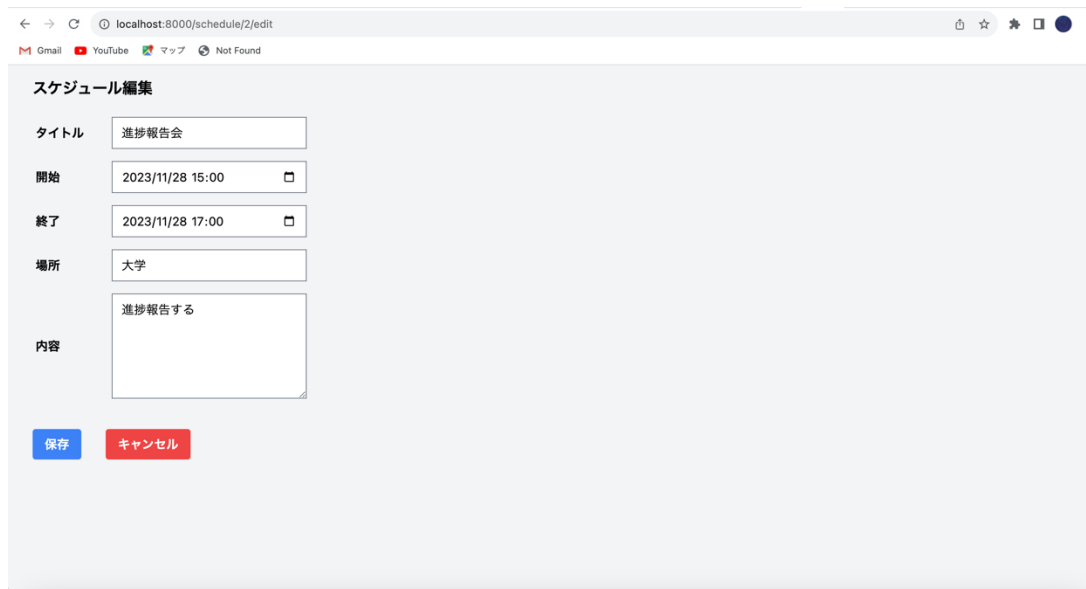


図 5 スケジュール編集画面

詳細画面で削除ボタンをクリックすれば、本当にそのスケジュールを削除しても良いかを確認するアラート画面が表示される。OK ボタンをクリックすればそのスケジュールが削除され、スケジュール一覧画面に遷移するようになっている。スケジュールの削除後はスケジュール一覧画面に遷移されるため、スケジュールが削除されたことを一目で確認することができる。

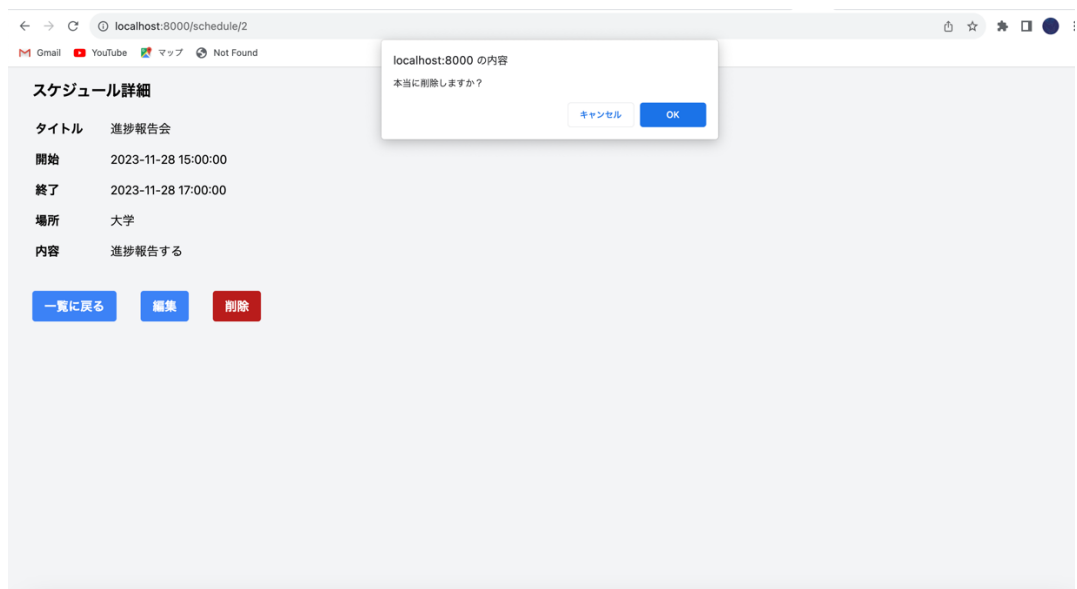


図 6 削除確認アラートの表示

登録ボタンをクリックすると登録しているユーザの新規スケジュールを登録することができる。全ての項目は入力必須項目になっているため、未入力項目があるとメッセージが表示されるようにしている。登録したスケジュールはスケジュール一覧画面に反映される。

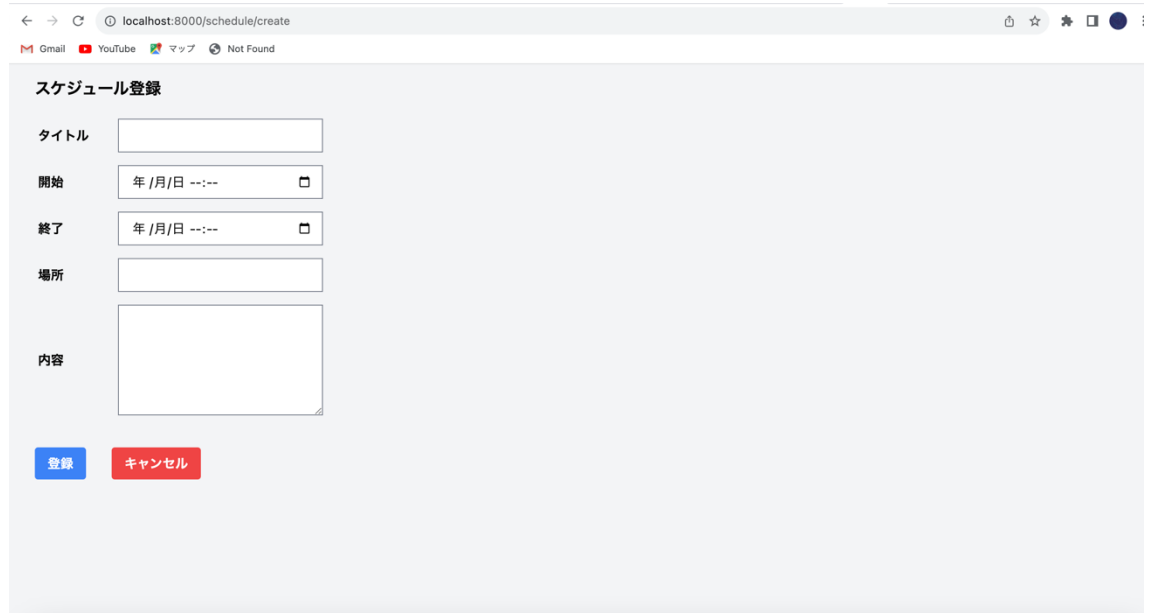


図 6 スケジュール登録画面

スケジュール一覧画面のログアウトボタンをクリックすると、ログアウトされ、ログイン画面に遷移する。また、ログインされていない状態で特定のページを見ようとする、ログイン画面にリダイレクトされ、ログインするように促される仕組みになっている。

その他、キャンセルボタンをクリックすると実行しようとしていることをやめ、ひとつ前のページに戻るようになっている。

## 2.2 データベースのテーブルとリレーション

今回のスケジュール管理用 Web アプリケーションでは Laravel データベースの users テーブルと schedules テーブルを主に使用している。users テーブルはユーザの管理、schedules テーブルはスケジュールの管理をしている。

行 0 - 2 の表示 (合計 3, クエリの実行時間: 0.0003 秒。)

```
SELECT * FROM `users`
```

	id	name	email	email_verified_at	password	remember_token
<input type="checkbox"/>	1	test	example@aiotech.ac.jp	NULL	\$2y\$12\$ztqQJ2yAjjAkY:7H51pBuQwlnp8qXqBUvi4RnZrCAp...	NULL
<input type="checkbox"/>	2	test2	example2@aiotech.ac.jp	NULL	\$2y\$12\$JkSohdagq2bt4kztWoObEup.hCqGUZErm18UXYmuB.1...	NULL
<input type="checkbox"/>	3	test3	example3@aiotech.ac.jp	NULL	\$2y\$12\$HcWfN/0BUGulZLQQvdK.v2RURhR2d1TPnmjRDhQmk...	NULL

図 7 users テーブル

行 0 - 2 の表示 (合計 3, クエリの実行時間: 0.0004 秒。)

```
SELECT * FROM `schedules`
```

	id	title	begin	end	place	content	created_at	updated_at	userid
<input type="checkbox"/>	2	進捗報告会	2023-11-28 15:00:00	2023-11-28 17:00:00	大学	進捗報告する	2023-11-27 14:39:19	2023-11-29 02:41:32	1
<input type="checkbox"/>	3	お茶会	2023-11-30 14:30:00	2023-11-30 15:30:00	喫茶店	お茶会	2023-11-27 14:43:04	2023-11-27 14:43:04	2
<input type="checkbox"/>	4	MTG	2023-12-08 10:00:00	2023-12-08 11:00:00	オンライン	MTG	2023-11-28 16:13:54	2023-11-28 16:13:54	1

図 8 schedules テーブル

users テーブルは Laravel の標準で用意されているものを使用している。schedules テーブルは今回用に独自に作成したものであり、テーブルの構成は以下のようになっている。

schedules テーブルの構成

- id increments(スケジュールを識別するための固有の ID)
- title text(スケジュールのタイトル)
- begin datetime(開始日時)

- end datetime(終了日時)
- place text(場所)
- content text(内容)
- created\_at timestamps(スケジュールの登録日時)
- updated\_at timestamps(スケジュールの更新日時)
- userid unsignedBigInteger(ユーザ ID)

新規にユーザを登録すると、データベースではユーザ ID(図 7 users テーブルの id カラム)が付与される。ログインしているユーザでスケジュールの登録を行うと、スケジュールのデータにログインユーザのユーザ ID(図 8 schedules テーブルの userid カラム)が付けられ、データが連携される。つまり、図 7 users テーブルの id カラムと図 8 schedules テーブルの userid カラムの値が一致するデータがそのユーザのスケジュールデータである。

スケジュールを一覧表示させるときは、ログインしているユーザのユーザ ID を見て表示する内容を変更している。

## 2.3 設計のコンセプト

設計のコンセプトは、無駄な動きのないシンプルな設計である。例えば、スケジュール内容を編集し、保存した後は編集内容が反映されているか確認することが予想される。そのために、保存したらスケジュールの詳細画面に自動的に遷移して内容を確認できるようにした。他に、スケジュールの登録・削除後は、本当にスケジュールが登録・削除されているか確認するという動作が予想される。よって、スケジュールの登録・削除後は自動的にスケジュール一覧画面に遷移し、スケジュールの有無を一目で確認できるようにした。これらのように、ある一つの動作をした後に、ユーザーは次にどのような動作をするかを予想して設計をすることにより、ユーザーに無駄な操作をさせないストレスフリーな設計を意識して作成した。

## 2.4 実行方法

1. 最初に実行する際は、ターミナルで `php artisan migrate` コマンドを実行し、マイグレーションの実行をする。
2. `php artisan serve` コマンドを実行して、サーバーを起動する。
3. `npm run dev` コマンドを実行して、CSS を反映させる

## 2.5 目標

最低限、PHP のフレームワークを用いて、課題の要求仕様を全て満たす実装を目標とした。余裕があれば追加仕様も実装したいと考えていた。



## 2.6 達成度

課題の要求仕様を全て満たす実装を行うことができたので、最低限の実装はすることができた。PHP のフレームワークである Laravel を初めて使用したが、MVC モデルの仕組みを理解して実装を行うことができたと思う。また Tailwind CSS を用いて、シンプルではあるが少しデザインを整えることもできた。

## 2.7 反省点

計画的に制作を進められなかった部分があり、追加仕様までは実装できなかった点が反省点である。