PHP フレームワーク Laravel の習得

○中村成美、太田芳博、伊藤康広、牧野輝、伊藤大作、島田啓史、藤原冨未治

情報通信技術支援室　情報システム構築技術グループ

概要

現在、我々が管理しているサーバには、古くから稼働し続けている Web アプリケーションが複数存在している。これらは当時の担当者が独自に開発を行っていたため、プログラムコードの書き方が統一されておらず、後任の担当者がプログラムの内容を理解するのに時間を要することが想定されている。そこで我々は、フレームワーク（アプリケーションを開発するときの土台として機能するソフトウェア）を用いて開発を行うことを考えた。フレームワークを導入することで、開発担当メンバー内でコーディングルールを統一化できるため、業務効率の大幅な向上が見込める。

今回は日本語のドキュメントが豊富で、学習コストも比較的低いと言われている PHP フレームワーク Laravel を用いた Web アプリケーション開発技術の習得に取り組むことにした。本報告では、開発環境と作成した Web アプリケーションについて報告を行う。

1　開発環境

Web アプリケーションの開発環境として、プログラム言語は PHP7.3、フレームワークは Laravel5.5（LTS）、パッケージ管理システムは Composer、データベースは MySQL 系を使用した。

フレームワークとして Laravel を採用した理由は以下のとおりである。

- 無料・オープンソースのフレームワークである。
- 古くから実績があり、高い信頼性を持つ Symfony のコンポーネントを実装の一部に利用している。
- 日本語のドキュメントが豊富なため、学習コストが低い。
- 新しいバージョンが定期的にリリースされており、今後も長期的なサポートが見込める。

2　作成した Web アプリケーションの概要

作成した ToDo リストアプリケーションの画面を図 1 に示す。作成した ToDo リストアプリケーションは、ユーザが登録したタスクを、一覧で表示する機能をもつ。ユーザはタスクを編集及び削除することもできる。

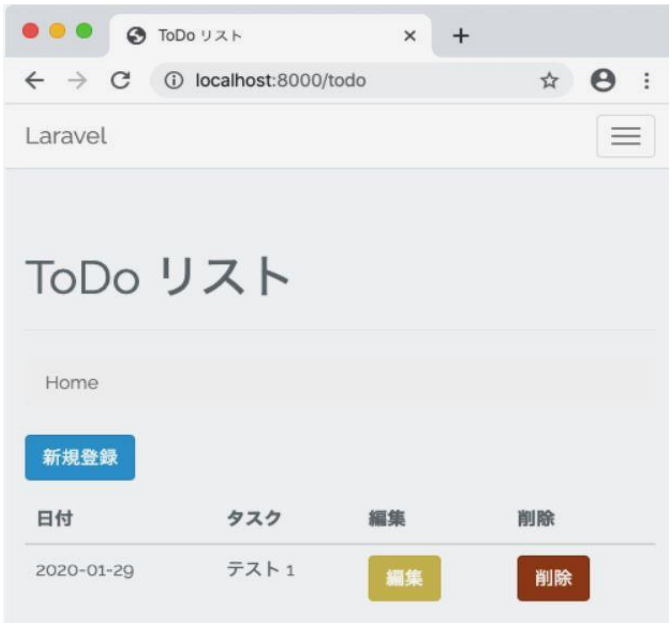

図 1. ToDo リスト表示画面

3　アプリケーションの開発について

3.1　Laravel における MVC モデル

今回、GUI アプリケーション実装のためのデザインパターンとして、MVC（Model / View / Controller）モデルを採用した。MVC モデルにおける、データ処理の流れを図 2 に示す。

ユーザからの HTTP リクエストは、URL、アクセスしてきた HTTP メソッド及びルート定義に基づいて適切な Controller が選択され、処理が振り分けられる。Controller は、routing から受け取ったリクエスト情報を処理するよう Model に指示する。Model はデータベースとの連携を行っており、必要なデータをデータベースから受け取り Controller へ返す。Controller は Model から返却されたデータを受け取り、View へ出力を指示する。View はユーザに対し、HTTP レスポンスを返す。

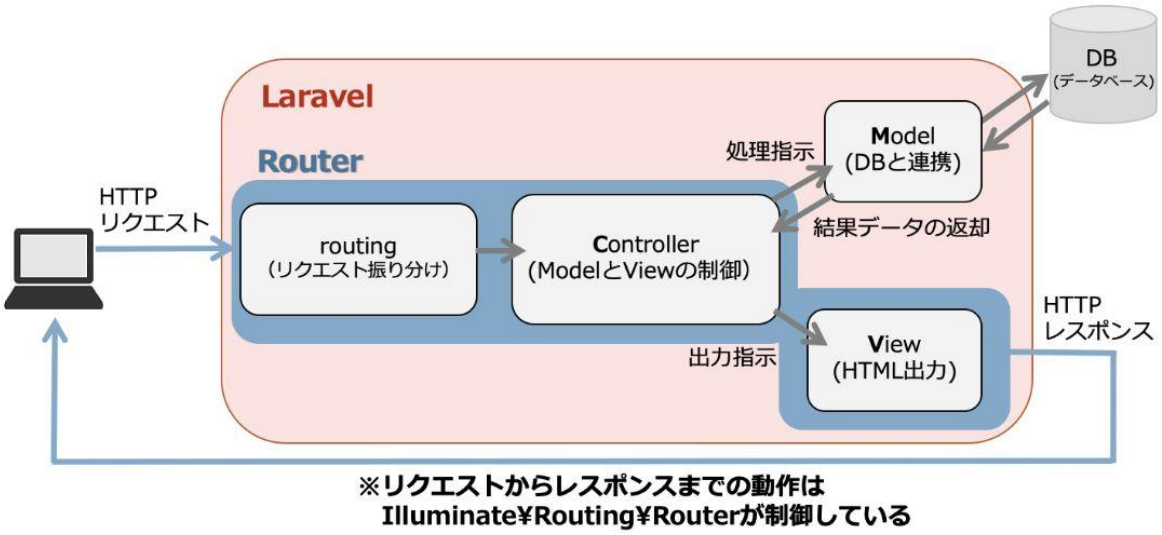

図 2. データ処理の流れ

3.2　プロジェクトの作成

PHP 向けのパッケージ管理ツールである Composer を用い、Laravel のバージョンを指定してプロジェクト todolist_application を作成した。作成されたプロジェクトフォルダの vendor 以下には、開発と実行に必要なライブラリがインストールされる。

```
$ composer create-project --prefer-dist laravel/laravel todolist_application "5.5.*"
```

3.3　Controller と View の作成

ToDo リストを扱う Controller として、TodoListsController を下記のコマンドで作成した。--resource オプションをつけて Controller を作成することにより、リソース操作に関するメソッドが自動で定義される。

```
$ php artisan make:controller TodoListsController --resource
```

次に、Controller と View の動作を理解するため、/todo にアクセスした時、Controller から渡される値を View で受け取り、表示する単純な機能を実装した。具体的には、TodoListsController に定義されたデータ表示用メソッド（index メソッド）の返値として、View 名と View に渡す文字列を代入した変数名を指定した。また、別途 View 名で指定したデータ表示用ファイルを用意した。

/todo という URL にアクセスした場合、TodoListsController が処理を行うようにルート設定を行った。プロジェクト内で下記のコマンドにより PHP 組み込みの Web サーバを起動した後、ブラウザから/todo にアクセスし、View に渡した文字列が表示されることを確認した。

```
$ php artisan serve
```

3.4　データベースとの連携

実際に作成する ToDo リストアプリケーションでは、データベースにタスクを保存する必要がある。この節ではデータベース連携について説明する。

3.4.1　データベースとの接続設定

データベースとの接続設定は、プロジェクト内の.env ファイルに記述した。図 3 にデータベース接続に関連する設定例を示す。

```
DB_CONNECTION=mysql ※データベースの種類
DB_HOST=（データベースサーバの IP アドレス）
DB_PORT=3306 ※ポート番号
DB_DATABASE=（データベース名）
DB_USERNAME=（データベースに接続するユーザ名）
DB_PASSWORD=（データベースに接続するユーザのパスワード）
```

図 3. .env ファイル（一部）

3.4.2　Model とマイグレーションの作成

Laravel のマイグレーションは、データベースのテーブル作成や定義変更を管理する機能である。マイグレーションファイルにテーブル構造を定義し、マイグレーションを実行すると、データベースに定義が反映される。また、実行したマイグレーションはデータベースの migrations テーブルで世代管理されるため、ロールバックすることも可能である。

まず、Model とマイグレーションファイルを同時に作成する。下記コマンドを実行した場合、Laravel の規約により、Model は“TodoLists”、マイグレーションファイルは“日付_create_todo_lists_table”、データベースのテーブルは“todo_lists”という名前になる。

```
$ php artisan make:model TodoLists --migration
```

作成されたマイグレーションファイルに todo_lists テーブル構造（表 1）を記述し、下記コマンドでマイグレーションを実行しデータベースにテーブルを作成した。

表 1. todo_lists テーブル構造

カラム名	カラムタイプ	MySQL のデータ型	備考
id	increments	AUTO_INCREMENT	主キー
date	string	VARCHAR	日付
message	string	VARCHAR	タスク
created_at	timestamp	TIMESTAMP	作成日時
updated_at	timestamp	TIMESTAMP	更新日時

```
$ php artisan migrate
```

3.4.3　Seeder

Seeder は、データベースに初期値を投入するための機能である。以下のコマンドで TodoListsTableSeeder ファイルを作成し初期値を記述した。

```
$ php artisan make:seeder TodoListsTableSeeder
```

次に、下記のコマンドを実行してデータベースの todo_lists テーブルに初期値を投入した。

```
$ php artisan db:seed
```

3.5　CRUD 操作

CRUD とは、Web アプリケーションの基本操作であり、Create（新規登録）、Read（データ取得）、Update（更新）、Delete（削除）を表している。

まず、図 1 で示した ToDo リスト表示画面（Read）について解説する。/todo に GET メソッドでアクセスすると、TodoListsController の index メソッドが呼び出される。index メソッドは、データベースから ToDo リストを全件取得し、View に渡す。View は受け取ったデータを HTTP レスポンスとしてブラウザに返し、図 1

が表示される。図 1 に表示されているタスクは、Seeder で投入した初期値である。

次に、新規登録（Create）について解説する。図 1 の新規登録ボタンを押して、/todo/create に GET メソッドでアクセスすると TodoListsController の create メソッドが呼びだされ、新規登録用のフォーム画面（図 4）が表示される。入力フォームにタスクを入力し、新規追加ボタンを押すと、POST メソッドにより/todo にデータが渡され、TodoListsController の store メソッドが呼び出される。呼び出された store メソッド内では、入力したタスクが todo_lists テーブルに登録される機能を実装している。タスクの登録完了後は、図 1 で示した ToDo リスト表示画面の View が表示される。

編集（Update）と削除（Delete）についても同様に、それぞれの機能に対応した TodoListsController 内のメソッドが呼び出され、処理が行われる。

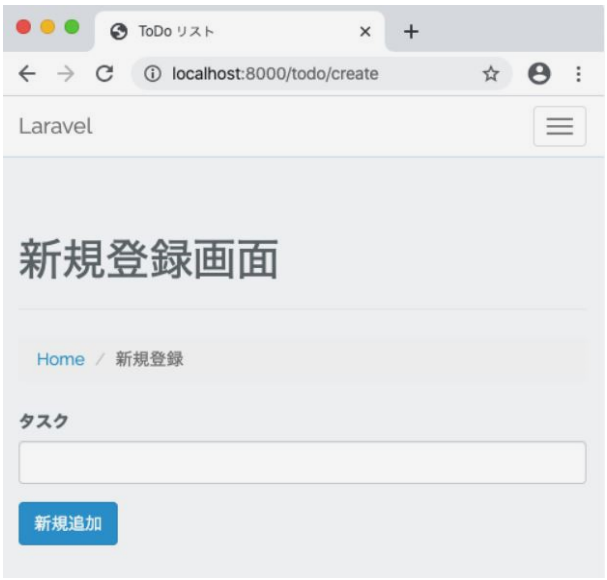

図 4. 新規登録画面

3.6　認証機能

Laravel では、ローカルのデータベースを用いた認証機能を標準で提供している。今回、特定の利用者のみが ToDo リストアプリケーションにアクセスできるようにするため、認証機能を実装した。

下記の認証機能実装用コマンドを実行することで、ログイン画面とユーザ登録に必要な View、Controller 及びルート定義が自動で生成される。

```
$ php artisan make:auth
```

次に、TodoListsController のコンストラクタに“ $this->middleware('auth'); ”と記述し、TodoListsController 内の全ての処理に対して認証をかけた。認証をかけた後、/todo にアクセスすると、認証が行われていないため、ToDo リストにアクセスできず、ログイン画面に転送されることを確認した。Middleware とは、アプリケーションへ送信される HTTP リクエストに対し、フィルタリングを行う仕組みである。auth Middleware は、認証済か否かの判断を行う。

TodoListsController に関連するルート定義一覧を表 2 に示す。TodoListsController に認証をかけたため、「Middleware」の項目に「auth」が表示されている。

表 2. ルート定義一覧

Method	URL	Name	Action	Middleware
POST	todo	todo.store	App¥Http¥Controllers ¥TodoListsController@store	web,auth
GET | HEAD	todo	todo.index	App¥Http¥Controllers ¥TodoListsController@index	web,auth
GET | HEAD	todo/create	todo.create	App¥Http¥Controllers ¥TodoListsController@create	web,auth
DELETE	todo/{todo}(*)	todo.destroy	App¥Http¥Controllers ¥TodoListsController@destroy	web,auth
PUT | PATCH	todo/{todo}(*)	todo.update	App¥Http¥Controllers ¥TodoListsController@update	web,auth
GET | HEAD	todo/{todo}(*)	todo.show	App¥Http¥Controllers ¥TodoListsController@show	web,auth
GET | HEAD	todo/{todo}(*) /edit	todo.edit	App¥Http¥Controllers ¥TodoListsController@edit	web,auth

(*) todo_lists テーブルの「id」が入る。例えば「id」が 10 番で登録されたタスクを削除した場合、HTTP レスポンスとしてブラウザに返る URL が /todo/10 となる。

4 まとめ

ToDo リストアプリケーションの開発を通して、Laravel の基本的な仕組みや操作手順を理解することができた。また、フレームワークは、コーディングルールの統一化に役立つだけでなく、開発時間の短縮に繋がることも実感できた。さらに、MVC モデルやデータベース連携、CRUD 操作、認証機能を実際のアプリケーション開発に適用することで、これらに関する知識を深めることができたことも有意義であった。今後は、Laravel を用いた開発のルール整備を行い Web アプリケーション開発・運用の効率化を実現したい。

参考文献

[1] PHP フレームワーク Laravel Web アプリケーション開発 バージョン 5.5LTS 対応
竹澤 有貴、栗生 和明、新原 雅司、大村 創太郎【共著】

[2] Laravel5.5 [https://readouble.com/laravel/5.5/ja/]