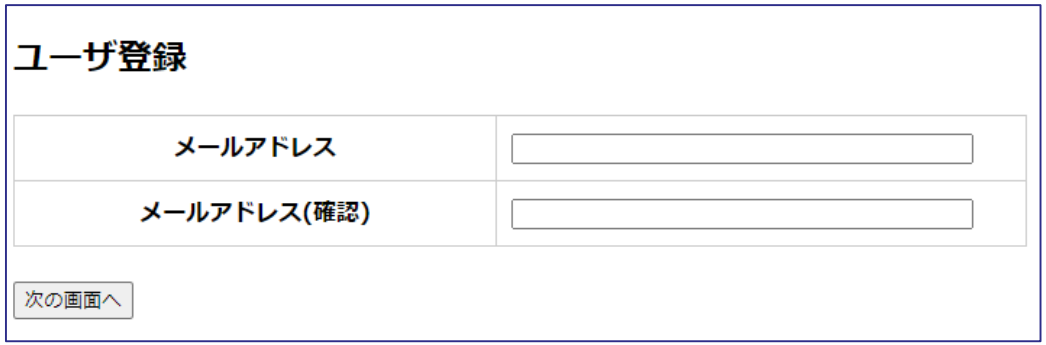
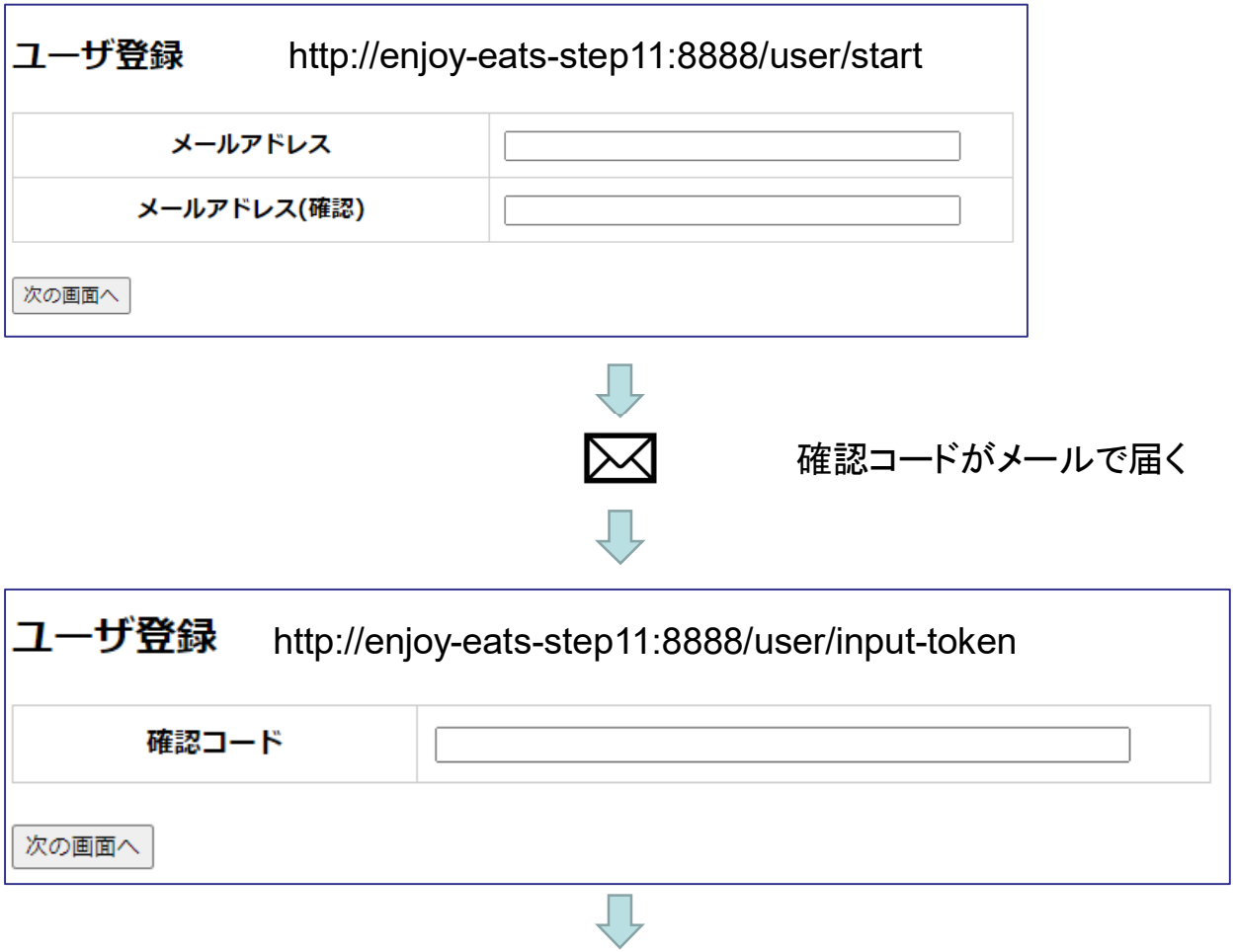
本ステップでおこなうこと

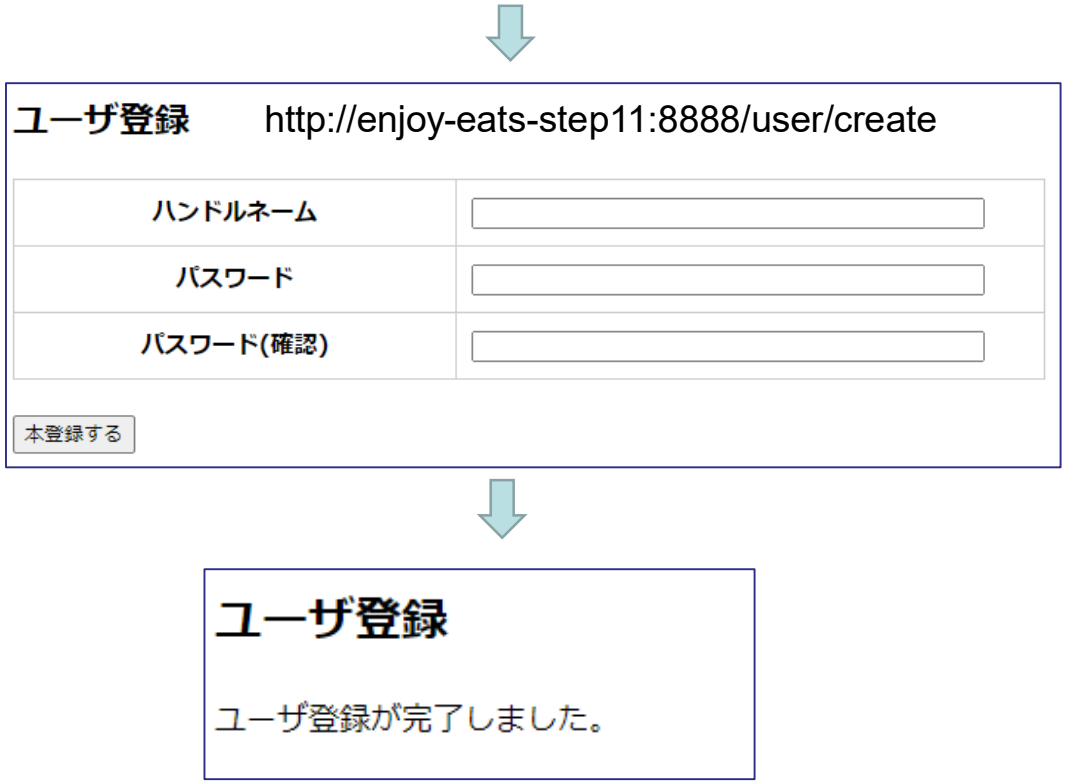
ユーザ登録ページを実装します。



ユーザ登録の流れ(1)

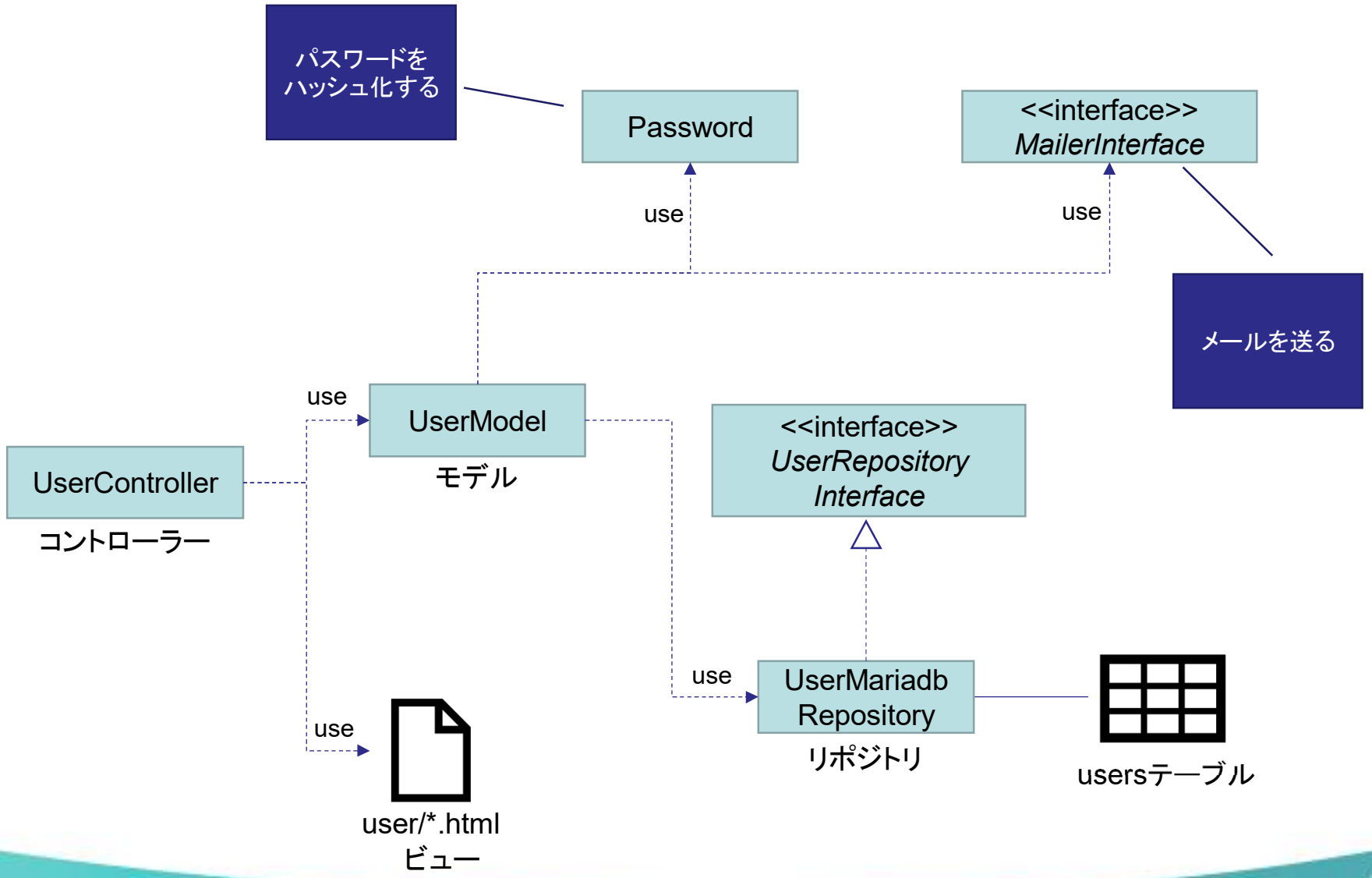

ユーザ登録
http://enjoy-eats-step11:8888/user/start
メールアドレス
メールアドレス(確認)
次の画面へ
確認コードがメールで届く
ユーザ登録
http://enjoy-eats-step11:8888/user/input-token
確認コード
次の画面へ

ユーザ登録の流れ(2)



http://enjoy-eats-step11:8888/user/created

本ステップのクラス構成



本ステップの変更ファイル一覧

●追加したファイル

- app/Models/UserModel.php
  → ユーザ登録ページ用のモデルクラス

- app/Repositories/UserMariadbRepository.php
  → usersテーブルに対応するリポジトリクラス

- app/Repositories/UserRepositoryInterface.php
  → usersテーブルに対応するリポジトリクラスのインターフェース

- app/Modules/Views/user/*.html
  → 各画面のHTMLファイル

- app/Libs/Password.php
  → パスワードハッシュ化のための共通クラス

本ステップの変更ファイル一覧

●変更したファイル

- app/Libs/Mailer/SwiftMailSender.php
  → メール送信処理を実装した

- app/Modules/User/Controllers/UserController.php

補足説明(1) - パスワード強度

- 辞書攻撃やリバートブルートフォース攻撃などへの対処のために、単純なパスワードを登録させないことが実アプリケーションでは必要になります。

- 単純なパスワードを登録させないための対策として、以下のようなものがあります:
  - 最低文字数をチェックする
  - 字種(大文字・小文字・数字・記号)をなるべく多く使わせる
  - bjeavons/zxcvbn-phpなどのパッケージを使い、強度を測る

補足説明(2) -
重複するメールアドレスを登録させない工夫

- UserModelクラスには、重複するメールアドレスを登録させないための
プログラム処理があります。

```
if ($datas['mail'] && $this->isMailExists($datas['mail'])) {
    $errors[] = "すでに本登録済のメールアドレスです。";
}
```

- ただ、バグや脆弱性により、上記のチェックをくぐり抜けられる可能性が
絶対にないとはいえません。

- このようなときに役立つのが、データベースのUNIQUE制約です。最悪
の場合でも、この制約が重複を防いでくれます。

```
CREATE TABLE users (
    id          SERIAL          PRIMARY KEY,
    (中略)
    mail        VARCHAR(190)    NOT NULL UNIQUE,
    password    VARCHAR(190)    NOT NULL'
);
```

参考情報

- PHP本格入門(上)
  「4-11 データを安全に取り扱う - ハッシュ化と暗号化」