

I. CONTROLLERS

DashboardController.php

```
app > Http > Controllers > DashboardController.php
1  <?php
2
3
4
5  namespace App\Http\Controllers;
6
7  use Illuminate\Http\Request;
8
9  class DashboardController extends Controller
10 {
11     public function index()
12     {
13         return view('dashboard');
14     }
15
16     public function show($id)
17     {
18         return view('user-dashboard', ['userId' => $id]); // in resources/views/user-dashboard.blade.php
19     }
20 }
```

Purpose:

The DashboardController is responsible for handling and displaying the general user dashboard as well as user-specific dashboards. It offers two main functionalities: loading the general dashboard view for authenticated users and showing a personalized dashboard for a specific user, identified by their id.

Methods:

- **index()** - Loads and displays the general dashboard view for authenticated users.
- **show(\$id)** - Displays the user-specific dashboard for a particular user, identified by the id parameter.

UserDashboardController.php

```
app > Http > Controllers > UserDashboardController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class UserDashboardController extends Controller
8  {
9      /**
10       * Show the user dashboard.
11       *
12       * @param int $userId
13       * @return \Illuminate\View\View
14       */
15     public function show($userId)
16     {
17         return view('user-dashboard', ['userId' => $userId]);
18     }
19 }
```

Purpose:

The UserDashboardController is responsible for managing the user-specific dashboard page. It is designed to show personalized content or data based on the user's ID passed as a parameter.

Methods:

show(\$userId)

Purpose: Displays the user dashboard for a specific user, identified by the userId parameter.

Functionality: This method retrieves the userId passed as a parameter and passes it to the user-dashboard view.

Parameter Handling:

\$userId: The show method accepts a userId as a parameter, which is used to load personalized content related to that specific user. This parameter is passed from the route, and it helps identify which user's data should be displayed on the dashboard.

HomeController.php

```
app > Http > Controllers > HomeController.php
1  <?php
2
3
4
5
6
7  namespace App\Http\Controllers;
8
9  class HomeController extends Controller
10 {
11     public function index()
12     {
13         return view('homepage'); // in resources/views/home.blade.php
14     }
15 }
```

Purpose:

The HomeController is responsible for handling the logic for the homepage of the application. Its main purpose is to process requests that target the homepage of the site and then render the appropriate view, which in this case is the homepage view located at resources/views/homepage.blade.php.

Methods:

index():

Purpose: This method is the default method for handling requests to the homepage of the site. It loads and returns the homepage view.

Functionality: When a request is made to the route that this controller is assigned to (usually the root URL /), the index() method is invoked. The method simply returns a view (homepage) to be rendered in the browser.

ProfileController.php

```
app > Http > Controllers > ProfileController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use App\Http\Requests\ProfileUpdateRequest;
6  use Illuminate\Http\RedirectResponse;
7  use Illuminate\Http\Request;
8  use Illuminate\Support\Facades\Auth;
9  use Illuminate\Support\Facades\Redirect;
10 use Illuminate\View\View;
11
12 class ProfileController extends Controller
13 {
14     /**
15      * Display the user's profile form.
16      */
17     public function edit(Request $request): View
18     {
19         return view('profile.edit', [
20             'user' => $request->user(),
21         ]);
22     }
23
24     /**
25      * Update the user's profile information.
26      */
27     public function update(ProfileUpdateRequest $request): RedirectResponse
28     {
29         $request->user()->fill($request->validated());
30
31         if ($request->user()->isDirty('email')) {
32             $request->user()->email_verified_at = null;
33         }
34
35         $request->user()->save();
36
37         return Redirect::route('profile.edit')->with('status', 'profile-updated');
38     }
39
40     /**
41      * Delete the user's account.
42      */
43     public function destroy(Request $request): RedirectResponse
44     {
45         $request->validateWithBag('userDeletion', [
46             'password' => ['required', 'current_password'],
47         ]);
48
49         $user = $request->user();
50
51         Auth::logout();
52
53         $user->delete();
54
55         $request->session()->invalidate();
56         $request->session()->regenerateToken();
57
58         return Redirect::to('/');
59     }
60 }
```

Purpose:

The ProfileController is responsible for handling actions related to the user's profile. It allows users to view and update their profile information, as well as delete their account if they choose to.

Methods:

- **edit(Request \$request): View**
 - Displays the user's profile form, where they can view and edit their profile details.

Functionality: This method is responsible for showing the profile editing form. It retrieves the currently authenticated user and passes the user object to the profile.edit view so the user can edit their profile.

- **update(ProfileUpdateRequest \$request): RedirectResponse**
 - Updates the authenticated user's profile information.

Functionality: This method is used to handle the updating of the user's profile details, such as name and email. It validates the request data using the ProfileUpdateRequest, updates the user's profile in the database, and handles the email verification logic (if the email has changed). After saving the updated information, it redirects the user back to the edit profile page with a success message.

- **destroy(Request \$request): RedirectResponse**
 - Deletes the authenticated user's account after confirming the current password.

Functionality: This method handles the deletion of the user's account. It first validates that the password entered by the user matches the current password. After validation, the user's account is deleted from the database, and the user is logged out. The session is invalidated to prevent the user from accessing the site after deletion. Finally, the user is redirected to the homepage.

Parameter Handling:

- **edit(Request \$request):**

The edit method takes a Request parameter, which is used to access the authenticated user (\$request->user()). This parameter is passed automatically by Laravel and represents the current HTTP request, including user data.

- **update(ProfileUpdateRequest \$request):**

The update method takes a ProfileUpdateRequest parameter, which is a custom request class that handles validation for updating the user's profile. It ensures that the incoming data is valid before updating the user's information.

- **destroy(Request \$request):**

The destroy method also takes a Request parameter to access the authenticated user and their session data. The password input field is validated to ensure the user has entered the correct password before their account is deleted.

Explanation of Key Features:

- **ProfileUpdateRequest:**

This is a custom request class used to validate the data being submitted when the user updates their profile. It ensures that the input meets the required format (e.g., valid email, correct password) before proceeding with the update.

- **Auth::logout():**

This function logs the user out of the application. It's used in the destroy method after the user account is deleted to ensure they can no longer access the site.

- **Redirect::route('profile.edit'):**

This is used to redirect the user back to the profile edit page with a success message after their profile is updated.

- **update(ProfileUpdateRequest \$request):**

The update method takes a ProfileUpdateRequest parameter, which is a custom request class that handles validation for updating the user's profile. It ensures that the incoming data is valid before updating the user's information.

II. ROUTES

Web.php

```
routes > web.php
1  <?php
2  use App\Http\Controllers\HomeController;
3  use App\Http\Controllers\UserDashboardController;
4  use App\Http\Controllers\ProfileController;
5  use Illuminate\Support\Facades\Route;
6
7  // Homepage Route
8  Route::get('/', [HomeController::class, 'index']);
9
10 // Dashboard Route
11 Route::get('/dashboard', function () {
12     return view('dashboard');
13 })->middleware(['auth', 'verified'])->name('dashboard');
14
15 // Profile Routes
16 Route::middleware('auth')->group(function () {
17     Route::get('/profile', [ProfileController::class, 'edit'])->name('profile.edit');
18     Route::patch('/profile', [ProfileController::class, 'update'])->name('profile.update');
19     Route::delete('/profile', [ProfileController::class, 'destroy'])->name('profile.destroy');
20 });
21
22 // User Dashboard Route
23 Route::get('/user-dashboard/{userId}', [UserDashboardController::class, 'show'])
24     ->middleware(['auth'])
25     ->name('user-dashboard');
26
27 require __DIR__.'/auth.php';
```

/: Handled by the *HomeController@index* method

/dashboard: Uses an anonymous function (direct view return), protected by the auth and verified middleware.

/profile: Handled by *ProfileController@edit*, *ProfileController@update*, and *ProfileController@destroy* methods with **auth** middleware.

/user-dashboard/{userId}: Handled by *UserDashboardController@show* with **auth** middleware.

III. BLADE FILES

Dashboard.blade.php

```
resources > views > dashboard.blade.php
1 <x-app-layout>
2   <x-slot name="header">
3     <h2 class="font-semibold text-xl text-gray-800 leading-tight">
4       {{ __('Dashboard') }}
5     </h2>
6   </x-slot>
7
8   <div class="py-12">
9     <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
10      <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
11        <div class="p-6 text-gray-900">
12          {{ __('You're logged in!') }}
13
14          <!-- Check if user is authenticated -->
15          @if (auth()->check())
16            <a href="{{ route('user-dashboard', ['userId' => auth()->user()->id]) }}"
17              class="text-blue-500 hover:underline mt-4 inline-block">
18              Go to User Dashboard
19            </a>
20          @else
21            <p>Please log in to access your dashboard.</p>
22          @endif
23        </div>
24      </div>
25    </div>
26  </div>
27 </x-app-layout>
```

Purpose:

- **Authenticated User:** If the user is authenticated (auth()->check()), a link to the **User Dashboard** is displayed. The URL is dynamically generated using the route helper, passing the authenticated user's ID as a parameter (auth()->user()->id).
- **Non-authenticated User:** If the user is not authenticated, a message prompts them to log in order to access the dashboard.

UserDashboard.blade.php

```
resources > views > user-dashboard.blade.php
1 <x-app-layout>
2   <x-slot name="header">
3     <h2 class="font-semibold text-xl text-gray-800 leading-tight">
4       {{ __('User Dashboard') }}
5     </h2>
6   </x-slot>
7
8   <div class="py-12">
9     <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
10      <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
11        <div class="p-6 text-gray-900">
12          <h1>Welcome to the user-dashboard, {{ $userId }}!</h1>
13
14          <p>This page is under development...</p>
15
16          <a href="{{ route('dashboard') }}" class="inline-block border border-blue-500 text-blue-700 font-semibold px-4 py-2 rounded hover:bg-blue-500 hover:text-white transition-colors duration-200 mt-4">
17            Back to Dashboard
18          </a>
19
20          <form action="{{ route('logout') }}" method="POST" style="margin-top: 10px;">
21            @csrf
22            <button type="submit" class="bg-red-500 text-white px-4 py-2 rounded hover:bg-red-600 transition-colors duration-200">
23              Logout
24            </button>
25          </form>
26        </div>
27      </div>
28    </div>
29  </div>
30 </div>
31 </x-app-layout>
```

The **welcome message** dynamically greets the user using the **{{ \$userId }}** placeholder, which would be replaced by the actual user ID passed from the controller.

Login.blade.php

```
resources > views > auth > login.blade.php
1 <x-guest-layout>
2 <!-- Session Status -->
3 <x-auth-session-status class="mb-4" :status="session('status')" />
4
5 <form method="POST" action="{{ route('login') }}">
6     @csrf
7
8     <!-- Email Address -->
9     <div>
10         <x-input-label for="email" :value="__('Email')" />
11         <x-text-input id="email" class="block mt-1 w-full" type="email" name="email" :value="old('email')" required autofocus autocomplete="username" />
12         <x-input-error :messages="$errors->get('email')" class="mt-2" />
13     </div>
14
15     <!-- Password -->
16     <div class="mt-4">
17         <x-input-label for="password" :value="__('Password')" />
18
19         <x-text-input id="password" class="block mt-1 w-full"
20             type="password"
21             name="password"
22             required autocomplete="current-password" />
23
24         <x-input-error :messages="$errors->get('password')" class="mt-2" />
25     </div>
26
27     <!-- Remember Me -->
28     <div class="block mt-4">
29         <label for="remember_me" class="inline-flex items-center">
30             <input id="remember_me" type="checkbox" class="rounded border-gray-300 text-indigo-600 shadow-sm focus:ring-indigo-500 name="remember" />
31             <span class="ms-2 text-sm text-gray-600">{{ __('Remember me') }}
32         </label>
33     </div>
34
35     <div class="flex items-center justify-end mt-4">
36         @if (Route::has('password.request'))
37             <a class="underline text-sm text-gray-600 hover:text-gray-900 rounded-md focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500" href="{{ route('password.request') }}">
38                 {{ __('Forgot your password?') }}
39             </a>
40         @endif
41
42         <x-primary-button class="ms-3">
43             {{ __('Log in') }}
44         </x-primary-button>
45     </div>
46 </form>
47 </x-guest-layout>
```

Session Status Component: Displays any session status messages like login success or errors.

Action Links and Button:

- **Forgot Password:** A link to reset the password.
- **Login Button:** Submits the login form.

Register.blade.php

```
resources > views > auth > register.blade.php
1 <x-guest-layout>
2 <form method="POST" action="{{ route('register') }}">
3     @csrf
4
5     <!-- Name -->
6     <div>
7         <x-input-label for="name" :value="__('Name')" />
8         <x-text-input id="name" class="block mt-1 w-full" type="text" name="name" :value="old('name')" required autofocus autocomplete="name" />
9         <x-input-error :messages="$errors->get('name')" class="mt-2" />
10     </div>
11
12     <!-- Email Address -->
13     <div class="mt-4">
14         <x-input-label for="email" :value="__('Email')" />
15         <x-text-input id="email" class="block mt-1 w-full" type="email" name="email" :value="old('email')" required autocomplete="username" />
16         <x-input-error :messages="$errors->get('email')" class="mt-2" />
17     </div>
18
19     <!-- Password -->
20     <div class="mt-4">
21         <x-input-label for="password" :value="__('Password')" />
22
23         <x-text-input id="password" class="block mt-1 w-full"
24             type="password"
25             name="password"
26             required autocomplete="new-password" />
27
28         <x-input-error :messages="$errors->get('password')" class="mt-2" />
29     </div>
30
31     <!-- Confirm Password -->
32     <div class="mt-4">
33         <x-input-label for="password_confirmation" :value="__('Confirm Password')" />
34
35         <x-text-input id="password_confirmation" class="block mt-1 w-full"
36             type="password"
37             name="password_confirmation" required autocomplete="new-password" />
38
39         <x-input-error :messages="$errors->get('password_confirmation')" class="mt-2" />
40     </div>
41
42     <div class="flex items-center justify-end mt-4">
43         <a class="underline text-sm text-gray-600 hover:text-gray-900 rounded-md focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500" href="{{ route('login') }}">
44             {{ __('Already registered?') }}
45         </a>
46
47         <x-primary-button class="ms-4">
48             {{ __('Register') }}
49         </x-primary-button>
50     </div>
51 </form>
52 </x-guest-layout>
```

Form: The registration form (<form>) uses the POST method to submit the data to the register route.

Action Links and Button:

- **Login Link:** Directs users to the login page if they already have an account.
- **Register Button:** Submits the form to create a new user.

Homepage.blade.php

```
resources > views > homepage.blade.php
1 <x-app-layout>
2   <x-slot name="header">
3     <h2 class="font-semibold text-xl text-gray-800 leading-tight">
4       {{ __('Home') }}
5     </h2>
6   </x-slot>
7
8   <div class="py-12">
9     <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
10      <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
11        <div class="p-6 text-gray-900">
12          <h1>WELCOME</h1>
13          <p>Enjoy your visit here.</p>
14
15          @auth
16            <p>You are logged in! Go to your <a href="{{ route('dashboard') }}" class="inline-block border border-blue-500 text-blue-700 font-semibold px-4 py-2 rounded hov
17          @else
18            <p>Please <a href="{{ route('login') }}" class="inline-block border border-blue-500 text-blue-700 font-semibold px-4 py-2 rounded hover:bg-blue-500 hover:text-w
19          @endauth
20        </div>
21      </div>
22    </div>
23  </div>
24 </x-app-layout>
25
```

Purpose: This conditional section checks if the user is authenticated using the @auth directive.

- If **authenticated**, it displays a message confirming login and a link to the dashboard.
- If **not authenticated**, it prompts the user to log in or register, with links that lead to the login and registration routes.

Delete-user-form.blade.php

```
resources > views > profile > partials > delete-user-form.blade.php
1 <section class="space-y-6">
2   <header>
3     <h2 class="text-lg font-medium text-gray-900">
4       {{ __('Delete Account') }}
5     </h2>
6
7     <p class="mt-1 text-sm text-gray-600">
8       {{ __('Once your account is deleted, all of its resources and data will be permanently deleted. Before deleting your account, please download any data or information that you wish to retain.') }}
9     </p>
10   </header>
11
12   <x-danger-button
13     x-data=""
14     x-on:click.prevent="$dispatch('open-modal', 'confirm-user-deletion')"
15   >{{ __('Delete Account') }}</x-danger-button>
16
17   <x-modal name="confirm-user-deletion" :show="$errors->userDeletion->isEmpty()" focusable>
18     <form method="post" action="{{ route('profile.destroy') }}" class="p-6">
19       @csrf
20       @method('delete')
21
22       <h2 class="text-lg font-medium text-gray-900">
23         {{ __('Are you sure you want to delete your account?') }}
24       </h2>
25
26       <p class="mt-1 text-sm text-gray-600">
27         {{ __('Once your account is deleted, all of its resources and data will be permanently deleted. Please enter your password to confirm you would like to permanently delete your account.') }}
28       </p>
29
30       <div class="mt-6">
31         <x-input-label for="password" value="{{ __('Password') }}" class="sr-only" />
32
33         <x-text-input
34           id="password"
35           name="password"
36           type="password"
37           class="mt-1 block w-3/4"
38           placeholder="{{ __('Password') }}"
39         />
40
41         <x-input-error :messages="$errors->userDeletion->get('password')" class="mt-2" />
42       </div>
43
44       <div class="mt-6 flex justify-end">
45         <x-secondary-button x-on:click="$dispatch('close')">
46           {{ __('Cancel') }}
47         </x-secondary-button>
48
49         <x-danger-button class="ms-3">
50           {{ __('Delete Account') }}
51         </x-danger-button>
52       </div>
53     </form>
54   </x-modal>
55 </section>
```

Purpose: This section allows the user to delete their account from the platform.

Update-password-form.blade.php

```
resources > views > profile > partials > update-password-form.blade.php
1 <section>
2 <header>
3 <h2 class="text-lg font-medium text-gray-900">
4     {{ __('Update Password') }}
5 </h2>
6
7 <p class="mt-1 text-sm text-gray-600">
8     {{ __('Ensure your account is using a long, random password to stay secure.') }}
9 </p>
10 </header>
11
12 <form method="post" action="{{ route('password.update') }}" class="mt-6 space-y-6">
13     @csrf
14     @method('put')
15
16     <div>
17         <x-input-label form="update_password_current_password" :value="__('Current Password')"/>
18         <x-text-input id="update_password_current_password" name="current_password" type="password" class="mt-1 block w-full" autocomplete="current-password" />
19         <x-input-error :messages="$errors->updatePassword->get('current_password')"/>
20     </div>
21
22     <div>
23         <x-input-label form="update_password_password" :value="__('New Password')"/>
24         <x-text-input id="update_password_password" name="password" type="password" class="mt-1 block w-full" autocomplete="new-password" />
25         <x-input-error :messages="$errors->updatePassword->get('password')"/>
26     </div>
27
28     <div>
29         <x-input-label form="update_password_password_confirmation" :value="__('Confirm Password')"/>
30         <x-text-input id="update_password_password_confirmation" name="password_confirmation" type="password" class="mt-1 block w-full" autocomplete="new-password" />
31         <x-input-error :messages="$errors->updatePassword->get('password_confirmation')"/>
32     </div>
33
34     <div class="flex items-center gap-4">
35         <x-primary-button>{{ __('Save') }}</x-primary-button>
36
37         @if (session('status') === 'password-updated')
38             <p>
39                 <x-data="{ show: true }">
40                     <x-show="show">
41                         <x-transition>
42                             <x-init="setTimeout(() => show = false, 2000)"
43                             class="text-sm text-gray-600"
44                         >{{ __('Saved.') }}</x-init>
45                     </x-show>
46                 </x-data>
47             </p>
48         @endif
49     </div>
50 </form>
51 </section>
```

Purpose: This section allows the user to update their account password.

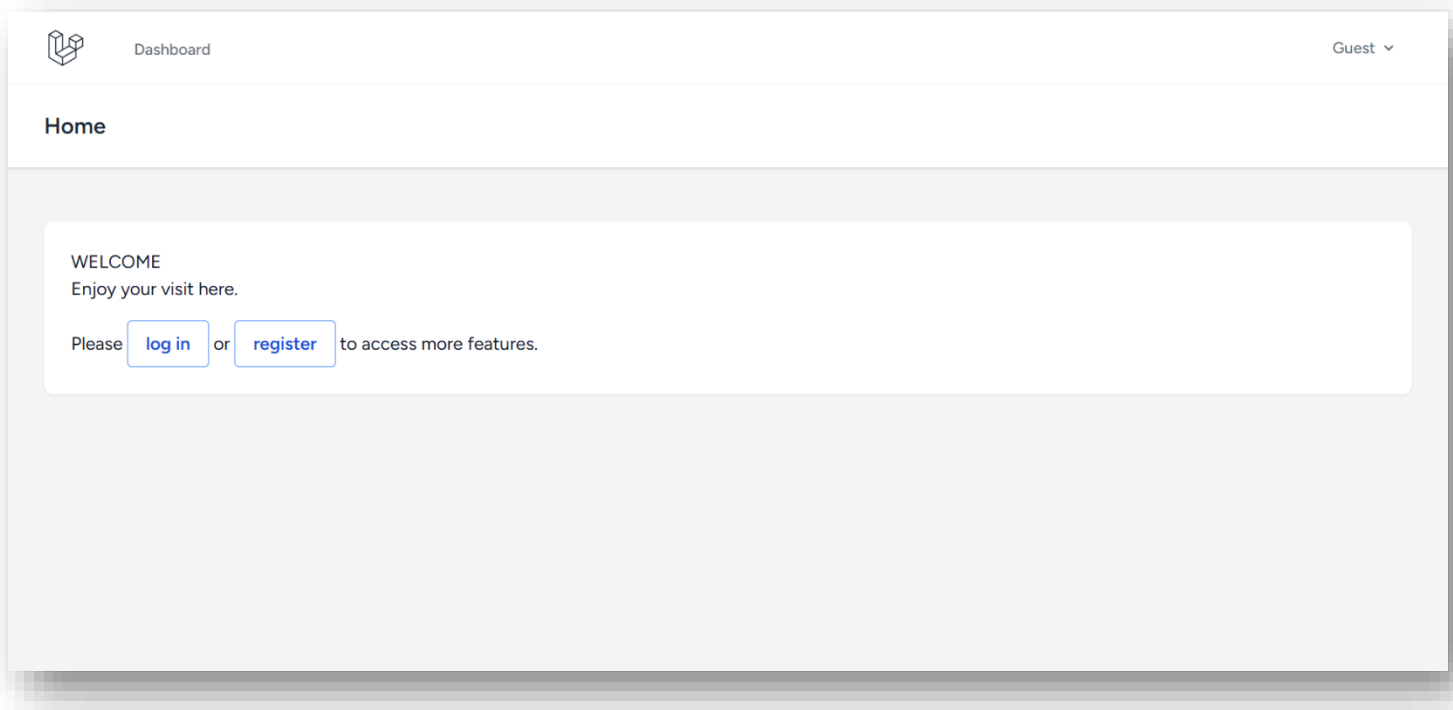
Update-profile-information-form.blade.php

```
resources > views > profile > partials > update-profile-information-form.blade.php
1 <section>
2 <header>
3 <h2 class="text-lg font-medium text-gray-900">
4     {{ __('Profile Information') }}
5 </h2>
6
7 <p class="mt-1 text-sm text-gray-600">
8     {{ __('Update your account's profile information and email address.') }}
9 </p>
10 </header>
11
12 <form id="send-verification" method="post" action="{{ route('verification.send') }}">
13     @csrf
14 </form>
15
16 <form method="post" action="{{ route('profile.update') }}" class="mt-6 space-y-6">
17     @csrf
18     @method('patch')
19
20     <div>
21         <x-input-label form="name" :value="__('Name')"/>
22         <x-text-input id="name" name="name" type="text" class="mt-1 block w-full" :value="old('name', $user->name)" required autofocus autocomplete="name" />
23         <x-input-error class="mt-2" :messages="$errors->get('name')"/>
24     </div>
25
26     <div>
27         <x-input-label form="email" :value="__('Email')"/>
28         <x-text-input id="email" name="email" type="email" class="mt-1 block w-full" :value="old('email', $user->email)" required autocomplete="username" />
29         <x-input-error class="mt-2" :messages="$errors->get('email')"/>
30
31         @if ($user instanceof Illuminate\Contracts\Auth\MustVerifyEmail && ! $user->hasVerifiedEmail())
32             <div>
33                 <p class="text-sm mt-2 text-gray-800">
34                     {{ __('Your email address is unverified.') }}
35
36                     <button form="send-verification" class="underline text-sm text-gray-600 hover:text-gray-900 rounded-md focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500">
37                         {{ __('Click here to re-send the verification email.') }}
38                     </button>
39                 </p>
40
41                 @if (session('status') === 'verification-link-sent')
42                     <p class="mt-2 font-medium text-sm text-green-600">
43                         {{ __('A new verification link has been sent to your email address.') }}
44                     </p>
45                 @endif
46             </div>
47         @endif
48     </div>
49
50     <div class="flex items-center gap-4">
51         <x-primary-button>{{ __('Save') }}</x-primary-button>
52
53         @if (session('status') === 'profile-updated')
54             <p>
55                 <x-data="{ show: true }">
56                     <x-show="show">
57                         <x-transition>
58                             <x-init="setTimeout(() => show = false, 2000)"
59                             class="text-sm text-gray-600"
60                         >{{ __('Saved.') }}</x-init>
61                     </x-show>
62                 </x-data>
63             </p>
64         @endif
65     </div>
66 </form>
67 </section>
```

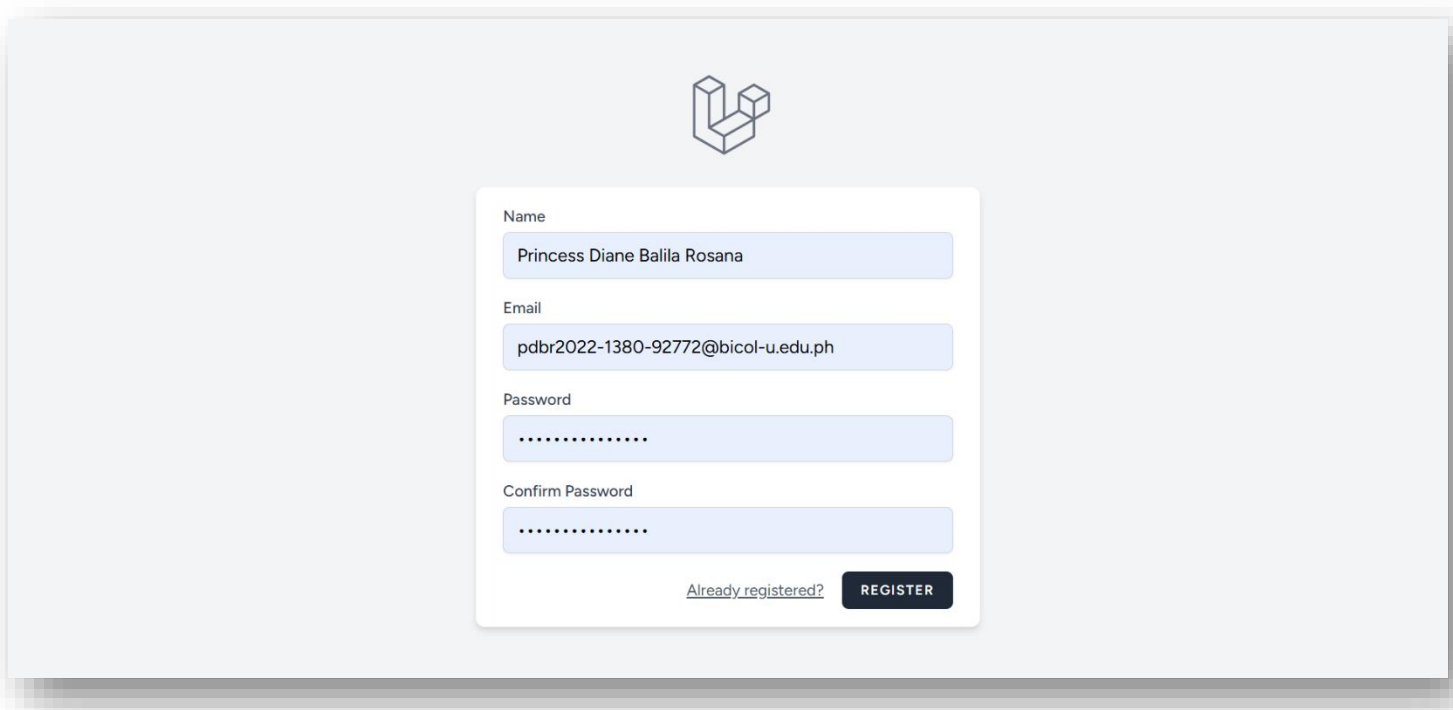
Purpose: This section allows the user to update their profile information, including their name and email address.

IV. RENDERED PAGES

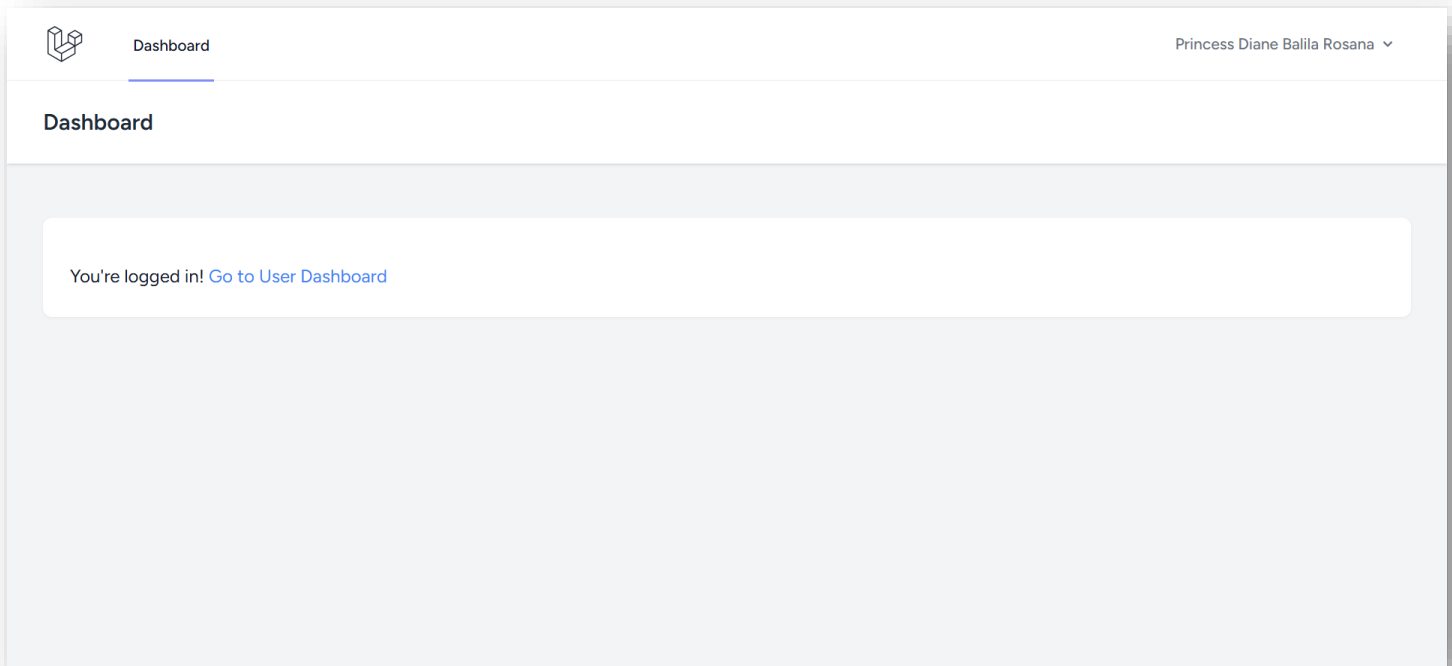
- <http://127.0.0.1:8000/>



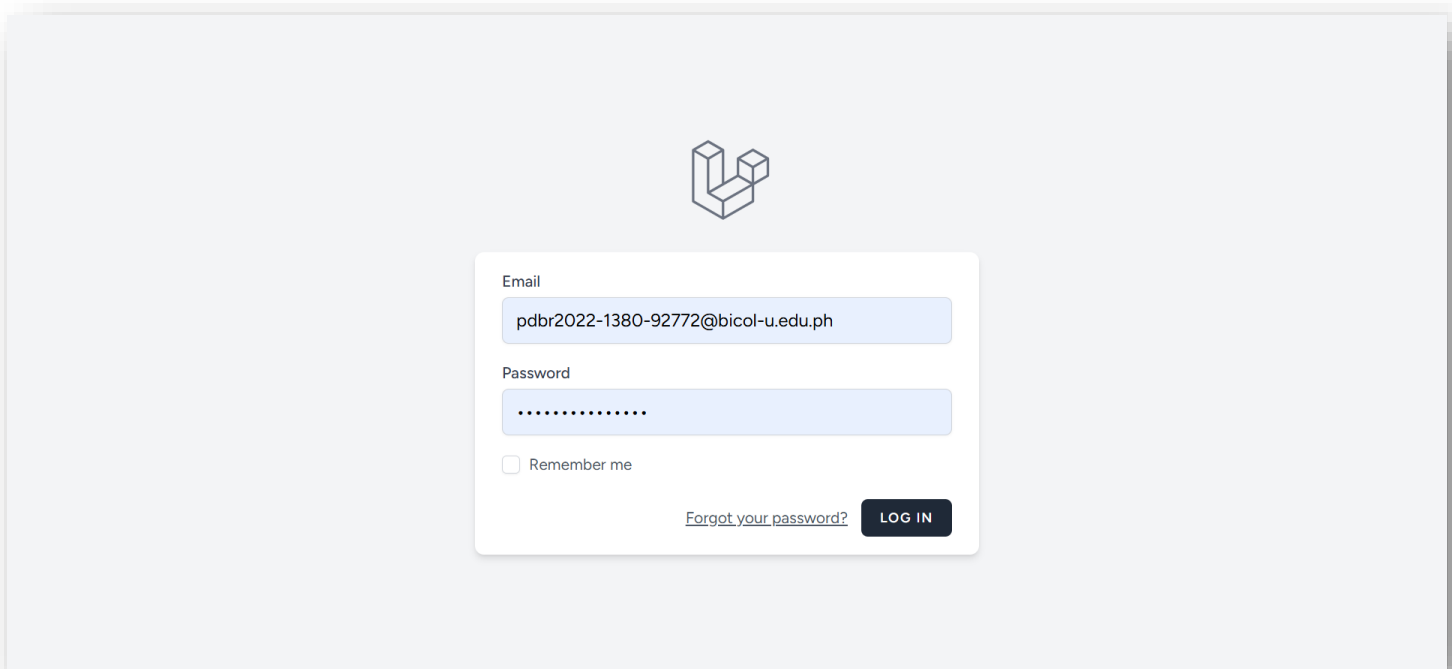
- <http://127.0.0.1:8000/register>



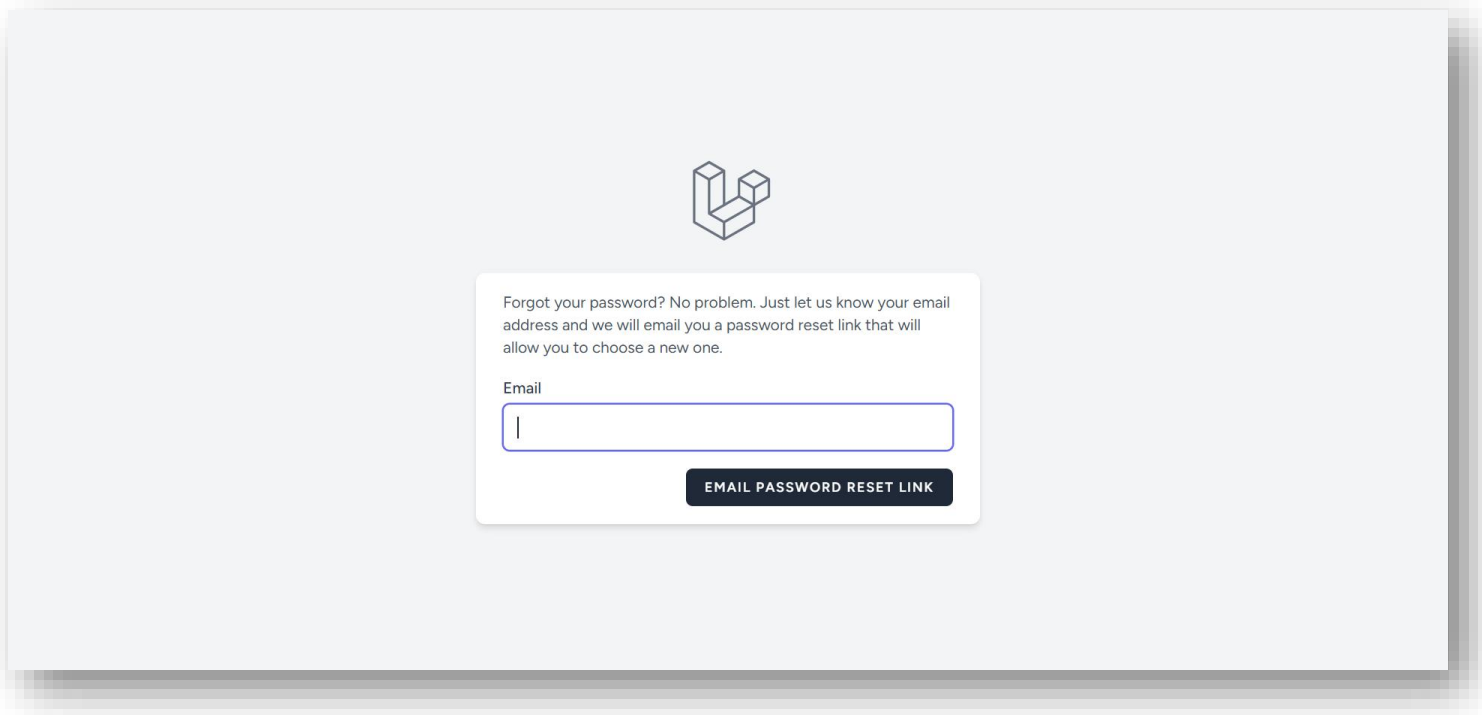
- **http://127.0.0.1:8000/dashboard**



- **http://127.0.0.1:8000/login**



- <http://127.0.0.1:8000/forgot-password>



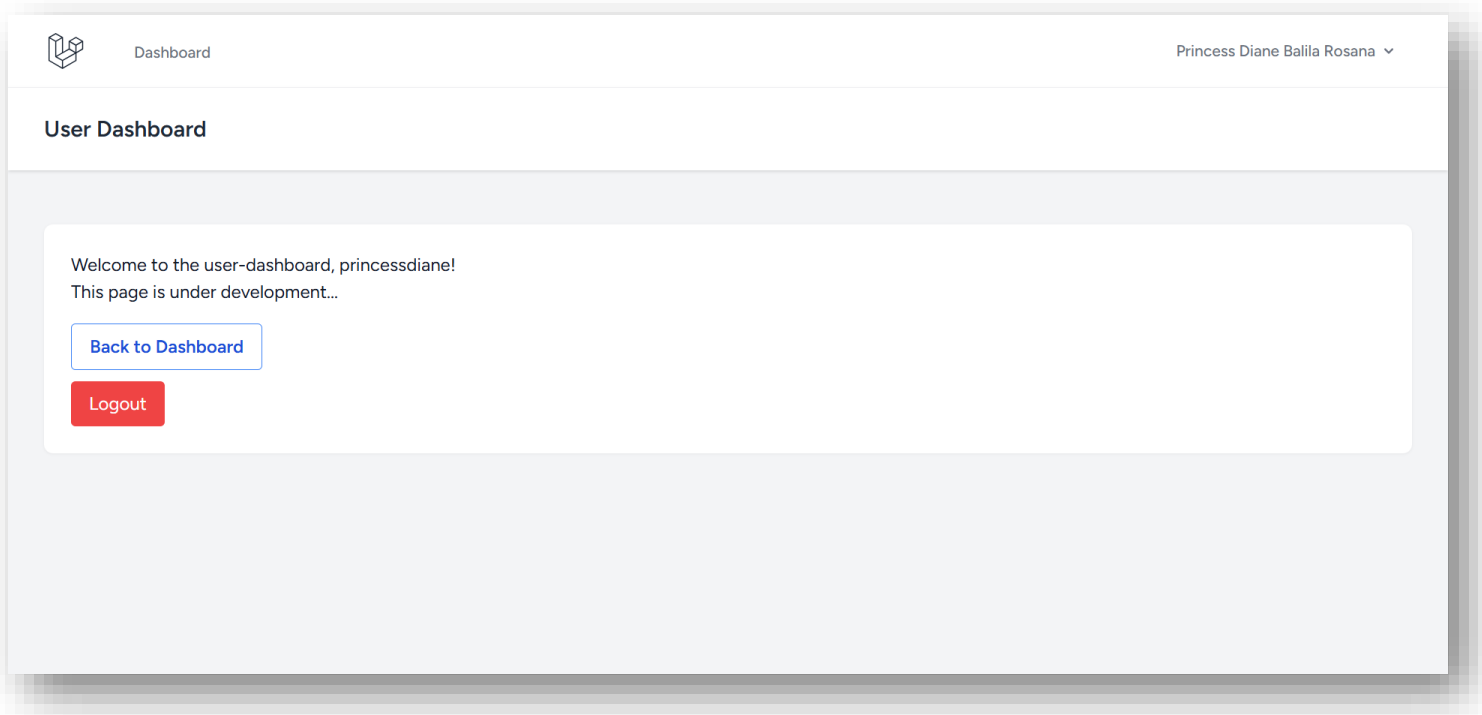
The image shows a web page for a forgot password form. At the top center is a logo consisting of three stacked cubes. Below the logo is a text box with the message: "Forgot your password? No problem. Just let us know your email address and we will email you a password reset link that will allow you to choose a new one." Below this text is a label "Email" followed by a text input field. At the bottom right of the form is a button labeled "EMAIL PASSWORD RESET LINK".

Forgot your password? No problem. Just let us know your email address and we will email you a password reset link that will allow you to choose a new one.

Email

EMAIL PASSWORD RESET LINK

- <http://127.0.0.1:8000/user-dashboard/princessdiane>



The image shows a user dashboard page. At the top left is a logo of three stacked cubes and the word "Dashboard". At the top right is the user's name "Princess Diane Balila Rosana" with a dropdown arrow. Below the header is a section titled "User Dashboard". Inside this section is a white box containing the text: "Welcome to the user-dashboard, princessdiane! This page is under development...". Below this text are two buttons: "Back to Dashboard" (blue outline) and "Logout" (red solid).

Dashboard

Princess Diane Balila Rosana ▾

User Dashboard

Welcome to the user-dashboard, princessdiane!
This page is under development...

[Back to Dashboard](#)

[Logout](#)

○ <http://127.0.0.1:8000/profile>



Dashboard

Princess Diane Balila Rosana ▾

Profile

Profile Information

Update your account's profile information and email address.

Name

Email

SAVE

Update Password

Ensure your account is using a long, random password to stay secure.

Current Password

New Password

Confirm Password

SAVE

Delete Account

Once your account is deleted, all of its resources and data will be permanently deleted. Before deleting your account, please download any data or information that you wish to retain.

DELETE ACCOUNT