

AI1	Dokumentacja projektu
Autor	Adrian Rojek, 125158
Kierunek, rok	Informatyka, II rok, st. stacjonarne (3,5-l)
Temat projektu	<i>Informator o rozkładzie autobusów</i>

Wstęp:

Bus-Info to nowoczesny system informatyczny zaprojektowany, aby usprawnić zarządzanie transportem publicznym. Aplikacja oferuje kompleksowe narzędzia dla administratorów, kierowców oraz pasażerów, ułatwiając planowanie tras, przydzielanie zadań, monitorowanie pojazdów oraz dostęp do informacji o rozkładach jazdy.

Dzięki intuicyjnemu interfejsowi użytkownika, Bus-Info zapewnia łatwy dostęp do wszystkich funkcji systemu. Administratorzy mogą zarządzać użytkownikami, trasami, przystankami oraz przypisywać trasy kierowcom. Kierowcy mają dostęp do swoich harmonogramów pracy oraz szczegółowych informacji o trasach. Pasażerowie mogą korzystać z aktualnych rozkładów jazdy oraz informacji o lokalizacji autobusów w czasie rzeczywistym.

Bus-Info to nie tylko narzędzie do zarządzania, ale również platforma komunikacji, która ułatwia przepływ informacji między wszystkimi uczestnikami systemu transportu publicznego.

Narzędzia i technologie:

Język programowania: PHP (wersja 8.2.12)

Framework backendowy: Laravel (wersja 11.8.0) <https://laravel.com/docs/11.x>

Framework frontendowy: Bootstrap <https://getbootstrap.com/>

Pakiet XAMPP: <https://www.apachefriends.org/download.html>

Menedżer zależności Composer: <https://getcomposer.org/Composer-Setup.exe>

Git (wersja 2.45.1): <https://git-scm.com/>

Środowisko programistyczne: <https://code.visualstudio.com/>

Baza danych:

1. Diagram ERD:

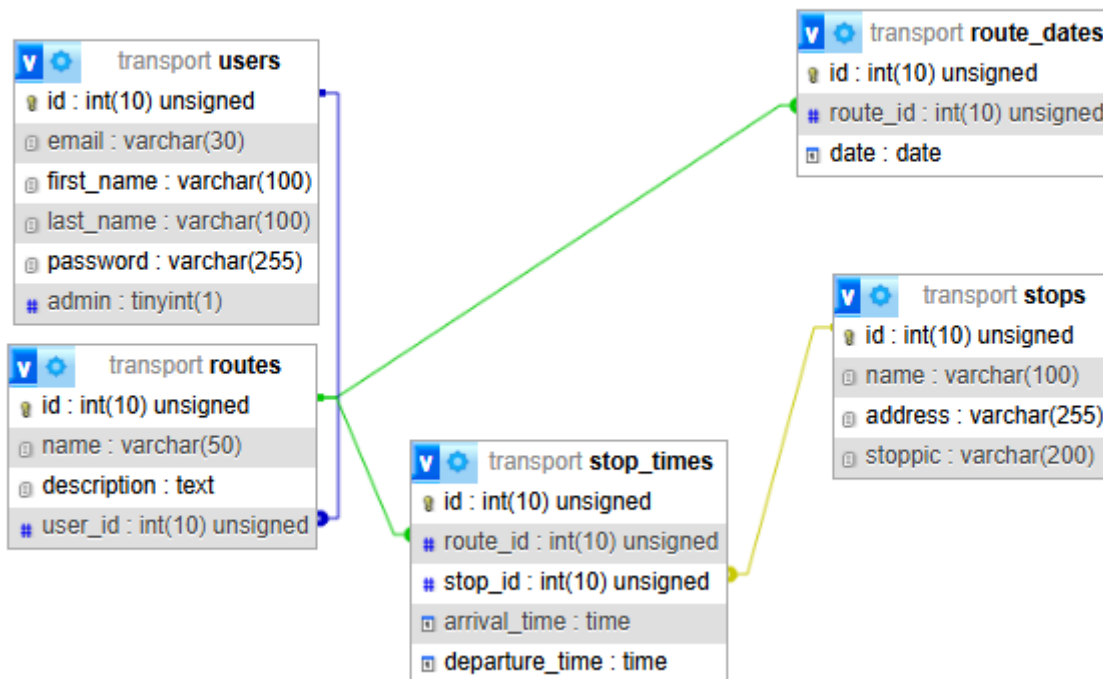


Diagram przedstawia model bazy danych dla systemu związanego z transportem publicznym. Składa się z pięciu tabel, które przechowują informacje o użytkownikach, trasach, przystankach, rozkładach jazdy oraz czasie przyjazdu i odjazdu na poszczególnych przystankach.

Relacje między tabelami:

- **transport_users** (użytkownicy transportu)
 - Tabela przechowuje informacje o użytkownikach systemu, takie jak identyfikator (id), adres e-mail (email), imię (first_name), nazwisko (last_name), hasło (password) oraz czy użytkownik jest administratorem (admin).
 - Tabela "transport_users" jest połączona z tabelą "transport_routes" relacją jeden do wielu, co oznacza, że jeden użytkownik może utworzyć wiele tras.
- **transport_routes** (trasy transportu)
 - Tabela przechowuje informacje o trasach, takie jak identyfikator (id), nazwa (name), opis (description) oraz identyfikator użytkownika, który utworzył trasę (user_id).
 - Tabela "transport_routes" jest połączona z tabelą "transport_stop_times" relacją jeden do wielu, co oznacza, że jedna trasa może mieć wiele przystanków z określonym czasem przyjazdu i odjazdu.
 - Tabela "transport_routes" jest połączona z tabelą "transport_route_dates" relacją jeden do wielu, co oznacza, że jedna trasa może obowiązywać w wielu różnych datach.
- **transport_route_dates** (daty tras)
 - Tabela przechowuje informacje o datach, w których obowiązuje dana trasa. Zawiera identyfikator (id), identyfikator trasy (route_id) oraz datę (date).

- **transport_stops** (przystanki transportu)
 - Tabela przechowuje informacje o przystankach, takie jak identyfikator (id), nazwa (name), adres (address) oraz zdjęcie (stoppic).
 - Tabela "transport_stops" jest połączona z tabelą "transport_stop_times" relacją jeden do wielu, co oznacza, że jeden przystanek może być częścią wielu tras z różnym czasem przyjazdu i odjazdu.
- **transport_stop_times** (czasy przystanków)
 - Tabela przechowuje informacje o czasie przyjazdu (arrival_time) i odjazdu (departure_time) na poszczególnych przystankach dla danej trasy. Zawiera identyfikator (id), identyfikator trasy (route_id), identyfikator przystanku (stop_id) oraz czas przyjazdu i odjazdu.

Aplikacja:

2.Seedery:

Seeder `DatabaseSeeder` wywołuje inne seedery, które odpowiadają za wypełnienie poszczególnych tabel danymi.


Oto struktura seederów i ich zawartość:

1. DatabaseSeeder.php:

```
database > seeders > DatabaseSeeder.php > ...
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6
7  0 references | 0 implementations
8  class DatabaseSeeder extends Seeder
9  {
10     0 references | 0 overrides
11     public function run()
12     {
13         $this->call([
14             StopsTableSeeder::class,
15             UsersTableSeeder::class,
16             RoutesTableSeeder::class,
17             StopTimesTableSeeder::class,
18             RouteDatesTableSeeder::class,
19         ]);
20     }
21 }
```

Główny seeder, który wywołuje wszystkie pozostałe seedery. Dzięki temu, jednym poleceniem (`php artisan db:seed`) możemy wypełnić całą bazę danymi.

2. StopsTableSeeder.php:

database > seeders >  StopsTableSeeder.php > ...

```
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use Illuminate\Support\Facades\DB;
7
8  1 reference | 0 implementations
9  class StopsTableSeeder extends Seeder
10 {
11     0 references | 0 overrides
12     public function run()
13     {
14         DB::table('stops')->insert([
15             ['id'=>1, 'name'=>'RejtanaSkrzyżowanie', 'address'=>'ul.Rejtana1',
16             ['id'=>2, 'name'=>'LisaKuliRondo', 'address'=>'ul.LisaKuli15', 'stoppic'=>
17             ['id'=>3, 'name'=>'DworzecGłównyPKP', 'address'=>'ul.Grottgera2', '
18             ['id'=>4, 'name'=>'Kilara02', 'address'=>'ul.Kilara02', 'stoppic'=>
19             ['id'=>5, 'name'=>'Pl.WolnościFontanna', 'address'=>'Pl.Wolności7'
20             ['id'=>6, 'name'=>'StaroniwaCmentarz', 'address'=>'ul.Krakowska20'
21             ['id'=>7, 'name'=>'ŁukasiewiczaPętla', 'address'=>'ul.IgnacegoŁuka
22             ['id'=>8, 'name'=>'GaleriaRzeszówGłównyWejście', 'address'=>'al.Pi
23             ['id'=>9, 'name'=>'MilleniumHallParking', 'address'=>'ul.Kopisto1'
24             ['id'=>10, 'name'=>'NoweMiastoOsiedle', 'address'=>'ul.Podwisłocz
25             ['id'=>11, 'name'=>'KwiatkowskiegoSkrzyżowanie', 'address'=>'ul.Kw
26             ['id'=>12, 'name'=>'SzpitalWojewódzkiSOR', 'address'=>'ul.Szopena2
27             ['id'=>13, 'name'=>'UniwersytetRzeszowskiWydziałPrawa', 'address'=>
28             ['id'=>14, 'name'=>'CentrumWystawienniczo-KongresoweG2AArena', 'ac
29             ['id'=>15, 'name'=>'PortLotniczyRzeszów-JasionkaTerminal', 'adres
30             ['id'=>16, 'name'=>'TyczynRynek', 'address'=>'ul.Mickiewicza1', 'st
31             ['id'=>17, 'name'=>'BiażowaZamek', 'address'=>'ul.Rynek1', 'stoppic
32             ['id'=>18, 'name'=>'ŁańcutZamek', 'address'=>'Pl.Sobieskiego1', 'st
33             ['id'=>19, 'name'=>'PrzeworskDworzecPKP', 'address'=>'ul.Jagiello
34             ['id'=>20, 'name'=>'LeżajskBazylika', 'address'=>'ul.Mickiewicza2'
35             ['id'=>21, 'name'=>'GłogówMałopolskiUrządMiasta', 'address'=>'ul.F
36             ['id'=>22, 'name'=>'KolbuszowaRynek', 'address'=>'ul.ObrońcówPoko
```

Wypełnia tabelę transport_stops danymi o przystankach.

Dane obejmują nazwę przystanku, adres oraz opcjonalnie zdjęcie.

3. UsersTableSeeder.php:

database > seeders > UsersTableSeeder.php > Database\Seeders\UsersTableSeed

```
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use Illuminate\Support\Facades\DB;
7  use Illuminate\Support\Facades\Hash;
8
9  1 reference | 0 implementations
10 class UsersTableSeeder extends Seeder
11 {
12     0 references | 0 overrides
13     public function run()
14     {
15         DB::table('users')->insert([
16             ['id'=>1, 'email'=>'jannowak@email.com',
17             'first_name'=>'Jan', 'last_name'=>'Nowak',
18             'password'=>'$2y$12$NHkuo.t5nAZoaceH6EwpE.
19             JNeYdOPR2BauV/sfyTjVrP522dCb0Je', 'admin'=>1],
20             ['id'=>2, 'email'=>'john@example.com',
21             'first_name'=>'John', 'last_name'=>'Doe',
22             'password'=>'$2y$12$NHkuo.t5nAZoaceH6EwpE.
23             JNeYdOPR2BauV/sfyTjVrP522dCb0Je', 'admin'=>1],
24             ['id'=>3, 'email'=>'krzy@email.com',
25             'first_name'=>'Krzysztof', 'last_name'=>'Kazik',
26             'password'=>'$2y$12$/aOXMlkFG/
27             cXGStB0tyZetM5Q70G0XaFwRi6fJhZ0U5KuuPAAwq',
28             'admin'=>0],
29             ['id'=>4, 'email'=>'p.nowak@email.com',
30             'first name'=>'Piotr', 'last name'=>'Nowak',
```

Wypełnia tabelę transport_users danymi o użytkownikach.

Wprowadzane są informacje takie jak imię, nazwisko, adres e-mail, hasło oraz rola (czy użytkownik jest administratorem).

4. RoutesTableSeeder.php:

```

database > seeders > RoutesTableSeeder.php > Database\Seeders\RoutesTableSee
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use Illuminate\Support\Facades\DB;
7
8  1 reference | 0 implementations
class RoutesTableSeeder extends Seeder
9  {
10     0 references | 0 overrides
    public function run()
11     {
12         DB::table('routes')->insert([
13             ['id' => 1, 'name' => 'Rejtana Skrzyżowanie - Pl
14             ['id' => 2, 'name' => 'Dworzec Główny PKP - Gale
15             ['id' => 3, 'name' => 'Nowe Miasto Osiedle - Por
16             ['id' => 4, 'name' => 'Tyczyn Rynek - Łańcut Zam
17             ['id' => 5, 'name' => 'Głogów Małopolski Urząd M
18         ]);
19     }
20

```

Wypełnia tabelę transport_routes danymi o trasach.

Dane obejmują nazwę trasy, opis oraz identyfikator użytkownika, który ją utworzył.

5. StopTimesTableSeeder.php:

database > seeders > [php](#) StopTimesTableSeeder.php > ...

```


1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use Illuminate\Support\Facades\DB;
7
8  1 reference | 0 implementations
9  class StopTimesTableSeeder extends Seeder
10 {
11     0 references | 0 overrides
12     public function run()
13     {
14         DB::table('stop_times')->insert([
15             ['id'=>1,'route_id'=>1,'stop_id'=>1,'arrival_time'=>'13:27:00'],'d
16             ['id'=>2,'route_id'=>1,'stop_id'=>2,'arrival_time'=>'13:33:00'],'d
17             ['id'=>3,'route_id'=>1,'stop_id'=>3,'arrival_time'=>'13:56:00'],'d
18             ['id'=>4,'route_id'=>1,'stop_id'=>4,'arrival_time'=>'14:02:00'],'d
19             ['id'=>5,'route_id'=>1,'stop_id'=>5,'arrival_time'=>'14:36:00'],'d
20             ['id'=>6,'route_id'=>5,'stop_id'=>21,'arrival_time'=>'11:10:00'],'
21             ['id'=>7,'route_id'=>5,'stop_id'=>22,'arrival_time'=>'13:20:00'],'
22             ['id'=>8,'route_id'=>5,'stop_id'=>23,'arrival_time'=>'15:23:00'],'
23             ['id'=>9,'route_id'=>5,'stop_id'=>24,'arrival_time'=>'16:56:00'],'
24             ['id'=>10,'route_id'=>5,'stop_id'=>25,'arrival_time'=>'21:44:00'],'
25             ['id'=>11,'route_id'=>4,'stop_id'=>16,'arrival_time'=>'13:23:00'],'
26             ['id'=>12,'route_id'=>4,'stop_id'=>17,'arrival_time'=>'15:15:00'],'
27             ['id'=>13,'route_id'=>4,'stop_id'=>18,'arrival_time'=>'16:23:00'],'
28             ['id'=>14,'route_id'=>3,'stop_id'=>10,'arrival_time'=>'11:03:00'],'
29             ['id'=>15,'route_id'=>3,'stop_id'=>13,'arrival_time'=>'12:05:00'],'
30             ['id'=>16,'route_id'=>3,'stop_id'=>14,'arrival_time'=>'14:33:00'],'
31             ['id'=>17,'route_id'=>3,'stop_id'=>15,'arrival_time'=>'15:00:00'],'
32             ['id'=>18,'route_id'=>2,'stop_id'=>8,'arrival_time'=>'16:55:00'],'
33             ['id'=>19,'route_id'=>3,'stop_id'=>11,'arrival_time'=>'14:54:00'],'

```

Wypełnia tabelę `transport_stop_times` informacjami o czasach przyjazdu i odjazdu na poszczególnych przystankach dla każdej trasy.

6. RouteDatesTableSeeder.php:


```

database > seeders >  RouteDatesTableSeeder.php > ...
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use Illuminate\Support\Facades\DB;
7
8  1 reference | 0 implementations
9  class RouteDatesTableSeeder extends Seeder
10 {
11     0 references | 0 overrides
12     public function run()
13     {
14         DB::table('route_dates')->insert([
15             ['id' => 1, 'route_id' => 1, 'date' => '2024-06-01'],
16             ['id' => 2, 'route_id' => 2, 'date' => '2024-06-02'],
17             ['id' => 3, 'route_id' => 2, 'date' => '2024-06-03'],
18             ['id' => 4, 'route_id' => 1, 'date' => '2024-06-04'],
19             ['id' => 5, 'route_id' => 1, 'date' => '2024-06-05'],
20             ['id' => 6, 'route_id' => 1, 'date' => '2024-06-06'],
21             ['id' => 7, 'route_id' => 3, 'date' => '2024-06-07'],
22             ['id' => 8, 'route_id' => 3, 'date' => '2024-06-08'],
23             ['id' => 9, 'route_id' => 3, 'date' => '2024-06-09'],
24             ['id' => 10, 'route_id' => 3, 'date' => '2024-06-10'],
25         ]);
26     }
27 }

```

Wypełnia tabelę `transport_route_dates` datami, w których poszczególne trasy są aktywne.

3. **Utworzenie użytkownika/ów pełniących rolę administratora, zarządzającego zasobami aplikacji:** Model `User` ma pole `admin`, które może służyć do określenia, czy użytkownik jest administratorem. W kontrolerze `AdminController` zaimplementowana jest logika ograniczająca dostęp do pewnych funkcji tylko dla administratorów. Middleware `IsAdmin` weryfikuje czy użytkownik ma uprawnienia administratora.

Pliki:

- **app/Models/User.php:** Zawiera definicję modelu `User` oraz metodę `isAdmin()`, która sprawdza, czy użytkownik jest administratorem.

```

1 reference | 0 overrides
public function isAdmin()
{
    return $this->admin;
}

```

- **app/Http/Controllers/AdminController.php:** Zawiera metody kontrolera, które są dostępne tylko dla administratorów (np. assignRoute, removeRoute, create).

```

1 reference | 0 overrides
public function assignRouteForm()
{
    $users = User::all();
    $routes = Route::all();
    return view('admin.assign-route', compact('users', 'routes'))
}

```

```

1 reference | 0 overrides
public function removeRoute(Request $request)
{
    $request->validate([
        'route_date_id' => 'required|exists:route_dates,id',
    ]);

    $routeDate = RouteDate::findOrFail($request->input('route_date_id'));

    $route = $routeDate->route;

    $routeDate->delete();

    $routeHasOtherDates = RouteDate::where('route_id', $route->id)->count() > 0;

    if (!$routeHasOtherDates) {
        $route->update(['user_id' => null]);
    }

    return redirect()->route('admin.assign')->with('success', 'Route removed successfully');
}

```

```

0 references | 0 overrides
public function create()
{
    $stops = Stop::all();
    $users = User::where('admin', false)->get();
    return view('admin.routes.create', compact('stops', 'users'))
}

```

- **app/Http/Middleware/IsAdmin.php:** Zawiera middleware `IsAdmin`, który chroni trasy dostępne tylko dla administratorów.

```

app > Http > Middleware > IsAdmin.php > ...
1  <?php
2
3  namespace App\Http\Middleware;
4
5  use Closure;
6  use Illuminate\Http\Request;
7
8  class IsAdmin
9  {
10     public function handle(Request $request, Closure $next)
11     {
12         if (!auth()->check() || !auth()->user()->isAdmin()) {
13             return redirect('/');
14         }
15         return $next($request);
16     }
17 }

```

- **routes/web.php:** Definiuje trasy, które są chronione middlewarem `IsAdmin`.

```

Route::middleware(\App\Http\Middleware\IsAdmin::class)->group(function() {
    Route::get('/admin', [AdminController::class, 'index'])->name('admin.index');
    Route::resource('routes', RouteController::class, ['as' => 'admin.routes']);
    Route::resource('stops', StopController::class, ['as' => 'admin.stops']);
    Route::get('assign-route', [AdminController::class, 'assignRoute'])->name('admin.assign-route');
    Route::post('assign-route', [AdminController::class, 'assignRoute']);
});

```

4. **Możliwość przeglądania ogólnie dostępnych zasobów:** Kontrolery `RouteController` i `StopController` mają metody `index` i `show`, które obsługują wyświetlanie tras i przystanków dla wszystkich użytkowników.

Pliki:

- **app/Http/Controllers/RouteController.php:** Zawiera metody `index` (wyświetlanie listy tras) i `show` (wyświetlanie szczegółów trasy).

```

2 references | 0 overrides
public function index()
{
    $routes = Route::all();
    return view('routes.index', compact('routes'));
}

2 references | 0 overrides
public function show($id)
{
    $route = Route::with('stopTimes.stop')->find($id);

    if (!$route) {
        return redirect()->route('routes.index')->with('error', 'Rout

    }

    return view('routes.show', compact('route'));
}

```

- **app/Http/Controllers/StopController.php:** Zawiera metody `index` (wyświetlanie listy przystanków)

```

1 reference | 0 overrides
public function index()
{
    $stops = Stop::all();
    return view('stops.index', compact('stops'));
}

```

- **resources/views/routes/index.blade.php:** Widok listy tras.

Available Routes

Select a Route

Choose...

- **resources/views/routes/show.blade.php:** Widok szczegółów trasy.

Rejtana Skrzyżowanie - Pl. Wolności Fontanna

Rejtana Skrzyżowanie - Lisa Kuli Rondo - Dworzec Główny PKP - Podpromie Kościół - Pl. Wolności Fontanna






Stop Name	Address	Arrival Time	Departure Time	Picture
Rejtana Skrzyżowanie	ul. Rejtana 1	13:27:00	13:28:00	
Lisa Kuli Rondo	ul. Lisa Kuli 15	13:33:00	13:34:00	
Dworzec Główny PKP	ul. Grottgera 2	13:56:00	13:57:00	
Kilara 02	ul. Kilara 02	14:02:00	14:03:00	
Pl. Wolności Fontanna	Pl. Wolności 7	14:36:00	14:37:00	

[Go back](#)

- <resources/views/stops/index.blade.php>: Widok listy przystanków.

Stops

[Add Stop](#)
[Go Back](#)

ID	Name	Address	Picture	Actions
41	Rejtana Skrzyżowanie	ul. Rejtana 1		Edit Delete
42	Lisa Kuli Rondo	ul. Lisa Kuli 15		Edit Delete
43	Dworzec Główny PKP	ul. Grottgera 2		Edit Delete
44	Kilara 02	ul. Kilara 02		Edit Delete
45	Pl. Wolności Fontanna	Pl. Wolności 7		Edit Delete

5. **Użytkownicy aplikacji, mogą się zalogować, zarządzać swoimi zasobami, zarządzać swoimi danymi:** Model `User` jest rozszerzeniem klasy `Authenticatable`, co oznacza, że używany jest wbudowany system uwierzytelniania Laravel. Kontrolery `Auth\LoginController` i `UserController` zawierają metody do logowania, rejestracji i zarządzania danymi użytkowników.

Pliki:

- app/Models/User.php:** Definiuje model użytkownika.

```

23 references | 0 implementations
class User extends Authenticatable
{
    use HasFactory, Notifiable;

    0 references
    protected $fillable = [
        'first_name', 'last_name', 'email', 'password', 'admin',
    ];

```

- **app/Http/Controllers/Auth/LoginController.php:** Zawiera metody związane z logowaniem.

```
1 reference | 0 overrides
public function login(Request $request)
{
    $request->validate([
        'email' => 'required|email|min:4',
        'password' => 'required|min:7',
    ]);

    $credentials = $request->only('email', 'password');

    if (Auth::attempt($credentials)) {
        if (Auth::user()->isAdmin()) {
            return redirect('/admin');
        }
        return redirect()->intended($this->redirectTo);
    }

    return back()->withErrors([
        'email' => 'The provided credentials do not match our records',
    ]);
}
```

- **app/Http/Controllers/UserController.php:** Zawiera metody do zarządzania danymi użytkowników (np. edycja profilu).

1 reference | 0 overrides

```
public function update(Request $request, User $user)
{
    if (auth()->user()->id !== $user->id && !auth()->user()->isAdmin) {
        abort(403, 'Unauthorized action.');
```

```
    }

    $request->validate([
        'first_name' => 'required|max:100|min:3',
        'last_name'  => 'required|max:100|min:3',
        'email'      => 'required|email|min:5|max:100',
        'password'   => 'nullable|confirmed|min:7',
    ]);

    $dataToUpdate = $request->only(['first_name', 'last_name', 'email', 'password']);

    if (auth()->user()->isAdmin()) {
        $dataToUpdate['admin'] = $request->input('admin');
    }

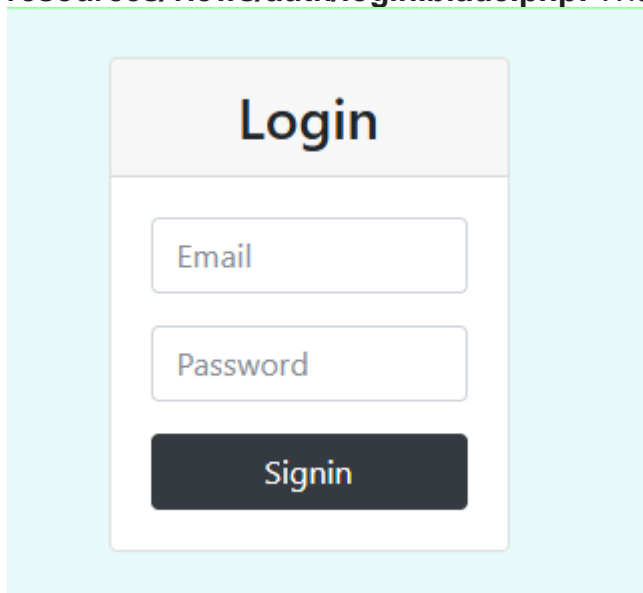
    if ($request->filled('password')) {
        $dataToUpdate['password'] = bcrypt($request->password);
    }

    $user->update($dataToUpdate);

    $redirectRoute = auth()->user()->isAdmin() ? 'users.index' : 'users.show';

    return redirect()->route($redirectRoute, $user)->with('success' => 'User updated successfully.');
```

- [resources/views/auth/login.blade.php](#): Widok formularza logowania.



The image shows a login form with a light blue background. At the top, there is a light gray header with the word "Login" in bold black text. Below the header, there are two white input fields with rounded corners. The first field is labeled "Email" and the second field is labeled "Password". Both fields have a light gray border and a light gray placeholder text. Below the input fields, there is a dark gray button with the word "Signin" in white text. The entire form is centered on the page.

7. W aplikacji dostępne są funkcjonalności CRUD: Kontrolery `RouteController`, `StopController`, `StopTimeController`, `RouteDateController` i `UserController` mają standardowe metody CRUD (create, read, update, delete) dla różnych zasobów.

Pliki:

- `app/Http/Controllers/RouteController.php`: CRUD dla tras.

```
1 reference | 0 overrides
public function store(Request $request)
{
    $request->validate([
        'name' => 'required|string|max:255',
        'description' => 'nullable|string',
        'stops' => 'required|array'
    ]);

    foreach ($request->stops as $stopId) {
        $stopExists = StopTime::where('stop_id', $stopId)->exists();
        if ($stopExists) {
            return redirect()->route('routes.create')->with('error'
        }
    }






    $route = Route::create($request->only('name', 'description'));
    $route->stops()->sync($request->stops);

    foreach ($request->stops as $key => $stopId) {
        $stopTime = new StopTime;
        $stopTime->route_id = $route->id;
        $stopTime->stop_id = $stopId;
        $stopTime->arrival_time = '00:00:00';
        $stopTime->departure_time = '00:01:00';
        $stopTime->save();
    }

    return redirect()->route('routes.index')->with('success', 'Rout
```

Rejtana Skrzyżowanie - Pl. Wolności Fontanna

Rejtana Skrzyżowanie - Lisa Kuli Rondo - Dworzec Główny PKP - Podpromie Kościół - Pl. Wolności Fontanna

Stop Name	Address	Arrival Time	Departure Time	Picture	Actions
Rejtana Skrzyżowanie	ul. Rejtana 1	13:27:00	13:28:00		Edit Delete
Lisa Kuli Rondo	ul. Lisa Kuli 15	13:33:00	13:34:00		Edit Delete
Dworzec Główny PKP	ul. Grottgera 2	13:56:00	13:57:00		Edit Delete
Kilara 02	ul. Kilara 02	14:02:00	14:03:00		Edit Delete
Pl. Wolności Fontanna	Pl. Wolności 7	14:36:00	14:37:00		Edit Delete

[Add Stop](#)[Delete Route](#)[Go back](#)

- [app/Http/Controllers/StopController.php](#): CRUD dla przystanków.

```

1 reference | 0 overrides
public function store(Request $request)
{
    $request->validate([
        'name' => 'required|max:255|min:3',
        'address' => 'required|max:255|min:3',
        'stoppic' => 'required|image|mimes:jpeg,png,jpg,gif|max:2048'
    ]);

    $stop = new Stop;
    $stop->name = $request->name;
    $stop->address = $request->address;

    $path = $request->file('stoppic')->store('public/stops');
    $fileName = basename($path);
    $stop->stoppic = $fileName;

    if ($stop->save()) {
        return redirect()->route('admin.stops.index')->with('success', 'Stop created');
    } else {
        return redirect()->route('stops.create')->with('error', 'Failed to create stop');
    }
}

```

Create Stop

Name

Address

Stop Picture

Wybierz plik

Nie wybrano pliku

Create

- **app/Http/Controllers/StopTimeController.php:** CRUD dla czasów przystanków.

0 references | 0 overrides

```
public function store(Request $request)
{
    $request->validate([
        'route_id' => 'required|exists:routes,id',
        'stop_id' => 'required|exists:stops,id',
        'arrival_time' => 'required|date_format:H:i:s',
        'departure_time' => 'required|date_format:H:i:s',
    ]);

    StopTime::create($request->all());
    return redirect()->route('stop-times.index')->with('success:');
}
```

Edit Stop and Time

Stop Name

Adress

Arrival Time



Stop Picture

Nie wybrano pliku

- **app/Http/Controllers/RouteDateController.php:** CRUD dla dat tras.


0 references | 0 overrides

```
public function store(Request $request)
{
    $request->validate([
        'route_id' => 'required|exists:routes,id',
        'date' => 'required|date',
    ]);

    RouteDate::create($request->all());
    return redirect()->route('route-dates.index')->with('success:');
}
```

Route Assignment

[Go Back](#)

User ID	Name and Surname	New Route	Assigned routes and days	Remove Route
1	Jan Nowak	<div>Choose the route ▾</div> <div>dd.mm.rrrr </div> <div>Accept</div>	Rejtana Skrzyżowanie - Pl. Wolności Fontanna: 2024-06-07 2024-06-12 2024-06-20 2024-06-21	<div>Choose the route to remove ▾</div> <div>Remove</div>

- **app/Http/Controllers/UserController.php:** CRUD dla użytkowników.

1 reference | 0 overrides

```
public function store(Request $request)
{
    if (!Auth::user()->isAdmin()) {
        abort(403, 'Unauthorized action.');
```

```
    }

    $request->validate([
        'first_name' => 'required|max:100|min:3',
        'last_name' => 'required|max:100|min:3',
        'admin' => 'required|boolean',
        'email' => 'required|min:5|max:100|email',
        'password' => 'required|min:7|max:100',
    ]);

    $input = $request->all();
    $input['password'] = bcrypt($request->password);
    $user = User::create($input);

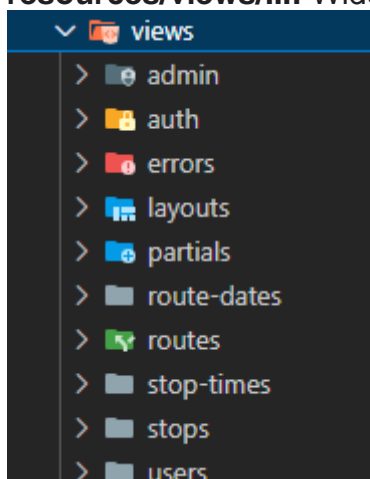
    return redirect()->route('users.show', $user)->with('succes
}
```

Users

[Create User](#)
[Go back](#)

ID	First Name	Last Name	Admin	Actions
1	Jan	Nowak	Yes	Show Edit Delete
3	John	Doe	Yes	Show Edit Delete
11	Krzysztof	Kazik	No	Show Edit Delete
13	Piotr	Nowak	No	Show Edit Delete

- **resources/views/...**: Widoki formularzy i list dla poszczególnych zasobów.



9. **Walidacja danych po stronie backendu:** W kontrolerach używana jest metoda `validate()` do sprawdzania poprawności danych wejściowych przed ich zapisaniem do bazy danych.

Pliki:

- **app/Http/Controllers/...**: Wszystkie kontrolery zawierają walidację danych w metodach `store` i `update`.

```
$request->validate([
    'route_id' => 'required|exists:routes,id',
    'stop_id' => 'required|exists:stops,id',
    'arrival_time' => 'required|date_format:H:i:s',
    'departure_time' => 'required|date_format:H:i:s',
]);
```

10. **Aplikacja posiada schludny responsywny GUI, spójny dla całej aplikacji:**

Uruchomienie aplikacji:

Należy mieć zainstalowanego XAMPP (wersja 8.2.12), Git (wersja 2.45.1), środowisko programistyczne np. Visual Studio Code, oraz Composer (wersja 2.7.6). Należy uruchomić XAMPP a następnie po otwarciu projektu w środowisku programistycznym należy wejść w terminal (Command Prompt) i wpisać „composer install” a następnie „php artisan migrate”. Po wykonaniu migracji należy uzupełnić bazę danymi, czyli wpisać polecenie „php artisan db:seed”. Jeżeli wszystko przebiegło pomyślnie należy wpisać komendę „php artisan serve”. Teraz możemy otworzyć przeglądarkę i wpisać „localhost:8000/”.

Widoki dla Administratora:

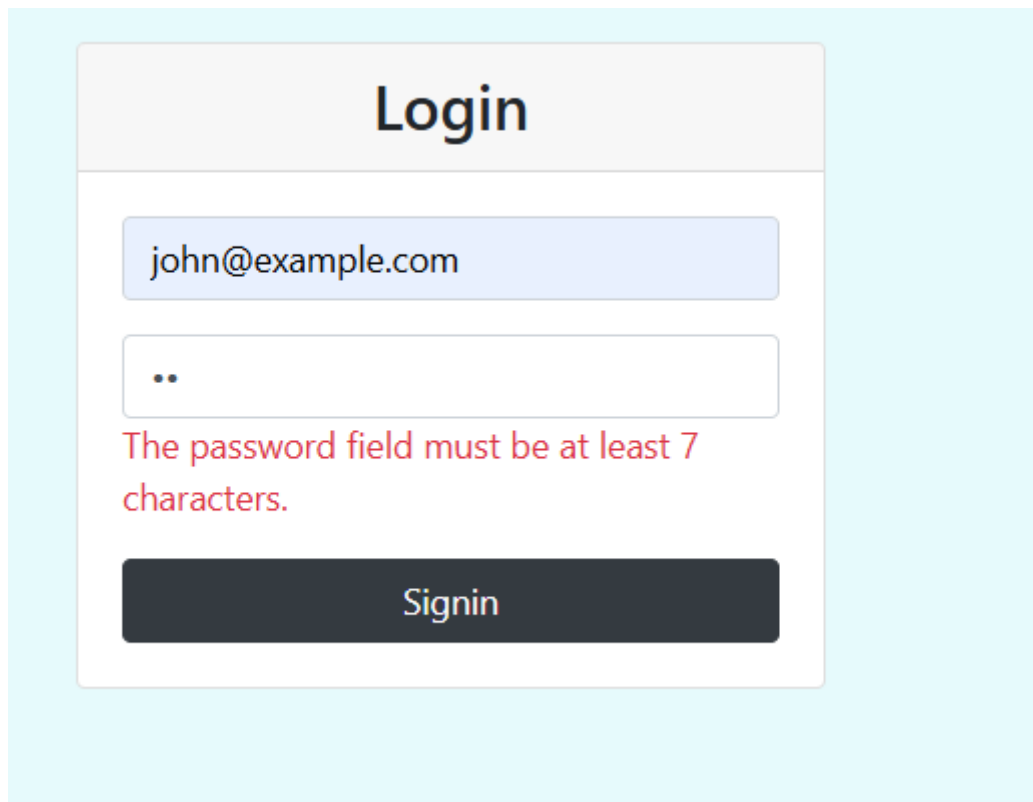
1. Widok logowania umożliwia użytkownikom dostęp do systemu poprzez podanie adresu e-mail oraz hasła.

Elementy widoku:

- **Login:** Nagłówek informujący o przeznaczeniu formularza.
- **Email:** Pole tekstowe do wprowadzenia adresu e-mail.
- **Password:** Pole tekstowe do wprowadzenia hasła (najlepiej z maskowaniem znaków).
- **Signin:** Przycisk zatwierdzający dane logowania i inicjujący proces uwierzytelniania użytkownika.

Działanie:

1. Użytkownik wprowadza swój adres e-mail i hasło w odpowiednich polach.
2. Po kliknięciu przycisku "Signin", system sprawdza poprawność wprowadzonych danych.
3. Jeśli dane są poprawne, użytkownik zostaje zalogowany i przekierowany do odpowiedniego widoku (np. strony głównej aplikacji).
4. Jeśli dane są niepoprawne, system wyświetla komunikat o błędzie i umożliwia ponowne wprowadzenie danych.

A login form UI mockup. It features a light blue background. At the top, a white box with a light gray header contains the word "Login" in bold black text. Below the header, there are two input fields. The first field is light blue and contains the email "john@example.com". The second field is white with a light gray border and contains two dots "••". Below the second field, a red error message reads "The password field must be at least 7 characters.". At the bottom of the form is a dark gray button with the text "Signin" in white.

2. Panel administratora systemu informacji autobusowej.

Użytkownicy

- **Dodaj użytkownika:** Umożliwia dodanie nowego użytkownika do systemu.
- **Pokaż użytkowników:** Wyświetla listę wszystkich użytkowników zarejestrowanych w systemie.

Trasy

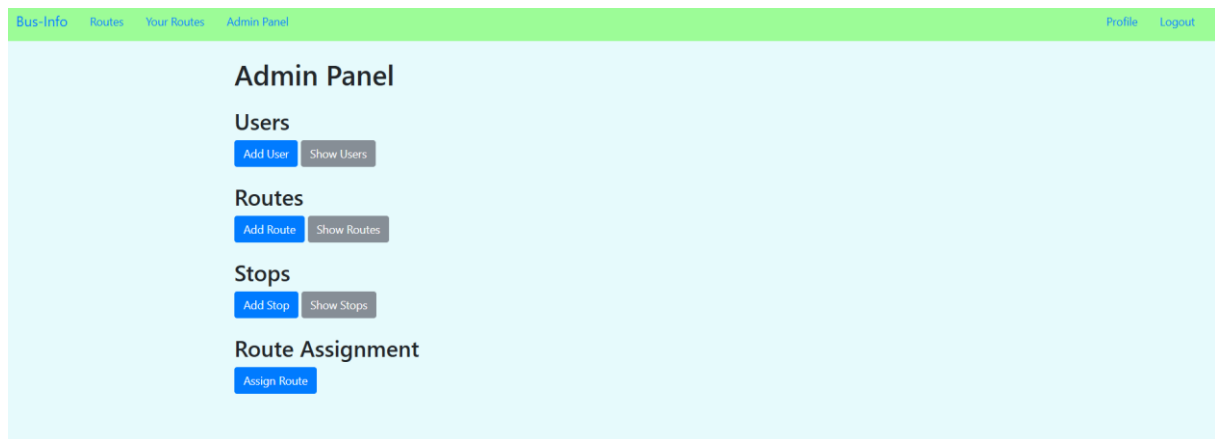
- **Dodaj trasę:** Umożliwia utworzenie nowej trasy autobusowej.
- **Pokaż trasy:** Wyświetla listę wszystkich tras autobusowych.

Przystanki

- **Dodaj przystanek:** Umożliwia dodanie nowego przystanku autobusowego.
- **Pokaż przystanki:** Wyświetla listę wszystkich przystanków autobusowych.

Przypisanie trasy

- **Przypisz trasę:** Umożliwia przypisanie trasy do konkretnego autobusu lub kierowcy.



3. Formularz tworzenia użytkownika w systemie Bus-Info.

Pola formularza:

- **First Name:** Pole do wprowadzenia imienia użytkownika.
- **Last Name:** Pole do wprowadzenia nazwiska użytkownika.
- **Email:** Pole do wprowadzenia adresu e-mail użytkownika.
- **Password:** Pole do wprowadzenia hasła użytkownika.
- **Admin:** Lista rozwijana pozwalająca określić, czy użytkownik ma być administratorem (tak/nie).

Przyciski:

- **Create:** Przycisk zatwierdzający utworzenie użytkownika z wprowadzonymi danymi.
- **Go Back:** Przycisk umożliwiający powrót do poprzedniego widoku (np. panelu administratora).

4. Widok przedstawia tabelę z listą użytkowników w systemie.

Zawartość tabeli:

- **ID:** Unikalny numer identyfikacyjny każdego użytkownika.

- **First Name:** Imię użytkownika.
- **Last Name:** Nazwisko użytkownika.
- **Admin:** Informacja czy użytkownik jest administratorem (tak/nie).
- **Actions:** Dostępne akcje dla każdego użytkownika:
 - **Show:** Wyświetlenie szczegółowych informacji o użytkowniku.
 - **Edit:** Edycja danych użytkownika.
 - **Delete:** Usunięcie użytkownika z systemu.

Przyciski:

- **Create User:** Przekierowanie do formularza tworzenia nowego użytkownika.
- **Go Back:** Przekierowanie do poprzedniego widoku (np. strony głównej panelu administratora).

Users				
Create User		Go back		
ID	First Name	Last Name	Admin	Actions
1	Jan	Nowak	Yes	Show Edit Delete
3	John	Doe	Yes	Show Edit Delete
11	Krzysztof	Kazik	No	Show Edit Delete
13	Piotr	Nowak	No	Show Edit Delete

5. Widok szczegółów użytkownika w systemie Bus-Info.

Zawartość:

- **User Details:** Nagłówek informujący o rodzaju wyświetlanych danych.
- **ID:** Unikalny numer identyfikacyjny użytkownika (w tym przypadku 1).
- **First Name:** Imię użytkownika (w tym przypadku Jan).
- **Last Name:** Nazwisko użytkownika (w tym przypadku Nowak).
- **Admin:** Informacja czy użytkownik posiada uprawnienia administratora (w tym przypadku Tak).
- **Email:** Adres e-mail użytkownika (w tym przypadku jannowak@email.com).

Przyciski:

- **Edit:** Przekierowuje do formularza edycji danych użytkownika.
- **Back to Users:** Przekierowuje do listy wszystkich użytkowników.

User Details

ID	1
First Name	Jan
Last Name	Nowak
Admin	Yes
Email	jannowak@email.com
<div><button>Edit</button><button>Back to Users</button></div>	

6. Formularz edycji użytkownika w systemie Bus-Info.

Pola formularza:

- **First Name:** Pole do edycji imienia użytkownika (wypełnione wartością "Jan").
- **Last Name:** Pole do edycji nazwiska użytkownika (wypełnione wartością "Nowak").
- **Email:** Pole do edycji adresu e-mail użytkownika (wypełnione wartością "jannowak@email.com").
- **Admin:** Lista rozwijana pozwalająca zmienić status administratora użytkownika (ustawiona na "Yes").

Przyciski:

- **Update:** Zatwierdza zmiany wprowadzone w danych użytkownika.
- **Go Back:** Powrót do poprzedniego widoku bez zapisywania zmian.

Edit User

First Name:

Last Name:

Email:

Admin:

UpdateGo back

7. Formularz edycji użytkownika (siebie) w systemie Bus-Info.

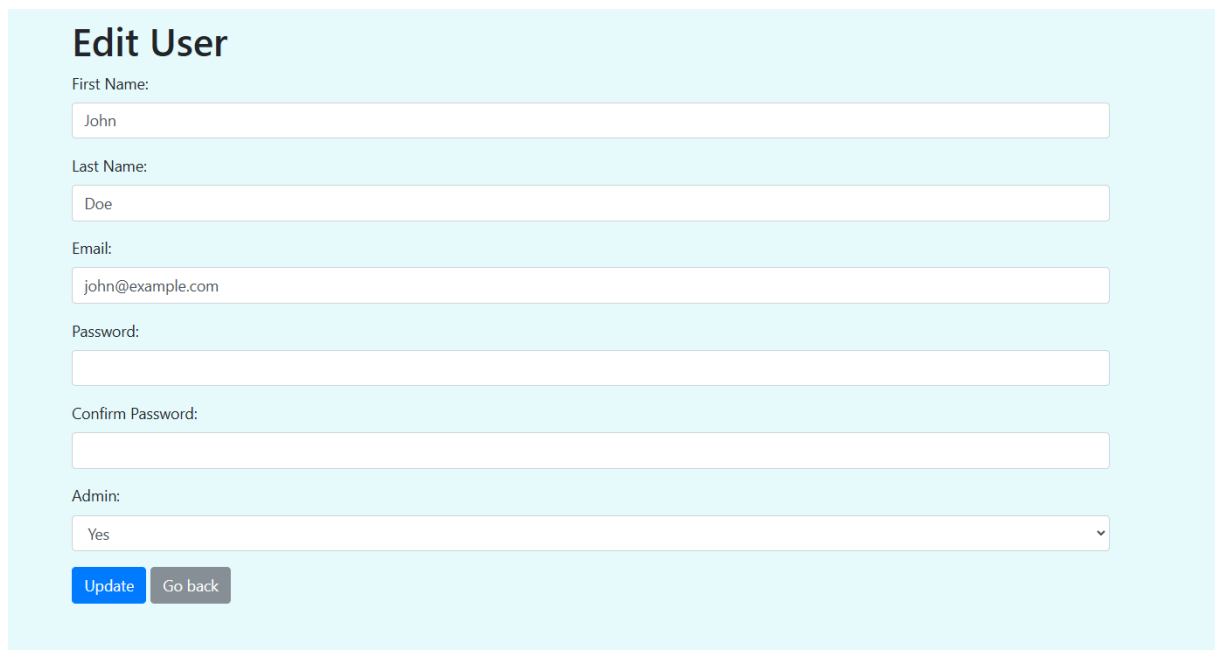
Pola formularza:

- **First Name:** Pole do edycji imienia użytkownika (wypełnione wartością "John").

- **Last Name:** Pole do edycji nazwiska użytkownika (wypełnione wartością "Doe").
- **Email:** Pole do edycji adresu e-mail użytkownika (wypełnione wartością "john@example.com").
- **Password:** Pole do wprowadzenia nowego hasła użytkownika (jeśli użytkownik chce zmienić hasło).
- **Confirm Password:** Pole do potwierdzenia nowego hasła użytkownika.
- **Admin:** Lista rozwijana pozwalająca zmienić status administratora użytkownika (ustawiona na "Yes").

Przyciski:

- **Update:** Zatwierdza zmiany wprowadzone w danych użytkownika.
- **Go Back:** Powrót do poprzedniego widoku bez zapisywania zmian.



Edit User

First Name:

Last Name:

Email:

Password:

Confirm Password:

Admin:

8. Formularz tworzenia nowej trasy autobusowej w systemie Bus-Info.

Pola formularza:

- **Name:** Pole tekstowe do wprowadzenia nazwy trasy (np. "Trasa nr 1", "Linia Zielona").
- **Description:** Pole tekstowe do wprowadzenia opisu trasy (np. "Trasa obsługująca centrum miasta", "Linia nocna").
- **Stops:** Lista przystanków autobusowych z możliwością zaznaczenia checkboxów przy przystankach, które mają znaleźć się na trasie.

Przyciski:

- **Create Route:** Przycisk zatwierdzający utworzenie nowej trasy z wprowadzonymi parametrami.

Działanie:

1. Użytkownik wprowadza nazwę i opis trasy.

2. Zaznacza przystanki, które mają znaleźć się na trasie.
3. Po kliknięciu przycisku "Create Route" system zapisuje nową trasę w bazie danych.

Create Route

Name

Description

Stops

- ☐ Staroniwa Cmentarz
- ☐ Łukasiewicza Pętla
- ☐ Millenium Hall Parking
- ☐ Szpital Wojewódzki SOR
- ☐ Przeworsk Dworzec PKP
- ☐ Leżajsk Bazylika

Create Route

9. Widok szczegółów trasy autobusowej w systemie Bus-Info.

Nagłówki:

- **Stop Name:** Nazwa przystanku.
- **Address:** Adres przystanku.
- **Arrival Time:** Planowany czas przyjazdu autobusu na przystanek.
- **Departure Time:** Planowany czas odjazdu autobusu z przystanku.
- **Picture:** Zdjęcie przystanku.
- **Actions:** Dostępne akcje dla każdego przystanku.

Przyciski:

- **Add Stop:** Dodanie nowego przystanku do trasy.
- **Delete Route:** Usunięcie całej trasy.
- **Go Back:** Powrót do poprzedniego widoku (np. listy tras).

Zawartość tabeli:

Tabela zawiera informacje o przystankach na trasie "Rejtana Skrzyżowanie - Pl. Wolności Fontanna", w tym:






- Nazwy przystanków (np. "Rejtana Skrzyżowanie", "Lisa Kuli Rondo").
- Adresy przystanków (np. "ul. Rejtana 1", "ul. Lisa Kuli 15").
- Planowane czasy przyjazdu i odjazdu (np. 13:27, 13:28).
- Zdjęcia przystanków.

Akcje dla przystanków:

Dla każdego przystanku dostępne są przyciski "Edit" (edycja danych przystanku) oraz "Delete" (usunięcie przystanku z trasy).

Rejtana Skrzyżowanie - Pl. Wolności Fontanna

Rejtana Skrzyżowanie - Lisa Kuli Rondo - Dworzec Główny PKP - Podpromie Kościół - Pl. Wolności Fontanna

Stop Name	Address	Arrival Time	Departure Time	Picture	Actions
Rejtana Skrzyżowanie	ul. Rejtana 1	13:27:00	13:28:00		Edit Delete
Lisa Kuli Rondo	ul. Lisa Kuli 15	13:33:00	13:34:00		Edit Delete
Dworzec Główny PKP	ul. Grottgera 2	13:56:00	13:57:00		Edit Delete
Kilara 02	ul. Kilara 02	14:02:00	14:03:00		Edit Delete
Pl. Wolności Fontanna	Pl. Wolności 7	14:36:00	14:37:00		Edit Delete

[Add Stop](#) [Delete Route](#) [Go back](#)

10. Formularz tworzenia nowego przystanku autobusowego w systemie Bus-Info.

Pola formularza:

- **Name:** Pole tekstowe do wprowadzenia nazwy przystanku (np. "Dworzec Główny", "Centrum Handlowe").
- **Address:** Pole tekstowe do wprowadzenia adresu przystanku (np. "ul. Warszawska 15", "Plac Centralny 3").
- **Stop Picture:** Przycisk do wyboru pliku ze zdjęciem przystanku (opcjonalnie).

Przyciski:

- **Create:** Przycisk zatwierdzający utworzenie nowego przystanku z wprowadzonymi danymi.

Działanie:

1. Użytkownik wprowadza nazwę i adres przystanku.
2. Opcjonalnie wybiera zdjęcie przystanku.
3. Po kliknięciu przycisku "Create" system zapisuje nowy przystanek w bazie danych.

Create Stop

Name

Address

Stop Picture

[Wybierz plik](#) Nie wybrano pliku

Create

11. Lista przystanków autobusowych w systemie Bus-Info.

Nagłówki:

- **ID:** Unikalny numer identyfikacyjny każdego przystanku.
- **Name:** Nazwa przystanku.
- **Address:** Adres przystanku.
- **Picture:** Zdjęcie przystanku (jeśli zostało dodane).
- **Actions:** Dostępne akcje dla każdego przystanku:
 - **Edit:** Edycja danych przystanku.
 - **Delete:** Usunięcie przystanku z systemu.

Przyciski:








- **Add Stop:** Przekierowanie do formularza dodawania nowego przystanku.
- **Go Back:** Powrót do poprzedniego widoku (np. widoku trasy, do której należą te przystanki).

Zawartość tabeli:

Tabela zawiera listę przystanków autobusowych, w tym:

- Nazwy przystanków (np. "Rejtana Skrzyżowanie", "Lisa Kuli Rondo").
- Adresy przystanków (np. "ul. Rejtana 1", "ul. Lisa Kuli 15").
- Zdjęcia przystanków (jeśli zostały dodane).

Dla każdego przystanku dostępne są przyciski "Edit" (edycja danych przystanku) oraz "Delete" (usunięcie przystanku z systemu).

Stops				
Add Stop Go Back				
ID	Name	Address	Picture	Actions
41	Rejtana Skrzyżowanie	ul. Rejtana 1		Edit Delete
42	Lisa Kuli Rondo	ul. Lisa Kuli 15		Edit Delete
43	Dworzec Główny PKP	ul. Grottgera 2		Edit Delete
44	Kilara 02	ul. Kilara 02		Edit Delete
45	Pl. Wolności Fontanna	Pl. Wolności 7		Edit Delete
46	Staroniwa Cmentarz	ul. Krakowska 20		Edit Delete
47	Łukasiewicza Pętla	ul. Ignacego Łukasiewicza 88		Edit Delete

12. Widok przypisania tras w systemie Bus-Info.

Nagłówek:

- **Route Assignment:** Tytuł strony wskazujący na jej funkcję.
- **Go Back:** Przycisk umożliwiający powrót do poprzedniej strony.

Tabela:

Tabela zawiera listę użytkowników wraz z możliwością przypisania im tras oraz usunięcia istniejących przypisań.

- **User and ID:** Unikalny numer identyfikacyjny oraz imię i nazwisko użytkownika.
- **New Route:**
 - Lista rozwijana do wyboru trasy, którą chcemy przypisać użytkownikowi.
 - Pole do wprowadzenia daty w formacie dd.mm.rrrr (dzień, miesiąc, rok).
 - Przycisk "Accept" zatwierdzający przypisanie trasy.
- **Assigned routes and days:** Lista tras już przypisanych do użytkownika wraz z datami.
- **Remove Route:**
 - Lista rozwijana do wyboru trasy, którą chcemy usunąć z przypisań użytkownika.
 - Przycisk "Remove" zatwierdzający usunięcie trasy.

Przykładowe dane:

- Użytkownik o ID 1 (Jan Nowak) ma przypisaną trasę "Rejtana Skrzyżowanie - Pl. Wolności Fontanna" w dniach 7, 12, 20 i 21 czerwca 2024 roku.
- Użytkownik o ID 3 (John Doe) ma przypisaną trasę "Nowe Miasto Osiedle - Port Lotniczy Rzeszów-Jasion" w dniach 7, 8, 9 i 13 czerwca 2024 roku.
- Użytkownik o ID 11 (Krzysztof Kazik) ma przypisaną trasę "Dworzec Główny PKP - Galeria Rzeszów Główny Wejście" w dniach 7 i 8 czerwca 2024 roku.
- Użytkownik o ID 13 (Piotr Nowak) nie ma jeszcze przypisanych tras.

Route Assignment

Go Back

User ID	Name and Surname	New Route	Assigned routes and days	Remove Route
1	Jan Nowak	<div><div>Choose the route</div><div>dd.mm.rrrr</div><div>Accept</div></div>	Rejtana Skrzyżowanie - Pl. Wolności Fontanna: 2024-06-07 2024-06-12 2024-06-20 2024-06-21	<div><div>Choose the route to remove</div><div>Remove</div></div>
3	John Doe	<div><div>Choose the route</div><div>dd.mm.rrrr</div><div>Accept</div></div>	Nowe Miasto Osiedle - Port Lotniczy Rzeszów-Jasion: 2024-06-07 2024-06-08 2024-06-09 2024-06-13	<div><div>Choose the route to remove</div><div>Remove</div></div>
11	Krzysztof Kazik	<div><div>Choose the route</div><div>dd.mm.rrrr</div><div>Accept</div></div>	Dworzec Główny PKP - Galeria Rzeszów Główny Wejści: 2024-06-07 2024-06-08	<div><div>Choose the route to remove</div><div>Remove</div></div>
13	Piotr Nowak	<div><div>Choose the route</div><div>dd.mm.rrrr</div><div>Accept</div></div>		<div><div>Choose the route to remove</div><div>Remove</div></div>

13. Widok "Twoje trasy" w systemie Bus-Info prezentuje informacje o przypisanych trasach kierowcy oraz statystyki dotyczące przepracowanych godzin.

Tabela "Twoje trasy":

- Zawiera listę przypisanych tras kierowcy.
- Kolumny:
 - **Nazwa:** Nazwa trasy.
 - **Opis:** Krótki opis trasy.
 - **Data:** Daty, w których trasa jest przypisana do kierowcy.
 - **Czas trwania:** Czas trwania trasy (obecnie brak danych).

Statystyki:

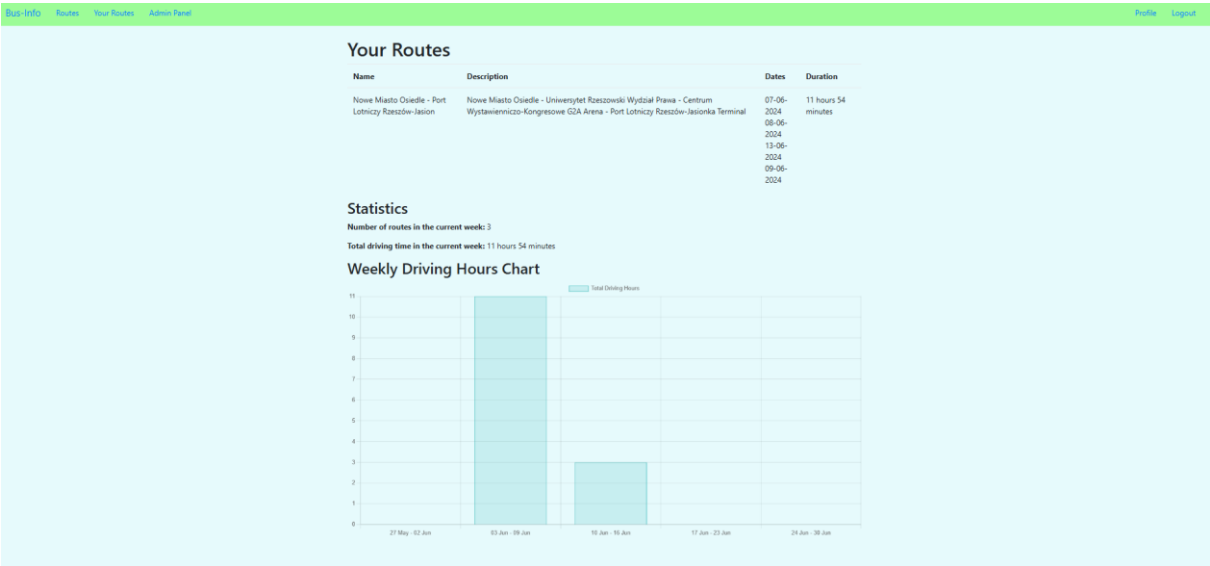
- **Liczba tras w bieżącym tygodniu:** Informuje o liczbie tras przypisanych do kierowcy w bieżącym tygodniu (w tym przypadku 2).
- **Całkowity czas jazdy w bieżącym tygodniu:** Pokazuje sumaryczny czas jazdy kierowcy w bieżącym tygodniu (w tym przypadku 11 godzin i 34 minuty).

Tygodniowy wykres godzin jazdy:

- Graficznie przedstawia rozkład godzin jazdy kierowcy w poszczególnych dniach tygodnia.
- W tym przypadku kierowca pracował najwięcej w poniedziałek (10 godzin), a w środę przepracował 2 godziny.

Przyciski:

- **Edytuj:** (obecnie nieaktywny) Prawdopodobnie umożliwia edycję przypisanych tras.
- **Usuń:** (obecnie nieaktywny) Prawdopodobnie umożliwia usunięcie przypisanych tras.





Widok trasy jako niezalogowany użytkownik:

Bus InfoRoutes

Log

Dworzec Główny PKP - Galeria Rzeszów Główne Wejści

Dworzec Główny PKP - Lisa Kuli Rondo - Kwiatkowskiego Skrzyżowanie - Galeria Rzeszów Główne Wejście

Stop Name	Address	Arrival Time	Departure Time	Picture
Kwiatkowskiego Skrzyżowanie	ul. Kwiatkowskiego 8	14:54:00	14:55:00	
Galeria Rzeszów Główne Wejście	al. Piłsudskiego 44	16:55:00	16:56:00	

Go back

Widoki zalogowanych użytkowników (nieAdministatorów):

Bus InfoRoutesYour Routes

ProfileLogout

Rejtana Skrzyżowanie - Pl. Wolności Fontanna

Rejtana Skrzyżowanie - Lisa Kuli Rondo - Dworzec Główny PKP - Podpromie Kościół - Pl. Wolności Fontanna

Stop Name	Address	Arrival Time	Departure Time	Picture
Rejtana Skrzyżowanie	ul. Rejtana 1	13:27:00	13:28:00	
Lisa Kuli Rondo	ul. Lisa Kuli 15	13:33:00	13:34:00	
Dworzec Główny PKP	ul. Grottgera 2	13:56:00	13:57:00	
Kilara 02	ul. Kilara 02	14:02:00	14:03:00	
Pl. Wolności Fontanna	Pl. Wolności 7	14:36:00	14:37:00	

Go back

Bus InfoRoutesYour Routes

ProfileLogout

Your Routes

Name	Description	Dates	Duration
Dworzec Główny PKP - Galeria Rzeszów Główne Wejści	Dworzec Główny PKP - Lisa Kuli Rondo - Kwiatkowskiego Skrzyżowanie - Galeria Rzeszów Główne Wejście	07-06-2024 08-06-2024	4 hours 4 minutes


Statistics

Number of routes in the current week: 3

Total driving time in the current week: 4 hours 4 minutes

Weekly Driving Hours Chart

Total Driving Hours



User Details

ID	11
First Name	Krzysztof
Last Name	Kazik
Admin	No
Email	krzy@email.com
<div>Edit Back to Users</div>	

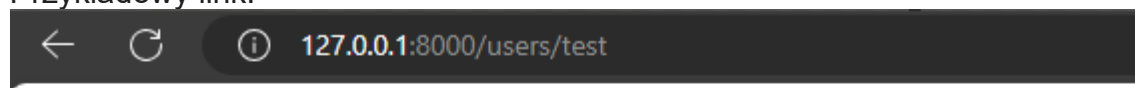
13. Dla kodów błędów HTTP zwracane są widoki z informacją o błędzie: Laravel domyślnie obsługuje błędy HTTP. Przykładowa strona w `resources/views/errors`.

Pliki:

- **`resources/views/errors/404.blade.php`**: Przykładowa strona błędu 404.

```
resources > views > errors > 404.blade.php
1  <!DOCTYPE html>
2  <html>
3  <head>
4  |   <title>404 Not Found</title>
5  </head>
6  <body>
7  |   <h1>404 Not Found</h1>
8  |   <p>The requested page could not be found.</p>
9  </body>
10 </html>
```

Przykładowy link:



404 Not Found

The requested page could not be found.

Podsumowanie:

Bus-Info to system informatyczny do zarządzania transportem publicznym. Aplikacja oferuje narzędzia dla administratorów, kierowców i pasażerów, ułatwiając planowanie tras, przydzielanie zadań, monitorowanie pojazdów i dostęp do informacji o rozkładach jazdy. System posiada intuicyjny interfejs użytkownika oraz umożliwia komunikację między wszystkimi uczestnikami transportu publicznego.