Aplikacje internetowe 1 (AI1) – laboratorium nr 7

Laravel – widoki z CRUD

Początek laboratorium:

• w *XAMPP* uruchomić *Apache* oraz *MySQL*, następnie przejść do *phpMyAdmin*,

• pobrać na pulpit archiwum `Lab007_AI1_start.zip`, w którym umieszczony jest projekt startowy do wykonania zadań oraz rozpakować to archiwum,

• przejść do rozpakowanego folderu oraz w przypadku posiadania innych ustawień niż domyślne (np. połączenia z bazą), wykonać ich zmianę w `.env.example` oraz `start.…`,

• uruchomić skrypt `start.bat` (*Windows,* 2x kliknięciem) lub `start.sh` (inne systemy, przez polecenie *bash start.sh*).

Zadania (*Laravel*):

Zadanie 7.1: *

Wyjaśnić następujące zagadnienia:
• operacje *CRUD*,
• *walidacja danych*,
• *walidacja danych* po stronie *backend'u*,
• *walidacja danych* po stronie *frontend'u*,
• *przekierowanie* (*redirect*),
• *przekierowanie* do *ścieżki nazwanej* w *Laravel'u*.

https://laravel.com/docs/11.x/redirects#redirecting-named-routes

Zadanie 7.2:

Otworzyć terminal *cmd (Command Prompt)* w *VSCode*.
Uruchomić serwer deweloperski *php* dla przy użyciu komendy `serve` *artisan'a*.

```
php artisan serve
```

W przeglądarce internetowej przejść pod adres: http://localhost:8000/trips

Zadanie 7.3:

Wykonać poniższe polecenie w celu wygenerowania nowego *kontrolera* dla modelu *Country* będącego „*zasobem*" wraz z klasami do ustalenia reguł walidacji dla wybranych rodzajów *żądań* (*app/Http/Requests/…*).

https://laravel.com/docs/11.x/controllers#resource-controllers
https://laravel.com/docs/11.x/controllers#generating-form-requests

```
php artisan make:controller CountryController --model=Country --resource --requests
```

Zadanie 7.4:
Ustawić *trasy* dla *kontrolera app/Http/Controllers/CountryController.php* dotyczące operacji *CRUD* na zasobie *Country*.
Wyświetlić obecnie skonfigurowane *trasowanie*.

https://laravel.com/docs/11.x/controllers#generating-form-requests

```
Route::resource('countries', CountryController::class);
```

```
php artisan route:list
```

```
GET|HEAD        countries ........................... countries.index › CountryController@index
POST            countries ........................... countries.store › CountryController@store
GET|HEAD        countries/create .................... countries.create › CountryController@create
GET|HEAD        countries/{country} ..................... countries.show › CountryController@show
PUT|PATCH       countries/{country} ................ countries.update › CountryController@update
DELETE          countries/{country} ............... countries.destroy › CountryController@destroy
GET|HEAD        countries/{country}/edit ............... countries.edit › CountryController@edit
```

Zadanie 7.5:
Uzupełnić funkcję *index* w *CountryController* o zwracanie widoku z pliku *resources\views\countries\index.blade.php* wraz z przekazaniem do niego zmiennej *countries* zawierającej kolekcję wszystkich krajów pobranych z bazy danych.
Następnie w tym widoku uzupełnić generowanie *HTMLowej* tabelki z krajami na podstawie zawartości kolekcji *countries*.

https://laravel.com/docs/11.x/eloquent#retrieving-models
https://laravel.com/docs/11.x/blade#loops

W przeglądarce internetowej przejść pod adres: http://localhost:8000/countries

Zadanie 7.6:
Uzupełnić funkcję *create* w *CountryController* o zwracanie widoku z pliku *resources\views\countries\create.blade.php*.
Następnie w tym widoku uzupełnić formularz o brakujące pole, w celu możliwości wpisania wszystkich danych nowego kraju oraz przesyłania ich żądaniem *POST* pod adres obsługiwany przez funkcję **`store`**.

W przeglądarce internetowej przejść pod adres: http://localhost:8000/countries/create

Zadanie 7.7:
Uzupełnić funkcję *store* w *CountryController* o dodanie do bazy danych nowego kraju, z danymi zebranymi z formularza z poprzedniego zadania.
Po udanym wstawieniu kraju do bazy → ma nastąpić przekierowanie pod *country.index*, po nieudanym → akcja domyślna.
W *app\Http\Requests\StoreCountryRequest.php*:
• w *authorize* zmienić *false* na *true*,
• w *rules* zaproponować i uzupełnić *reguły walidacji* nowych krajów np. opcjonalności, typu danych, unikalne nazwy, maks. długość nazwy, zakres min/maks. powierzchni.

https://laravel.com/docs/11.x/requests#retrieving-input
https://laravel.com/docs/11.x/eloquent#inserts
https://laravel.com/docs/11.x/redirects#redirecting-named-routes

https://laravel.com/docs/11.x/validation#quick-writing-the-validation-logic
https://laravel.com/docs/11.x/validation#creating-form-requests

Sprawdzić działanie formularza dodawania nowego kraju:
• podając kraj z prawidłowymi danymi,
• podając kraj z nieprawidłowymi danymi np. brakującymi wartościami, ujemną powierzchnią.

Zadanie 7.8:
W *resources\views\countries\index.blade.php* zmodyfikować tabelkę tak, aby była w niej dodatkowa kolumna zawierająca link „*Edycja*” w każdym wierszu.

```
<td><a href="{{route('countries.edit', $country->id)}}">Edycja</a></td>
```

lub

```
<td><a href="{{route('countries.edit', $country)}}">Edycja</a></td>
```

Zadanie 7.9:
Uzupełnić funkcję *edit* w *CountryController* o zwracanie widoku z pliku *resources\views\countries\edit.blade.php* wraz z przekazaniem do niego danych kraju automatycznie zmapowanego i pobranego z bazy danych, wybranego do edycji (w celu wstępnego uzupełniania pól formularza danymi tego kraju).
Następnie uzupełnić formularz edycji danych tego kraju w celu przesyłania ich *żądaniem POST* (z ukrytym polem **_method** z wartością „PUT” – specyficznie w *Laravel'u*) pod adres obsługiwany przez funkcję *update* z przekazaniem *id* kraju lub całego obiektu kraju.

https://laravel.com/docs/11.x/routing#implicit-binding
https://laravel.com/docs/11.x/blade#method-field

W przeglądarce internetowej przejść pod adres: http://localhost:8000/countries/1/edit

Zadanie 7.10:
Uzupełnić funkcję *update* w *CountryController* o aktualizację w bazie danych konkretnego kraju danymi przyjmowanymi w *żądaniu*.
Po udanej edycji kraju → ma nastąpić przekierowanie pod *country.index*, po nieudanej → akcja domyślna.
W *app\Http\Requests\UpdateCountryRequest.php*:
• w *authorize* zmienić *false* na *true*,
• w *rules* zaproponować i uzupełnić *reguły walidacji* nowych krajów np. opcjonalności, typu danych, unikalne nazwy, maks. długość nazwy, zakres min/maks. powierzchni. Uwzględnić wykluczenie sprawdzania unikatowości wobec tego samego obiektu.

https://laravel.com/docs/11.x/eloquent#updates

```
public function update(UpdateCountryRequest $request, Country $country)
{
    $input = $request->all();
    $country->update($input);
    return redirect()->route('countries.index');
}
```

https://www.csrhymes.com/2019/06/22/using-the-unique-validation-rule-in-laravel-form-request.html

```
return [
    'name' => 'required|unique:countries,name,'.$this->country->id.'|max:50',
    //...,
];
```

Sprawdzić działanie formularza edycji danego kraju:
• podając prawidłowe dane,
• podając nieprawidłowe dane np. brakujące wartości, ujemna powierzchnia.

Zadanie 7.11:

Uzupełnić funkcję *destroy* w *CountryController* o usunięcie z bazy danych konkretnego wybranego kraju.
Po udanym usunięciu kraju → ma nastąpić przekierowanie pod *country.index*, po nieudanym → akcja domyślna.

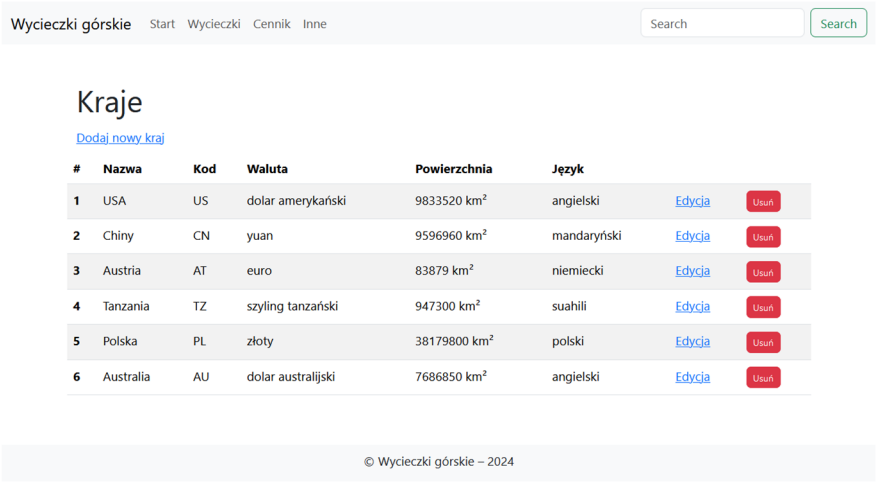
https://laravel.com/docs/11.x/eloquent#deleting-models

Zadanie 7.12:

W *resources\views\countries\index.blade.php* zmodyfikować tabelkę tak, aby była w niej dodatkowa kolumna zawierająca (mały) formularz z przyciskiem *Usuń* dotyczący usuwania danego kraju w każdym wierszu.
Przesłanie formularza ma być żądaniem *POST* (z ukrytym polem `_method` z wartością „DELETE" – specyficznie w *Laravel'u*) pod adres obsługiwany przez funkcję *destroy* z przekazaniem *id* kraju.

```
<td>
    <form method="POST" action="{{ route('countries.destroy', $country->id) }}">
        @csrf
        @method('DELETE')
        <input type="submit" class="btn btn-danger" value="Usuń"
        style="--bs-btn-padding-y: .25rem; --bs-btn-padding-x: .5rem; --bs-btn-font-size: .75rem;" />
    </form>
</td>
```

Sprawdzić działanie usuwania danego kraju:
• usuwając kraj z tabeli,
• usuwając kraj z tabeli ze zmodyfikowanym w *narzędziach dewelopera (F12)* nieistniejącym identyfikatorem:



Zadania (*Laravel c.d.*):

Zadanie 7.13: *

Spróbować usunąć pierwszy kraj. Wyjaśnić zaistniałą sytuację. Zaproponować rozwiązanie problemu.

Wykorzystać sposób wyświetlania błędów w sesji umieszczony w szablonie *resources\views\shared\session-error.blade.php*.

https://laravel.com/docs/11.x/errors

Zadanie 7.14: *

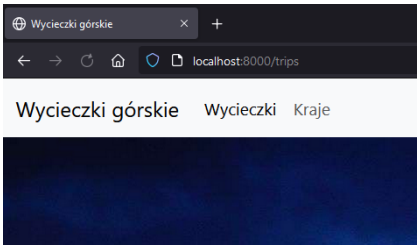
W *resources\views\countries\index.blade.php* zmodyfikować tabelkę tak, aby w kolumnie zawierającej *id* krajów były one linkami, a kliknięcie przenosiłoby do podstrony z informacjami o danym kraju (wykorzystanie funkcji *show* oraz widoku *show.blade.php*). Zaprojektować podstronę w tym widoku, np. karta/tabelka, itp.

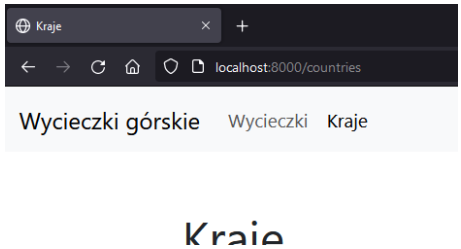
Zadanie 7.15: *

Zmodyfikować *navbar*, tak aby były w nim dwa linki:

- do strony głównej wycieczek,
- do tabelki z krajami.

Używając obiektu *request* w zależności od przeglądanej strony wyróżnić odpowiedni link.



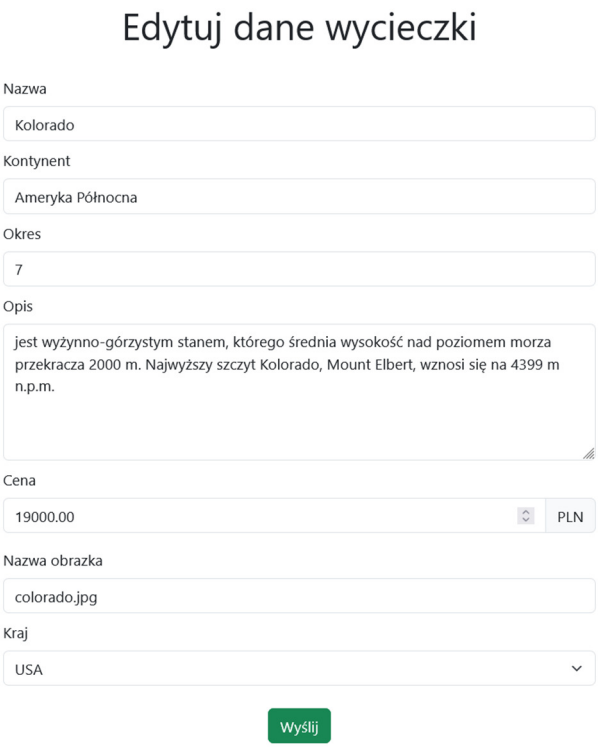


Zadanie 7.16: *

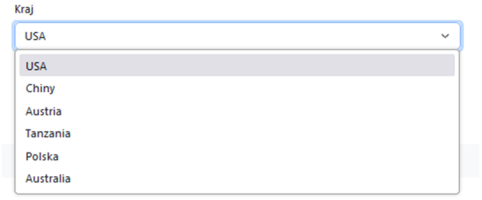
Dodać do *TripController* dwie nowe funkcje: *edit* i *update*, w których analogicznie zaprogramować edycję danej wycieczki. Ustawić dla nich *routing*.
W formularzu ma być pole *select* do wygodnego dla użytkownika wyboru kraju (na podstawie krajów obecnych dostępnie w bazie danych), w którym odbywa się wycieczka.
Umieścić załączenia szablonów *shared.session-error* i *shared.validation-error*.

```
Route::get('/trips/{id}/edit', 'edit')->name('trips.edit');
Route::put('/trips/{id}', 'update')->name('trips.update');
```

http://localhost:8000/trips/1/edit



https://developer.mozilla.org/en-US/docs/Web/HTML/Element/select



Zadanie 7.17: *

Do poprzedniego zadania dodać: reguły walidacji i ich obsługę, ale tym razem innym sposobem: bezpośrednie określenie ich w funkcji *update* (zamiast klas *...TripRequest*) wraz z przekazaniem ich do *validate*.

http://localhost:8000/trips

```
$request->validate([
    'name' => 'required|string|unique:trips,name,'.$id',
    //...,
]);
```

Zadanie 7.18: *

W *resources\views\trips\index.blade.php* zmodyfikować tabelkę tak, aby była w niej dodatkowa kolumna zawierająca w każdym wierszu link „*Edycja*”, prowadzący do podstrony edycji danej wycieczki.

Cena	
19000.00 PLN	Edycja
24000.00 PLN	Edycja
22000.00 PLN	Edycja

* – zadania/podpunkty do samodzielnego dokończenia/wykonania,

* – zadania/podpunkty dla zainteresowanych.

Do spakowania projektu należy wykorzystać skrypt *archiwizacja.bat*. Po rozpakowaniu projektu należy użyć ponownie *start.bat*.

Po zakończonym laboratorium należy skasować wszystkie pobrane oraz utworzone przez siebie pliki z komputera w sali laboratoryjnej.

Wersja pliku: v1.0