3rd LAB

"LARAVEL"

We will build a simple Laravel project during the 3rd and 4th labs.

- ➔ We will create two database tables: countries and cities.
- ➔ Each city must belong to a country.
- ➔ We will do basic database operations (show, insert, delete, etc.) in our project

---

"Do as much as you can in the 3rd lab"

PRE-INSTALLATION

1. **php:** php version should be >= **7.3.0**.
   Please check both (a) and (b):
   a. Go to Command Prompt and execute command `php -v`
      If php version is older than 7.3.0, then EDIT the system environment variables (quick web search).
      You should add or edit path containing "php.exe", e.g.
      `WAMPDIRECTORY\bin\php\php7.3.0`
      Then, restart Command Prompt to use the new version.
   b. Left click to WAMPSERVER tray icon and set the php version to >= 7.3.0.
2. **composer:** Your system should have **composer** installed. For testing, you can Open Command Prompt and execute command `composer -V`
   You can download it from https://getcomposer.org/Composer-Setup.exe

INSTALLATION

Open Command Prompt and go to the root director of WAMPSERVER (typically c:\wamp64\):

WAMPDIRECTORY\www

By using composer, install Laravel installer:

```
composer global require "laravel/installer"
```

Create your Laravel project under directory **WAMPDIRECTORY\www** by using your name (keeping it short and using only English letters is highly recommended)

"YOURNAME" will refer to your name throughout this document.
Please replace "YOURNAME" with your name when you copy-paste.

```
laravel new YOURNAME
```

```
cd YOURNAME
```

OPEN the following file

C:\wamp64\bin\apache\apache<version>\conf\httpd.conf

Make sure there is no # in front of a line

Include conf/extra/httpd-vhosts.conf

EDIT the following file

C:\wamp64\bin\apache\apache<version>\conf\extra\httpd-vhosts.conf

You should have the following:

```
<VirtualHost *:80>
  ServerName localhost
  ServerAlias localhost
  DocumentRoot "${INSTALL_DIR}/www"
  <Directory "${INSTALL_DIR}/www/">
    Options +Indexes +Includes +FollowSymLinks +MultiViews
    AllowOverride All
    Require local
  </Directory>
</VirtualHost>


<VirtualHost *:80>
    ServerName yourname.test
    DocumentRoot "${INSTALL_DIR}/www/yourname/public"
    <Directory "${INSTALL_DIR}/www/yourname/public/">
        AllowOverride All
  </Directory>
</VirtualHost>
```

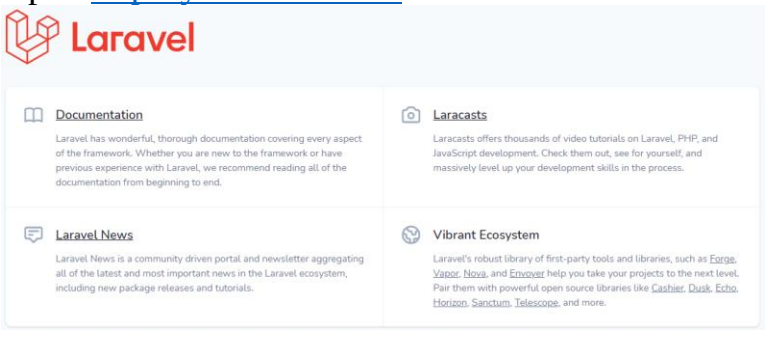
Assign a domain name to your project

EDIT (for example, run Notepad++ as administrator) → C:\Windows\System32\drivers\etc\hosts

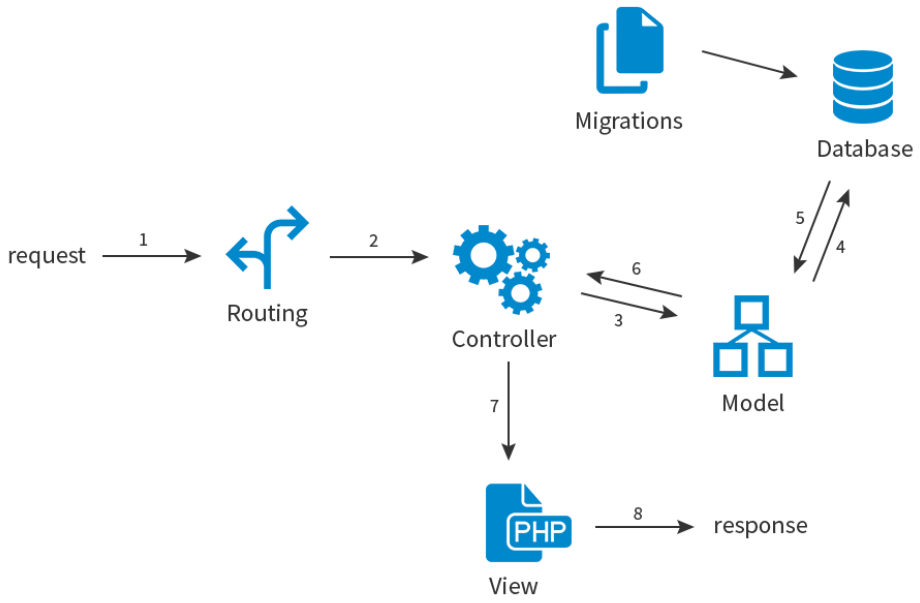
Add the following line at the end of a file

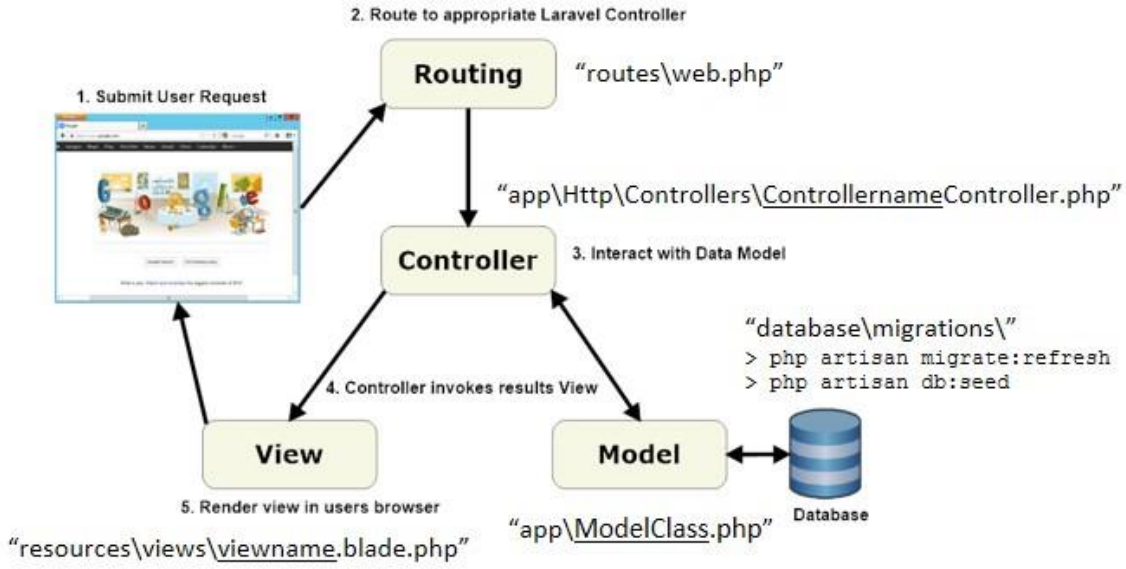
```
127.0.0.1 yourname.test
```

Open http://yourname.test/ in the browser. You should see:



GENERAL STRUCTURE





DATABASE

By using “phpmyadmin”,

create a database (YOURNAME-db) and
a user (YOURNAME-user / “pick a passoword”) with all privileges on a new database YOURNAME-db.

Configure the system accordingly:

EDIT → ..\www\YOURNAME\.env

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=YOURNAME-db
DB_USERNAME=YOURNAME-user
DB_PASSWORD=yourpassword
```

EDIT → ..\www\YOURNAME\config\database.php

```
'connections' => [
...
        'mysql' => [
...
            'engine' => 'InnoDB',
```

MIGRATION

In order to avoid “length error”:

EDIT → ..\YOURNAME\app\Providers\AppServiceProvider.php

```
use Illuminate\Support\Facades\Schema;
use Illuminate\Support\ServiceProvider;
…
public function boot()
    {
        Schema::defaultStringLength(191);
    }
```

Create migration files:

```
php artisan make:migration create_countries_table
```

```
php artisan make:migration create_cities_table
```

Configure migration files:

EDIT → ..\YOURNAME\database\migrations*_create_countries_table.php

```
    public function up()
    {
        Schema::create('countries', function (Blueprint $table) {
            $table->id();
            $table->timestamps();
            $table->string('country_name',50)->unique;
            $table->string('country_code',10);
        });
    }
```

EDIT → ..\YOURNAME\database\migrations*_create_cities_table.php

```
    public function up()
    {
        Schema::create('cities', function (Blueprint $table) {
            $table->id();
            $table->timestamps();
            $table->foreignId('country_id')->constrained();
            $table->string('city_name',100);
        });
    }
```

Create database tables:

```
php artisan migrate
```

Check created database tables in phpMyAdmin:

- cities
- countries
- failed_jobs
- migrations
- password_resets
- users

MODEL

Create your models working between the database and controllers:

```
php artisan make:model Country
```

```
php artisan make:model City
```

EDIT → ..\YOURNAME\app\Models\Country.php

```
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
// City model usage
use App\Models\City;

class Country extends Model
{
    use HasFactory;

    // to allow mass assignment of these fields
    protected $fillable=['country_name','country_code'];

    public function cities() { // FK relationship
        return $this->hasMany(City::class);
    }

}}
```

EDIT → ..\YOURNAME\app\Models\City.php

```
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
// Country model usage
use App\Models\Country;

class City extends Model
{
    use HasFactory;

    public function country() { // FK relationship
        return $this->belongsTo(Country::class);
    }
}
```

SEEDER

Let's prepare and insert some default data: ***3 countries and 3 cities***.

EDIT → ..\YOURNAME\database\seeders\DatabaseSeeder.php

```
    public function run()
    {
        DB::statement('SET FOREIGN_KEY_CHECKS=0;');
        City::truncate();
        Country::truncate();

        DB::table('countries')->insert([
        'country_name' => 'Latvia',
        'country_code' => 'LV',
        ]);

        $country = new Country();
        $country->country_name = 'Finland';
        $country->country_code = 'FI';
        $country->save();

        $country = Country::create(['country_name' => 'Germany',
            'country_code' => 'DE']);

        $country = Country::where('country_name','Latvia')->first();
        $country->cities()->create(['city_name' => 'Riga']);

        $city = new City();
        $city->city_name = 'Helsinki';
        $country = Country::where('country_name','Finland')->first();
        $country->cities()->save($city);

        $country = Country::where('country_name','Germany')->first();
        $city = new City();
        $city->city_name = 'Berlin';
        $city->country()->associate($country);
        $city->save();

        DB::statement('SET FOREIGN_KEY_CHECKS=1;');
    }
```

Add usages to **..\YOURNAME\database\seeds\DatabaseSeeder.php**

```
use App\Models\City;
use App\Models\Country;
use Illuminate\Database\Seeder;
use Illuminate\Support\Facades\DB;
```

```
php artisan db:seed
```

You can check the database and tables on "phpmyadmin" for the result.

If you need, you can re-build your tables: `php artisan migrate:refresh --seed`

CONTROLLER (Country)

Create a controller for “country”:

```
php artisan make:controller CountryController --resource
```

Showing all countries

1. What to do when calling “index” method of “County Controller”:

EDIT → ..\YOURNAME\app\Http\Controllers\CountryController.php

```
use App\Models\Country;
use App\Models\City;
...
public function index()
{
	$countries=Country::all();
	return view('countries', compact('countries'));
}
```

2. Method “index” calls a view for showing all countries.
   All the records from the database are retrieved by Model “Country” and then they are passed to the following view.

CREATE A VIEW → ..\YOURNAME\resources\views\countries.blade.php

```
<!DOCTYPE html>
<html>
<head>
<title>Countries</title>
</head>
<body>
@if (count($countries)==0)
	<p color='red'> There are no records in the database!</p>
@else
	<table style="border: 1px solid black">
		<tr>
			<td> Country Id </td>
			<td> Country Name </td>
			<td> Country Code </td>
			<td> cities </td>
			<td>  </td>
		</tr>
	@foreach ($countries as $country)
		<tr>
			<td> {{ $country->id }} </td>
			<td> {{ $country->country_name }} </td>
			<td> {{ $country->country_code }} </td>
			<td> <input type="button" value="show"
onclick="showCities({{ $country->id }})"> </td>
		<td>  </td>
		  </td>
	@endforeach
```

```
        </table>
@endif
<p> <input type="button" value="New Country"> </p>

<script>
        function showCities(countryID) {
                window.location.href="/city/country/"+countryID;
        }
</script>
</body>
</html>
```

3. Call the "index" method as the default initial page:

EDIT → ..\YOURNAME\routes\web.php

```
use App\Http\Controllers\CountryController;
...

// Comment out the following three lines
// Route::get('/', function () {
//     return view('welcome');
// });

Route::redirect('/', 'country');

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

4. Test your project by calling http://YOURNAME.test
   You should see the table of countries.

Delete a country

Modify the system to delete a country by its id.

EDIT → ..\YOURNAME\app\Http\Controllers\CountryController.php

```
    public function destroy($id)
    {
        City::where('country_id',$id)->delete();
        Country::findOrFail($id)->delete();
        return redirect('country/');
    }
```

EDIT A VIEW ..\YOURNAME\resources\views\countries.blade.php

```
<!DOCTYPE html>
<html>
<head>
<title>Countries</title>
</head>
<body>
@if (count($countries)==0)
        <p color='red'> There are no records in the database!</p>
@else
        <table style="border: 1px solid black">
                <tr>
```

```
                    <td> Country Id </td>
                    <td> Country Name </td>
                    <td> Country Code </td>
                    <td> cities </td>
                    <td>  </td>
             </tr>
      @foreach ($countries as $country)
             <tr>
                    <td> {{ $country->id }} </td>
                    <td> {{ $country->country_name }} </td>
                    <td> {{ $country->country_code }} </td>
                    <td> <input type="button" value="show"
onclick="showCities({{ $country->id }})"> </td>
                    <td> <form method="POST" action="{{
action([App\Http\Controllers\CountryController::class, 'destroy'], $country-
>id) }}">@csrf @method('DELETE')<input type="submit" value="delete"></form>
</td>
             </td>
      @endforeach
      </table>
@endif
<p> <input type="button" value="New Country"> </p>

<script>
      function showCities(countryID) {
             window.location.href="/city/country/"+countryID;
      }
</script>
</body>
</html>
```

Refresh the home page and press the delete button for a country (e.g. Finland) on the webpage.

(If needed, you can add new records by using “phpmyadmin” or by running the seeder.)

CONTROLLER (City)

Create a controller for “city”:

```
php artisan make:controller CityController --resource
```

Showing the cities of a country

EDIT → ..\YOURNAME\routes\web.php

```
use App\Http\Controllers\CityController;

...
Route::resource('city', CityController::class, ['except' => ['index',
'create']]);
Route::get('city/country/{id}', [CityController::class, 'index']);
```

EDIT → ..\YOURNAME\app\Http\Controllers\CityController.php

```
use App\Models\City;
use App\Models\Country;
...
public function index($id)
{
	$cities=City::where('country_id','=',$id)->get();
	return view('cities',['country_id'=>$id,'cities'=>$cities]);
}
```

CREATE A VIEW → ..\YOURNAME\resources\views\cities.blade.php

```
<!DOCTYPE html>
<html>
<head>
<title>Cities</title>
</head>
<body>
@if (count($cities)==0)
	<p color='red'> There is no city in the database!</p>
@else
	<table border="1">
		<tr>
			<td> City Id </td>
			<td> City Name </td>
			<td> Country Id </td>
			<td>  </td>
		</tr>
	@foreach ($cities as $city)
		<tr>
			<td> {{ $city->id }} </td>
			<td> {{ $city->city_name }} </td>
			<td> {{ $city->country_id }} </td>
			<td>  </td>
		</td>
	@endforeach
	</table>
@endif
<p> <input type="button" value="New City" onclick="newCity( {{ $country_id }}
)"> </p>
```

```
<script>
	function newCity(countryID) {
		window.location.href="/city/country/"+countryID+"/create";
	}
</script>
</body>
</html>
```

Test your project by clicking “show” button for a country (e.g. Latvia):

http://YOURNAME.test/city/country/1

Add a new city for a country – create a form

EDIT → ..\YOURNAME\routes\web.php

```
Route::get('city/country/{id}/create', [CityController::class, 'create']);
```

EDIT → ..\YOURNAME\app\Http\Controllers\CityController.php

```
public function create($id)
    {
       $country = Country::findOrFail($id);
       return view('city_new',compact('country'));
    }
```

CREATE A VIEW → ..\YOURNAME\resources\views\city_new.blade.php

```
<!DOCTYPE html>
<html>
<head>
<title>New city</title>
</head>
<body>
We will add a city for <b>{{ $country->country_name }}</b>:
<form method="POST" action="{{
action([App\Http\Controllers\CityController::class, 'store']) }}">
    @csrf
    <input type="hidden" name="country_id" value="{{ $country->id }}">
       <label for="city_name">City Name: </label>
       <input type="text" name="city_name" id="city_name">
       <input type="submit" value="add">
</form>
</body>
</html>
```

Test your project by clicking “New City” button:

http://YOURNAME.test/city/country/1/create

Inserting a new city

EDIT → ..\YOURNAME\app\Http\Controllers\CityController.php

```
    public function store(Request $request)
    {
```

```
        $city = new City();
        $city->city_name=$request->city_name;
        $city->country_id=$request->country_id;
        $city->save();
        return redirect('city/country/'.$city->country_id);
    }
```

Test your project by adding new cities.

Delete a city

Modify the system to delete a city by its id.

EDIT A VIEW ..\YOURNAME\resources\views\cities.blade.php

```
<!DOCTYPE html>
<html>
<head>
<title>Cities</title>
</head>
<body>
@if (count($cities)==0)
        <p color='red'> There is no city in the database!</p>
@else
        <table border="1">
                <tr>
                        <td> City Id </td>
                        <td> City Name </td>
                        <td> Country Id </td>
                        <td>  </td>
                </tr>
        @foreach ($cities as $city)
                <tr>
                        <td> {{ $city->id }} </td>
                        <td> {{ $city->city_name }} </td>
                        <td> {{ $city->country_id }} </td>
                        <td> <form method="POST" action="{{
action([App\Http\Controllers\CityController::class, 'destroy'], $city->id)
}}">@csrf @method('DELETE')<input type="submit" value="delete"></form> </td>
                </td>
        @endforeach
        </table>
@endif
<p> <input type="button" value="New City" onclick="newCity( {{ $country_id }}
)"> </p>
<script>
        function newCity(countryID) {
                window.location.href="/city/country/"+countryID+"/create";
        }
</script>
</body>
</html>
```

EDIT → ..\YOURNAME\app\Http\Controllers\CityController.php

```
    public function destroy($id)
    {
```

```
        $country_id = City::findOrFail($id)->country_id;
        City::findOrFail($id)->delete();
        return redirect('city/country/'.$country_id);
    }
```

Refresh the home page and press the delete button for a city (e.g. Riga) on the webpage showing cities for a country.

Lab 3 Task

1. Create your own table using a migration (make a foreign key referencing cities or countries table). At the beginning of the class, use the last digit of your student ID number and a laboratory group date and time to get the definition of a table that you have to create.

2. Modify the system to add new records and delete records from your own table

    a. Add a model for your table

    b. Add a controller and implement the necessary methods (index, create, store, destroy)

    c. Add views:

        i. to show all records from your table with a link/button to add a new record and to delete a record

        ii. to add new record

    d. Add routes to ..\YOURNAME\routes\web.php

3. Test the system