

Web Engineering Lab (B.Sc Engg Part-4, 2021)

Solution by Momen Khandoker Opi

Create an Employee Information System using the Laravel web framework. You can find the initial project in the following GitHub repository.

https://github.com/m-r-kushal/lab4222_23_g1.git

Objective 1

Clone the repository to your computer and place it in your webroot folder. Take necessary steps to make the initial application up and running. (E.g., create a MySQL database for the application, update environment variables in the .env file, install project dependencies). Finally, visit the web application's root URL and show the home page.

Solution: When you clone a Laravel project from GitHub, you'll need to set it up on your local environment before you can run it. Here's a step-by-step guide:

(Note: you must install Laragon, Composer, and npm first)

1. Clone the Repository
`git clone <project-url>`
2. Navigate to the Project Folder
`cd <project-folder>`
3. Install PHP Dependencies
`composer install`
4. Install Node.js Dependencies
`npm install`
5. Set Up Environment Configuration
`cp .env.example .env`

This copies the example environment file (.env.example) to a new .env file, which will store the project's environment-specific settings.

6. Generate Application Key
`php artisan key:generate`

This generates a unique application key and adds it to the .env file. The key is used for encryption and other security-related tasks in Laravel.

7. Start the Development Serve

`php artisan serve`

By default, the project should now be accessible at <http://localhost:8000>.

Objective 2

Create a new git branch named dev and generate migration and seeder classes to create a table for employee information following the Laravel naming convention and insert test data.

Table attributes:

- id (primary key)
- name (string:255)
- job_title (string:100)
- joining_date (date)
- salary (float)
- email (string:255, optional)
- mobile_no (string)
- address(text)

Solution: To generate migration and seeder classes, you should first create a database using phpmyadmin, HeidiSQL, or any other database client.

1. Put the database name, username, and password in the env folder like this:

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=my_database_name
DB_USERNAME=root
DB_PASSWORD=
```

2. Generate a Migration with model, controller, factory, and seeder. When you create these together, these are automatically connected.

Note: If you create one by one then you need to connect the model, migration, and factory manually. (Not recommended)

Single Command to Generate Everything You Need:

`php artisan make:model <ModelName> -mcrfs`

<ModelName>: Replace this with the name of your model, typically in singular form (example: **Employee**).

-m: Generates a migration file along with the model. This file will be located in the **database/migrations** directory.

-c: Creates a controller for the model. This will be placed in the **app/Http/Controllers** directory.

- r: Indicates that the controller should be a resource controller, which includes methods for handling typical CRUD operations (index, create, store, show, edit, update, destroy).
- f: Generates a factory for the model, placed in the `database/factories` directory. Factories are useful for generating test data.
- s: Creates a seeder for the model, found in the `database/seeds` directory. Seeders are used to insert data into the database.

3. Define the Table Structure

- Open the generated migration file located in the `database/migrations` directory.
- In the `up()` method, define the columns of the table:

```
public function up()
{
    Schema::create('employees', function (Blueprint $table) {
        $table->id();
        $table->string('name', 255);
        $table->string('job_title', 100);
        $table->date('joining_date');
        $table->double('salary');
        $table->string('email', 255)->unique()->nullable();
        $table->string('mobile_no')->unique();
        $table->text('address');
        $table->timestamps();
    });
}
```

4. Run the Migration

```
php artisan migrate
```

This command connects to your configured database server (as specified in the `.env` file) and updates the database by applying the changes defined in the migration files.

5. Go to the EmployeeFactory file from `database/factories` and update the file to generate test data

```
public function definition()
{
    return [
        'name' => $this->faker->name,
        'job_title' => $this->faker->jobTitle,
        'joining_date' => $this->faker->date,
        'salary' => $this->faker->randomFloat(2, 1000, 10000),
        'email' => $this->faker->unique()->safeEmail,
        'mobile_no' => $this->faker->unique()->phoneNumber,
        'address' => $this->faker->address,
    ];
}
```

6. Goto EmployeeSeeder file from `database/seeder` and update the file to run the factory in the seeder

```
<?php

namespace Database\Seeders;

use App\Models\Employee;
use Illuminate\Database\Seeder;

class EmployeeSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        Employee::factory(10)->create();
    }
}
```

(Note: Don't forget to add `use App\Models\Employee;`)

7. Run the Seeder

`php artisan db:seed --class=<seeder_name>` (for Employee model seeder name will be EmployeeSeeder)

Finally 10 test data will be added to the database.

Objective 3

Check Kushal Sir's online video

Video Resource by Kushal Sir

- Laravel 11 Class -Recorded on 24th June 2024

<https://youtu.be/zyiozjn7dkg>

Artificial Intelligence Lab

Solution by Akif Islam

We predicts - the final lab question will be similar to this class test question below:

AI-Lab TestI Full Marks: 20 Time: 02 Hours

1. Design a Customize Convolutional Neural Network (CNN) for Handwritten Digit Classification with the following specifications:

- a) Generate a CNN model with:
 - i. Two CNN hidden layers (Conv2D) of sizes 32, 64 followed by
 - ii. ReLU Activation and
 - iii. MaxPooling2D with Kernel size (3, 3), and Stride= (1,1)
- b) Use Flatten Layers to convert the feature map into 1D with a Dense layer of size 64 followed by an output Dense Layer of size 10 with SoftMax Activation Function.
- c) Display the generated CNN with the required number of parameters.
- d) Use the MNIST database for training and testing.
- ~~e) Adopt Data augmentation (rotation, shift, shear, zoom) with the MNIST dataset.~~
- f) Train two CNNs using the original MNIST dataset and augmented MNIST dataset.
- ~~g) Use the test MNIST dataset as well as the augmented test MNIST dataset to predict the accuracy of the two trained CNNs.~~
- h) Compare and plot the prediction accuracy of the two trained CNNs.

Solution Notebook

https://colab.research.google.com/drive/1ZM8PAWcfR2SYhPk3_4VZ8KJI79cvuBaD?usp=sharing

Distributed DBMS Lab

Solution by Md. Ahanaf Arif Khan

Installation

1. Download and Install Docker Desktop (Personal):

<https://www.docker.com/products/docker-desktop/>

2. Run Docker Desktop
3. Create a new directory anywhere (eg: /home/user/hadoop or E:\hadoop)
4. Open terminal from within the newly created directory.
5. **Run the command below to download the container:**

```
docker run -p 9870:9870 -p 8088:8088 -v ./home/hadoop/data -it --name=hadoop  
macio232/hadoop-pseudo-distributed-mode
```

6. Done.

Work

1. Start the container:

```
docker start hadoop
```

2. Get inside the container:

```
docker exec -it hadoop /bin/bash
```

3. Done.

Note to Troubleshoot

- If the container exits early, then delete the container using docker desktop or docker cli and run the command in step 5 of installation again.
- The created directory is mapped to the container in /home/hadoop/data for handling data communication.
- The container is an Ubuntu 16.04 OS, so most linux commands will work.

Resources to Learn More

1. <https://medium.com/geekculture/hdfs-commands-cheat-sheet-1cd7bf22e795> (for command)
2. <https://www.youtube.com/watch?v=qgBu8Go1SyM> (word count)
3. <https://www.tutorialspoint.com/hive/index.htm> (hive QL commands)

Cryptography Lab

<https://github.com/Ankar-Kumar/cryptoLab>