Laravel 11 CRUD Application

In this Lab Activity, We will create step by step laravel 11 crud operations.

CRUD Meaning: CRUD is an acronym that comes from the world of computer programming and refers to the four functions that are considered necessary to implement persistent storage in your application: create, read, update, and delete.

We will create a product CRUD application using Laravel 11 in this example. We will create a products table with name and detail columns using Laravel 11 migration. Then, we will create routes, a controller, views, and model files for the product module. We will use Bootstrap 5 for design. So, let's follow the steps below to create CRUD operations with Laravel 11.

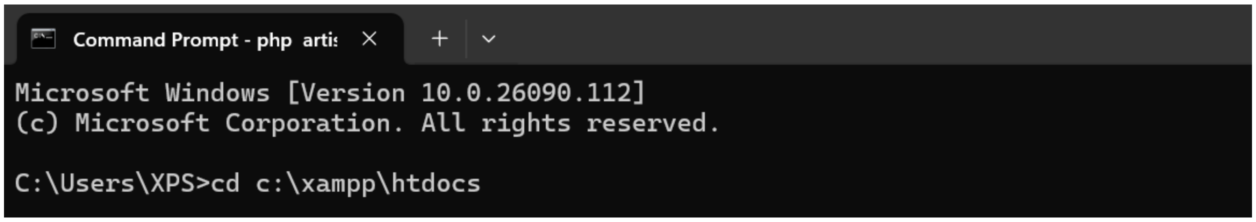
Step for Laravel 11 CRUD Operation Example

- **Step 1:** Install Laravel 11
- **Step 2:** MySQL Database Configuration
- **Step 3:** Create Migration
- **Step 4:** Create Form Request Validation Class
- **Step 5:** Create Controller and Model
- **Step 6:** Add Resource Route
- **Step 7:** Update AppServiceProvider
- **Step 8:** Add Blade Files
- **Run Laravel App**

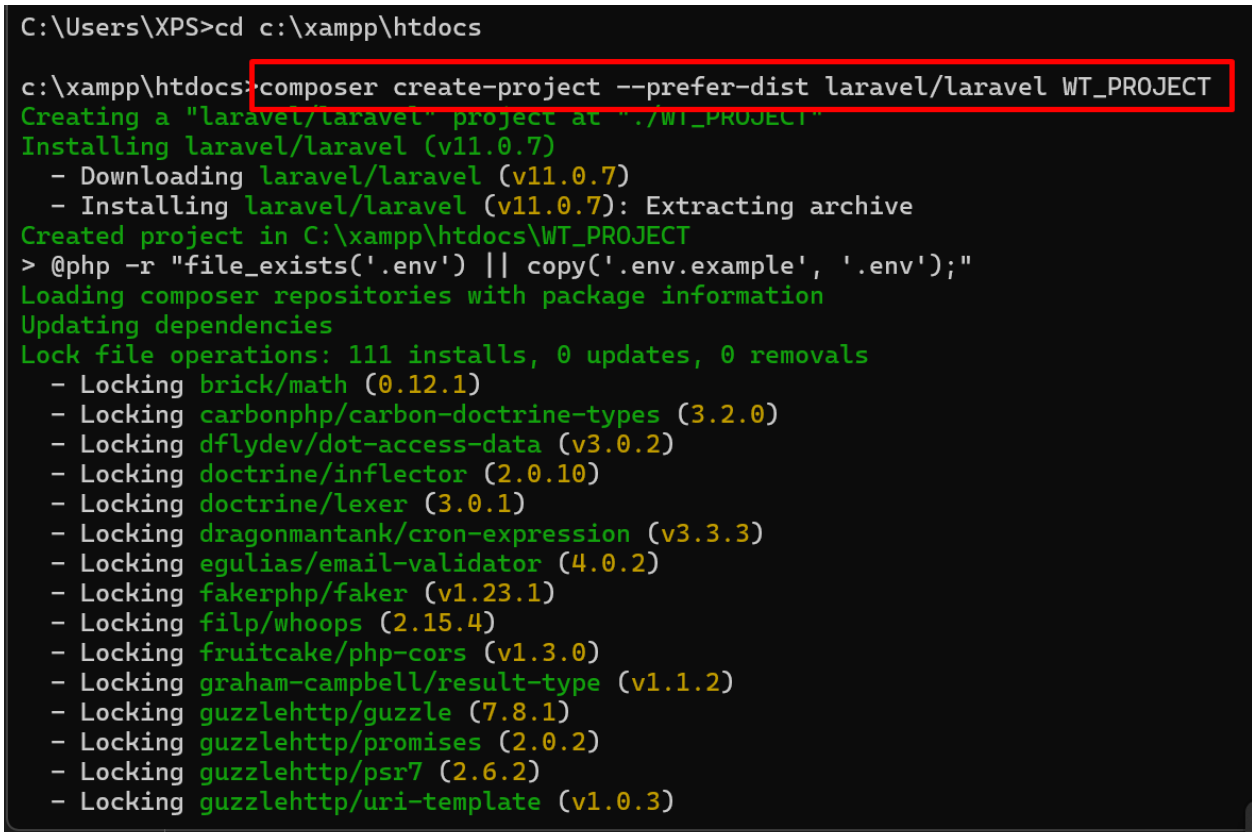
Step 1: Install Laravel 11

First of all, we need to get a fresh Laravel 11 version application using the command below because we are starting from scratch. So, open your terminal or command prompt and run the command below:

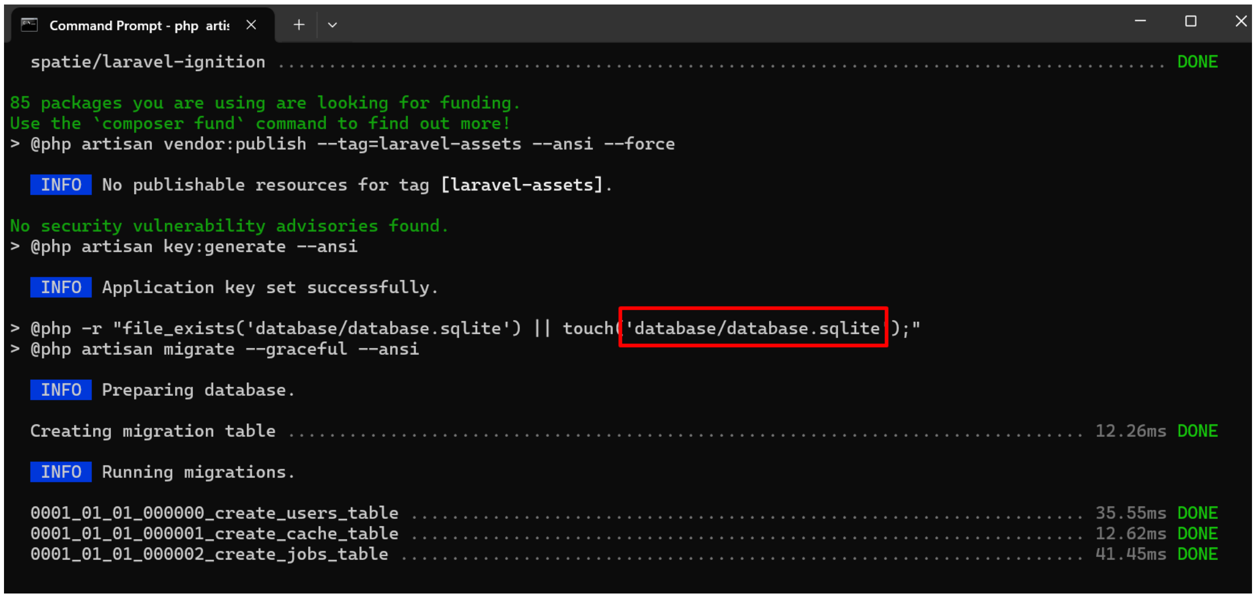
```
composer create-project laravel/laravel Crud-Application
```



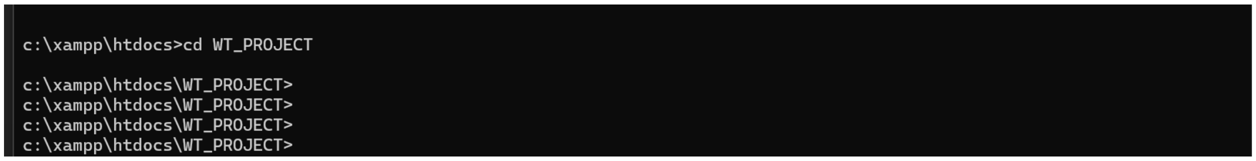
```
Microsoft Windows [Version 10.0.26090.112]
(c) Microsoft Corporation. All rights reserved.

C:\Users\XPS>cd c:\xampp\htdocs
```



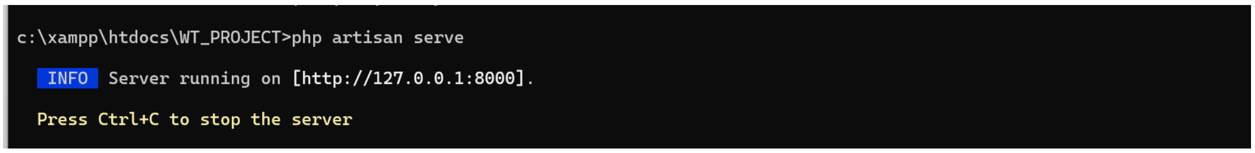
```
C:\Users\XPS>cd c:\xampp\htdocs

c:\xampp\htdocs>composer create-project --prefer-dist laravel/laravel WT_PROJECT
Creating a "laravel/laravel" project at "./WT_PROJECT"
Installing laravel/laravel (v11.0.7)
  - Downloading laravel/laravel (v11.0.7)
  - Installing laravel/laravel (v11.0.7): Extracting archive
Created project in C:\xampp\htdocs\WT_PROJECT
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
Lock file operations: 111 installs, 0 updates, 0 removals
  - Locking brick/math (0.12.1)
  - Locking carbonphp/carbon-doctrine-types (3.2.0)
  - Locking dflydev/dot-access-data (v3.0.2)
  - Locking doctrine/inflector (2.0.10)
  - Locking doctrine/lexer (3.0.1)
  - Locking dragonmantank/cron-expression (v3.3.3)
  - Locking egulias/email-validator (4.0.2)
  - Locking fakerphp/faker (v1.23.1)
  - Locking filp/whoops (2.15.4)
  - Locking fruitcake/php-cors (v1.3.0)
  - Locking graham-campbell/result-type (v1.1.2)
  - Locking guzzlehttp/guzzle (7.8.1)
  - Locking guzzlehttp/promises (2.0.2)
  - Locking guzzlehttp/psr7 (2.6.2)
  - Locking guzzlehttp/uri-template (v1.0.3)
```



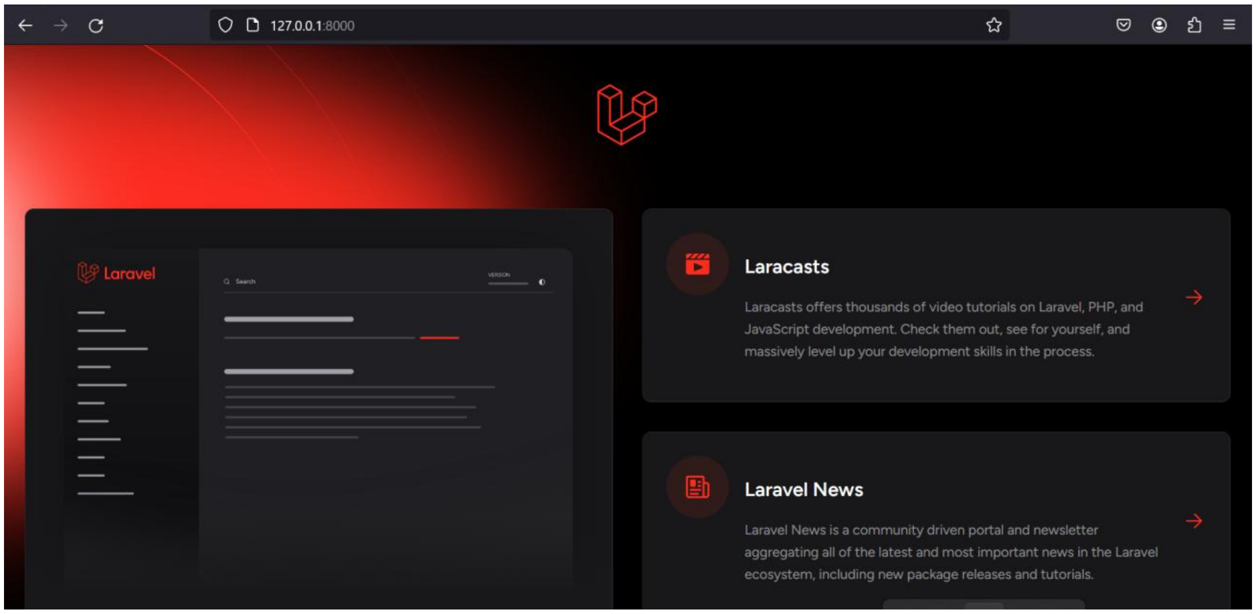
```
  spatie/laravel-ignition ........................................................................ DONE

85 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
> @php artisan vendor:publish --tag=laravel-assets --ansi --force

  INFO  No publishable resources for tag [laravel-assets].

No security vulnerability advisories found.
> @php artisan key:generate --ansi

  INFO  Application key set successfully.

> @php -r "file_exists('database/database.sqlite') || touch('database/database.sqlite');"
> @php artisan migrate --graceful --ansi

  INFO  Preparing database.

  Creating migration table ................................................................ 12.26ms DONE

  INFO  Running migrations.

  0001_01_01_000000_create_users_table .................................................. 35.55ms DONE
  0001_01_01_000001_create_cache_table .................................................. 12.62ms DONE
  0001_01_01_000002_create_jobs_table ................................................... 41.45ms DONE
```

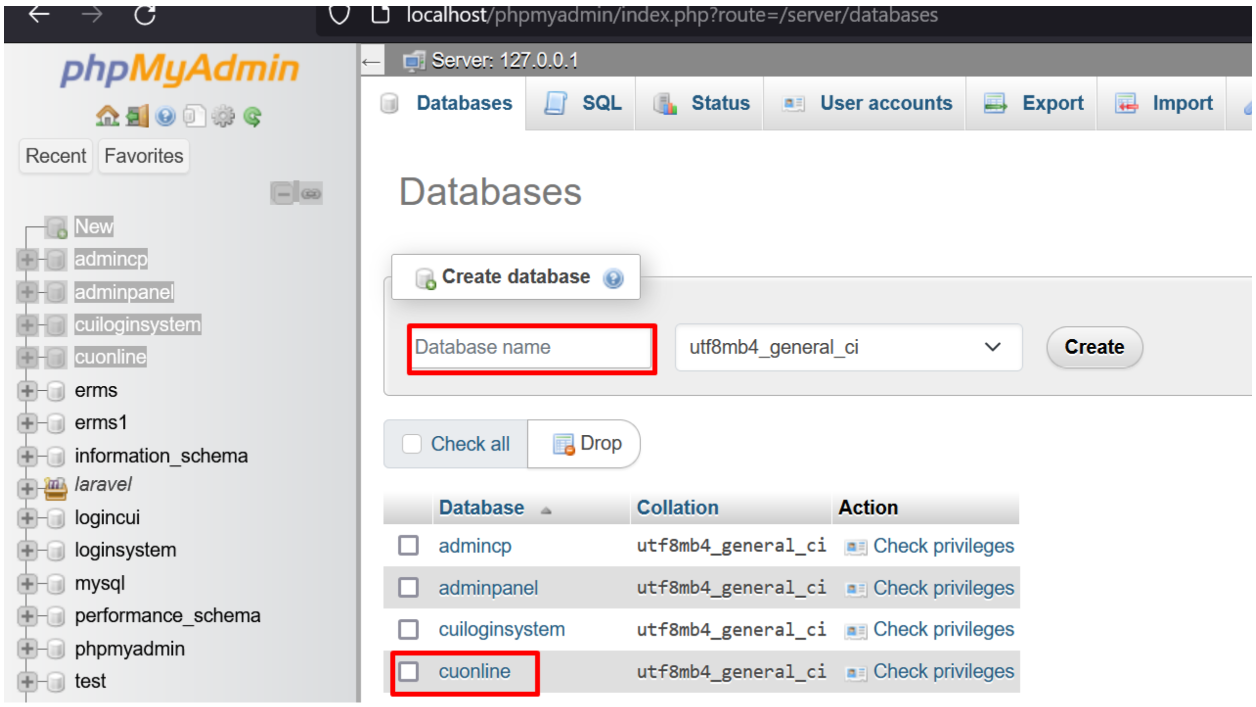


```
c:\xampp\htdocs>cd WT_PROJECT

c:\xampp\htdocs\WT_PROJECT>
c:\xampp\htdocs\WT_PROJECT>
c:\xampp\htdocs\WT_PROJECT>
c:\xampp\htdocs\WT_PROJECT>
```



```
c:\xampp\htdocs\WT_PROJECT>php artisan serve

  INFO  Server running on [http://127.0.0.1:8000].

  Press Ctrl+C to stop the server
```


127.0.0.1:8000
Laravel
Laracasts
Laracasts offers thousands of video tutorials on Laravel, PHP, and JavaScript development. Check them out, see for yourself, and massively level up your development skills in the process.
Laravel News
Laravel News is a community driven portal and newsletter aggregating all of the latest and most important news in the Laravel ecosystem, including new package releases and tutorials.



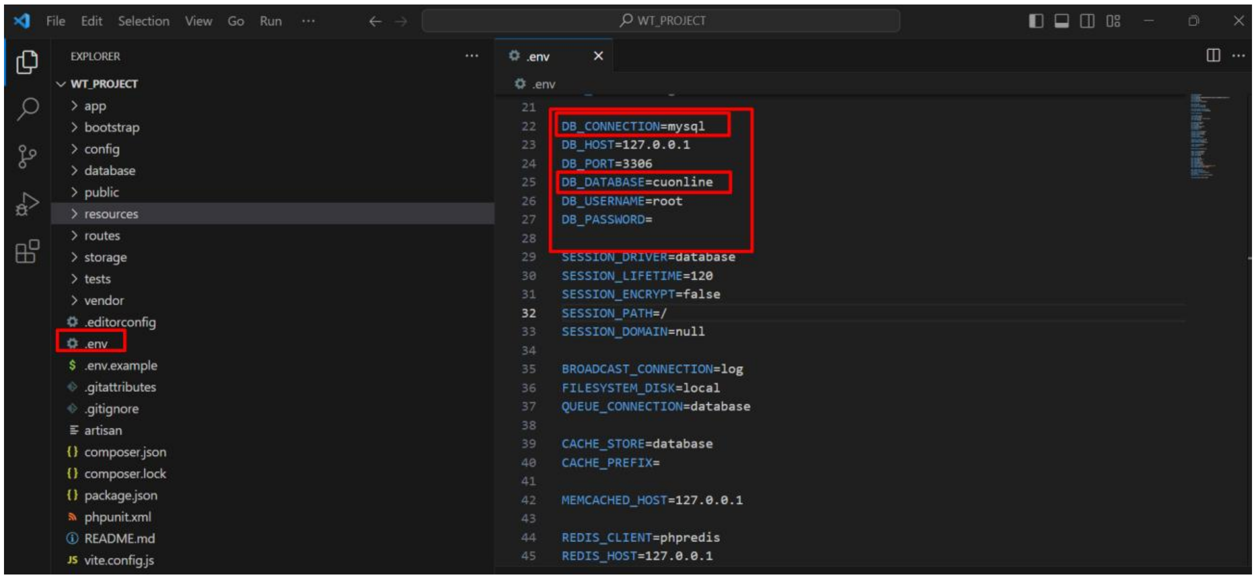
	Database	Collation	Action
☐	admincp	utf8mb4_general_ci	Check privileges
☐	adminpanel	utf8mb4_general_ci	Check privileges
☐	cuiloginsystem	utf8mb4_general_ci	Check privileges
☐	cuonline	utf8mb4_general_ci	Check privileges

Step 2: MySQL Database Configuration

In Laravel 11, there is a default database connection using SQLite, but if we want to use MySQL instead, we need to add a MySQL connection with the database name, username, and password to the `.env` file.

.env

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

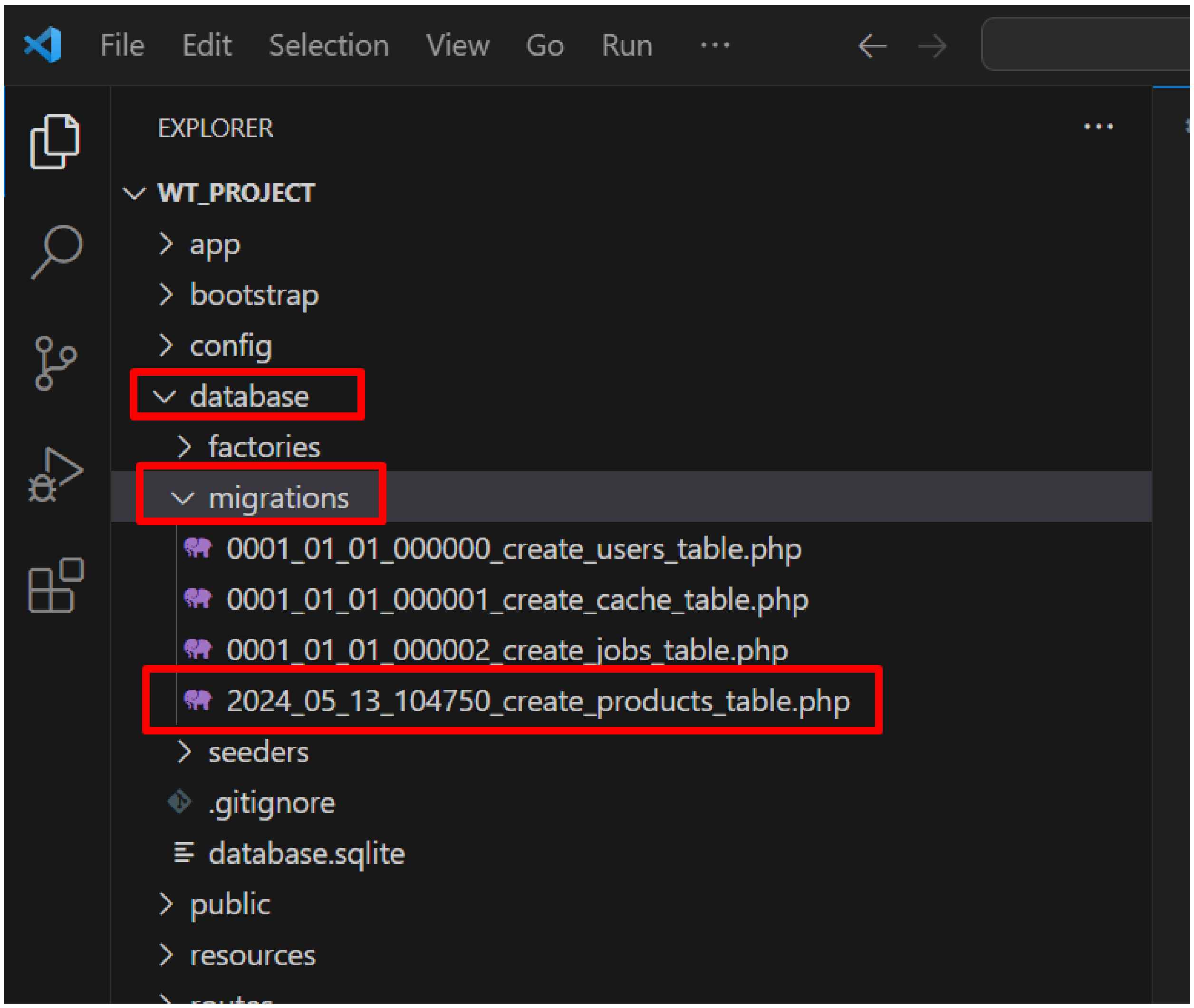


```

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=cuonline
DB_USERNAME=root
DB_PASSWORD=

SESSION_DRIVER=database
SESSION_LIFETIME=120
SESSION_ENCRYPT=false
SESSION_PATH=/
SESSION_DOMAIN=null

BROADCAST_CONNECTION=log
FILESYSTEM_DISK=local
QUEUE_CONNECTION=database

CACHE_STORE=database
CACHE_PREFIX=

MEMCACHED_HOST=127.0.0.1

REDIS_CLIENT=phpredis
REDIS_HOST=127.0.0.1
```

Step 3: Create Migration

In the third step, we will create a "products" table with "name" and "details" columns using Laravel migration. So, let's use the following command to create a migration file:

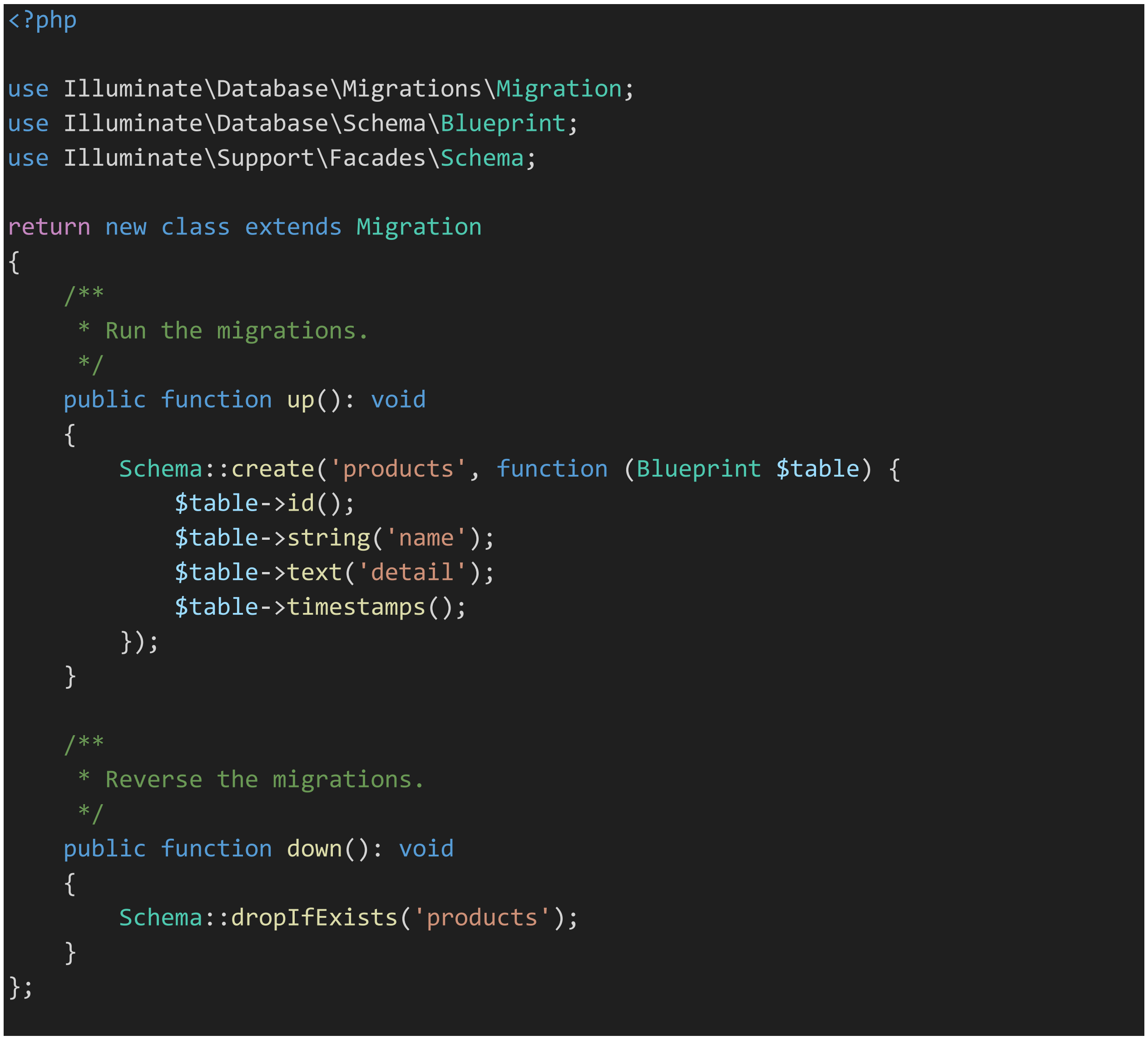
```
php artisan make:migration create_products_table --create=products
```


File
Edit
Selection
View
Go
Run
EXPLORER
WT_PROJECT
app
bootstrap
config
database
factories
migrations
0001_01_01_000000_create_users_table.php
0001_01_01_000001_create_cache_table.php
0001_01_01_000002_create_jobs_table.php
2024_05_13_104750_create_products_table.php
seeders
.gitignore
database.sqlite
public
resources

After executing the above command, you will find a file in the following path: "database/migrations".

You have to put the code below in your migration file to create the products table.

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('products', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->text('detail');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('products');
    }
};
```



```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('products', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->text('detail');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('products');
    }
};
```

Now you have to run this migration by the following command:

```
php artisan migrate
```

Step 4: Create Form Request Validation Class

In this step, we will create a form request validation class for the `store()` and `update()` methods in the controller. In this class, we will define validation rules and use it in the controller file. So, let's create it.

```
php artisan make:request ProductStoreRequest
```

Just put the below code in your request class:

app/Http/Requests/ProductStoreRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class ProductStoreRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     */
    public function authorize(): bool
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array|string>
     */
    public function rules(): array
    {
        return [
            'name' => 'required',
            'detail' => 'required'
        ];
    }
}
```

Now, let's do the same thing for the Update Request Class.

```
php artisan make:request ProductUpdateRequest
```

Just put the below code in your request class:

app/Http/Requests/ProductUpdateRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class ProductUpdateRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     */
    public function authorize(): bool
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array|string>
     */
    public function rules(): array
    {
        return [
            'name' => 'required',
            'detail' => 'required'
        ];
    }
}
```

Step 5: Create Controller and Model

In this step, now we should create a new resource controller named ProductController. So run the below command to create the new controller. Below is the controller for creating the resource controller.

```
php artisan make:controller ProductController --resource --model=Product
```

After the following command, you will find a new file at this path: "app/Http/Controllers/ProductController.php".

In this controller, seven methods will be created by default as follows:

1)index()

2)create()

3)store()

4)show()

5)edit()

6)update()

7)destroy()

So, let's copy the code below and put it in the ProductController.php file.

app/Http/Controllers/ProductController.php

```php
<?php

namespace App\Http\Controllers;

use App\Models\Product;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Http\Response;
use Illuminate\View\View;
use App\Http\Requests\ProductStoreRequest;
use App\Http\Requests\ProductUpdateRequest;

class ProductController extends Controller
{
    /**
     * Display a listing of the resource.
     */
    public function index(): View
    {
        $products = Product::latest()->paginate(5);

        return view('products.index', compact('products'))
                    ->with('i', (request()->input('page', 1) - 1) * 5);
    }

    /**
     * Show the form for creating a new resource.
     */
    public function create(): View
    {
        return view('products.create');
    }

    /**
     * Store a newly created resource in storage.
     */
    public function store(ProductStoreRequest $request): RedirectResponse
    {
        Product::create($request->validated());

        return redirect()->route('products.index')
                         ->with('success', 'Product created successfully.');
    }
    /**
     * Display the specified resource.
     */
    public function show(Product $product): View
    {
        return view('products.show',compact('product'));
    }

    /**
     * Show the form for editing the specified resource.
     */
    public function edit(Product $product): View
    {
        return view('products.edit',compact('product'));
    }

    /**
     * Update the specified resource in storage.
     */
    public function update(ProductUpdateRequest $request, Product $product): 
RedirectResponse
    {
        $product->update($request->validated());

        return redirect()->route('products.index')
                        ->with('success','Product updated successfully');
    }

    /**
     * Remove the specified resource from storage.
     */
    public function destroy(Product $product): RedirectResponse
    {
        $product->delete();

        return redirect()->route('products.index')
                        ->with('success','Product deleted successfully');
    }
}
```

So, let's update the Product model code as follows:

app/Models/Product.php

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Product extends Model
{
    use HasFactory;

    protected $fillable = [
        'name',
        'detail',
    ];
}
```

Step 6: Add Resource Route

Here, we need to add a resource route for the product CRUD application. So, open your `routes/web.php` file and add the following route.

routes/web.php

```
<?php

use Illuminate\Support\Facades\Route;

use App\Http\Controllers\ProductController;

Route::get('/', function () {
    return view('welcome');
});

Route::resource('products', ProductController::class);
```

Step 7: Update AppServiceProvider

Here, we will use bootstrap 5 for pagination. so, we need to import it on AppServiceProvider.php file. let's update it.

app/Provides/AppServiceProvider.php

```
<?php

namespace App\Providers;

use Illuminate\Support\ServiceProvider;
use Illuminate\Pagination\Paginator;

class AppServiceProvider extends ServiceProvider
{
    /**
     * Register any application services.
     */
    public function register(): void
    {

    }

    /**
     * Bootstrap any application services.
     */
    public function boot(): void
    {
        Paginator::useBootstrapFive();
    }
}
```

Step 8: Add Blade Files

In the last step, we need to create only blade files. So, primarily, we have to create a layout file and then a new folder called "products." After that, we create blade files for the CRUD app. So, finally, you have to create the following blade files below:

1) layout.blade.php

2) index.blade.php

3) create.blade.php

4) edit.blade.php

5) show.blade.php

So let's just create the following file and put the code below in it.

resources/views/products/layout.blade.php

```
<!DOCTYPE html>
<html>
<head>
    <title>Laravel 11 CRUD Application </title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css
" rel="stylesheet">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.5.1/css/all.min.css" />
</head>
<body>

<div class="container">
    @yield('content')
</div>

</body>
</html>
```

resources/views/products/index.blade.php

```
@extends('products.layout')

@section('content')

<div class="card mt-5">
  <h2 class="card-header">Laravel 11 CRUD Example</h2>
  <div class="card-body">

        @session('success')
            <div class="alert alert-success" role="alert"> {{ $value }}
</div>
        @endsession

        <div class="d-grid gap-2 d-md-flex justify-content-md-end">
            <a class="btn btn-success btn-sm" href="{{
route('products.create') }}"> <i class="fa fa-plus"></i> Create New
Product</a>
        </div>

        <table class="table table-bordered table-striped mt-4">
            <thead>
                <tr>
                    <th width="80px">No</th>
                    <th>Name</th>
                    <th>Details</th>
                    <th width="250px">Action</th>
                </tr>
            </thead>

            <tbody>
            @forelse ($products as $product)
                <tr>
                    <td>{{ ++$i }}</td>
                    <td>{{ $product->name }}</td>
                    <td>{{ $product->detail }}</td>
                    <td>
                        <form action="{{ route('products.destroy',$product-
>id) }}" method="POST">

                            <a class="btn btn-info btn-sm" href="{{
route('products.show',$product->id) }}"><i class="fa-solid fa-list"></i>
Show</a>

                            <a class="btn btn-primary btn-sm" href="{{
route('products.edit',$product->id) }}"><i class="fa-solid fa-pen-to-
square"></i> Edit</a>

                            @csrf
                            @method('DELETE')

                            <button type="submit" class="btn btn-danger btn-
sm"><i class="fa-solid fa-trash"></i> Delete</button>
                        </form>
                    </td>
                </tr>
            @empty
                <tr>
                    <td colspan="4">There are no data.</td>
                </tr>
            @endforelse
            </tbody>

        </table>
        {!! $products->links() !!}

  </div>
</div>
@endsection
```

resources/views/products/create.blade.php

```
@extends('products.layout')

@section('content')

<div class="card mt-5">
  <h2 class="card-header">Add New Product</h2>
  <div class="card-body">

    <div class="d-grid gap-2 d-md-flex justify-content-md-end">
        <a class="btn btn-primary btn-sm" href="{{ route('products.index')
}}"><i class="fa fa-arrow-left"></i> Back</a>
    </div>

    <form action="{{ route('products.store') }}" method="POST">
        @csrf

        <div class="mb-3">
            <label for="inputName" class="form-
label"><strong>Name:</strong></label>
            <input
                type="text"
                name="name"
                class="form-control @error('name') is-invalid @enderror"
                id="inputName"
                placeholder="Name">
            @error('name')
                <div class="form-text text-danger">{{ $message }}</div>
            @enderror
        </div>

        <div class="mb-3">
            <label for="inputDetail" class="form-
label"><strong>Detail:</strong></label>
            <textarea
                class="form-control @error('detail') is-invalid @enderror"
                style="height:150px"
                name="detail"
                id="inputDetail"
                placeholder="Detail"></textarea>
            @error('detail')
                <div class="form-text text-danger">{{ $message }}</div>
            @enderror
        </div>
        <button type="submit" class="btn btn-success"><i class="fa-solid fa-
floppy-disk"></i> Submit</button>
    </form>
  </div>
</div>
@endsection
```

resources/views/products/edit.blade.php

```
@extends('products.layout')

@section('content')

<div class="card mt-5">
  <h2 class="card-header">Edit Product</h2>
  <div class="card-body">

    <div class="d-grid gap-2 d-md-flex justify-content-md-end">
        <a class="btn btn-primary btn-sm" href="{{ route('products.index')
}}"><i class="fa fa-arrow-left"></i> Back</a>
    </div>

    <form action="{{ route('products.update',$product->id) }}" method="POST">
        @csrf
        @method('PUT')

        <div class="mb-3">
            <label for="inputName" class="form-
label"><strong>Name:</strong></label>
            <input
                type="text"
                name="name"
                value="{{ $product->name }}"
                class="form-control @error('name') is-invalid @enderror"
                id="inputName"
                placeholder="Name">
            @error('name')
                <div class="form-text text-danger">{{ $message }}</div>
            @enderror
        </div>

        <div class="mb-3">
            <label for="inputDetail" class="form-
label"><strong>Detail:</strong></label>
            <textarea
                class="form-control @error('detail') is-invalid @enderror"
                style="height:150px"
                name="detail"
                id="inputDetail"
                placeholder="Detail">{{ $product->detail }}</textarea>
            @error('detail')
                <div class="form-text text-danger">{{ $message }}</div>
            @enderror
        </div>
        <button type="submit" class="btn btn-success"><i class="fa-solid fa-
floppy-disk"></i> Update</button>
    </form>

  </div>
```

```
</div>
@endsection
```

resources/views/products/show.blade.php

```
@extends('products.layout')

@section('content')

<div class="card mt-5">
  <h2 class="card-header">Show Product</h2>
  <div class="card-body">

    <div class="d-grid gap-2 d-md-flex justify-content-md-end">
        <a class="btn btn-primary btn-sm" href="{{ route('products.index')
}}"><i class="fa fa-arrow-left"></i> Back</a>
    </div>

    <div class="row">
        <div class="col-xs-12 col-sm-12 col-md-12">
            <div class="form-group">
                <strong>Name:</strong> <br/>
                {{ $product->name }}
            </div>
        </div>
        <div class="col-xs-12 col-sm-12 col-md-12 mt-2">
            <div class="form-group">
                <strong>Details:</strong> <br/>
                {{ $product->detail }}
            </div>
        </div>
    </div>

  </div>
</div>
@endsection
```

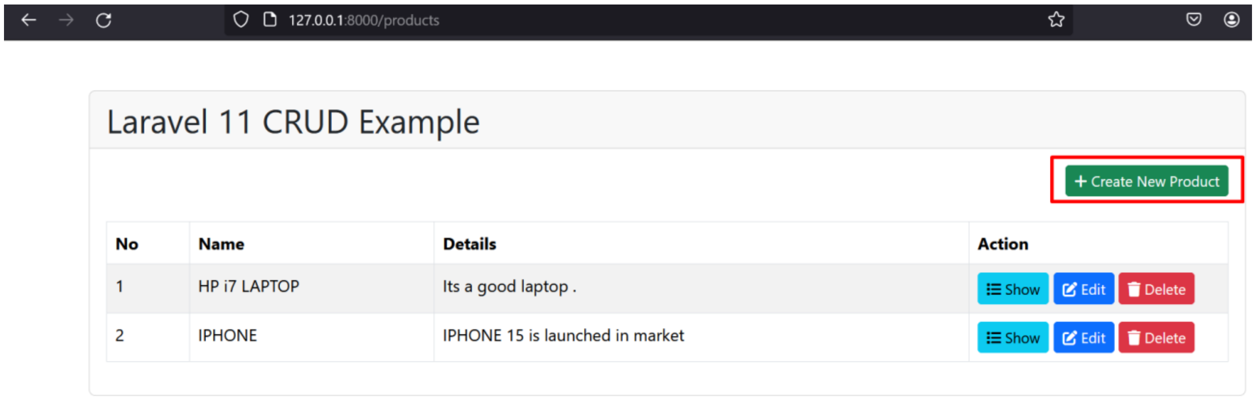
Run Laravel App:

All the required steps have been done, now you have to type the given below command and hit enter to run the Laravel app:

```
php artisan serve
```

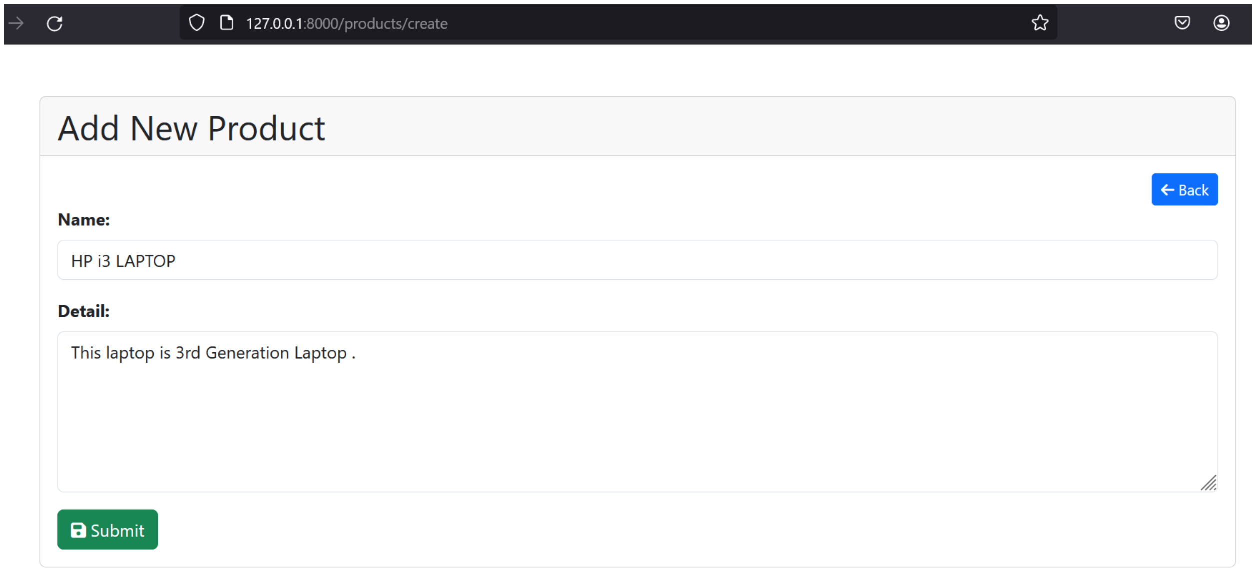
Now, Go to your web browser, type the given URL and view the app output:

```
http://localhost:8000/products
```


127.0.0.1:8000/products

Laravel 11 CRUD Example

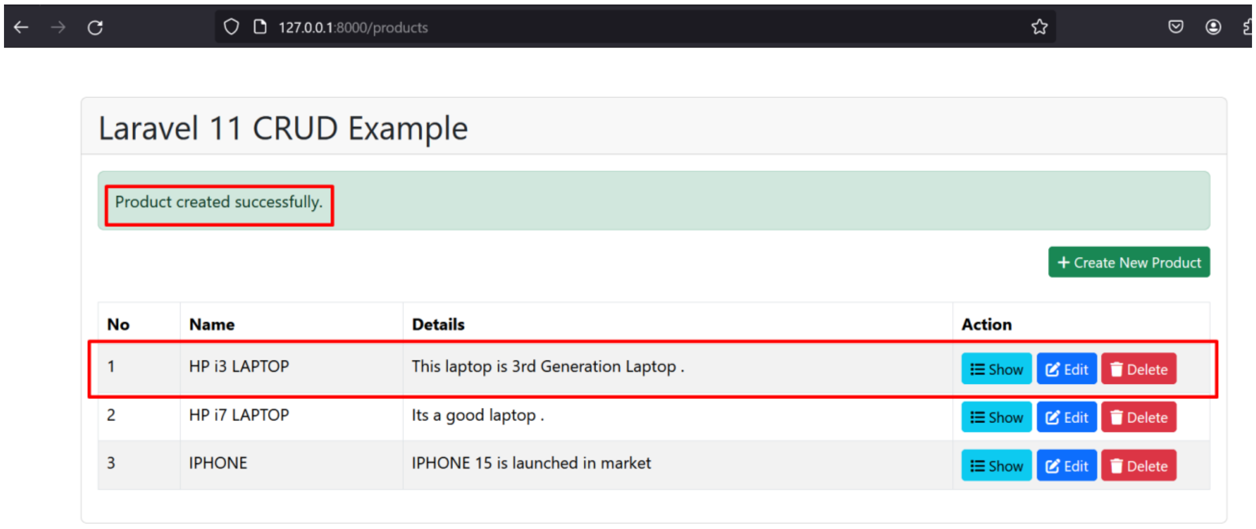
No	Name	Details	Action
1	HP i7 LAPTOP	Its a good laptop .	Show Edit Delete
2	IPHONE	IPHONE 15 is launched in market	Show Edit Delete


127.0.0.1:8000/products/create

Add New Product

Name:
HP i3 LAPTOP

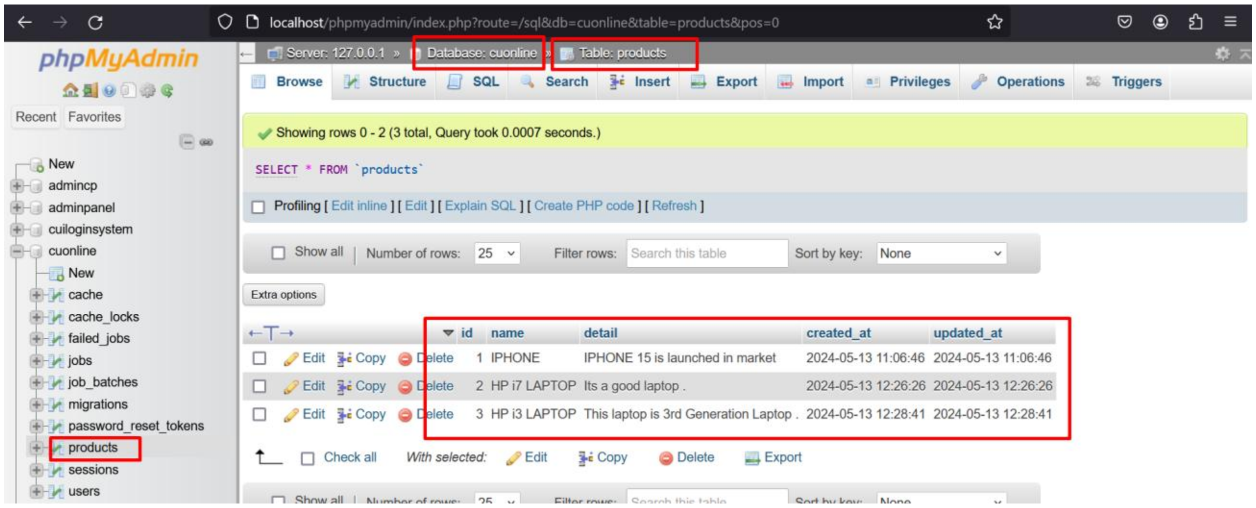
Detail:
This laptop is 3rd Generation Laptop .


127.0.0.1:8000/products

Laravel 11 CRUD Example

Product created successfully.

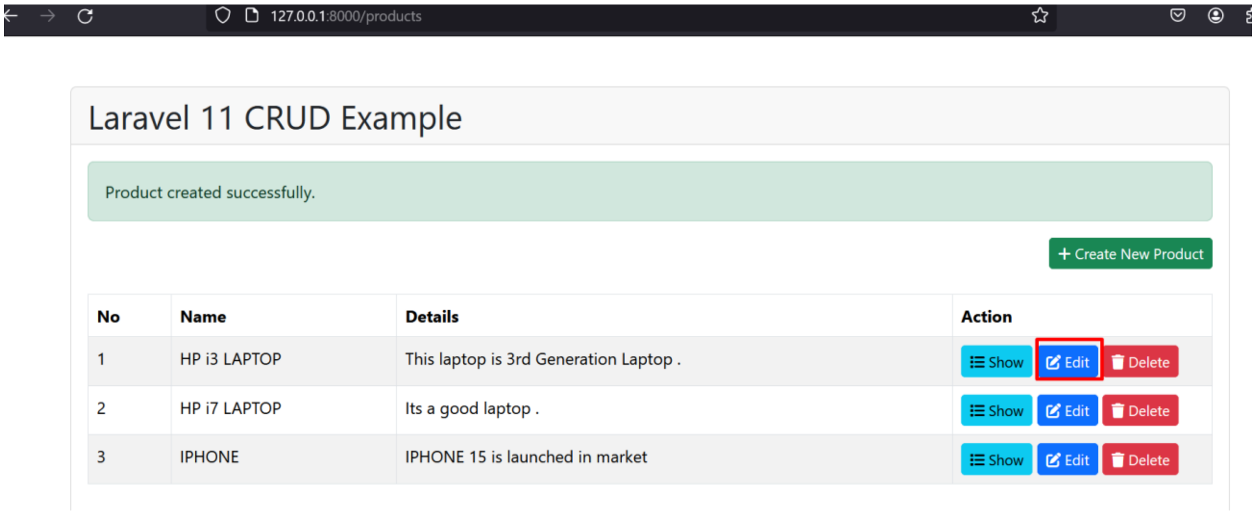
No	Name	Details	Action
1	HP i3 LAPTOP	This laptop is 3rd Generation Laptop .	Show Edit Delete
2	HP i7 LAPTOP	Its a good laptop .	Show Edit Delete
3	IPHONE	IPHONE 15 is launched in market	Show Edit Delete


localhost/phpmyadmin/index.php?route=/sql&db=cuonline&table=products&pos=0
Server: 127.0.0.1 » Database: cuonline » Table: products
Browse Structure SQL Search Insert Export Import Privileges Operations Triggers
New
admincp
adminpanel
cuiloginsystem
cuonline
New
cache
cache_locks
failed_jobs
jobs
job_batches
migrations
password_reset_tokens
products
sessions
users

Showing rows 0 - 2 (3 total, Query took 0.0007 seconds.)

```
SELECT * FROM `products`
```

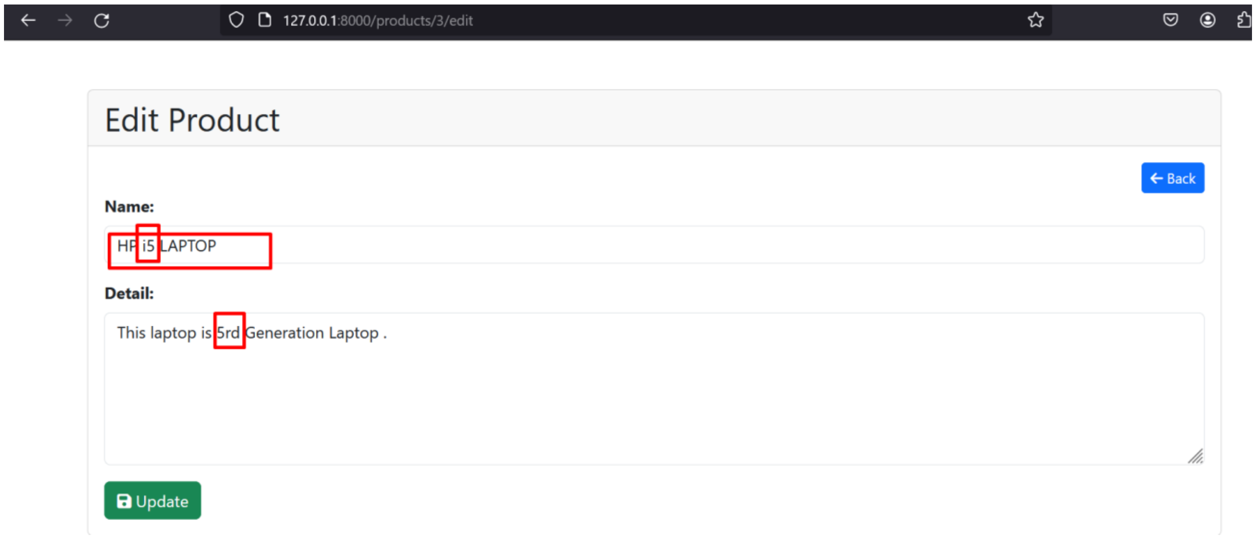
id	name	detail	created_at	updated_at
1	IPHONE	IPHONE 15 is launched in market	2024-05-13 11:06:46	2024-05-13 11:06:46
2	HP i7 LAPTOP	Its a good laptop .	2024-05-13 12:26:26	2024-05-13 12:26:26
3	HP i3 LAPTOP	This laptop is 3rd Generation Laptop .	2024-05-13 12:28:41	2024-05-13 12:28:41



Laravel 11 CRUD Example

Product created successfully.

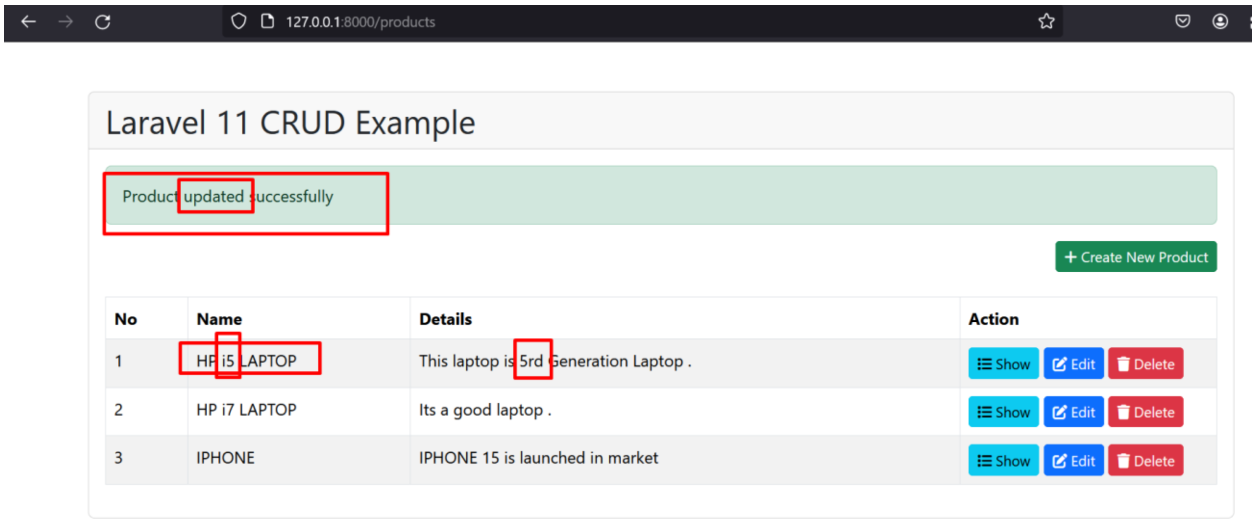
No	Name	Details	Action
1	HP i3 LAPTOP	This laptop is 3rd Generation Laptop .	Show Edit Delete
2	HP i7 LAPTOP	Its a good laptop .	Show Edit Delete
3	IPHONE	IPHONE 15 is launched in market	Show Edit Delete



Edit Product

Name: HP i5 LAPTOP

Detail: This laptop is 5rd Generation Laptop .

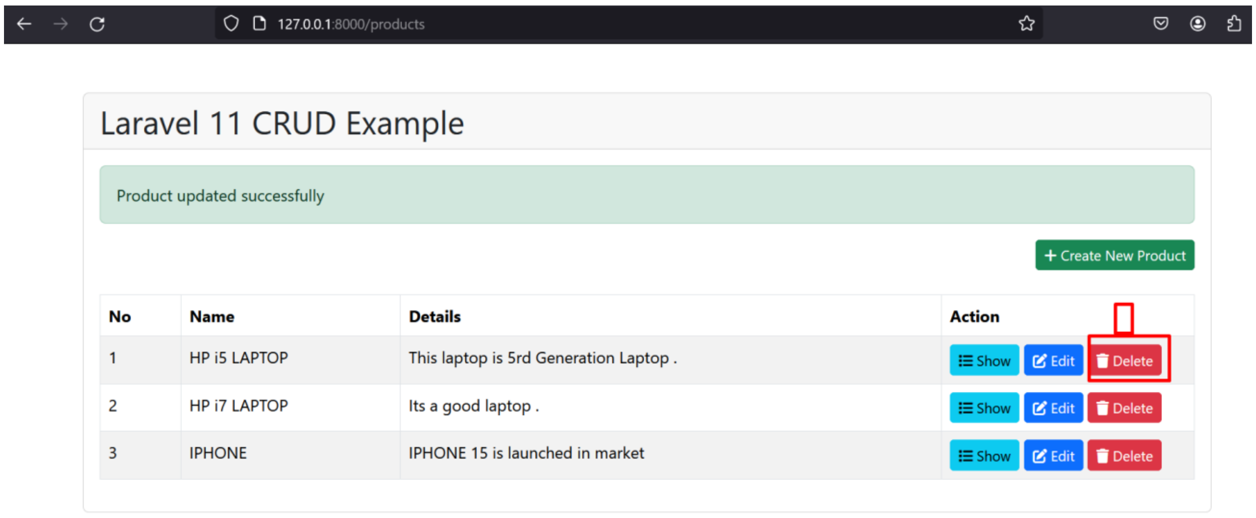


127.0.0.1:8000/products

Laravel 11 CRUD Example

Product updated successfully

No	Name	Details	Action
1	HP i5 LAPTOP	This laptop is 5rd Generation Laptop .	Show Edit Delete
2	HP i7 LAPTOP	Its a good laptop .	Show Edit Delete
3	IPHONE	IPHONE 15 is launched in market	Show Edit Delete

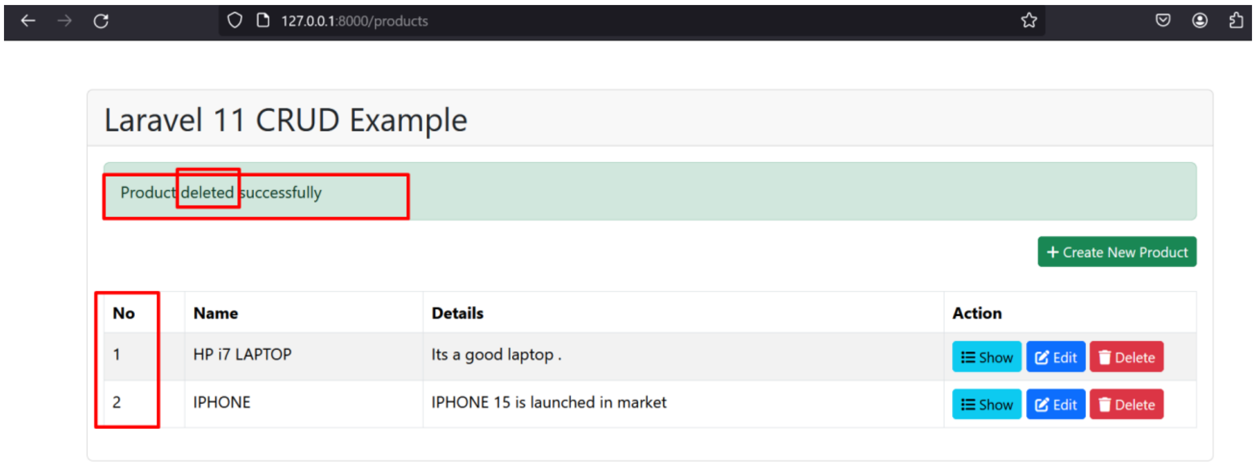


127.0.0.1:8000/products

Laravel 11 CRUD Example

Product updated successfully

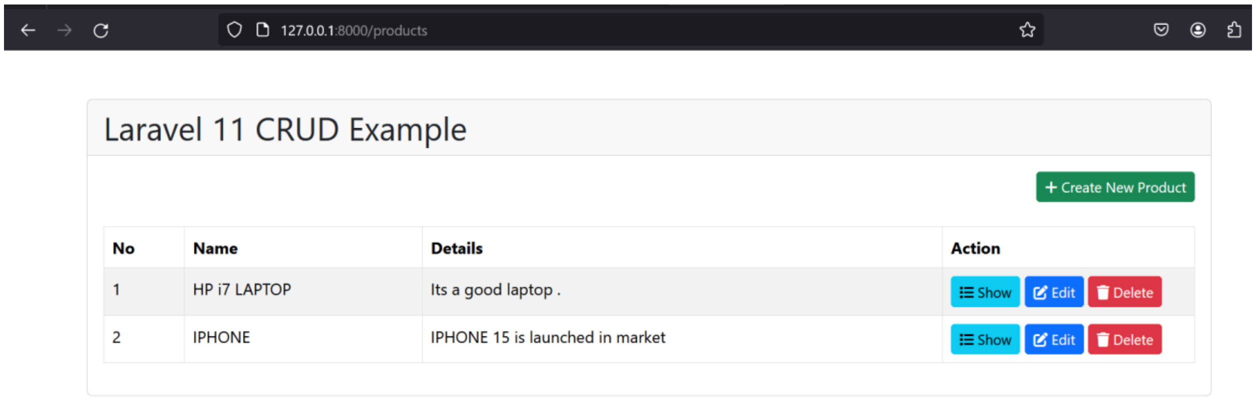
No	Name	Details	Action
1	HP i5 LAPTOP	This laptop is 5rd Generation Laptop .	Show Edit Delete
2	HP i7 LAPTOP	Its a good laptop .	Show Edit Delete
3	IPHONE	IPHONE 15 is launched in market	Show Edit Delete


127.0.0.1:8000/products

Laravel 11 CRUD Example

Product deleted successfully

No	Name	Details	Action
1	HP i7 LAPTOP	Its a good laptop .	Show Edit Delete
2	IPHONE	IPHONE 15 is launched in market	Show Edit Delete


127.0.0.1:8000/products

Laravel 11 CRUD Example

No	Name	Details	Action
1	HP i7 LAPTOP	Its a good laptop .	Show Edit Delete
2	IPHONE	IPHONE 15 is launched in market	Show Edit Delete