### Project: \*\*Project-Eloquent\*\*

#### \*\*Project Overview:\*\*

"Project-Eloquent" is a Laravel-based application built to manage and manipulate user data using the Eloquent ORM. The application provides essential CRUD operations along with advanced features like pagination, search across multiple fields, and dynamic data management.

---

#### \*\*Process of Building the Project:\*\*

1. \*\*Install Laravel:\*\*

Start by creating a new Laravel project using Composer:

```bash

composer create-project --prefer-dist laravel/laravel Project-Eloquent

cd Project-Eloquent

```

2. \*\*Set Up Database:\*\*

Configure your `.env` file with the database credentials:

```plaintext

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=your\_database

DB\_USERNAME=your\_username

DB\_PASSWORD=your\_password

```

3. \*\*Create Model, Migration, and Controller:\*\*

Generate a model with a migration file and resource controller:

```bash

php artisan make:model User -mcr

```

This command will generate:

- `User.php` model

- `create\_users\_table.php` migration

- `UserController.php` resource controller

4. \*\*Define Database Schema:\*\*

Update the migration file in `database/migrations/xxxx\_xx\_xx\_create\_users\_table.php`:

```php

public function up()

{

Schema::create('users', function (Blueprint $table) {

$table->id();

$table->string('name');

$table->string('email')->unique();

$table->integer('age')->nullable();

$table->string('city')->nullable();

$table->timestamps();

});

}

```

Run the migration to create the table:

```bash

php artisan migrate

```

5. \*\*Implement CRUD in the Controller:\*\*

Inside `UserController.php`, implement the CRUD methods (`index`, `create`, `store`, `edit`, `update`, `destroy`).

6. \*\*Set Up Routes:\*\*

In `routes/web.php`, define resource routes:

```php

use App\Http\Controllers\UserController;

Route::resource('user', UserController::class);

```

7. \*\*Create Views:\*\*

Use Blade templates to create views:

- `layout.blade.php`: A master layout

- `home.blade.php`: Displays the list of users with search and pagination

- `adduser.blade.php`: Form for adding a new user

- `edituser.blade.php`: Form for editing user data

8. \*\*Search and Pagination:\*\*

Implement search in the `index` method of `UserController`:

```php

public function index(Request $request)

{

$query = User::query();

if ($request->has('search')) {

$query->where('name', 'like', '%' . $request->search . '%')

->orWhere('email', 'like', '%' . $request->search . '%')

->orWhere('age', 'like', '%' . $request->search . '%')

->orWhere('city', 'like', '%' . $request->search . '%');

}

$users = $query->paginate(10);

return view('home', compact('users'));

}

```

---

#### \*\*Flow of the Application:\*\*



### \*\*Explanation of the Application Flow\*\*

The diagram above illustrates the flow of the "Project-Eloquent" Laravel application. Here’s a breakdown of the core components and their interactions:

1. \*\*Routes:\*\*

- Routes define the entry points of the application. They map URLs to controller actions (like `index`, `create`, `store`, `show`, `edit`, `update`, and `destroy`).

2. \*\*Controllers:\*\*

- Controllers handle the incoming requests. They process user input, retrieve data from models, and pass data to the views. Each controller method corresponds to a specific task, such as displaying data, creating new entries, or updating existing records.

3. \*\*Models:\*\*

- Models represent the application's data structure. They interact with the database using Laravel's Eloquent ORM, making it easy to perform CRUD operations on the `users` table.

4. \*\*Views:\*\*

- Views are Blade templates that present data to the user. They handle the front-end display of information, forms, and user interactions.

5. \*\*Database:\*\*

- The database is where all user data is stored. The model interacts with the database using Eloquent ORM, enabling seamless data retrieval, updates, and deletions.

### \*\*Functionalities:\*\*

- \*\*CRUD Operations:\*\* Basic functionality includes creating, reading, updating, and deleting user records.

- \*\*Pagination:\*\* Users can navigate through a large dataset easily.

- \*\*Search Functionality:\*\* The application supports searching across multiple fields (name, email, age, city).

- \*\*Form Validation:\*\* The application supports validating form data from both (client side & server side) for all fields (name, email, age, city).