Step 1: Create a New Laravel Project

If you don't have a Laravel project, you can create one with:

```
composer create-project --prefer-dist laravel/laravel laravelname
```

Step 2: Set Up Database

Make sure your .env file is configured with the correct database settings.

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

Then, run the migration to create the default tables:

```
php artisan migrate
```

Step 3: Create a Model and Migration

Create a model with a migration for the entity you want to manage. For this example, let's use Post as an entity:

```
php artisan make:model Post -m
```

This will create two files:

- app/Models/Post.php (Model)
- database/migrations/YYYY_MM_DD_create_posts_table.php (Migration)

In the migration file, define the fields that you want in the posts table.

```
public function up()
{
    Schema::create('posts', function (Blueprint $table) {
        $table->id();
        $table->string('title');
        $table->text('content');
        $table->timestamps();
    });
}
```

Run the migration to create the table:

```
php artisan migrate
```

Step 4: Create a Controller

Generate a controller for handling CRUD operations:

```
php artisan make:controller PostController
```

In PostController.php, define methods for CRUD operations.

Step 5: Define CRUD Methods in Controller

Edit the PostController.php file:

```
<?php

namespace App\Http\Controllers;

use App\Models\Post;

use Illuminate\Http\Request;

class PostController extends Controller

{

  // Display a listing of the posts

  public function index()

  {

    $posts = Post::all();

    return view('posts.index', compact('posts'));

  }

  // Show the form for creating a new post

  public function create()

  {
```

```
  return view('posts.create');
}
// Store a newly created post in the database
public function store(Request $request)
{
   $request->validate([
      'title' => 'required|max:255',
      'content' => 'required',
   ]);
   Post::create([
      'title' => $request->title,
      'content' => $request->content,
   ]);
   return redirect()->route('posts.index');
}
// Show the form for editing the specified post
public function edit(Post $post)
{
   return view('posts.edit', compact('post'));
}
// Update the specified post in the database
public function update(Request $request, Post $post)
{
        $request->validate([
            'title' => 'required|max:255',
            'content' => 'required',
        ]);
        $post->update([
            'title' => $request->title,
            'content' => $request->content,
        ]);
        return redirect()->route('posts.index');
    }
    // Remove the specified post from the database
    public function destroy(Post $post)
    {
        $post->delete();
        return redirect()->route('posts.index');
    }
}
```

Step 6: Define Routes

Define the routes for the CRUD operations in routes/web.php:

```
use App\Http\Controllers\PostController;
Route::resource('posts', PostController::class);
```

This single line will generate routes for all the CRUD operations:

- GET /posts for index

- GET /posts/create for create
- POST /posts for store
- GET /posts/{post}/edit for edit
- PUT/PATCH /posts/{post} for update
- DELETE /posts/{post} for destroy

Step 7: Create Views for CRUD Operations

Create views for displaying and managing posts. You can place these views in the resources/views/posts folder.

1. **Index View (resources/views/posts/index.blade.php)**

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Posts</title>

</head>

<body>

  <h1>Posts</h1>

  <a href="{{ route('posts.create') }}">Create New Post</a>

  <ul>

    @foreach ($posts as $post)

      <li>

        {{ $post->title }}

        <a href="{{ route('posts.edit', $post->id) }}">Edit</a>

        <form action="{{ route('posts.destroy', $post->id) }}" method="POST"
style="display:inline;">
```

```
                @csrf

                @method('DELETE')

                <button type="submit">Delete</button>

            </form>

        </li>

    @endforeach

  </ul>

</body>
```

2. **Create View (resources/views/posts/create.blade.php)**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Create Post</title>
</head>
<body>
  <h1>Create Post</h1>
  <form action="{{ route('posts.store') }}" method="POST">
    @csrf
    <label for="title">Title:</label>
    <input type="text" name="title" id="title" required>
    <label for="content">Content:</label>
    <textarea name="content" id="content" required></textarea>
    <button type="submit">Save</button>
  </form>
</body>
</html>
```

3. **Edit View (resources/views/posts/edit.blade.php)**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Edit Post</title>
```

```
</head>
<body>
  <h1>Edit Post</h1>
  <form action="{{ route('posts.update', $post->id) }}" method="POST">
    @csrf
    @method('PUT')
    <label for="title">Title:</label>
    <input type="text" name="title" id="title" value="{{ $post->title }}" required>
    <label for="content">Content:</label>
    <textarea name="content" id="content" required>{{ $post->content }}</textarea>
    <button type="submit">Update</button>
  </form>
</body>
</html>
```

Step 8: Test the Application

Now you can test the CRUD operations by visiting the following URLs:

- **List all posts**: http://your-app-url/posts
- **Create a post**: http://your-app-url/posts/create
- **Edit a post**: http://your-app-url/posts/{id}/edit