Day 2 Course Content

Hour 1:

- Introduction to Controllers
    - What are controllers?
    - Creating a basic controller
    - Returning views from controllers

Hour 2:

- Introduction to Views
    - What are views?
    - Using Blade templating engine
    - Creating and using Blade templates

Introduction to Controller

What are Controllers?

- In the MVC framework, the letter 'C' stands for Controller. It acts as a directing traffic between Views and Models.
- Key Responsibilities of Controllers
  - Request Handling: Controllers receive and process HTTP requests from users.
  - Business Logic: They contain the logic for handling user actions and processing data.
  - Model Interaction: Controllers interact with models to retrieve or manipulate data.
  - Response Generation: They return responses to the client, which could be views, JSON responses, redirects, etc.

Types of Controller

- **Basic Controller:**

  Command:

  php artisan make:controller BasicController

- **Resource Controller:**

  Command:

  php artisan make:controller UserController --resource

Creating a Basic Controller

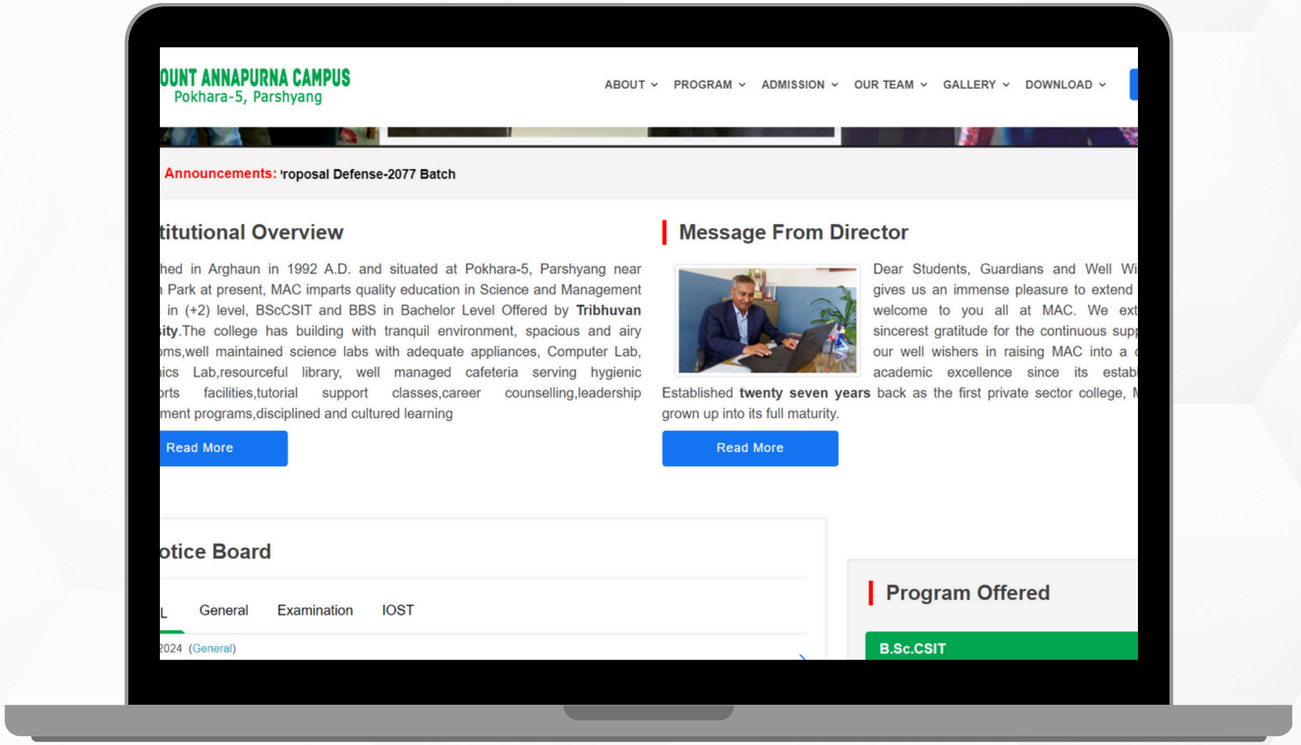
- Use Artisan CLI to create a controller.
- Command: php artisan make:controller MyController
- This creates a new file in the app/Http/Controllers directory.

```bash
php artisan make:controller MyController
```

Introduction to view

What is view?

- Views are templates used to display data.
- They separate the presentation logic from the application logic.



Returning View From Controllers

- **Controllers can return views.**

```php
<?php

namespace App\Http\Controllers;

class MyController extends Controller
{
    public function index()
    {
        return view('welcome'); // This will return the resources/views/welcome.blade.php
    }
}
```

Fun Time

What is the golden rule in programming?

Fun Time

What is the golden rule in programming?

"If it works, don't touch it."

Stay Motivated!

"Nothing is impossible. The word itself says 'I'm possible!"

– Audrey Hepburn

Using Blade Templating Engine

- Blade is Laravel's powerful templating engine.
- It makes it easy to create dynamic web pages.

Blade Syntax Overview

- **Blade templates use .blade.php extension.**

```
php
Copy code


@extends('layout')


@section('content')
<h1>Hello, World!</h1>
@endsection
```

Creating and Using Blade Templates

- **Create a Blade template: resources/views/myview.blade.php**
- **Example:**

```
php
Copy code
<!DOCTYPE html>
<html>
<head>
  <title>My First Blade Template</title>
</head>
<body>
  <h1>Hello, World!</h1>
</body>
</html>
```

Including Blade Templates in Views

- **Use @include to include templates.**
- **Example:**

```
php
Copy code:
@include('header')
<h1>Welcome</h1>
@include('footer')
```

Blade Template Directives

- Condition Directives
  - @if, @elseif, @else, and @endif
- Loop Directives
  - @foreach, @endforeach, @forelse, and @endforelse
  - @while and @endwhile,@for and @endfor
- Component and Layout Directives
  - @include,@extends,@yield and @section

Hands-on: Create a Basic Controller and View

1. Controller
 -› what is controller
 -› basic controller
 -› create basic controller

2. Views
 -› what are views
 -› Blade template
 -› Create blade file

3. Routing from controller
  -› return view from controller
    e.g public function index(){
 return view('first')
 }

4. Blade directives

@include, @extends, @yield, @section

@include -› Arko file ma bhaya ko content lai current file ma include garni

@extends -› kun file lai extends garni (main content lai kun file ma rakhni)

@yield -› main content lai rakhni tham or area or second i.e. another file bata aya ko content lai current file ma rakhni

@section -› Inside this we write the main content of the file.

e.g.

in views › app.blade.php

```
@include('layouts.header')
@include('layouts.navbar')
 @yield('main-section')
@include('layouts.footer')
```

in first.blade.php

```
@extends('app')
@section('main-section')

// main content here

@endsection
```