BUC3131 · ADVANCED WEB APPLICATION DEVELOPMENT

Topic 1

Introduction to Advanced Web Development and Web Frameworks

Department of Information Systems
Makerere University Business School

Mutebi Bashir · Balunywa Ali · Joy Tiko · Dr. Ali Mwase

ROADMAP

What We Will Cover

Six ideas, building in order, from how the web works to how Laravel is structured.

How the web works

01

Client, server, frontend, backend

What a framework is

02

And why we rarely build from a blank page

The frameworks landscape

03

Laravel, Django, Rails, Spring, React

Why this course chooses Laravel

04

Concrete reasons, not tradition

The MVC pattern

05

Model, View, Controller — and Routes

Meet Laravel

06

Where it came from, what it gives you

From Static Pages to Advanced Applications

A static site tells a visitor something. An advanced application has a conversation with them.

STATIC WEBSITE

— Content is fixed until a developer edits the files

— No user accounts or personalisation

— Data changes only by manually editing a file

— No notifications

— Example: a printed flyer, shown on a screen

ADVANCED WEB APPLICATION

— Interacts with a database to store and update data

— Users register, log in, and manage a profile

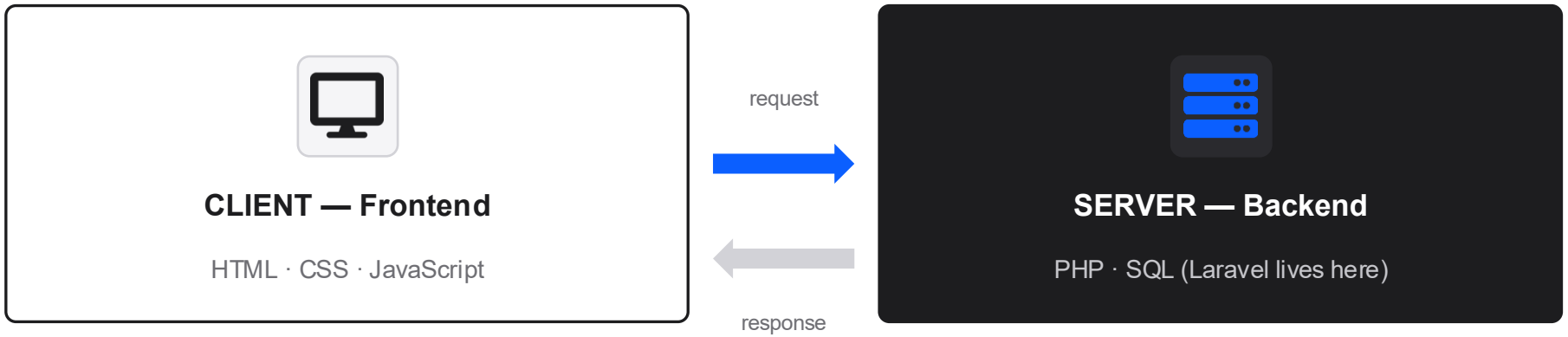
— Data updates automatically through the database

— Sends automated email / in-app notifications

— Example: a student portal that logs in and remembers

Every Application Has Two Sides

The client asks; the server answers. Laravel lives on the server side.



Laravel is a backend framework — but it connects smoothly with whatever frontend you build.

Why Advanced Web Development Matters

Almost every sector now depends on web applications for efficiency and data management.

Business

E-commerce, inventory, payments

Education

Student portals, online learning

Healthcare

Booking, patient management

Government / NGOs

Service delivery, reporting

Uganda's digital economy

Business, education, government going digital

Employability

Market-ready skills, global standards

What Exactly Is a Framework?

A collection of tools, libraries, and predefined structures that help developers build applications more efficiently and consistently.

Rather than starting from a blank page, a developer starts from a ready-made foundation — one that already knows how to route a request, talk to a database, and defend against common attacks.

- Reduces repetitive coding
- Enforces good practice
- Built-in security, routing, database & templating tools
- Easier to maintain and scale

A Ready-Made Toolkit

You still need skill to use the tools well — but the toolkit saves time, reduces mistakes, and pushes you toward good practice by default.

One Problem, Many Toolkits

Popular frameworks, grouped by the language they are written in.

PHP	Laravel · Symfony · CodeIgniter
Python	Django · Flask
JavaScript / TypeScript	Angular · React · Vue.js · Express.js
Java & C#	Spring Boot · ASP.NET Core
Ruby & Go	Ruby on Rails · Gin

Why Use a Framework at All?

Faster development

Prebuilt auth, routing, DB connections

Improved security

Built-in defence against SQLi, XSS, CSRF

Better maintainability

Clean, modular structure (MVC)

Scalability

Efficient handling as usage grows

Community support

Documentation, tutorials, shared solutions

A house built without a framework: cutting your own wood, moulding your own bricks. Possible — but slow and error-prone.

So — Why Laravel?

Six concrete reasons this course is built around Laravel.

Clean, readable syntax

Less code, easier teamwork

MVC architecture

Organised, scalable structure

Built-in security & auth

SQLi, XSS, CSRF protection by default

Eloquent ORM

Expressive PHP instead of raw SQL

Rich package ecosystem

Spatie · Livewire · Laravel PDF

Real-world relevance

Widely used in Uganda and globally

Meet Laravel

2011

YEAR CREATED

Created by Taylor Otwell

An open-source PHP framework celebrated for simplicity, elegance and performance.

Think of Laravel as a well-built car

You could build a car from scratch — cut your own metal, forge your own engine — but that would take forever. Laravel gives you the car already built; you fuel it, drive it, and customise it to your style.

It follows the Model-View-Controller (MVC) pattern and streamlines routing, authentication, and database interactions.

MVC: Three Roles, One Clear Job Each

Model – View – Controller: a way of organising code so every part has one responsibility.

MODEL

Talks to the database. Saves, fetches, and updates records; enforces the rules about that data.

The accountant who keeps records.

VIEW

What the user actually sees — the layout, text, buttons and forms in the browser.

The designer of the system.

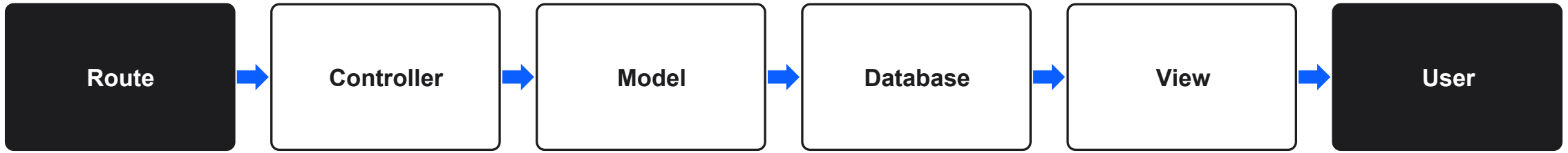
CONTROLLER

Receives the request, asks the Model for data if needed, and tells the View what to show.

The manager in the middle.

The Request Lifecycle

Routes are the GPS of the application — they match a URL to the Controller action that should handle it.



Almost every bug in this course traces back to exactly one of these six steps going wrong.

Laravel's Core Toolkit

MVC is the architecture. These are the tools you will actually use, week after week.

Eloquent ORM

Work with the database using expressive PHP, not raw SQL.

Blade Templating

Laravel's lightweight templating engine for building Views.

Artisan CLI

The command-line tool for migrations, testing and code generation.

Authentication

Ready-made registration, login and password handling.

Routing

A clean way to map URLs to the Controller actions that handle them.

RECAP

Key Takeaways

01	Advanced web apps are dynamic, database-driven, secure, maintainable and scalable — not just static pages
02	A framework is a ready-made, tested foundation — it saves time, but you still need to learn to use it well
03	Laravel is chosen for its clean syntax, security, Eloquent ORM, ecosystem and real-world relevance in Uganda
04	MVC separates data (Model), presentation (View) and logic (Controller) — Routes direct traffic between them
05	Every request follows one repeatable lifecycle: Route → Controller → Model → Database → View → User

COMING UP NEXT — TOPIC 2

Setting Up the Development Environment

Installing and configuring Laravel, understanding the file structure, using Composer, and setting up a local development environment.

Bring one question

About MVC, or about a framework from today's landscape slide you'd like compared to Laravel.