

Intermediate Laravel Applications and Linux Deployment

Routing, validation, authentication and REST APIs

FA-0637 · 3 days

Software Development · DevOps, Cloud & Infrastructure

Develop a small Laravel application with validated requests, authentication and REST endpoints, then rehearse deployment in a prepared Linux environment.

Introduction

This three-day course builds on practical Laravel fundamentals. It uses one prepared application to explore routing, controllers, middleware, validation, authentication and REST APIs before a guided Linux deployment rehearsal.

Participants implement selected features and review deployment considerations rather than attempting every authentication package or a complete production environment. Application security and performance depend on configuration, testing and operations; the course does not promise mastery, industrial readiness or guaranteed improvements.

Learning outcomes

The course teaches participants to:

- Use route constraints, model binding and resource controllers for a small CRUD workflow.
- Trace middleware and implement a selected request-validation rule.
- Distinguish authentication from authorisation and apply a simple access-control check.
- Build and exercise JSON REST endpoints with appropriate validation and error handling.
- Choose an API authentication approach based on the application's actual requirements.

- Inspect a Linux deployment's runtime, document root, permissions, secrets and debug configuration.
- Rehearse a version-controlled deployment and identify service/caching considerations in a training environment.

Prerequisites

- Good grasp and experience with Laravel fundamentals (routing, MVC, Eloquent ORM).
- Basic command-line proficiency in POSIX systems.
- Knowledge of HTML, CSS, JavaScript basics.
- Understanding PHP basics and databases (MySQL preferred).

A prepared compatible Laravel/PHP/Composer and database stack, plus an authorised disposable Linux server or equivalent training environment.

Training outline

1. Day 1 — Routing, Controllers and Requests

Routing

- Route parameters and constraints
- Named routes and URL generation
- Route grouping with middleware for security and access control
- Routing exercises for the prepared application.

Resource Controllers

- Creating and managing resource controllers
- Leveraging route model binding
- Efficient usage of resourceful routing for CRUD operations

Middleware and Lifecycle

- Trace the middleware/request lifecycle and inspect the selected release's configuration.
- Create and test a small custom middleware component.
- Middleware groups and parameterization for controlled access

Form Requests and Validation

- Fundamentals of form validation in Laravel
- Creating and using custom request classes for validation
- Effective handling and displaying of validation errors

2. Day 2 — Authentication and APIs

User Authentication and Access Control

- Use a maintained starter kit compatible with the selected release; recognise Breeze/Jetstream in older applications rather than treating them as the universal current default.
- User registration, login, and password management
- Distinguish authentication from authorisation; implement a small role/permission example and review customisation limits.

RESTful APIs

- Constructing API routes and controllers
- JSON response structure and request handling with clear validation.
- Consistent error responses and logging that does not expose secrets.

API Authentication

- Compare Sanctum for appropriate first-party/simple API use with Passport when OAuth2 is required. Discuss token/JWT concepts without equating all packages or implementing multiple authentication stacks.
- Practise one selected authentication flow; inspect transport, token lifecycle and access checks.
- Managing tokens and handling authentication errors effectively

3. Day 3 — Linux Deployment Rehearsal

- Prepare a supported Linux distribution and a compatible PHP/runtime stack; use Ubuntu or another suitable supported environment rather than an unspecified legacy CentOS release.
- Git-based deployment workflow and introductory automation considerations, not an unreviewed production pipeline.
- Utilizing SSH and CLI commands for efficient server management
- Review caching, configuration and security settings; measure performance rather than promise optimisation.

- Survey NGINX, Supervisor and Redis roles and use only components required by the prepared training application.

Inspect the prepared application's public document root, runtime compatibility, writable directories, secret handling and production debug setting. Build, deploy and review a sample change in the training environment; document remaining operational requirements.