LARAVEL TRAINING PLAN

Practical Backend Development Bootcamp

Recommended Duration	6 weeks / 30 sessions / 45-60 training hours
Intensive Option	12 training days with condensed labs and daily homework
Target Level	Beginner to intermediate PHP developers
Stack Target	Laravel 13.x, PHP 8.3+, Composer, MySQL/PostgreSQL, Git

Training philosophy
This plan is built around real development work: every module ends with a practical task, every week has a deliverable, and the final project simulates a production-ready Laravel system with roles, APIs, testing, queues, and deployment readiness.

Prepared for: OH Tech Solutions / Laravel Training Program
Date: May 2026

1. Program Overview

Goal: take trainees from basic Laravel understanding to building, testing, securing, and deploying a real backend application. The plan balances fundamentals, business logic, APIs, code quality, and production readiness.

2. Learning Outcomes

- Create and configure Laravel projects using Composer, environment files, routing, controllers, views, and service providers.
- Design relational database structures using migrations, seeders, factories, Eloquent models, relationships, scopes, and query optimization basics.
- Build secure authentication, authorization, policies, roles, middleware, validation, and form request layers.
- Create REST APIs with resources, authentication tokens, pagination, filtering, and consistent response formats.
- Use queues, jobs, notifications, events, mail, file storage, scheduling, and caching in practical business workflows.
- Apply testing, debugging, logging, code style, Git workflow, deployment preparation, and environment separation.
- Complete a final project that can be used as a portfolio case study or internal company starter system.

3. Prerequisites

Required	Recommended	Setup Before Day 1
Basic PHP	OOP in PHP	PHP 8.3+
HTML/CSS basics	Basic JavaScript	Composer
SQL basics	Git basics	MySQL or PostgreSQL
Command line basics	REST API concepts	VS Code / PhpStorm
Problem-solving mindset	MVC concept awareness	Postman or Insomnia

Version note
This plan targets Laravel 13.x. The practical concepts also apply to Laravel 11/12, but new projects should use a supported release and compatible PHP version.

4. Recommended 6-Week Structure

Week	Main Focus	Core Topics	Lab Deliverable	Assessment
1	Laravel Foundations	Setup, MVC, routing, controllers, Blade, config, request lifecycle	Mini content website with dynamic pages	Project setup + route/controller quiz
2	Database & Eloquent	Migrations, seeders, factories, models, relationships, query builder	Products/ categories module with CRUD	Database design review
3	Auth, Validation & Roles	Auth, middleware, validation, form requests, policies, file upload	Multi-role admin panel basics	Security checklist review
4	APIs & Business Logic	REST APIs, API resources, Sanctum-style tokens, services, repositories	Public API for products/orders/users	API testing challenge
5	Advanced Laravel Workflows	Queues, jobs, notifications, mail, events, caching, scheduling, storage	Automated email + queued reports workflow	Debugging and performance task
6	Testing, Deployment & Capstone	Pest/PHPUnit, Pint, logging, deployment checklist, envs, final project	Final CRM/LMS-style Laravel app	Final presentation + code review

5. Daily Session Plan

Format per session: 15 minutes concept explanation, 20-30 minutes live coding, 30-45 minutes trainee lab, 10 minutes review and questions.

Session	Topic	What Trainees Learn	Output
1	Environment & Laravel installation	Install PHP, Composer, Laravel installer, configure .env, run first project	A clean Laravel app committed to Git
2	Routing, controllers & request lifecycle	Route methods, route groups, parameters, named routes, controller actions	Simple pages with route/controller structure
3	Blade & layouts	Layouts, components, partials, assets, forms, CSRF	Responsive dashboard shell
4	Configuration & project organization	Config files, env separation, helper files, folder structure, naming	Clean project skeleton
5	Migrations & seeders	Schema design, indexes, foreign keys, seeders, factories	Users/products/ categories database
6	Eloquent models & relationships	HasOne, HasMany, BelongsTo, Many-to-Many, eager loading	Products linked to categories and orders
7	CRUD implementation	Resource controllers, forms, validation, flash messages, pagination	Product CRUD module
8	Validation & Form Requests	Validation rules, custom messages, authorization in form requests	Clean create/update validation layer

9	Authentication	Starter kits, login/register flow, password reset, profile update	Authenticated dashboard
10	Roles, policies & middleware	Role design, gates, policies, route protection, admin/user separation	Multi-role access control
11	File upload & storage	Local/public disks, validation, naming, image/file records	Upload product image or user document
12	Services and business logic	Service classes, action classes, avoiding fat controllers	Order service / registration service
13	REST API fundamentals	API routes, JSON responses, status codes, resources, collections	Products API with pagination
14	API authentication & security	Token-based auth, rate limiting, CORS, API validation	Protected user API endpoints
15	Filtering, search & query optimization	Scopes, filters, sorting, eager loading, indexes	Searchable product/order list
16	Events, listeners & notifications	Domain events, notifications database/mail channels	Notify admin after new order
17	Queues & jobs	Queue drivers, retry, failed jobs, supervisor concept	Queued email/report job
18	Mail & scheduled tasks	Mailables, templates, scheduler, cron concept	Daily summary email task
19	Caching and	Cache facade,	Optimized dashboard

	performance basics	remember, config/route/view cache, N+1 review	query
20	Testing fundamentals	Feature tests, unit tests, factories, test database	Tests for auth + CRUD
21	Debugging & logging	Logs, exceptions, Telescope-style workflows, error handling	Debugging exercise report
22	Security checklist	CSRF, XSS, mass assignment, authorization, file upload risks	Security audit on project
23	Code quality & Git workflow	Pint, naming, branches, pull requests, commits, reviews	Clean PR with fixes
24	Deployment preparation	Env files, permissions, storage link, queues, scheduler, backups	Deployment checklist
25	Capstone planning	Requirements, DB design, routes, API list, roles, milestones	Final project technical plan
26	Capstone build 1	Core auth, roles, dashboard, base models	Working skeleton
27	Capstone build 2	CRUD modules, validation, file upload	Main modules complete
28	Capstone build 3	API, queues, notifications, reports	Advanced features complete
29	Testing and refactoring	Feature tests, policy tests, performance fixes,	Tested project

		code review	
30	Final demo and review	Presentation, feedback, next steps, portfolio packaging	Final presentation + GitHub repository

6. Optional 12-Day Intensive Track

Use this version when the training must be delivered quickly. Each day should include 3-4 hours of guided training and 1-2 hours of homework.

Day	Focus	Deliverable
1	Setup, Laravel basics, routing, controllers, Blade	Working Laravel app with dashboard layout
2	Migrations, seeders, factories, Eloquent basics	Database schema and sample data
3	Relationships, CRUD, pagination, validation	Products/categories CRUD
4	Authentication, roles, middleware, policies	Admin/user protected areas
5	File uploads, storage, clean controllers, services	Upload workflow + service layer
6	REST APIs, resources, filtering, API auth	Protected API endpoints
7	Queues, jobs, mail, notifications, scheduler	Queued notification/report workflow
8	Caching, performance, debugging, logs	Optimized dashboard and error report
9	Testing, factories, feature tests, code style	Minimum test suite
10	Deployment, environment, permissions, production checklist	Deployment-ready configuration checklist
11	Capstone build and review	Core capstone features
12	Final polish, demo, code review,	Final project presentation

	next steps	

7. Final Capstone Project

Suggested project: Mini CRM / LMS Management System

- Multi-role authentication: admin, staff, and normal user.
- Dashboard with summary cards and filtered tables.
- CRUD modules: users, courses/products, categories, orders/registrations, payments or notes.
- File upload: profile image, attachment, or course material.
- API endpoints for mobile/frontend integration with pagination and validation.
- Queued notification or email after a key action.
- Basic reports with date filtering and totals.
- Feature tests for authentication, CRUD, authorization, and API responses.
- Deployment checklist and GitHub README with installation steps.

Capstone acceptance rule
The final project should not be judged only by whether it works. It should also be reviewed for structure, naming, security, validation, authorization, test coverage, and readability.

8. Practical Exercises

Exercise	Skill	Task	Success Criteria
Route Map	Routing	Create grouped admin/user routes with names and middleware.	Routes are readable and protected.
Database Design	Migrations	Design products, categories, orders, users, and order_items tables.	Correct relationships and indexes.
CRUD Module	Controllers + Blade	Build a complete CRUD with validation and flash messages.	No duplicate logic in views/controllers.
Authorization Gate	Security	Prevent normal users from editing admin-only resources.	Unauthorized requests return correct response.
API Challenge	REST APIs	Create filtered, paginated API endpoint with resource output.	Consistent JSON format and status codes.
Queue Workflow	Queues	Send a queued email or generate a queued report.	Job retries and failure handling are configured.
Testing Task	Testing	Write tests for login, create, update, delete, and API list.	Tests pass locally and in CI-ready command.
Deployment Checklist	DevOps	Prepare production environment checklist.	Includes env, cache, storage, queues, scheduler, backups.

9. Assessment Rubric

Area	Weight	What to Check	Excellent Standard

Laravel fundamentals	15%	Routes, controllers, views, config, project structure	Clean MVC structure and readable route naming
Database & Eloquent	20%	Migrations, relationships, queries, seeders	Correct schema and optimized relationship usage
Security & authorization	20%	Validation, policies, middleware, mass assignment, uploads	No obvious authorization or validation gaps
API quality	15%	Resources, status codes, filtering, pagination, auth	Consistent, documented API responses
Advanced workflows	10%	Queues, jobs, notifications, caching, scheduler	Feature works asynchronously and safely
Testing & code quality	10%	Tests, Pint, naming, Git commits	Readable code with meaningful tests
Final presentation	10%	Demo, README, deployment checklist, explanation	Clear explanation of architecture and decisions

10. Instructor Delivery Notes

- Start every session by connecting the concept to a real business use case such as CRM, LMS, ERP, e-commerce, or admin dashboards.
- Avoid teaching Laravel as isolated commands. Show where each concept belongs inside a production project.
- Review Git commits weekly. Code review is part of the training, not a separate activity.
- Use one shared capstone domain throughout the training so trainees see how concepts connect.
- Give trainees starter tasks first, then improvement tasks: validation, authorization, API, tests, performance, and deployment.
- Require a README for every trainee project with setup steps, env variables, database commands, and screenshots.

11. Recommended Tools

Category	Tools	Purpose
Core	PHP 8.3+, Composer, Laravel Installer	Create and run Laravel projects
Database	MySQL or PostgreSQL, TablePlus/phpMyAdmin	Build and inspect database structures
Editor	VS Code or PhpStorm	Development and debugging
API	Postman or Insomnia	Test API endpoints
Local Dev	Laravel Herd, Valet, Laragon, Sail, Docker	Local environment options
Quality	Laravel Pint, Pest/PHPUnit	Style and testing
Deployment	Nginx/Apache, Supervisor, Cron, GitHub Actions optional	Production readiness

12. Production Readiness Checklist

Environment: APP_ENV=production, APP_DEBUG=false, Correct APP_URL, Separate database credentials, Secure file permissions

Performance: config:cache, route:cache, view:cache, optimized queries, pagination for large lists

Security: CSRF enabled, authorization policies, validated uploads, no secrets in Git, rate limits on sensitive endpoints

Operations: queue worker configured, scheduler cron configured, logs monitored, backups enabled, rollback plan documented

13. Official References

- Laravel Documentation: https://laravel.com/docs
- Laravel 13.x Release Notes: https://laravel.com/docs/13.x/releases
- Laravel 13.x Upgrade Guide: https://laravel.com/docs/13.x/upgrade
- Laravel Framework Releases on GitHub: https://github.com/laravel/framework/releases

End of Training Plan