

Django Web Development

Course Duration ~75 Hours · 10 Weeks · Live Online · Build-Along · Deployed Applications

Curriculum reviewed: July 2026 — recent Django releases changed the framework materially; this syllabus tracks the current line

[ORM in Depth](#)
[Admin Back Office](#)
[HTMX & Partials](#)
[Security](#)
[Background Work](#)
[Deployment](#)

In short: Ship the whole product yourself. Django is batteries included — ORM, migrations, auth, admin, forms, security defaults, templating — one framework, one developer, a whole application, with two decades of production behind it. Two things most courses undersell get real modules here: the **admin as a production back office** generated from your models — something no other framework in our catalogue offers — and **server-rendered interactivity**: HTMX plus template partials now deliver SPA-like behaviour without a separate frontend codebase, a JSON API layer, or client-side state management — for small teams and internal tools, often the correct answer. The honest decision rule: choose Django for an application with users, permissions, a database and an internal back office; choose our FastAPI course for APIs and AI/ML serving; choose the Next.js/React stack when the frontend is the product. The syllabus tracks recent releases: a built-in background task framework taught honestly against a full queue, template partials, native Content Security Policy support, composite primary keys, and maturing async support — plus LTS and upgrade planning as first-class material. Across 10 modules you ship a multi-tenant SaaS app, an admin back office, an HTMX dashboard, a content platform, an API layer, an N+1 optimisation workshop, a zero-downtime migration and a defended capstone. The AI question, answered honestly: AI writes Django fluently and gets N+1 queries, table-locking migrations, misplaced permission checks and disabled security defaults wrong — the value is architecture, review, debugging and operational judgment.

Duration	10 weeks · ~75 hours · live + hands-on	Format	Live online · build-along · deployed applications
Prerequisite	Working Python and basic HTML/CSS. No web framework or database experience required.	Curriculum	10 modules, from architecture to deployment, operations and capstone
Portfolio	7 deployed, production-style applications plus a defended capstone	Ideal for	Fresh IT graduates, founders and freelancers shipping alone, and teams planning a Django LTS upgrade

Why Learn Django When AI Writes Views?

AI generates Django views and models fluently and gets the things that bite wrong — query patterns that trigger N+1 problems, migrations that lock a production table, permission checks in the wrong layer, security defaults quietly disabled. The value is architecture, review, debugging and operational judgment. You'll use AI tooling as an accelerator you review, and learn the failure patterns generated Django code repeats — so you ship its speed without shipping its mistakes.

Course Highlights

- Every module leads with what you build and ship, not what you "learn about"
- The ORM in depth — N+1 kills, zero-downtime migrations, composite keys, reading real queries
- The admin as a production back office — customised, optimised, secured
- SPA-like interactivity without an SPA: HTMX plus template partials, taught honestly
- Background work with judgment: the built-in task framework vs a full queue — the boundary taught
- Security by default — and how people accidentally switch it off; native CSP covered
- LTS and upgrade planning as first-class material for teams on older projects
- AI tooling as an accelerator you review, never a substitute for understanding

Hands-on Projects & Portfolio

- **Multi-Tenant SaaS Application** — custom user model, roles, permissions and billing hooks
- **Internal Back-Office Tool** — built on a heavily customised admin, delivered fast
- **Interactive Dashboard** — HTMX and template partials: no SPA, no separate API

- **Content Platform** — search, media handling, caching and SEO-correct rendering
- **API Layer Addition** — authentication, pagination and docs on an existing application
- **Query Optimisation Workshop** — find and fix N+1 problems and slow pages with profiling evidence
- **Zero-Downtime Migration** — on a table with real data volume, without locking production
- **Capstone** — production application, deployed, secured, monitored and defended on architecture

Tools & Stack

Python · Django · Django REST Framework · HTMX · PostgreSQL · Redis · Celery · Alpine.js · Tailwind CSS · pytest · factory_boy · Gunicorn · Uvicorn · Nginx · Docker · Sentry · GitHub Actions · AWS · Git

Career Outcomes

Django's advantage is completeness and stability — two decades in production, settled conventions, and a single developer able to deliver a full application with authentication, an admin back office and a database layer without assembling a stack from parts. That makes it disproportionately common in startups, agencies, internal tooling, government and content-driven products — and Django skills date far more slowly than frontend framework skills. This course targets:

- Python Web Developer · Django Developer · Backend Developer · Full-Stack Developer
- Software Engineer in product teams · Technical founder shipping alone · Freelance/agency developer delivering complete applications

Who This Course Is For

- Final-year students and fresh IT graduates targeting Python or full-stack roles
- Python developers moving from scripts, automation or data work into applications
- **Developers who want to ship a complete product alone** — founders, freelancers, small teams
- Backend developers from PHP, Java, .NET or Node adding Python web development
- Data scientists and analysts who need a real application around their work
- **Teams maintaining older Django projects planning an LTS upgrade** — upgrade planning is first-class material here

Module 1: Django Architecture & Project Setup

7 Hours

YOU SHIP

- A running multi-app project with environment-split settings, clean URL routing and a structure that survives growth — the skeleton every later module builds on

TOPICS

- The request/response cycle; apps and project structure that survives growth
- Settings management across environments; URLs and views; modern dependency management
- The design philosophy behind the framework's opinions

Assessment: Quiz + Lab

Module 2: The ORM in Depth

9 Hours

YOU SHIP

- The query optimisation workshop — N+1 problems found and fixed with evidence — plus a zero-downtime migration executed on a table with real data volume

TOPICS

- Models, fields and relationships; migrations including data migrations and zero-downtime patterns
- Querysets and lazy evaluation; select_related and prefetch_related; aggregation and annotation; transactions; raw SQL escape hatches
- Indexing; composite primary keys; reading the queries your code actually generates

Assessment: Quiz + Lab

Module 3: Views, Forms & Templates

7 Hours

YOU SHIP

- A content platform slice with forms, validation, file uploads and template partials — with a deliberate call on where generic views help and where they obscure

TOPICS

- Function and class-based views; the generic view hierarchy — when it helps versus when it obscures
- Forms and formsets; validation; file uploads
- The template language; context processors; template partials; static and media file handling

Assessment: Quiz + Lab**Module 4: Authentication, Authorisation & the Admin**

8 Hours

YOU SHIP

- An internal back-office tool on a heavily customised admin — inlines, actions, optimised lists — over a custom user model with roles, groups and object-level access

TOPICS

- The user model and customising it before it's too late; sessions and login flows; password management
- Permissions and groups; object-level access
- The admin as a production back office: customisation, inlines, actions, list optimisation — and the security considerations of exposing it at all

Assessment: Quiz + Lab**Module 5: Server-Rendered Interactivity**

7 Hours

YOU SHIP

- An interactive dashboard with partial updates, live form validation in fragments and no full reloads — built with HTMX and template partials, no SPA and no separate API

TOPICS

- Modern interactive interfaces without an SPA: HTMX for partial updates, combined with template partials
- Progressive enhancement; handling forms and validation errors in fragments
- An honest account of when this approach is better than a JavaScript frontend — and when it isn't

Assessment: Quiz + Lab**Module 6: Building APIs with Django**

7 Hours

YOU SHIP

- An API layer added to an existing application — authentication, permissions, pagination, filtering, versioning and docs — plus a written toolkit choice including the "use FastAPI instead" option

TOPICS

- When an application needs an API layer; serialisers, viewsets and routers
- Authentication and permissions for APIs; pagination, filtering, versioning and documentation
- The established REST toolkit vs lighter, typed alternatives — and vs choosing FastAPI instead

Assessment: Quiz + Lab**Module 7: Asynchronous & Background Work**

8 Hours

YOU SHIP

- A notification and processing pipeline split deliberately: fire-and-forget work on the built-in task framework, scheduled and chained work on a full queue — with the boundary documented

TOPICS

- Async views and async ORM usage and their current limits
- The built-in background task framework — honest scope: fire-and-forget email and webhooks — versus a full task queue for scheduling, chaining, rate limiting and result storage
- Caching strategies; signals and their misuse; scheduled jobs; email and notifications; long-running work that must not block a request

Assessment: Quiz + Lab

Module 8: Security & Correctness

7 Hours

YOU SHIP

- A security hardening pass on a deliberately weakened app: re-enable the disabled defaults, add CSP, fix upload handling, pass the deployment checklist

TOPICS

- What the framework protects you from by default — and how people accidentally switch it off; CSRF, XSS and SQL injection
- Content Security Policy and where it sits in middleware order; clickjacking and HTTPS settings
- Secrets management; dependency auditing; rate limiting; file upload safety; the deployment checklist

Assessment: Quiz + Lab

Module 9: Testing, Performance & Quality

7 Hours

YOU SHIP

- A fully tested feature — views, models, permissions, query-count assertions — plus a slow page diagnosed and fixed with profiling evidence and a caching layer

TOPICS

- The test framework and pytest; factories and fixtures; testing views, models and permissions; database query assertions
- Coverage that means something; profiling and query optimisation; caching layers; database tuning
- Load testing; diagnosing a slow page with evidence

Assessment: Quiz + Lab

Module 10: Deployment, Operations & Capstone

8 Hours

YOU SHIP

- Your capstone: one production-grade application — containerised, statically served, migrated under traffic, logged, monitored, CI/CD-deployed — defended on architecture, plus an LTS upgrade plan

TOPICS

- WSGI and ASGI servers; static and media files in production; containerisation; database migrations under real traffic
- Environment configuration; logging and error tracking; monitoring; scaling patterns; CI/CD
- Upgrade planning across LTS lines — stepwise, not version skipping; capstone design, build, deploy and defense

Assessment: Capstone Project + Review