

FastAPI Development

Course Duration ~60 Hours · 8 Weeks · Live Online · Build-Along · Deployed Services

Curriculum reviewed: July 2026 — FastAPI, Pydantic and SQLAlchemy move fast; this syllabus tracks the current versions

Async Correctness

Pydantic

Dependency Injection

AI/ML Serving

Security

Observability

In short: Build the Python APIs everything else calls — typed, async, documented and fast enough to sit in front of a model. Every model, agent and RAG pipeline eventually needs to be an API somebody else can call: this is that layer, which is why serving AI and ML workloads is a first-class module — token-by-token streaming, server-sent events, long-running inference and job handoff, timeouts and cancellation, backpressure against rate-limited providers, cost and token accounting, batching and warm-up, and exposing capabilities as tools for agents. The headline technical theme is **async correctness**: the most common production failure in FastAPI apps is blocking the event loop — a sync driver or SDK call inside an async route that works on a laptop and collapses under load. You'll learn when async def is right and when a regular def route is, what the threadpool does behind the scenes, why async database sessions are the only sane default, and how to diagnose a blocked loop with evidence. Across 9 modules you ship a typed CRUD API, an OAuth2-secured service, an LLM gateway with cost tracking, a model-serving endpoint proven under load, a long-running job API, and an observable capstone. Our AI courses mention FastAPI as their serving layer — this is the deep treatment; our Node.js course is the same discipline in a different ecosystem. The AI question, answered honestly: AI scaffolds endpoints convincingly and gets the async model wrong in ways that only appear under concurrency — the value is architecture, review, debugging and operational judgment.

Duration	8 weeks · ~60 hours · live + hands-on	Format	Live online · build-along · deployed services
Prerequisite	Working Python. Type hints, async and web fundamentals taught from first principles. No prior API or backend experience required.	Curriculum	9 modules, from modern Python to deployment and capstone — AI/ML serving first-class
Portfolio	7 deployed, production-style services plus a defended capstone	Ideal for	Fresh IT graduates, Python developers shipping scripts or notebooks, and data scientists/ML engineers serving their own models

Why Learn FastAPI When AI Scaffolds Endpoints?

AI scaffolds FastAPI endpoints convincingly and gets the async model wrong in ways that only appear under concurrency — a blocking call inside an async route looks fine locally and collapses at load. 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 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"
- Async correctness as a headline theme — the blocking-call failure most tutorials never explain
- AI/ML serving as a first-class module: streaming, cancellation, backpressure, cost accounting
- Dependency injection taught as the framework's best architectural feature
- Async database layer by default — with the honest account of when SQLAlchemy convenience becomes constraint
- Real failure modes named: validation confusion, CORS, serialisation surprises
- Observability from day one: structured logs, correlation IDs, tracing with OpenTelemetry
- AI tooling as an accelerator you review, never a substitute for understanding

Hands-on Projects & Portfolio

- **Typed CRUD API** — async database access, migrations, layered structure and full test coverage
- **OAuth2-Secured Service** — roles, scopes, refresh tokens and rate limiting
- **LLM Gateway API** — streaming responses, timeouts, cancellation, retries and per-request cost tracking

- **Model-Serving Endpoint** — warm-up, batching and concurrency limits under load test
- **Long-Running Job API** — submit, poll, webhook callback, and a worker doing the actual work
- **Async-Correctness Workshop** — find the blocking calls in a slow service and fix them, with before/after load-test evidence
- **Observable Deployment** — structured logs, metrics and tracing on a deployed service
- **Capstone** — production API, containerised, monitored and defended on architecture

Tools & Stack

Python · FastAPI · Pydantic · SQLAlchemy · SQLModel · Alembic · PostgreSQL · Redis · Celery or ARQ · uv · Uvicorn · httpx · pytest · Testcontainers · Ruff · Docker · Kubernetes · OpenTelemetry · nginx · AWS · GitHub Actions · Git

Career Outcomes

FastAPI has become the default choice for new Python APIs outside full Django applications — a position strengthened by the fact that most AI and ML work happens in Python, which means the team that trains the model usually also ships the API in front of it. Python developers who can build and operate a production service, rather than only write scripts or notebooks, are disproportionately employable right now. This course targets:

- Backend Developer (Python) · API Engineer · Python Developer · Full-Stack Developer
- ML/AI Platform Engineer · MLOps Engineer · Data Engineer building service layers · Microservices Developer

Who This Course Is For

- Final-year students and fresh IT graduates targeting Python backend roles
- Python developers who write scripts, notebooks or automation and want to ship services
- **Data scientists and ML engineers who need to serve their own models** — a large and underserved audience
- Django and Flask developers moving to an async, typed, API-first framework
- Node, Java and .NET backend developers adding Python to their stack
- AI engineers building agent, RAG or LLM applications that need a real API layer

Module 1: Modern Python for APIs

6 Hours

YOU SHIP

- A modern, linted, lockfiled project skeleton — typed throughout, structured to scale past one file — that every later service builds on

TOPICS

- Type hints as the foundation the framework is built on; the ASGI model and how it differs from WSGI
- `async/await` and the event loop; concurrency versus parallelism
- Project setup with a modern package manager and lockfiles; linting and formatting; a structure that scales

Assessment: Quiz + Lab

Module 2: FastAPI Fundamentals

6 Hours

YOU SHIP

- A modular API with routers, response models and OpenAPI documentation good enough to hand to another team — plus deliberate startup logic via lifespan

TOPICS

- Routing, path and query parameters, request bodies, status codes and response models
- Routers for modular applications; application lifespan and startup logic
- The automatically generated OpenAPI documentation — and making it genuinely useful

Assessment: Quiz + Lab

Module 3: Data Validation with Pydantic

6 Hours

YOU SHIP

- A validation layer with custom validators and clean schema separation — request vs response vs database — whose failures read as messages an API consumer can act on

TOPICS

- Models, field constraints and custom validators; nested and recursive models
- Serialisation and response shaping; separating request schemas from response schemas from database models
- Settings and configuration management; actionable validation errors

Assessment: Quiz + Lab**Module 4: Dependency Injection**

7 Hours

YOU SHIP

- A refactor that replaces globals and singletons with injected dependencies — sessions, auth, shared logic — making the same code testable overnight

TOPICS

- The dependency system as the framework's best architectural feature; dependencies for database sessions, authentication and shared logic
- Sub-dependencies and caching; scoped resources and cleanup; class-based dependencies
- Injection for testability instead of globals or singletons

Assessment: Quiz + Lab**Module 5: Async Database Layer**

7 Hours

YOU SHIP

- A production data layer on async sessions — modern ORM style, eager loading that kills N+1s, migrations, transactions — behind a repository pattern, with a written single-model vs separate-schema decision

TOPICS

- Async sessions and connection pooling — why the synchronous session silently negates the framework's main advantage
- The modern ORM style; relationships and eager loading; migrations; transactions; converting ORM objects into validated response models
- Repository patterns; the single-model approach and when its convenience becomes a constraint

Assessment: Quiz + Lab**Module 6: Authentication, Authorisation & Security**

7 Hours

YOU SHIP

- An OAuth2-secured service — token issuance, refresh and validation, roles and scopes, dependency-injected auth, rate limiting — hardened against the OWASP API risks with CORS done correctly

TOPICS

- OAuth2 flows; token issuance, validation and refresh; password hashing; API keys and service-to-service auth
- Role and scope based access; dependency-injected auth; CORS done correctly; rate limiting at application and gateway layers
- Input hardening; secrets management; the OWASP API risks

Assessment: Quiz + Lab**Module 7: Serving AI & ML Workloads**

9 Hours

YOU SHIP

- An LLM gateway API — token-by-token streaming, timeouts and client cancellation, retries and circuit breaking, per-request cost tracking — plus a model-serving endpoint with warm-up, batching and concurrency limits proven under load test

TOPICS

- Streaming token-by-token responses; server-sent events; background tasks versus a real job queue for long-running inference; timeouts and client cancellation
- Concurrency limits and backpressure against rate-limited providers; retries and circuit breaking; caching and cost accounting; batching; model loading and warm-up
- File and multimodal uploads; exposing internal capabilities as tools for agents over emerging agent-integration protocols

Assessment: Quiz + Lab

Module 8: Testing, Observability & Performance**6 Hours****YOU SHIP**

- The async-correctness workshop: find the blocking calls in a slow service and fix them with before/after load-test evidence — on a service fully tested and traced end-to-end

TOPICS

- The test client and async testing; pytest fixtures and factories; database testing with containers; mocking external providers; contract testing
- Structured logging and correlation IDs; metrics and distributed tracing with OpenTelemetry
- Profiling; load testing; diagnosing a blocked event loop with evidence

Assessment: Quiz + Lab

Module 9: Deployment & Capstone**6 Hours****YOU SHIP**

- Your capstone: one production-grade API — small container image, health checks, graceful shutdown, horizontally scaled, CI/CD-deployed — monitored and defended on architecture

TOPICS

- ASGI servers, worker models and process management; container images that stay small; health checks and graceful shutdown; reverse proxies
- Environment configuration; horizontal scaling; serverless and its cold-start trade-offs; CI/CD
- Deployment targets compared honestly; capstone design, build, deploy and defense

Assessment: Capstone Project + Review