REST APIs mit Python und FastAPI

Moderne REST APIs mit Python entwickeln

01 INTRO

REST APIs bilden die Schnittstelle zwischen Anwendungen, Services und externen Systemen. FastAPI bietet dafür einen modernen Python-Ansatz mit Type Hints, automatischer Validierung und integrierter API-Dokumentation.

Der Kurs zeigt den vollständigen Weg von einer einfachen HTTP-Schnittstelle bis zu einer strukturierten und getesteten API. Neben FastAPI selbst stehen API-Design, Datenmodelle, Fehlerbehandlung, Persistenz, Sicherheit und der Betrieb einer API im Mittelpunkt.

DAUER

3 Tage

FORMAT

Präsenz · Live Online · Inhouse

VORKENNTNISSE

Sichere Python-Grundkenntnisse

02 TRAININGSPROGRAMM

Inhalte

HTTP und REST

- HTTP als Grundlage von Web APIs
- Requests und Responses
- HTTP-Methoden
- Status Codes
- Header
- Ressourcen und URLs
- REST-Prinzipien
- JSON als Austauschformat

Einstieg in FastAPI

- FastAPI-Projekt einrichten
- ASGI und Uvicorn
- erste Endpoints
- Path Operations
- Path- und Query-Parameter
- Request und Response
- automatische OpenAPI-Dokumentation
- Swagger UI und ReDoc

Datenmodelle und Validierung

- Pydantic-Modelle
- Type Hints als API-Vertrag
- Request Bodies
- Pflichtfelder und optionale Felder
- Validierungsregeln
- verschachtelte Modelle
- Serialisierung und Deserialisierung
- unterschiedliche Request- und Response-Modelle

REST-Schnittstellen entwerfen

- Ressourcen modellieren
- CRUD-Operationen
- sinnvolle URLs
- HTTP-Methoden korrekt einsetzen
- Status Codes auswählen
- Filterung, Sortierung und Pagination
- Idempotenz
- konsistente API-Konventionen

Fehlerbehandlung

- HTTP Exceptions
- Fehlerantworten gestalten
- Validierungsfehler
- eigene Exception Handler
- einheitliche Fehlerstrukturen
- interne Fehler nicht nach außen tragen
- Logging von Fehlern

Struktur größerer FastAPI-Anwendungen

- Router
- Module und Packages
- APIRouter
- Trennung von API und Fachlogik
- Service-Schicht
- Konfiguration
- Dependency Injection mit Depends
- wiederverwendbare Abhängigkeiten

Datenbankanbindung

- Persistenz in Web APIs
- SQL-Datenbanken anbinden
- SQLAlchemy im Überblick
- Sessions und Transaktionen
- CRUD mit persistenten Daten
- Datenbankmodelle und API-Modelle trennen
- Fehlerfälle bei Datenbankzugriffen

Asynchrone Verarbeitung

- synchron und asynchron in Python
- async und await
- asynchrone Endpoints
- blockierende Operationen erkennen
- I/O-bound und CPU-bound Aufgaben
- wann async sinnvoll ist
- typische Fehler bei asynchronem Code

Authentifizierung und Autorisierung

- Authentifizierung und Autorisierung unterscheiden
- API Keys
- Bearer Tokens
- OAuth2-Grundlagen
- JWT im Überblick
- geschützte Endpoints
- Benutzer und Rollen
- Passwörter sicher behandeln

APIs testen

- Endpoints automatisiert testen
- pytest und FastAPI
- Test Client
- Request- und Response-Daten prüfen
- Status Codes testen
- Fehlerfälle testen
- Dependencies für Tests ersetzen
- Datenbankzugriffe in Tests kontrollieren

API-Dokumentation

- OpenAPI
- automatisch erzeugte Dokumentation
- Metadaten für Endpoints
- Response Models
- Beispiele
- Tags
- Beschreibungen
- API-Verträge nachvollziehbar halten

Betrieb und Deployment

- Entwicklungs- und Produktionsbetrieb unterscheiden
- ASGI-Server
- Konfiguration über Umgebungsvariablen
- Logging
- CORS
- Containerisierung mit Docker im Überblick
- Health Checks
- grundlegende Betriebsaspekte

03 DURCHFÜHRUNG

Rahmen und Organisation

Dauer	3 Tage
Durchführung	Präsenz oder Live Online, vorzugsweise als Inhouse-Schulung
Schwerpunkt	Inhouse-Schulungen für Unternehmen und Teams
Sprache	Deutsch, englische Fachbegriffe und Dokumentation
Unterlagen	Kursunterlagen, Beispiele und Übungsaufgaben
Technik	Eigener Rechner mit aktueller Python-Version und Entwicklungsumgebung

> **Inhouse-Schulungen**
> Die Schwerpunkte können an vorhandene Kenntnisse, eingesetzte Technologien und konkrete Anforderungen des Teams angepasst werden.

04 ZIELGRUPPE

Für wen ist der Kurs gedacht?

Der Kurs richtet sich an Python-Entwicklerinnen und -Entwickler, die REST APIs und Backend-Services mit FastAPI entwickeln möchten.

Er eignet sich sowohl für Entwickler, die erstmals eine Web API implementieren, als auch für Teilnehmer mit Erfahrungen aus anderen Web- oder API-Frameworks, die FastAPI und dessen Python-spezifischen Ansatz kennenlernen möchten.

Die Teilnehmer sollten mit folgenden Themen sicher umgehen können:

- grundlegende Python-Syntax
- Funktionen
- Module und Packages
- Klassen und Objekte
- Exceptions
- Type Hints in Grundzügen

> **HTTP und REST**
> Grundkenntnisse von HTTP und REST sind hilfreich, aber nicht erforderlich.

05 LERNZIELE

Was Sie nach dem Kurs können

Nach dem Kurs können die Teilnehmer:

- REST-Schnittstellen fachlich und technisch strukturieren
- APIs mit FastAPI implementieren
- Requests und Responses mit Pydantic modellieren und validieren
- geeignete HTTP-Methoden und Status Codes einsetzen
- größere FastAPI-Anwendungen mit Routern und Dependencies strukturieren
- Datenbanken an eine API anbinden
- synchrone und asynchrone Verarbeitung sinnvoll unterscheiden
- grundlegende Authentifizierungs- und Autorisierungsmechanismen umsetzen
- APIs mit pytest automatisiert testen
- OpenAPI-Dokumentation gezielt gestalten
- typische Fehler in API-Design und Implementierung erkennen
- eine FastAPI-Anwendung für den Betrieb vorbereiten

06 FAQ

Häufige Fragen

Muss ich FastAPI bereits kennen?

Nein. Der Kurs beginnt mit den Grundlagen und entwickelt daraus schrittweise eine strukturierte API.

Muss ich REST und HTTP bereits kennen?

Nein. Die für die Entwicklung benötigten HTTP- und REST-Grundlagen werden zu Beginn behandelt.

Wird eine Datenbank verwendet?

Ja. Die Anbindung einer relationalen Datenbank wird praktisch behandelt. Der Schwerpunkt liegt dabei auf der Integration in die API und nicht auf einer vollständigen Einführung in SQL oder Datenbankentwicklung.

Wird auch async/await behandelt?

Ja. Gerade bei FastAPI ist wichtig zu verstehen, wann asynchrone Verarbeitung einen Vorteil bietet und wann sie unnötig oder sogar problematisch ist.

Wird die API automatisiert getestet?

Ja. Automatisierte API-Tests gehören zum Kurs. pytest wird dabei eingesetzt, ohne die Inhalte von PY-004 „Python Testing mit pytest“ vollständig zu wiederholen.

Wird Docker behandelt?

Die Containerisierung und der Betrieb einer FastAPI-Anwendung werden im Überblick behandelt. Docker selbst ist kein Schwerpunkt des Kurses.

Kann der Kurs an eine vorhandene API oder Systemarchitektur angepasst werden?

Ja. Bei Inhouse-Schulungen können Architektur, Authentifizierung, Datenbanktechnologie und Beispiele an die vorhandene technische Umgebung angepasst werden.

UC IT Service · Ulrich Cuber · kontakt@uc-it.de
https://www.uc-it.de/coaching/python-fastapi/