TRAINING PROGRAM

Python Engineering Course

Beginner to Backend Developer — A comprehensive 8-week journey to master Python and build production-ready backend systems

DURATION	LEVEL	FORMAT
8 Weeks	**Beginner**	**Hands-on**

Table of Contents

Course Objective

Program Structure

Phase 1: Python Fundamentals

- Topics & Tools
- Project Ideas

Phase 2: Intermediate Python

Phase 3: Python for Backend

Phase 4: Database Integration

Phase 5: Advanced Backend

Phase 6: Deployment & DevOps

Final Project

Recommended Stack

Weekly Execution Plan

Evaluation Metrics

Learning Outcomes

Course Objective

Transform Learners into Python Developers

This comprehensive course is designed to take beginners and transform them into capable Python backend developers who can:

- Write clean, maintainable, and efficient Python code
- Build robust REST APIs using FastAPI framework
- Design and work with relational databases effectively
- Create real-world backend systems ready for production

Program Structure

Table 1 Program Overview

Attribute	Details
Duration	8 Weeks (2 Months)
Weekly Schedule	2 Days Learning + 2 Days Practice + 1 Day Demo
Total Phases	6 Progressive Phases
Final Deliverable	Capstone Project

Weekly Rhythm

The program follows a consistent weekly pattern: Monday-Tuesday for learning new concepts, Wednesday-Thursday for hands-on practice and building, and Friday for demos and discussions.

Phase 1: Python Fundamentals

Duration: Week 1–2

TOPICS COVERED

- Python basics: syntax, variables, and data types
- Control flow: if/else statements and loops
- Functions and modules
- Data structures: lists, tuples, sets, and dictionaries
- File handling operations

TOOLS & ENVIRONMENT

- Python 3.x (latest stable version)
- VS Code or PyCharm IDE
- Terminal/Command line basics

Hands-on Projects

- **CLI Calculator:** Build a command-line calculator with arithmetic operations
- **To-Do List:** Create a terminal-based task manager
- **File Reader:** Develop a log file reader with read/write capabilities

Learning Outcome

Students will establish a strong Python foundation and gain the ability to write basic scripts and automate simple tasks.

Phase 2: Intermediate Python

Duration: Week 3

TOPICS COVERED

- Object-Oriented Programming: classes, objects, inheritance
- Exception handling and error management
- Modules and packages organization
- Virtual environments (venv) management
- Working with JSON and external APIs

Hands-on Projects

- **Contact Manager:** OOP-based contact book with CRUD operations
- **API Data Fetcher:** Fetch and display weather or news data
- **JSON Config System:** Build a configuration management system

Learning Outcome

Students will write structured, reusable code and understand real-world data handling patterns.

Phase 3: Python for Backend

Duration: Week 4–5

TOPICS COVERED

- Introduction to backend development concepts
- FastAPI framework fundamentals
- REST API design (GET, POST, PUT, DELETE)
- Request and response handling
- Middleware basics and CORS

TOOLS & ENVIRONMENT

- FastAPI framework
- Uvicorn ASGI server
- Postman for API testing

Hands-on Projects

- **Simple User API:** CRUD operations for user management
- **Notes API:** Create, read, update, delete notes endpoints
- **Authentication System:** Basic login/logout functionality

Learning Outcome

Students will be able to build functional backend APIs and understand web architecture fundamentals.

Phase 4: Database Integration

Duration: Week 6

TOPICS COVERED

- SQL basics and database design
- ORM concepts with SQLAlchemy
- CRUD operations with databases
- Connecting FastAPI with PostgreSQL/SQLite
- Migrations and schema management

TOOLS & ENVIRONMENT

- SQLite for development
- PostgreSQL for production
- SQLAlchemy ORM
- Alembic for migrations

Hands-on Projects

- **User Management System:** Full database-backed user CRUD
- **Blog API:** Posts and comments with database persistence

Learning Outcome

Students will confidently work with databases and build persistent applications with data integrity.

Phase 5: Advanced Backend Concepts

Duration: Week 7

TOPICS COVERED

- JWT authentication and authorization
- Pagination and filtering strategies
- Background tasks with Celery
- Logging and error handling best practices
- API rate limiting and security

Hands-on Projects

- **Secure API:** Complete login/signup with JWT tokens
- **Role-based Access:** Admin and user permission system

Learning Outcome

Students will build production-level backend features with security and scalability in mind.

Phase 6: Deployment & DevOps Basics

Duration: Week 8

TOPICS COVERED

- API deployment strategies
- Docker containerization basics

- Environment variables and secrets management
- Cloud hosting options (Railway, AWS, Render)
- CI/CD pipeline introduction

TOOLS & ENVIRONMENT

- Docker and Docker Compose
- GitHub for version control
- Cloud platforms (Railway/AWS/Render)

Hands-on Projects

- **Deploy Notes API:** Full deployment to cloud platform
- **Dockerize Python App:** Create production-ready containers

Learning Outcome

Students will deploy real applications and understand production environment setup.

Final Project

At the end of the course, students will choose and complete one capstone project that demonstrates their mastery of all learned concepts.

Choose Your Capstone Project

1. Task Management API

- User registration and authentication
- Task CRUD operations
- Task assignments and status tracking

2. Blog Backend System

- Post creation and management
- Comments system
- User profiles and permissions

3. E-commerce Backend

- Product catalog management
- Order processing system
- User accounts and cart

4. API for Mobile App

- Authentication endpoints
- Data synchronization
- Push notification support

Recommended Technology Stack

Backend

- Python 3.11+
- FastAPI framework
- Uvicorn ASGI server
- Pydantic for validation

Database

- PostgreSQL (production)
- SQLite (development)
- SQLAlchemy ORM
- Alembic migrations

Authentication

- JWT tokens
- Passlib for hashing
- OAuth2 with Password flow

DevOps & Deployment

- Docker
- GitHub Actions
- Railway / AWS / Render

Weekly Execution Plan

Table 2 Standard Weekly Schedule

Day	Activity	Focus
Monday	Learning	New concepts and theory
Tuesday	Learning	Deep dive with examples
Wednesday	Practice	Guided exercises
Thursday	Build	Independent project work
Friday	Demo	Presentation and discussion

Evaluation Metrics

Student progress will be evaluated based on the following criteria:

Table 3 Evaluation Criteria

Metric	Description	Weight
Code Quality	Clean, readable, well-documented code	25%
API Functionality	Working endpoints with proper responses	25%
Database Integration	Proper ORM usage and data persistence	20%
Deployment Success	Live, accessible application	20%
Documentation	README, API docs, comments	10%

Learning Outcomes

Upon successful completion of this 8-week course, you will be able to:

- Write production-level Python code following best practices and PEP 8 standards
- Design and build RESTful APIs with FastAPI framework
- Implement authentication and authorization systems using JWT
- Work with relational databases using SQLAlchemy ORM
- Containerize applications with Docker for consistent deployments
- Deploy backend applications to cloud platforms
- Understand and implement backend security best practices

Career Path

This course provides the foundation for roles such as Python Developer, Backend Engineer, API Developer, and Full-Stack Developer. Combined with frontend and AI skills, graduates can pursue Full AI Product Engineering positions.