

2-Month Python Training Program, Feb, 25

Training Schedule

Month 1: Git, Python, SQL, Multi-Threaded Media Review System

Topics	Sub-Topics	Days Required	Date	Session Recordings/Resources
Git Basics	Cloning, Branching, Merging, Rebase	1 Day	17 Feb 2025	#00 GIT Concepts.mp4
Python Fundamentals	Syntax, Data Types, Control Flow, Functions, Modules	3 Days	20 Feb 2025	#01 Python_Introduction_DataType.mp4, #02 Python DataType_Session.mp4, #04 Python_Functions.mp4, #05 Iterator_Generator_Container.mp4, #06 Decorators.mp4, #09 Exceptions_ContextManagers.mp4
Python OOP	Classes, Inheritance, Polymorphism, Encapsulation	2 Days	24 Feb 2025	#07 Python_OOPS_Session1.mp4 #08 Python_OOPS_Session2.mp4
Debugging & Logging	Debugging Techniques, Logging Frameworks	1 Day	25 Feb 2025	1. https://sematext.com/blog/python-logging/ 2. https://realpython.com/python-logging/

Multithreading & Multiprocessing	Threading Module, Multiprocessing, AsyncIO	2 Days	27 Feb 2025	#10 MultiThreading_MultiProcessing_Session1.mp4, #11 MultiThreading_MultiProcessing_Session2.mp4
Unit Testing	Pytest, unittest, Mocking	1 Day	28 Feb 2025	#12 Pytest_Session1.mp4, #13 Pytest_Session2.mp4
Packaging & Code Quality	Virtual Environments, PEP8, Linting	1 Day	03 Mar 2025	<ol style="list-style-type: none"> 1. https://realpython.com/python-code-quality/ 2. https://www.digitalocean.com/blog/top-python-best-practices-for-better-code-quality/ 3. https://packaging.python.org/en/latest/tutorials/packaging-projects/ 4. https://www.pyopensci.org/python-package-guide/tutorials/intro.html #03 Python_VirtualEnv.mp4
SQL	CRUD, Joins, Aggregations, Indexing, Transactions, Views, Stored Procedures	4 Days	07 Mar 2025	#14 SQL.mp4
Mini Project: Multi-	SQLite, Redis Caching,	5 Days	14 Mar	See The details below

Threaded Media Review System	Multithrea ding, Design Patterns		20 25	
---------------------------------------	-------------------------------------------	--	----------	--

Month 1 Mini Project: Multi-Threaded Media Review System

Objective

Develop a **CLI-based Media Review System** that allows users to **review movies, web shows, and songs**, store ratings, and retrieve recommendations. This project will test **Git, Python, SQL, Caching, Multithreading, and Design Patterns**.

Technologies Used

- **Python** (Core development)
- **SQLite** (Relational Database for data storage)
- **Redis** (Caching frequently accessed reviews)
- **Multithreading** (Handling multiple review submissions concurrently)
- **Factory Pattern** (Managing different media types)
- **Observer Pattern** (Real-time updates on new reviews)
- **Git** (Version control and collaboration)

Features & Functionalities

- **User Management:** Add and track users reviewing media.
- **Media Storage & Reviews:** Users can review and rate Movies, Web Shows, and Songs.
- **Multithreading for Concurrent Submissions:** Ensures multiple users can submit reviews simultaneously.
- **Redis Caching for Fast Data Retrieval:** Frequently accessed reviews are stored in memory.
- **Factory Pattern for Media Types:** A structured approach to handling different media categories.
- **Observer Pattern for Notifications:** Users get alerts when a new review is added for their favorite media.
- **Git Version Control:** Maintain structured workflow using Git best practices.

CLI-Based Commands

Operation	Command
View All Media	python media_review.py --list
Add a Review	python media_review.py --review <media_id> <rating> <comment>
Search by Title	python media_review.py --search <title>
Get Top-Rated	python media_review.py --top-rated
Get Recommendations	python media_review.py --recommend <user_id>

Month 2: Docker, DSA, Software Engineering, Web Frameworks, Pub/Sub, Advanced Real-Time Discussion Forum

Topics	Days Required	Date	Session Recordings/Resources	Status
Docker	2 days	18 Mar 2025	#17 Docker_Session1.mp4 #18 Docker_Session2.mp4	
Data Structures & Algorithms	5 Days	25 Mar 2025	Refer learn portal	
Software Engineering Best Practices	4 Days	31 Mar 2025	https://testdriven.io/blog/clean-code-python/ , https://github.com/heykarimoff/solid.python , https://github.com/faif/python-patterns	

Python Web Frameworks	5 Days	07 Apr 2025	#19 Flask_Session1.mp4 #20 Flask_Session2.mp4 #FastAPI_Session.mp4	
Mini Project: Book Rental Microservice	4 Days	11 Apr 2025	See Details Below	

Project Details :

Project Title: *Book Rental Microservice*

Description:

Build a microservice for managing a simple book rental system. The system should include two main services:

1. **Book Service** - Handles book information (CRUD operations).
2. **User Service** - Manages user information (CRUD and rental history).

Both services should communicate through **REST APIs**.

Features:

Book Service:

- Add, update, delete, and list books.
- Each book should have fields like:
 - id (UUID)
 - title
 - author
 - genre
 - available_copies

User Service:

- Add, update, delete, and list users.
- Rent a book:

- Update the available_copies in the Book Service.
 - Save the rental record to the database.
- Fields for users:
 - id (UUID)
 - name
 - email
 - rented_books (store book IDs or details).

Tech Stack:

- **Framework:** FastAPI
- **Database:** SQLite or PostgreSQL (depending on comfort level).
- **Communication:** REST APIs.
- **ORM:** SQLAlchemy or Tortoise ORM.

Requirements:

1. Database Design:

- Create a separate database for each service (simulating microservices).
- Define tables for books, users, and rentals.

2. Endpoints:

- Implement endpoints for all CRUD operations.
- Add a rent_book endpoint in the **User Service** that calls the **Book Service** to decrement available_copies.

3. Validation:

- Validate input data using Pydantic models.
- Ensure users cannot rent a book if available_copies is 0.

4. Asynchronous Programming:

- Use async def for all database and API calls.

5. Optional Features (if time permits):

- Add JWT-based authentication for sensitive operations (like renting books).
- Include Docker support for both services.

4 MONTHS TRAINING PROGRAMMING + 2 MONTHS

Month 3: AWS, Relational & NoSQL Databases, CI/CD, AI-Powered Automated Resume Screener

Topics	Days Required	Date	Session Recordings/Resources	Status
AWS	4 Days	03 Apr 2025	https://learn.epam.com/detailsPage?id=63188ae7-095a-45a7-9e7f-21bcbcbba61d4	
Relational Databases	3 Days	08 Apr 2025	<p>Database design : https://support.microsoft.com/en-us/office/database-design-basics-eb2159cf-1e30-401a-8084-bd4f9c9ca1f5</p> <p>SQL Databases Comparison : https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems</p> <p>ACID Properties : https://www.geeksforgeeks.org/acid-properties-in-dbms/</p> <p>Postgres Isolation Level : https://www.postgresql.org/docs/current/transaction-iso.html</p> <p>Indexing : https://use-the-index-luke.com/</p> <p>Query Optimizations : https://www.geeksforgeeks.org/best-practices-for-sql-query-optimizations/</p> <p>Using psycopg2 to Connect to PostgreSQL : https://www.datacamp.com/tutorial/tutorial-postgresql-python</p>	

			SQLAlchemy : https://auth0.com/blog/sqlalchemy-orm-tutorial-for-python-developers/	
Non-Relational Databases	2 Days	10 Apr 2025	Intro to NoSQL databases - https://www.altexsoft.com/blog/nosql-databases/ MongoDB Basics - https://www.sitepoint.com/an-introduction-to-mongodb/ PyMongo - https://www.mongodb.com/resources/languages/pymongo-tutorial Redis - https://blog.logrocket.com/guide-to-fully-understanding-redis/ DynamoDB - https://medium.com/@_amanarora/dynamodb-2efdbcaebabc Cassandra - https://www.datastax.com/blog/introduction-to-apache-cassandra-the-lamborghini-of-the-nosql-world	
CI/CD	3 Days	15 Apr 2025	CI/CD Fundamentals : https://about.gitlab.com/blog/2025/01/06/ultimate-guide-to-ci-cd-fundamentals-to-advanced-implementation/ GitHub Actions for CI/CD : https://realpython.com/github-actions-python/ CI/CD Deployment Strategies : https://blog.mergify.com/real-world-ci-cd-pipeline-examples-devops-success/	

			Security in CI/CD Pipelines : https://cycode.com/blog/ci-cd-pipeline-security-best-practices/ Infrastructure as Code (IaC) with Terraform : https://developer.hashicorp.com/terraform/tutorials/aws-get-started/infrastructure-as-code CI/CD Monitoring & Observability : https://www.infoq.com/articles/ci-cd-observability/	
Mini Project: AI-Powered Automated Resume Screener	8 Days	25 Apr 2025	Details are added below	

AI-Powered Automated Resume Screener

Objective

Build a system that allows users to **upload resumes**, automatically **extract key details**, and **rank candidates** based on **job relevance** using AI and search technologies.

Technologies Used

- **FastAPI/Flask** – REST API for resume submission and retrieval
- **PostgreSQL** – Structured storage for jobs and candidate metadata
- **MongoDB** – NoSQL storage for raw resume content
- **OpenSearch** – Full-text search and ranking of resumes
- **Redis** – Caching frequent searches
- **GitHub Actions + Docker** – CI/CD pipeline for deployment
- **Python Libraries** – For parsing, scoring (e.g., spaCy, TF-IDF)

Features

- Resume upload and parsing (PDF/DOCX)
- AI-powered keyword extraction and matching
- Search resumes by skills, experience, and location
- Rank candidates by relevance to a job description
- Cache frequently queried results with Redis

API Operations

- POST /upload_resume – Upload and parse a resume
- GET /candidates?skills=Python – Filter candidates by skill
- GET /rank_candidates?job_id=123 – Get ranked candidates
- GET /resume/{id} – View parsed resume details

Month 4: Generative AI, Machine Learning, Generative AI-Powered Content Generator

Topics	Days Required	Date	Session Recordings/Resources	Status
Machine Learning	5 Days	16 May 2025	https://epam-my.sharepoint.com/:w:/p/sumit_kumar3/EaKKSV9yV_FDuAmDrF2pmLABnLPZKljW2QBfLuF8di0Fcw?e=4TlxpO	
Generative AI	7 Days	30 May 2025	https://videoportal.epam.com/playlist/oYVzXAY0	

Mini Proje ct	8 Day s	12 Ju n 20 25	Details are added below	
---------------------	---------------	---------------------------	-------------------------	--

Hate Speech Detection Assistant

Overview

This project is designed to help interns build a hate speech detection tool using GenAI techniques. The application will classify user input, retrieve related policies, explain the decision, and suggest a moderation action — all within a simple web interface.

Objective

Develop a web-based assistant that:

- Accepts user-submitted text
- Classifies it as Hate, Toxic, Offensive, Neutral, or Ambiguous
- Retrieves relevant policy/guideline documents using Hybrid RAG
- Explains the classification using LLMs
- Recommends an action (e.g., remove, warn, flag)

Core Components

1. HateSpeechDetectionAgent

- Uses OpenAI to classify the input
- Returns a label and a short explanation

1. HybridRetrieverAgent

- Retrieves the top relevant .txt policy files using sentence-transformer + FAISS

1. PolicyReasoningAgent

- Combines classification and retrieved docs to justify the decision via OpenAI

1. ActionRecommenderAgent

- Maps the classification to an action (e.g., ban, warn, flag, allow)

1. ErrorHandlerAgent

- Handles errors gracefully and informs the user

UI Requirements

- Built in Streamlit
- Input: textarea for content
- Output:
 - Classification label
 - Reason
 - Retrieved policy snippets
 - Recommended action