MockAttempt

TECH ACADEMY • OFFICIAL COURSE SYLLABUS BROCHURE

MockAttempt Academy Pvt. Ltd. • Live Architect Mentorship, AI Assessments & Client Production

Admissions & Support:
support@mockattempt.com
Phone / WhatsApp: +91 98765 43210
Course Code: PYTH • **Updated:** 01 Nov 2026
https://mockattempt.com/courses/python-training

PYTH • PROGRAMMING & BACKEND **BEGINNER TO JOB-READY PYTHON DEVELOPER**

100% LIVE INDUSTRY EXPERT COHORT + INTERNSHIP LETTER

Python Training

Master Python from scratch to OOP, DSA, Decorators, Regex, Web Scraping, SQLite/MySQL & Tkinter GUI — with runnable programs, 10×50 MCQ Mock Tests per topic, and Live Voice + Gemini AI Technical Interviews at the most affordable fee.

Duration: 16–18 Weeks • Live + Hands-On Projects (205 Hrs) • **Modules:** 16 • **Live Topics:** 52 • **Labs & Tasks:** 104 • **Mocks / AI Vivas:** 376 / 376 • **Projects:** 16 Module + 1 Live Client Prod

Fixed Non-Negotiable Tuition Fee: Base Fee **₹3,499** + 18% GST (₹630) = **Total Incl. GST: ₹4,129**

Live Batch Tracks: Weekday Live Online Batch (Mon – Fri) (Monday to Friday (5 Live Interactive Classes / Week) • 1.5 Hours / Day (60 Mins Live Architecture & Coding + 30 Mins Hands-On Lab) • 17–19 Weeks Fast-Track) | Weekend Intensive Live Online Batch (Sat – Sun) (Saturday & Sunday (Designed for Working Professionals & College Students) • 3.0 Hours / Day (2 Hrs Deep-Dive Live Coding + 1 Hr Guided Project Build) • 19–21 Weeks Weekend Track)

Core Technologies & Engineering Stack: Python 3.12 • Data Structures & OOP • File/JSON/CSV ETL • Algorithms • Regex & Scraping • MySQL & SQLite3 • Tkinter GUI

1. PROGRAM MASTERY OUTCOMES & ELIGIBILITY

End-to-End Engineering Outcomes:

1. Write clean, PEP-8 compliant Python 3 programs from fundamentals to advanced OOP, Decorators & Generators
2. Build 10 portfolio-ready Python Capstone Projects across CLI automation, ETL pipelines, Web Scraping, Databases, and Tkinter Desktop GUI
3. Practice 340 Topic-Wise Timed Mock Exams containing 17,000 curated Python MCQs with explanations
4. Clear 340 Spoken Voice + Google Gemini AI Technical Interviews with real-time Speech-to-Text verification
5. Python 3.12+ static type hints, dataclasses, Enums, and modular package layouts (`src/` layout with `pyproject.toml`).
6. Memory-safe iterators, generators, and context managers (`with`) for large file, stream, and database workloads.
7. Structured exception hierarchies, environment-driven configs, and structured JSON logging instead of `print()` debugging.
8. Concurrency mastery: knowing exactly when to use `asyncio` (I/O-bound) vs `multiprocessing` (CPU-bound).

Eligibility & Prerequisites: Zero prior programming experience required — starts from installing Python & writing your first program • Any basic laptop/PC (4 GB RAM is sufficient for all 10 Python modules and projects) • No prior Python industry experience required — Module 1 starts from environment setup, core mental models, and plain-English fundamentals. • Basic computer & internet familiarity on a Windows, macOS, or Linux laptop (8 GB+ RAM recommended for smooth live coding labs). • Ideal for B.Tech / BCA / MCA / B.Sc IT students, non-CS beginners, career switchers, and working developers upgrading to 2026 production standards. • Commitment to attend live Weekday or Weekend classes, complete daily hands-on lab checklists, and practice topic-wise Mock Tests & AI Interviews.

2. DETAILED MODULE & TOPIC-WISE SYLLABUS (16 MODULES • 52 LIVE TOPICS)

MODULE 1 OF 16 • 10 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 1: Python Fundamentals, Setup, Syntax & Operators

Get started with CPython architecture, interactive vs script execution, indentation rules, keywords, and all 7 Python operator families.

Objectives: ✓ Understand Python interpreter workflow • ✓ Write PEP-8 indented scripts • ✓ Master arithmetic, bitwise, logical, identity & membership operators

1.1 What is Python — Architecture, Features & Applications [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply What is Python — Architecture, Features & Applications in real Python systems — specifically: History of Python, CPython bytecode execution, dynam...

1.2 Python Syntax, Indentation, Identifiers & Comments [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Syntax, Indentation, Identifiers & Comments in real Python systems — specifically: Interactive REPL vs .py script mode, 4-space indentation...

1.3 Python Operators & Precedence (Arithmetic, Bitwise, Logical, Identity) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Operators & Precedence (Arithmetic, Bitwise, Logical, Identity) in real Python systems — specifically: Deep dive into //, **, %, comparison...

Module 1 Hands-On Capstone Project (100 Marks): Smart CLI Personal Expense, GST & Loan EMI Calculator — Build an interactive command-line financial toolkit in Python 3 that computes GST tax splits (CGST/SGST), monthly loan EMIs using exponentiation operators, and budget alerts.

MODULE 2 OF 16 • 12 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 2: Variables, Data Types, Type Casting & String Manipulation

Master variable unpacking, LEGB scope, built-in numeric/sequence/mapping types, explicit type conversion, and string slicing & methods.

Objectives: ✓ Unpack multiple variables & manage local/global scope • ✓ Convert safely across int, float, str, bool • ✓ Slice, format, and clean text using Python str methods

2.1 Python Variables, Multiple Unpacking & Scope Rules [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Variables, Multiple Unpacking & Scope Rules in real Python systems — specifically: Object references, id(), multiple assignment, tuple unpa...

2.2 Python Built-In Data Types (Numeric, Sequence, Set, Mapping, Boolean) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Built-In Data Types (Numeric, Sequence, Set, Mapping, Boolean) in real Python systems — specifically: Complete tour of int, float, complex,...

2.3 Implicit & Explicit Type Casting in Python [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Implicit & Explicit Type Casting in Python in real Python systems — specifically: Automatic numeric widening vs explicit conversion constructors i...

2.4 Python Strings — Indexing, Slicing, Formatting & Built-In Methods [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Strings — Indexing, Slicing, Formatting & Built-In Methods in real Python systems — specifically: Unicode immutability, negative indexing,...

Module 2 Hands-On Capstone Project (100 Marks): Automated Resume & Invoice TextSanitizer CLI — Create a Python text-processing utility that extracts normalized emails, phone numbers, skills, and invoice totals from messy raw text strings using slicing, split/join, and safe type castin...

MODULE 3 OF 16 • 11 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 3: Control Statements, Branching & Loop Constructs

Build decision trees and iterative algorithms with if-elif-else, match-case, for-range/enumerate, while loops, and break/continue/pass.

Objectives: ✓ Write clean conditional logic & ternary expressions • ✓ Iterate pythonically with enumerate() and zip() • ✓ Control loop termination with break, continue, and loop-else

3.1 Python Conditional Statements (if, elif, else & Ternary Expressions) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Conditional Statements (if, elif, else & Ternary Expressions) in real Python systems — specifically: Decision branching, chained comparison...

3.2 Python for Loops, range(), enumerate() & zip() Iteration [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python for Loops, range(), enumerate() & zip() Iteration in real Python systems — specifically: Iterating over sequences, range(start, stop, step)...

3.3 Python while Loops, Sentinel Patterns & Convergence Algorithms [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python while Loops, Sentinel Patterns & Convergence Algorithms in real Python systems — specifically: Condition-driven iteration, sentinel input l...

3.4 Loop Control Statements — break, continue, pass & Loop-else [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Loop Control Statements — break, continue, pass & Loop-else in real Python systems — specifically: Early loop exit with break, skipping invalid it...

Module 3 Hands-On Capstone Project (100 Marks): Interactive Banking ATM & Student Grade Analytics Engine — Develop a menu-driven ATM PIN & transaction simulator alongside a batch student grade classifier using while loops, retry limits, break/continue guards, and summary statistics.

MODULE 4 OF 16 • 14 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 4: Core Python Data Structures (Lists, Tuples, Sets & Dictionaries)

Master dynamic arrays (list), immutable records (tuple), hash-based unique sets (set), and O(1) key-value hash tables (dict).

Objectives: ✓ Manipulate lists with O(1) append/pop and Timsort • ✓ Leverage tuples as hashable dict keys • ✓ Perform set algebra & O(1) dictionary lookups

4.1 Python Lists — Dynamic Arrays, Slicing, Sorting & Shallow vs Deep Copy [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Lists — Dynamic Arrays, Slicing, Sorting & Shallow vs Deep Copy in real Python systems — specifically: Dynamic resizing arrays, append/exte...

4.2 Python Tuples — Immutability, Packing/Unpacking & Hashability [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Tuples — Immutability, Packing/Unpacking & Hashability in real Python systems — specifically: Fixed-size immutable records, single-element...

4.3 Python Sets & Frozensets — Hash Uniqueness & Set Algebra [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Sets & Frozensets — Hash Uniqueness & Set Algebra in real Python systems — specifically: O(1) membership lookup, automatic deduplication, u...

4.4 Python Dictionaries — Hash Maps, Safe .get(), Views & Merging [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Dictionaries — Hash Maps, Safe .get(), Views & Merging in real Python systems — specifically: Insertion-ordered key-value hash maps (Python...

Module 4 Hands-On Capstone Project (100 Marks): E-Commerce Inventory, Cart & Deduplicated Coupon Engine — Build an in-memory store backend using dicts for SKU catalog lookup, lists for ordered cart line-items, tuples for immutable price snapshots, and sets for unique applied coupon codes.

MODULE 5 OF 16 • 12 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 5: Modular Python — Functions, Lambdas, Comprehensions & Packages

Write reusable, expressive Python with *args/**kwargs, closures, lambda functional pipelines, comprehensions, and custom packages.

Objectives: ✓ Master positional, keyword-only, *args, and **kwargs parameters • ✓ Transform datasets concisely via comprehensions & map/filter/sorted • ✓ Organize multi-file packages using __name__ == "__main__"

5.1 Python Functions — Parameters, *args, **kwargs, Type Hints & Docstrings [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Functions — Parameters, *args, **kwargs, Type Hints & Docstrings in real Python systems — specifically: Defining functions with def, positi...

5.2 Python Lambda Anonymous Functions with map(), filter() & sorted() [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Lambda Anonymous Functions with map(), filter() & sorted() in real Python systems — specifically: Single-expression anonymous functions, cu...

5.3 List, Dictionary & Set Comprehensions in Python [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply List, Dictionary & Set Comprehensions in Python in real Python systems — specifically: Declarative sequence construction, filtering clauses, neste...

5.4 Python Modules, Packages, Standard Library & __name__ Guard [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Modules, Packages, Standard Library & __name__ Guard in real Python systems — specifically: Importing modules, from ... import, sys.path re...

Module 5 Hands-On Capstone Project (100 Marks): Modular Employee Payroll, Tax Slab & Analytics Package — Architect a reusable multi-module Python package that computes net salaries, HRA/PF deductions, progressive tax slabs, and department-wise leaderboard reports using comprehensions and higher...

MODULE 6 OF 16 • 14 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 6: Object-Oriented Programming (OOPs) in Python

Design extensible domain models using classes, __init__, instance vs class attributes, super() MRO inheritance, properties, and ABCs.

Objectives: ✓ Model real-world entities with classes, __init__, and __repr__ • ✓ Apply single/multiple inheritance & C3 Linearization MRO • ✓ Enforce invariants via @property encapsulation and abc.ABC

6.1 Python Classes, Objects, self, Instance vs Class Attributes [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Classes, Objects, self, Instance vs Class Attributes in real Python systems — specifically: Blueprints vs instances, explicit self binding,...

6.2 Python Constructors (__init__ vs __new__), Dunder Methods & Lifecycle [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Constructors (__init__ vs __new__), Dunder Methods & Lifecycle in real Python systems — specifically: Object allocation (__new__) vs initia...

6.3 Inheritance in Python — super(), Method Overriding & MRO [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Inheritance in Python — super(), Method Overriding & MRO in real Python systems — specifically: Single, multilevel, hierarchical, and multiple inh...

6.4 Encapsulation, @property Accessors & Abstract Base Classes (abc) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Encapsulation, @property Accessors & Abstract Base Classes (abc) in real Python systems — specifically: Protected (_attr) vs name-mangled private...

Module 6 Hands-On Capstone Project (100 Marks): OOP Digital Library & Multi-Tier Banking Ledger System — Design an object-oriented Banking & Library domain featuring abstract base accounts, Savings/Current subclasses with overdraft rules, @property balance guards, and transaction audit logs.

MODULE 7 OF 16 • 11 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 7: Exception Handling, File I/O, CSV & JSON Serialization

Build resilient programs with try-except-else-finally, custom exceptions, context managers (with open), pathlib, CSV, and JSON.

Objectives: ✓ Handle specific exceptions and raise custom domain errors • ✓ Read and write UTF-8 files safely using context managers • ✓ Serialize & validate nested JSON API payloads

7.1 Python Exception Handling — try, except, else, finally & Custom Errors [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Exception Handling — try, except, else, finally & Custom Errors in real Python systems — specifically: Exception hierarchy, catching specif...

7.2 Python File Handling — Context Managers (with open), Modes & pathlib [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python File Handling — Context Managers (with open), Modes & pathlib in real Python systems — specifically: File modes (r, w, a, rb), automatic de...

7.3 Working with JSON in Python — loads(), dumps(), Custom Encoders & APIs [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Working with JSON in Python — loads(), dumps(), Custom Encoders & APIs in real Python systems — specifically: Mapping Python dicts/lists to JSON o...

Module 7 Hands-On Capstone Project (100 Marks): Fault-Tolerant Server Log Analyzer & JSON/CSV ETL Pipeline — Create a Python ETL pipeline that streams raw application log files, isolates malformed lines into an error audit log without crashing, aggregates HTTP status counts, and exports clean JSON...

MODULE 8 OF 16 • 10 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 8: Searching & Sorting Algorithms in Python (DSA)

Crack coding rounds with iterative/recursive Binary Search, bisect, Merge Sort, Quick Sort, and Python Timsort complexity analysis.

Objectives: ✓ Implement O(log n) Binary Search & boundary variants • ✓ Code Merge Sort & Quick Sort with recurrence analysis • ✓ Compare algorithmic trade-offs using time.perf_counter()

8.1 Linear & Binary Search in Python (Iterative, Recursive & bisect) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Linear & Binary Search in Python (Iterative, Recursive & bisect) in real Python systems — specifically: Halving sorted search spaces in O(log n),...

8.2 Divide & Conquer Sorting — Merge Sort, Quick Sort & Timsort Internals [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Divide & Conquer Sorting — Merge Sort, Quick Sort & Timsort Internals in real Python systems — specifically: Stable O(n log n) Merge Sort, partiti...

Module 8 Hands-On Capstone Project (100 Marks): High-Speed Product Catalog Search & Sorting Benchmark Suite — Build an algorithmic search and ranking benchmark tool on 50,000 synthetic e-commerce products comparing Linear Scan vs Binary Search (bisect) and Merge Sort vs Quick Sort vs native Timsort.

MODULE 9 OF 16 • 13 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 9: Advanced Python — Decorators, Generators, Regex & Web Scraping

Write senior-level Python with higher-order @decorators, memory-efficient yield generators, compiled regular expressions, and HTML scraping.

Objectives: ✓ Build reusable @decorators preserving metadata via functools.wraps • ✓ Stream million-row pipelines in O(1) RAM using yield generators • ✓ Extract structured data using re patterns and HTML parsers

9.1 Python Closures & @Decorators (functools.wraps, Timing & Auth Guards) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Closures & @Decorators (functools.wraps, Timing & Auth Guards) in real Python systems — specifically: First-class functions, lexical closur...

9.2 Python Iterators & Generators — yield, next(), StopIteration & Lazy Streams [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Iterators & Generators — yield, next(), StopIteration & Lazy Streams in real Python systems — specifically: Iterator protocol (__iter__/__n...

9.3 Python Regular Expressions (re Module) & Structured HTML Extraction [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Regular Expressions (re Module) & Structured HTML Extraction in real Python systems — specifically: Raw strings r"...", character classes,...

Module 9 Hands-On Capstone Project (100 Marks): Automated Tech Job & Price Tracker Web Scraper with Regex Cleaner — Build an ethical Python scraper and HTML parser that extracts course/job titles, cleans messy salary/price strings via compiled regular expressions, streams records lazily via generators, an...

MODULE 10 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 10: Python Databases (MySQL, SQLite3) & Tkinter Desktop GUI

Connect Python applications to relational SQL databases safely using parameterized DB-API 2.0 queries and build Tkinter desktop interfaces.

Objectives: ✓ Prevent SQL injection 100% using parameterized placeholders • ✓ Manage transactions (commit/rollback) in SQLite3 & MySQL • ✓ Build event-driven Tkinter desktop CRUD forms

10.1 Python MySQL Database Connectivity, Parameterized Queries & Transactions [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python MySQL Database Connectivity, Parameterized Queries & Transactions in real Python systems — specifically: PEP-249 DB-API 2.0 workflow, mysql...

10.2 Python SQLite3 — Embedded Relational Database, Row Factory & CRUD [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python SQLite3 — Embedded Relational Database, Row Factory & CRUD in real Python systems — specifically: Zero-configuration built-in sqlite3 modul...

10.3 Python Tkinter GUI — Widgets, Grid Geometry, StringVar & Event Callbacks [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Python Tkinter GUI — Widgets, Grid Geometry, StringVar & Event Callbacks in real Python systems — specifically: Building cross-platform desktop ap...

Module 10 Hands-On Capstone Project (100 Marks): Desktop Student Admission, Fee Receipt & Attendance Manager (Tkinter + SQLite3) — Build a complete standalone desktop application using Tkinter (ttk Treeview + forms) backed by a persistent SQLite3/MySQL relational database with parameterized CRUD queries, search filters,...

MODULE 11 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Python Asyncio Event Loop, Coroutines & Concurrent Network I/O

Production-grade Python Training engineering module covering Python Asyncio Event Loop, Coroutines & Concurrent Network I/O with hands-on labs and capstone build.

Objectives: ✓ async/await Coroutines, Tasks & asyncio.gather() — Purpose: Fetch 100+ partner HTTP endpoints concurrently in 200ms instead of 20 sequential seconds. • ✓ Async Context Managers (async with) & Semaphores — Purpose: Bound concurrent outbound sockets cleanly so Python scripts never trigger 429 rate-limit bans. • ✓ Threading vs Multiprocessing vs Python GIL Offloading — Purpose: Use ProcessPoolExecutor for CPU-heavy parsing and asyncio for high-volume network I/O.

11.1 async/await Coroutines, Tasks & asyncio.gather() [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply async/await Coroutines, Tasks & asyncio.gather() in real Python systems — specifically: Fetch 100+ partner HTTP endpoints concurrently in 200ms in...

11.2 Async Context Managers (async with) & Semaphores [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Async Context Managers (async with) & Semaphores in real Python systems — specifically: Bound concurrent outbound sockets cleanly so Python script...

11.3 Threading vs Multiprocessing vs Python GIL Offloading [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Threading vs Multiprocessing vs Python GIL Offloading in real Python systems — specifically: Use ProcessPoolExecutor for CPU-heavy parsing and asy...

Module 11 Hands-On Capstone Project (100 Marks): Async Multi-Endpoint API Scraper & Webhook Dispatcher — Design, implement, test, and document 'Async Multi-Endpoint API Scraper & Webhook Dispatcher' applying every production concept from Module 11 (Python Asyncio Event Loop, Coroutines & Concur...

MODULE 12 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Static Type Checking (mypy), Dataclasses, Pydantic & Clean Packaging

Production-grade Python Training engineering module covering Static Type Checking (mypy), Dataclasses, Pydantic & Clean Packaging with hands-on labs and capstone build.

Objectives: ✓ Python Type Hints, Generics, Protocol & mypy Verification — Purpose: Catch NoneType and dict key typos before runtime across large multi-developer Python repos. • ✓ Immutable @dataclass(frozen=True, slots=True) & Enum — Purpose: Cut per-object RAM usage by 40% and prevent accidental state mutation across functions. • ✓ Modern Packaging with pyproject.toml, Virtualenvs & CLI Entrypoints — Purpose: Publish clean, reproducible Python CLI tools and internal wheels for enterprise teams.

12.1 Python Type Hints, Generics, Protocol & mypy Verification [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Python Type Hints, Generics, Protocol & mypy Verification in real Python systems — specifically: Catch NoneType and dict key typos before runtime...

12.2 Immutable @dataclass(frozen=True, slots=True) & Enum [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Immutable @dataclass(frozen=True, slots=True) & Enum in real Python systems — specifically: Cut per-object RAM usage by 40% and prevent accidental...

12.3 Modern Packaging with pyproject.toml, Virtualenvs & CLI Entrypoints [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Modern Packaging with pyproject.toml, Virtualenvs & CLI Entrypoints in real Python systems — specifically: Publish clean, reproducible Python CLI...

Module 12 Hands-On Capstone Project (100 Marks): Type-Safe Python Domain SDK & Installable pyproject.toml Package — Design, implement, test, and document 'Type-Safe Python Domain SDK & Installable pyproject.toml Package' applying every production concept from Module 12 (Static Type Checking (mypy), Datacl...

MODULE 13 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Automated Testing with Pytest: Fixtures, Parametrize & Mocking

Production-grade Python Training engineering module covering Automated Testing with Pytest: Fixtures, Parametrize & Mocking with hands-on labs and capstone build.

Objectives: ✓ Pytest Fixtures, Scopes (function/module/session) & tmp_path — Purpose: Share clean database and temp-file setups across hundreds of tests without boilerplate. • ✓ @pytest.mark.parametrize Edge-Case Matrices — Purpose: Test 20 boundary inputs (empty strings, negative ₹, Unicode, None) in a single 8-line test. • ✓ monkeypatch & unittest.mock for External API/Clock Isolation — Purpose: Test retry logic and billing cutoffs deterministically without calling live servers.

13.1 Pytest Fixtures, Scopes (function/module/session) & tmp_path [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Pytest Fixtures, Scopes (function/module/session) & tmp_path in real Python systems — specifically: Share clean database and temp-file setups acro...

13.2 @pytest.mark.parametrize Edge-Case Matrices [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply @pytest.mark.parametrize Edge-Case Matrices in real Python systems — specifically: Test 20 boundary inputs (empty strings, negative ₹, Unicode, No...

13.3 monkeypatch & unittest.mock for External API/Clock Isolation [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply monkeypatch & unittest.mock for External API/Clock Isolation in real Python systems — specifically: Test retry logic and billing cutoffs determini...

Module 13 Hands-On Capstone Project (100 Marks): 95%-Coverage Pytest Suite with Deterministic Gateway Mocks — Design, implement, test, and document '95%-Coverage Pytest Suite with Deterministic Gateway Mocks' applying every production concept from Module 13 (Automated Testing with Pytest: Fixtures,...

MODULE 14 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Python Memory Profiling, Generators, Security & Production Logging

Production-grade Python Training engineering module covering Python Memory Profiling, Generators, Security & Production Logging with hands-on labs and capstone build.

Objectives: ✓ Lazy Generators (yield / yield from) & itertools Pipelines — Purpose: Process 10GB CSV/log files line-by-line using < 25MB constant Python RAM. • ✓ Secure Secrets Handling, Subprocess Safety & SQL Parameterization — Purpose: Eliminate command-injection, pickle-deserialization, and hardcoded-key vulnerabilities. • ✓ Structured JSON Logging, Rotating Handlers & Custom Exception Hierarchies — Purpose: Emit searchable production logs with correlation IDs and clean error recovery.

14.1 Lazy Generators (yield / yield from) & itertools Pipelines [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Lazy Generators (yield / yield from) & itertools Pipelines in real Python systems — specifically: Process 10GB CSV/log files line-by-line using <...

14.2 Secure Secrets Handling, Subprocess Safety & SQL Parameterization [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Secure Secrets Handling, Subprocess Safety & SQL Parameterization in real Python systems — specifically: Eliminate command-injection, pickle-deser...

14.3 Structured JSON Logging, Rotating Handlers & Custom Exception Hierarchies [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Structured JSON Logging, Rotating Handlers & Custom Exception Hierarchies in real Python systems — specifically: Emit searchable production logs w...

Module 14 Hands-On Capstone Project (100 Marks): Zero-Leak Streaming Log Processor & Audit CLI — Design, implement, test, and document 'Zero-Leak Streaming Log Processor & Audit CLI' applying every production concept from Module 14 (Python Memory Profiling, Generators, Security & Produc...

MODULE 15 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Modern Python Type Safety, Packaging & CLI/Automation Engineering

Self-explanatory list-and-bullet production mastery module covering Modern Python Type Safety, Packaging & CLI/Automation Engineering in Python Training, paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken Gemini AI Viva.

Objectives: ✓ Strict Type Hints, Protocols, TypedDict & Static Checking with mypy/pyright — Purpose: Catch subtle NoneType and schema bugs in editors and CI pipelines before Python code ever runs. • ✓ Modern pyproject.toml Packaging, Virtual Environments (uv/venv) & Ruff Linting — Purpose: Structure reproducible, dependency-pinned Python projects that install cleanly on any server. • ✓ Resilient CLI Tools, Structured Logging & Retry/Timeout HTTP Automation — Purpose: Automate enterprise file, API, and reporting workflows with clean exit codes and audit logs. • ✓ Build, verify, and document 'Typed Python CLI Data Validator & Wheel Package with ruff + mypy CI' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + Google Gemini AI Technical Interviews.

15.1 Strict Type Hints, Protocols, TypedDict & Static Checking with mypy/pyright [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Strict Type Hints, Protocols, TypedDict & Static Checking with mypy/pyright in real Python systems — specifically: Catch subtle NoneType and schem...

15.2 Modern pyproject.toml Packaging, Virtual Environments (uv/venv) & Ruff Linting [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Modern pyproject.toml Packaging, Virtual Environments (uv/venv) & Ruff Linting in real Python systems — specifically: Structure reproducible, depe...

15.3 Resilient CLI Tools, Structured Logging & Retry/Timeout HTTP Automation [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Resilient CLI Tools, Structured Logging & Retry/Timeout HTTP Automation in real Python systems — specifically: Automate enterprise file, API, and...

Module 15 Hands-On Capstone Project (100 Marks): Typed Python CLI Data Validator & Wheel Package with ruff + mypy CI — Architect, implement, test, and document 'Typed Python CLI Data Validator & Wheel Package with ruff + mypy CI' applying every bullet-wise engineering concept from Module 15 (Modern Python Ty...

MODULE 16 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Python Memory Profiling, Security Hygiene & Coding Interview Mastery

Self-explanatory list-and-bullet production mastery module covering Python Memory Profiling, Security Hygiene & Coding Interview Mastery in Python Training, paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken Gemini AI Viva.

Objectives: ✓ Memory Profiling with tracemalloc, __slots__ & Generator Streaming Pipelines — Purpose: Process multi-gigabyte CSV/JSON datasets in constant RAM without Out-Of-Memory crashes. • ✓ Secure Python Practices: Subprocess Safety, Secrets Handling & Safe Deserialization — Purpose: Prevent command injection, pickle exploits, and credential leaks in production Python scripts. • ✓ Python Internals, Edge-Case Traps & Live Technical Interview Problem Solving — Purpose: Master mutable default traps, GIL trade-offs, and clean Pythonic algorithmic solutions. • ✓ Build, verify, and document 'tracemalloc Memory Auditor & High-Throughput Stream Transformation Engine' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + Google Gemini AI Technical Interviews.

16.1 Memory Profiling with tracemalloc, __slots__ & Generator Streaming Pipelines [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Memory Profiling with tracemalloc, __slots__ & Generator Streaming Pipelines in real Python systems — specifically: Process multi-gigabyte CSV/JSO...

16.2 Secure Python Practices: Subprocess Safety, Secrets Handling & Safe Deserialization [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Secure Python Practices: Subprocess Safety, Secrets Handling & Safe Deserialization in real Python systems — specifically: Prevent command injecti...

16.3 Python Internals, Edge-Case Traps & Live Technical Interview Problem Solving [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Python Internals, Edge-Case Traps & Live Technical Interview Problem Solving in real Python systems — specifically: Master mutable default traps,...

Module 16 Hands-On Capstone Project (100 Marks): tracemalloc Memory Auditor & High-Throughput Stream Transformation Engine — Architect, implement, test, and document 'tracemalloc Memory Auditor & High-Throughput Stream Transformation Engine' applying every bullet-wise engineering concept from Module 16 (Python Mem...

3. LIVE CLIENT PRODUCTION PROJECT, GLOBAL CERTIFICATIONS & CAREER SCOPE

End-of-Course Live Client Production Project + Official Internship Confirmation Letter

Project: DataFlow Automated ETL Reconciliation, CLI Ops & Reporting Engine • **Confirmed Role:** Python Software & Automation Engineering Intern

Domain & Production Infra: FinTech Accounting & Automated Back-Office Operations Client (Python 3.12 + Poetry/uv + Pydantic + SQLite/PostgreSQL + Scheduled Linux Cron/Docker)

- Engineered modular Python 3.12 packages using strict type annotations, dataclasses, custom exceptions, and context managers.
- Built memory-safe generator pipelines processing 500k+ CSV/JSON ledger records with automated anomaly detection.
- Integrated REST API clients with retry/backoff policies, SQLite/PostgreSQL persistence, and structured logging.
- Shipped pytest unit suites (90%+ coverage) and containerized CLI workflows to live client production.

Aligned Global Vendor Certifications: **PCEP-30-02** (PCEP™ – Certified Entry-Level Python Programmer) • **PCAP-31-03** (PCAP™ – Certified Associate in Python Programming) • **PCPP1-32-101** (PCPP1™ – Certified Professional in Python Programming 1)

Target Job Roles: Junior Python Developer, Backend Automation Engineer, Python Scripting & QA Automation Analyst, Associate Software Engineer (Python)

Realistic Annual CTC Scope: **Fresher / Entry-Level (0–1 Yr):** ₹4.5 LPA – ₹8.0 LPA **Mid-Level Python Engineer (1–3 Yrs):** ₹8.5 LPA – ₹16.5 LPA **Senior Backend / AI Specialist (3–6+ Yrs):** ₹18.0 LPA – ₹34.0+ LPA

MockAttempt Academy Pvt. Ltd. • Email: support@mockattempt.com • Phone/WhatsApp: +91 98765 43210 • Web: https://mockattempt.com/courses/python-training

Try 2 Free Topic-Wise 50-MCQ Mock Tests & Spoken Gemini AI Interviews daily on every topic above before enrolling.