Professional Learning Roadmap

This 4.5-month roadmap is designed to help you build a complete skill set in **Python development, automation testing, and database management** - from beginner to professional level. You'll start with core Python programming concepts, data structures, and object-oriented design, then advance into real-world testing using **Pytest** and **Playwright** for both API and UI automation. Along the way, you'll strengthen your understanding of **SQL** for data validation and reporting. The final phase focuses on **project integration, CI/CD pipelines, and teaching mastery**, preparing you to apply your skills confidently in professional environments or as a trainer.

MODULE 1: Python Environment & Introduction

Objective: Learn to set up the Python environment and understand how Python programs run.

Topics Covered:

- Install Python (Windows/Linux/Mac)
- IDE setup (VS Code, PyCharm)
- Environment variables
- pip & virtual-env usage
- Writing & executing `.py` files
- Python syntax rules (indentation, naming)
- Comments (single-line, multi-line)
- Using `input()` and `print()`
- Basic error types & how to fix them

Mini Task: Create a Python script that asks for your name and prints a greeting.

MODULE 2: Data Types, Variables & Operators

Objective: Understand and work with Python's fundamental data types.

Topics Covered:

- Variable declaration and naming rules
- Primitive data types (`int`, `float`, `str`, `bool`, `complex`)
- Type checking (`type()`)
- Type conversion (`int()`, `str()`, etc.)
- Operators:
  - Arithmetic (+, -, *, /, %, **, //)
  - Comparison (==, !=, >, <, >=, <=)
  - Logical (and, or, not)
  - Assignment (+=, -=, etc.)
  - Bitwise operators
- Operator precedence and associativity

Mini Project: Build a simple calculator that supports +, -, *, / operations.

MODULE 3: Conditional Logic

Objective: Use conditions to control the program flow.

Topics Covered:

- `if`, `elif`, `else` syntax
- Nested `if` statements
- Comparison chaining (e.g., `18 <= age < 65`)
- Truthy & Falsy values
- Common logical patterns (login check, eligibility check)

Practice: Write a grade calculator using marks input and conditionals.

MODULE 4: Loops and Iterations

Objective: Repeat code efficiently with loops.

Topics Covered:

- `for` loop and `range()`
- `while` loop
- Loop control: `break`, `continue`, `pass`
- Nested loops
- Looping over strings, lists, and dicts
- `enumerate()` for index iteration

Project: Display multiplication tables (1–10) and print number patterns.

MODULE 5: Functions

Objective: Write reusable blocks of code.

Topics Covered:

- Defining functions with `def`
- Parameters & arguments
- Return values
- Default parameters
- Keyword arguments
- Variable-length arguments (`*args`, `**kwargs`)
- Scope of variables (local/global)
- Docstrings & annotations

Mini Project: Build a function-based calculator or string utility functions.

MODULE 6: Data Structures – Lists, Tuples, Sets, Dictionaries

Objective: Store and manipulate collections of data.

Topics Covered:

- List creation and indexing
- Common list methods (`append`, `remove`, `sort`, `reverse`)
- Tuples: immutability, unpacking

- Sets: unique elements, set operations (union, intersection)
- Dictionaries: key-value storage, iteration
- Nested data structures
- List & dict comprehensions

Project: Manage student marks using a list of dictionaries.

MODULE 7: Strings & Regular Expressions

Objective: Handle and validate textual data.

Topics Covered:

- String creation & slicing
- String methods (`split`, `join`, `replace`, `find`, `count`)
- f-Strings & formatting (`format()`)
- Escape sequences
- Regex basics: `re.match()`, `re.search()`, `re.findall()`
- Regex patterns: `\d`, `\w`, `\s`, anchors, quantifiers
- Validation examples (email, phone, password)

Mini Project: Extract all emails from a text file using regex.

MODULE 8: File Handling

Objective: Store and read persistent data from files.

Topics Covered:

- File modes: read, write, append
- Using `open()` and `with` statement
- Reading & writing text files
- Reading CSV with `csv` module
- JSON handling (`json.load()`, `json.dump()`)
- Exception-safe file handling

Project: Log analyzer — read log.txt and extract failed login attempts.

MODULE 9: Exception Handling & Debugging

Objective: Prevent crashes and handle runtime errors properly.

Topics Covered:

- `try`, `except`, `else`, `finally`
- Catching multiple exceptions
- Custom exception classes
- Using `raise`
- Debugging with `print()` and `pdb`
- Common error types (IndexError, KeyError, TypeError)

Practice: Build a safe division calculator that catches ZeroDivisionError.

MODULE 10: Object-Oriented Programming (OOP)

Objective: Build scalable, reusable class-based systems.

Topics Covered:

- Creating classes & objects
- Constructors (`__init__`)
- Instance & class variables
- Instance & class methods
- Inheritance (single, multiple)
- Method overriding
- Polymorphism & encapsulation
- Dunder methods (`__str__`, `__len__`)

Project: Create a `BankAccount` class with deposit, withdraw, and balance features.

MODULE 11: Advanced Python Concepts

Objective: Use advanced features to write professional-grade code.

Topics Covered:

- Lambda functions
- Map, Filter, Reduce
- Decorators
- Generators (`yield`)
- Modules & Packages (`import`, `__init__.py`)
- Logging with `logging` module
- ConfigParser usage
- Virtual environments (venv)

Mini Project: Logging system for a data processing script.

MODULE 12: SQL – Database Fundamentals

Objective: Learn to use SQL to store and query data.

Topics Covered:

- RDBMS concepts (tables, schema, keys)
- SQL syntax and structure
- CRUD operations: INSERT, SELECT, UPDATE, DELETE
- Constraints: PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL
- Data types (INT, VARCHAR, DATE)
- Sorting and filtering (ORDER BY, WHERE, LIKE)

Practice: Create a “students” database with sample records.

MODULE 13: SQL – Advanced Queries

Objective: Perform powerful and optimized data analysis.

Topics Covered:

- Aggregate functions: SUM, AVG, COUNT, MAX, MIN
- GROUP BY and HAVING
- JOINS: INNER, LEFT, RIGHT, FULL
- Subqueries
- Views and Aliases
- Indexes & Query Optimization

Mini Project: Build a sales report using joins between product and order tables.

MODULE 14: Pytest – Basics

Objective: Learn Python testing fundamentals.

Topics Covered:

- Installing pytest
- Writing and running test cases
- Assert statements
- Test naming conventions
- Setup and teardown with fixtures
- Parameterization (`@pytest.mark.parametrize`)
- Skipping & marking tests

Practice: Write test cases for calculator functions.

MODULE 15: Pytest – Advanced Framework

Objective: Build a structured testing framework.

Topics Covered:

- conftest.py & fixture scope (function, class, module)
- Hooks (`pytest_runtest_setup`, `pytest_configure`)
- Pytest.ini configuration
- Logging integration
- Reporting tools: `pytest-html`, Allure
- Tag-based test runs
- Custom markers & plugins

Mini Project*:* End-to-end test suite for login validation.

MODULE 16: Playwright – Fundamentals

Objective: Automate web applications using Playwright.

Topics Covered:

- Install & setup Playwright
- Record & run test scripts
- Locators (CSS, XPath, role-based)
- Page navigation & element interactions
- Assertions
- Waits: implicit, explicit
- Browser contexts

Practice: Automate login/logout scenario for a demo site.

MODULE 17: Playwright – Advanced Automation

Objective: Perform complex UI automation with Pytest integration.

Topics Covered:

- Handling frames, alerts, dropdowns
- File upload/download
- Screenshots & video capture
- Parallel execution (`pytest-xdist`)
- API testing via Playwright
- Headless vs headed runs
- Fixtures + Page Object Model (POM) design

Mini Project*:* Automate an e-commerce checkout flow.

MODULE 18: API Automation with Python

Objective: Test REST APIs effectively using Python.

Topics Covered:

- REST API basics
- HTTP methods: GET, POST, PUT, DELETE
- `requests` library
- JSON handling
- Status codes & response validation
- Pytest + API testing integration
- Authentication (Bearer, Basic Auth)
- Schema validation

Project: API validation between frontend (Playwright) and backend (requests).

MODULE 19: Test Framework Integration + CI/CD

Objective: Automate continuous integration & testing workflows.

Topics Covered:

- Test framework structure (UI + API + DB)
- Environment config files (YAML, JSON)
- Version control with Git & GitHub
- Continuous Integration using GitHub Actions
- Jenkins pipeline basics
- Parallel execution & report generation
- Test result artifacts

Capstone Project: Unified automation suite for full-stack web testing.

MODULE 20: Teaching & Professional Mastery

Objective: Transition into a confident professional or trainer.

Topics Covered:

- Teaching best practices: Explain → Demo → Practice → Review
- Public speaking for tech topics
- Resume & portfolio building
- GitHub profile enhancement
- Mock interviews & code reviews
- Common QA interview questions
- Creating video demos for LinkedIn

Final Project: Present a full end-to-end automation demo and record it for your teaching portfolio.