Training Plan: AI & Enterprise App Development with Python

(12 Weeks | 3 hours/day | 5 days/week)

Module & Week	Day	Lecture Topic (1 Hour)	Hands-on Exercise (1 Hour)
Module 1: Core Python & Data			
Week 1	Mon	**Introduction to Python:** History, use cases, and setting up the development environment (Python, VS Code).	**Setup & "Hello, World!":** Install Python and VS Code. Write and run your first script. Explore the terminal and basic commands.
	Tue	**Python Syntax, Variables & Data Types:** Numbers, strings, booleans. Type casting.	**Variable Playground:** Create variables of different types. Practice string formatting and perform basic arithmetic operations.
	Wed	**Basic Operators:** Arithmetic, assignment, comparison, and logical operators.	**Simple Calculator:** Build a script that takes two numbers as input and performs all arithmetic operations on them.
	Thu	**Data Structures (Part 1):** Lists and Tuples. Indexing, slicing, methods.	**List Manipulation:** Create a to-do list application. Allow users to add, remove, and view items.
	Fri	**Data Structures (Part 2):** Dictionaries and Sets. Key-value pairs, methods, use cases.	**Contact Book:** Build a simple contact book using a dictionary to store names and phone numbers.
Week 2	Mon	**Control Flow (Part 1):** if, elif, else statements. Conditional logic.	**Number Guessing Game:** Create a game where the program picks a random number and the user has to guess it.
	Tue	**Control Flow (Part 2):** for and while loops. break, continue.	**Data Aggregation:** Loop through a list of numbers to calculate their sum and average.

	Wed	**Functions:** Defining functions, parameters, arguments, return statement.	**Refactor Previous Exercises:** Convert the calculator and to-do list logic into reusable functions.
	Thu	**Introduction to Object-Oriented Programming (OOP):** Classes and objects.	**Basic Class Creation:** Define a Car class with attributes (color, brand) and methods (start, stop).
	Fri	**Python Modules & Standard Library:** math, datetime, random.	**Password Generator:** Build a tool that generates a random, secure password using the random and string modules.
Week 3	Mon	**File I/O:** Reading from and writing to text files (.txt, .csv).	**Log File Analyzer:** Write a script to read a log file and count the number of error and warning messages.
	Tue	**Virtual Environments:** Understanding venv and its importance for dependency management.	**Project Setup:** Create a new project directory with its own virtual environment. Install packages like requests.
	Wed	**Introduction to SQL & Relational Databases:** Core concepts (tables, rows, columns, keys).	**Database Design:** On paper, design a simple schema for a blog (Users, Posts, Comments tables with relationships).
	Thu	**Basic SQL Queries:** SELECT, FROM, WHERE, ORDER BY.	**Querying a Sample DB:** Use an online SQL playground (e.g., SQLite Online) to run basic queries against a sample database.
	Fri	**Advanced SQL Queries:** JOIN, GROUP BY, HAVING.	**Relational Queries:** Write queries to join the Users and Posts tables to find out who wrote which post.
Week 4	Mon	**Data Definition Language (DDL):** CREATE, ALTER, DROP tables.	**Build Your Database:** Write SQL statements to create the tables you designed for the blog schema.
	Tue	**Data Manipulation Language (DML):** INSERT, UPDATE, DELETE data.	**Populate Your Database:** Write SQL statements to insert sample users, posts, and comments into your blog database.
	Wed	**Integrating Python with SQL (Part 1):** Using libraries like sqlite3.	**Connect and Read:** Write a Python script to connect to your SQLite blog database and fetch all posts.

	Thu	**Integrating Python with SQL (Part 2):** Executing DML commands from Python.	**Dynamic Insertion:** Create a Python script that allows a user to input data for a new blog post and inserts it into the database.
	Fri	**Module 1 Review & Project:** Consolidating Python and SQL skills.	**CLI Blog Manager:** Build a command-line application in Python that can add, view, and delete posts from your SQL database.
Module 2: AI & LLM Ecosystem			
Week 5	Mon	**Foundations of AI:** History, types of AI (ANI, AGI), and key terminology.	**AI Use Case Research:** Identify and document three real-world applications of AI, explaining the problem they solve.
	Tue	**Introduction to Machine Learning:** Supervised, unsupervised, and reinforcement learning.	**Model Categorization:** Given a list of problems, categorize them into supervised or unsupervised learning tasks.
	Wed	**Deep Learning & Neural Networks:** Basic concepts of neurons, layers, and activation functions.	**Neural Network Diagram:** Draw a simple diagram of a neural network that could classify images of cats and dogs.
	Thu	**Introduction to Large Language Models (LLMs):** What they are, how they work at a high level.	**Prompt Engineering Basics:** Experiment with a public LLM (like Gemini) to see how different prompts affect the output quality.
	Fri	**The Transformer Architecture:** High-level overview of self-attention, encoders, and decoders.	**Attention Mechanism Explained:** Write a short explanation of how the attention mechanism helps an LLM understand context.
Week 6	Mon	**Text Embeddings:** Representing words and sentences as vectors.	**Vector Similarity:** Use a pre-trained model (via a library) to find the cosine similarity between different words/sentences.
	Tue	**Vector Databases:** What they are and why they are needed for AI applications.	**Explore Vector DB Options:** Research and compare two popular vector databases (e.g., Pinecone, ChromaDB).
	Wed	**Retrieval-Augmented Generation (RAG):** The concept and architecture.	**RAG Workflow Diagram:** Create a flowchart that illustrates the step-by-step process of a RAG query.

	Thu	**Setting up a RAG Pipeline (Part 1):** Loading and chunking documents.	**Document Processing:** Write a Python script to load a text file and split it into smaller, overlapping chunks.
	Fri	**Setting up a RAG Pipeline (Part 2):** Creating embeddings and storing in a vector DB.	**Vector Store Creation:** Use a library like ChromaDB to create embeddings from your text chunks and store them locally.
Week 7	Mon	**Introduction to LangChain:** Core concepts (Chains, Agents, Tools).	**Install LangChain:** Set up a new virtual environment and install LangChain and its dependencies.
	Tue	**LangChain: Models, Prompts, and Parsers:** Integrating LLMs and structuring inputs/outputs.	**First LLMChain:** Build a simple chain that takes a topic and generates a short explanation using an LLM.
	Wed	**LangChain: Building a Basic RAG Chain:** Combining a retriever and a generation model.	**Query Your Documents:** Build a LangChain RAG chain that answers questions based on the document you vectorized last week.
	Thu	**LangChain: Agents and Tools:** Giving LLMs access to external tools (e.g., search).	**Simple Agent:** Create an agent that can use a calculator tool to answer math questions.
	Fri	**Introduction to LangGraph:** Moving from chains to cyclical graphs for multi-step reasoning.	**LangGraph vs. LangChain:** Write a comparison outlining when to use LangGraph over a standard LangChain agent.
Week 8	Mon	**LangGraph: Building a Basic Graph:** Nodes, edges, and state management.	**Simple Two-Step Graph:** Create a graph where the first node generates a question and the second node answers it.
	Tue	**LangGraph: Multi-Agent Collaboration Concepts:** How multiple agents can work together.	**Agent Roles Design:** Design a two-agent system on paper: one "researcher" agent and one "writer" agent for creating a blog post.
	Wed	**LangGraph: Implementing a Two-Agent System:** Passing state between different agent nodes.	**Code the Two-Agent System:** Implement the researcher/writer agent system using LangGraph.
	Thu	**Evaluating LLM Applications:** Metrics and strategies for testing RAG and agentic systems.	**Evaluation Plan:** Create a simple evaluation plan for your RAG application, including sample questions and ideal answers.

	Fri	**Module 2 Review & Project:** Consolidating AI and LangChain skills.	**Conversational RAG Agent:** Build a conversational agent using LangChain that can answer questions about a specific document.
Module 3: Enterprise Dev & AI			
Week 9	Mon	**Intro to Enterprise Web Frameworks:** Django vs. Flask vs. Frappe.	**Framework Comparison:** Create a table comparing the pros and cons of Django, Flask, and Frappe for business applications.
	Tue	**Frappe Framework: Introduction & Architecture:** "Everything is a DocType" philosophy.	**Install Bench & Frappe:** Set up the Frappe development environment by installing the bench CLI.
	Wed	**Frappe Framework: DocTypes:** Understanding fields, naming, and types (System, Standard, Child).	**Create a "Library Member" DocType:** Use the Frappe UI to create a DocType for managing library members.
	Thu	**Frappe Framework: UI & Views:** List View, Form View, and basic customizations.	**Customize Member Form:** Add fields for first name, last name, email, and membership date to your DocType.
	Fri	**Frappe Framework: Creating a Custom App:** The structure of a Frappe app.	**Build a "Library Management" App:** Use bench to create a new, reusable app to house your custom DocTypes.
Week 10	Mon	**Frappe: Linking DocTypes:** Using the "Link" field type for relationships.	**Create "Book" DocType:** Create a Book DocType and link it to a "Library Member" to show who has borrowed it.
	Tue	**Frappe: Controllers & Client Scripts:** Adding custom logic to the UI.	**Validation Script:** Write a client script to validate that the member's email address is in the correct format.
	Thu	**Frappe: Server Scripting (Part 1):** Introduction to Python scripting on the backend.	**Default Value Script:** Write a server script to set the default membership start date to the current date.
	Fri	**Frappe: Server Scripting (Part 2):** DocEvents (on_submit, on_update).	**"Book Issued" Logic:** Write a server script that automatically sets a book's status to "Issued" when it's linked to a member.
	Wed	**Frappe: Reporting:** Building basic reports with the Report Builder.	**"Overdue Books" Report:** Create a simple report that shows all books that have been borrowed for more than 30 days.

Week 11	Mon	**Frappe: REST API:** Understanding Frappe's built-in API for DocTypes.	**API Exploration with Postman:** Use a tool like Postman to fetch, create, and update "Book" records via the API.
	Tue	**Frappe: Custom API Endpoints:** Creating whitelisted Python functions.	**Custom "Check Status" API:** Write a whitelisted Python function that returns the status of a specific book.
	Wed	**Frappe: Permissions & User Roles:** Managing access control.	**Create Roles:** Define "Librarian" and "Member" roles and configure permissions for your DocTypes.
	Thu	**Frappe: Hooks:** Using hooks.py to extend core functionality.	**Email on Submit:** Use a hook to automatically send a welcome email to a new member when their document is submitted.
	Fri	**Frappe Best Practices:** Code organization, deployment, and maintenance.	**Code Review:** Review all the custom scripts written so far and refactor them according to best practices.
Week 12	Mon	**Integrating LangChain into Frappe:** Strategy and architecture.	**Plan the Integration:** Design a feature to add an "AI Book Recommender" to the Library Management app.
	Tue	**Frappe-LangChain (Part 1):** Setting up a custom script that calls a LangChain chain.	**Backend Integration:** Create a whitelisted Frappe server script that takes a book title and uses LangChain to find similar books.
	Wed	**Frappe-LangChain (Part 2):** Creating a custom button in the UI to trigger the AI feature.	**Frontend Trigger:** Add a button to the Book form that calls your server script and displays the AI-generated recommendations.
	Thu	**Frappe-LangChain (Part 3):** Using Frappe data to build a RAG source for LangChain.	**AI-Powered Search:** Create a RAG pipeline using all book descriptions in your Frappe database to answer natural language queries.
	Fri	**Final Project Showcase & Review:** Presenting the final integrated application.	**Build & Present:** Finalize the "AI Library Manager" Frappe app, ensuring all features work. Prepare a short presentation.