Python Fast-Track: Data & Web

Tagline: *Go from zero to deploying interactive data web apps in 8 weeks—using purely Python.*

Here is your detailed, 8-week, 5-days-a-week curriculum.

Phase 1: Core Python Mastery (Weeks 1-4)

Your first month is dedicated to mastering the syntax, logic, and data structures of Python.

Week 1: Fundamentals, Variables, & I/O

- **Day 1: The Ecosystem:** Computer architecture basics, compilation vs. interpretation, installing Python, and configuring your IDE (VS Code or IDLE). Write your first "pre-program."
- **Day 2: The print() Function & Literals:** Deep dive into print(), arguments, escape/newline characters. Understand literals (integers, floats, strings, booleans).
- **Day 3: Math & Operators:** Python as a calculator. Basic operators, priority rules, and shortcut operators.
- **Day 4: Variables & Comments:** Creating variables, naming conventions, dynamic typing, reassigning values, and using comments for code documentation.
- **Day 5: User Interaction:** The input() function, type casting (converting strings to ints/floats), and string operators.
    - *Lab:* Build a simple interactive currency/unit converter.

Week 2: Control Flow & Logic

- **Day 1: Decisions:** Equality and comparison operators. Master if, if-else, and if-elif-else conditional statements.
- **Day 2: Computer Logic:** Logical expressions (and, or, not), logical values vs. bits, and basic bitwise operators.
- **Day 3: The while Loop:** Setting up while loops, avoiding infinite loops, and using the else branch in loops.
- **Day 4: The for Loop:** Iterating with for, mastering the range() function, and using break and continue to control flow.
- **Day 5: Algorithm Practice:** Implement Collatz's hypothesis and the "Vowel Eater" lab to solidify loop logic.

Week 3: Data Structures

- **Day 1: Lists (Basics):** Creating lists, indexing, negative indices, and accessing content. Adding (append, insert) and removing elements.
- **Day 2: Lists (Advanced):** Powerful list slices, in and not in operators, and navigating 2D/multidimensional arrays. Understand the Bubble Sort algorithm.
- **Day 3: Tuples:** Sequence types, understanding mutability (and why Tuples are immutable), and combining them with lists.
- **Day 4: Dictionaries:** Creating key-value pairs, accessing data, and using dictionary methods/functions.
- **Day 5: Data Modeling Lab:** Build a complex data structure (like a list of dictionaries) to represent a real-world dataset (e.g., a music playlist or student roster).

Week 4: Functions & Error Handling

- **Day 1: Function Basics:** Decomposition, defining functions, and understanding positional vs. keyword arguments.
- **Day 2: Returns & Scopes:** The return instruction, handling None, and understanding local vs. global scopes (the global keyword).
- **Day 3: Advanced Functions:** Multi-parameter functions (BMI, Factorials, Fibonacci) and an introduction to recursion.
- **Day 4: Exceptions:** Handling errors with the try-except branch, dealing with multiple exceptions, and default exceptions.
- **Day 5: Debugging & Testing:** Print debugging, testing methodologies, and handling "bad" data gracefully.
    - *Milestone:* Month 1 Core Python Assessment.

Phase 2: The Data Layer (Week 5)

Transitioning from basic Python scripts to managing persistent data using a relational database.

Week 5: MySQL Integration

- **Day 1: Database Fundamentals:** Relational DB concepts, tables, columns, rows. Install MySQL Server and MySQL Workbench.
- **Day 2: Core SQL:** Writing basic queries: CREATE TABLE, INSERT, and SELECT with WHERE clauses.
- **Day 3: Advanced SQL:** Updating records (UPDATE), deleting records (DELETE), and understanding Primary/Foreign Keys and basic JOINs.
- **Day 4: Connecting Python to MySQL:** Install mysql-connector-python. Learn to establish a connection, create cursors, and execute SQL queries directly from a Python script.
- **Day 5: Database Lab:** Build a Python Command-Line Interface (CLI) app that performs CRUD (Create, Read, Update, Delete) operations on a MySQL database.

Phase 3: The Interactive Frontend (Week 6)

Turning your Python logic into beautiful, interactive web applications without HTML/CSS.

Week 6: Streamlit Mastery

- **Day 1: Streamlit Basics:** Installation, understanding how Streamlit runs (top-to-bottom execution), and basic text elements (st.title, st.write, st.markdown).
- **Day 2: Input Widgets:** Gathering user data with st.text_input, st.number_input, st.selectbox, st.checkbox, and st.button.
- **Day 3: Displaying Data:** Rendering lists and dictionaries as clean tables using st.dataframe and st.table. Introduction to simple charts (st.bar_chart).
- **Day 4: Layouts & Containers:** Organizing the UI using st.columns, st.sidebar, st.tabs, and st.expander.
- **Day 5: Session State (Crucial):** Understanding st.session_state to remember user interactions (like keeping a user logged in or tracking a counter) across app reruns.

Phase 4: Fullstack Integration & Capstone (Weeks 7-8)

Wiring the UI, the backend logic, and the database together into a cohesive product.

Week 7: Connecting Streamlit & MySQL

- **Day 1: Secure Connections:** Setting up Streamlit to connect to MySQL securely. Fetching data from the DB and displaying it in a Streamlit DataFrame.
- **Day 2: Creating Data (The 'C' in CRUD):** Building Streamlit forms (st.form) that take user input, validate it, and write it to the MySQL database.

- **Day 3: Updating & Deleting (The 'U' & 'D'):** Adding interactive elements allowing users to select an existing record, edit its values, or delete it from the database entirely.
- **Day 4: Error Handling & UX:** Using Python try-except blocks to handle database connection drops. Displaying success/error messages using st.success() and st.error().
- **Day 5: Integration Lab:** Build a fully functional, single-page Dashboard (e.g., a Task Manager) that reads and writes to MySQL in real-time.

Week 8: The Capstone Project

- **Day 1: Architecture & Design:** Define your final project scope. Design the MySQL database schema and mock up the Streamlit UI flow.
- **Day 2: Database & Backend Build:** Execute table creation in MySQL. Write all the Python helper functions needed to query the database.
- **Day 3: Frontend UI Build:** Build out the Streamlit interface, connect it to your helper functions, and test the user flow.
- **Day 4: Polish & Data Visualization:** Add dynamic charts to your dashboard, refine the layout (columns/sidebars), and ensure the UI looks professional.
- **Day 5: Final Presentation:** Project freeze. Walk through your fully functional Python/MySQL/Streamlit web application.

This plan gives you a steady ramp-up through logic and data structures before diving into the fast-paced world of web apps and databases.