Transitioning from manual testing to **Python automation testing** is a smart career move. Here’s a structured breakdown of the **skills** you’ll need to develop, grouped into logical categories:

**✅ 1. Core Python Skills**

You must understand the basics of Python before diving into automation:

* Variables, data types, loops, and conditionals
* Functions and modules
* Exception handling
* File handling (read/write files)
* OOP concepts (classes, inheritance, etc.)
* List, dict, set, tuple operations

🔹 Resources: w3schools Python, [Python Docs](https://docs.python.org/3/)

**✅ 2. Automation Tools & Libraries**

These tools are widely used in Python-based test automation:

**📌 Selenium with Python**

* WebDriver basics (launching browser, locating elements, actions)
* Waits (implicit, explicit)
* Handling dropdowns, alerts, frames
* Taking screenshots, logging

**📌 PyTest *(most popular framework)***

* Writing and running tests
* Fixtures
* Parametrize
* Assertions
* Markers (@pytest.mark)

**📌 Unittest *(alternative to PyTest)***

* Test classes, setup/teardown
* Test discovery

**📌 Other Useful Libraries**

* requests – API testing
* openpyxl or pandas – reading Excel or CSV
* json – handling JSON data (common in API tests)
* logging – for debug and reports

**✅ 3. Testing Fundamentals**

Even though you're moving from manual, ensure you're strong in:

* SDLC and STLC
* Test case design techniques (boundary value, equivalence, etc.)
* Bug life cycle
* Black box testing

**✅ 4. Version Control**

* **Git** basics (clone, commit, push, pull, branch, merge)
* GitHub or GitLab for code collaboration

**✅ 5. CI/CD Basics**

Understanding of:

* Jenkins or GitHub Actions for test automation pipelines
* How to integrate test scripts in CI/CD

**✅ 6. Reporting**

* Use pytest-html, Allure, or logging module to generate test reports
* Knowing how to share logs and screenshots when tests fail

**✅ 7. Basic Knowledge of APIs**

* Understand how REST APIs work
* Test APIs using requests library or Postman
* Validate response codes, JSON payloads

**✅ 8. Databases**

* Basic SQL for validating backend data
* Use sqlite3, mysql-connector-python to connect Python to DB

**✅ 9. Soft Skills**

* Clear communication of automation logic
* Collaboration with developers and other testers
* Writing clear, maintainable code and test cases

**🧠 Optional (but beneficial)**

* **Behavior-Driven Development (BDD)** – e.g., using behave
* **Docker** – containerize your automation framework
* **Cloud testing tools** – BrowserStack, Sauce Labs, etc.

**🧭 Suggested Learning Path:**

1. Python basics (2–4 weeks)
2. Selenium + PyTest (3–4 weeks)
3. API testing + reporting + Git (2–3 weeks)
4. Build a framework (mini project)
5. Learn CI/CD basics and integrate

Would you like a **custom learning plan or roadmap** based on your current level?

**You said:**

yes

**ChatGPT said:**

Updated saved memory

Great! Here's a **custom 8-week learning roadmap** to transition you from **manual testing** to a **Python automation tester**. You can adjust the pace based on your availability.

**🗺️ 8-Week Roadmap to Become a Python Automation Tester**

**🔰 Week 1–2: Core Python Fundamentals**

Goal: Build a strong foundation in Python

**Topics:**

* Variables, data types, operators
* Loops, conditionals
* Functions, scope, modules
* Lists, dictionaries, tuples, sets
* File handling (read/write to files)
* Exception handling
* Object-Oriented Programming (classes, objects, inheritance)

**Practice:**

* Solve 5–10 problems daily on HackerRank or [LeetCode Easy]

**🧪 Week 3: PyTest Basics + Assertions**

Goal: Learn unit and automation test structure in PyTest

**Topics:**

* Installing and running pytest
* Writing simple test functions
* Assertions (assert, assert in, etc.)
* Fixtures (setup/teardown)
* Parametrize tests
* Organizing tests into files/folders

**Practice:**

* Create a sample test project
* Write tests for functions you wrote earlier

**🌐 Week 4–5: Selenium Web Automation with Python**

Goal: Automate real websites and build test scenarios

**Topics:**

* Install and set up selenium
* WebDriver basics (ChromeDriver)
* Locators (id, class, XPath, CSS)
* Actions (click, send\_keys, etc.)
* Handling alerts, frames, dropdowns
* Waits (explicit/implicit)
* Page Object Model (POM) basics

**Practice:**

* Automate a login test on a demo website (e.g., https://the-internet.herokuapp.com/login)
* Build your first small framework using POM

**🔌 Week 6: API Testing + Reporting**

Goal: Understand and test APIs, add reporting

**Topics:**

* HTTP methods (GET, POST, PUT, DELETE)
* API status codes, headers, body
* Python requests library
* Validate response JSON
* Use pytest for API tests
* Add reporting: pytest-html, logging, screenshots

**Practice:**

* Write tests for public APIs (e.g., https://jsonplaceholder.typicode.com)
* Generate and share a sample HTML report

**🛢️ Week 7: Database Testing + Git**

Goal: Learn DB test integration + use version control

**Topics:**

* SQL basics (SELECT, JOIN, WHERE)
* Connect to a database using Python (sqlite3 or mysql-connector-python)
* Validate data from DB after a UI/API test
* Git basics: clone, commit, push, pull
* Create a GitHub repository for your code

**Practice:**

* Write tests that read data from DB and assert results

**🔁 Week 8: Final Project + CI/CD Introduction**

Goal: Combine everything into one framework

**Final Project:**

* Build a hybrid test framework using:
  + Selenium + PyTest + POM
  + API tests using requests
  + DB validation
  + Logging and HTML reports
* Upload it to GitHub

**Bonus:**

* Jenkins or GitHub Actions: run test suite automatically
* Set up test triggers and view results in reports

**📚 Resources**

* Python: [RealPython](https://realpython.com), w3schools
* Selenium: selenium.dev
* PyTest: docs.pytest.org
* APIs: [Postman](https://www.postman.com/) for learning, requests for Python
* SQL: [SQLBolt](https://sqlbolt.com/)

Would you like a **downloadable checklist** or **Notion/Excel tracker** to manage your weekly goals?

Top of Form

Tools

Bottom of Form

ChatGPT can make mistakes. Check important