Week 3 – Live Session FAQ

1. What is Week 3 actually about in this programme?

Week 3 is about moving from “writing small scripts” to “using and organising real-world Python code”. In the live sessions, you will mainly:

- Learn how to **handle errors** using try / except (and related patterns) so your programs fail more gracefully instead of just crashing.
- Work with **Python libraries**:
    - Standard libraries like math, random, datetime, os, json, string.
    - Third-party libraries like requests, numpy, pandas, matplotlib, seaborn.
- Build and run a small **CLI (command-line) calculator** spread across multiple files to illustrate project structure and modularisation.
- Use **NumPy and pandas** to work with arrays and tabular data.
- Do basic **data visualisation** in Python using Matplotlib and Seaborn.

The focus is on getting comfortable with “real code + real libraries” that you can reuse in later weeks.

---

2. How does Week 3 connect to Weeks 1–2 and to the later agentic AI content?

There’s a clear progression:

- Weeks 1–2: basic Python, control flow, and functions; getting used to reading and running code.
- Week 3:
    - Shows how real projects are built from **modules, libraries, and packages** instead of one long file.
    - Introduces standard and third-party libraries you’ll see again when you connect Python to APIs, data, and eventually agents.
    - Emphasises **project structure** and modular functions that later become “tools” or “skills” your agents call.

Later agentic weeks assume you can:

- Import and use libraries without fear.
- Read and navigate multi-file projects.
- Understand how code, data, and external APIs fit together.

Week 3 is where that starts to feel normal.

3. What exactly do you mean by “error handling”, and what will we do with it?

Error handling is about anticipating that code can fail and dealing with it cleanly. In Week 3 you will:

- Review common error types (like ValueError, TypeError, ZeroDivisionError) and how they show up.
- Use try / except blocks to catch and respond to predictable problems, such as invalid user input or network issues.
- See else and finally in context, and where they make sense.
- Apply these patterns in practical examples (e.g., validating numbers for the calculator, handling failed HTTP requests).

You won’t be building a full logging/monitoring system; the goal is basic, robust patterns you can reuse everywhere else.

4. What are Python libraries, and which ones will we actually see in class?

A library is pre-written code you can import instead of reinventing it yourself. This week you’ll see both:

- **Standard libraries** (come with Python):
    - math – mathematical functions.
    - random – random numbers and sampling.
    - datetime – dates and times.
    - os – interacting with the operating system.
    - json / string – handling text and JSON data.
- **Third-party libraries** (installed with pip):
    - requests – calling HTTP APIs.
    - numpy – numerical arrays and fast math.
    - pandas – tabular data and analysis.
    - matplotlib and seaborn – plotting and visualisation.

You will see concrete examples of when and how to import these, not just lists of names.

5. How deep do we go into APIs and web requests in Week 3?

The focus is on **basic, hands-on usage**, not full API integrations. You will:

- Use requests to make simple HTTP calls to a **public API**.

- Inspect the **status code** and handle obvious errors.
- Work with JSON responses using the json module (e.g., convert to Python dictionaries and read a few fields).
- See how requests and json can be combined in small scripts.

You will **not** be building production-grade integrations, dealing with auth flows, or managing complex rate-limiting or retries here. This week is about understanding the pattern and trying it once in a controlled way.

---

6. What is the CLI calculator, and why are we spending time on it?

The CLI calculator is a small command-line program that gives you a realistic but manageable project to structure. It is used to:

- Apply your knowledge of **functions** for each operation (add, subtract, multiply, divide).
- Practise **modularisation** by splitting code across files like:
    - operations.py – arithmetic functions.
    - utils.py – helper functions (e.g., input validation with try / except).
    - main.py – the user menu and overall flow.
- Reinforce clean **project structure** and good habits (clear entry point, reusable helpers, separate logic).

It's not about making a fancy app; it's about understanding how a small but real Python project is organised.

---

7. What do you mean by "project structure" in this week?

Project structure is simply **how you organise your files and modules** so the code is maintainable. In Week 3, you will:

- See examples of splitting logic into multiple .py files.
- Use import to pull in functions from those files instead of copy-pasting.
- Learn why things like a dedicated utils.py or operations.py help you:
    - avoid repetition,
    - keep each file small and focused,
    - make testing and debugging easier.

You're not packaging a library for PyPI; you're taking the first steps from "one monolithic script" to "structured code".

---

8. How much NumPy and pandas will we actually cover?

The goal is to give you a **practical first pass**, not full data-science training. You will:

- Learn what a **NumPy array** is and why it's faster and more convenient than plain lists for numerical work.
- See basic array operations (e.g., element-wise math, simple statistics).
- Learn pandas core objects: **Series** and **DataFrame**.
- Load small datasets into a DataFrame and perform simple operations:
    - selecting columns and rows,
    - filtering based on conditions,
    - sorting and removing duplicates,
    - basic summary statistics.

Enough to make "NumPy and pandas in agent workflows" feel familiar later, not intimidating.

---

9. What will we do with data visualisation in Matplotlib and Seaborn?

You'll work at the **intro level** of plotting, aimed at getting you comfortable with the tools:

- In Matplotlib:
    - line plots,
    - bar charts,
    - histograms,
    - scatter plots.
- In Seaborn:
    - slightly higher-level plots with nicer defaults (e.g., count plots, distribution plots).
- You'll see how to:
    - feed DataFrames into these libraries,
    - adjust basic parameters (titles, labels, simple styling).

We're not doing dashboarding, plotly, or full "data storytelling" here—just enough plotting to interpret and inspect data in a notebook.

---

10. Do I need a strong maths or statistics background for Week 3?

No. The maths and stats used here are **basic**, such as:

- Simple arithmetic, min/max, averages.
- Reading distributions and basic trends from simple plots.

What matters more is:

- Being comfortable reading small code snippets.
- Following how data flows through arrays, DataFrames, and plots.

If you can follow Week 2's logic and loops, you're in range for Week 3.

---

11. Will I be creating my own Python library or package in this week?

Not in the full "publish to PyPI" sense. What you will do is:

- Group related functions into modules like operations.py or utils.py.
- Import those modules into other files using import and from module import ....
- Think of these modules as your **personal mini-libraries** inside a project.

Formal packaging, versioning, and distribution are beyond this week's scope. Here, you're just learning how to structure code in a way that could later be packaged if needed.

---

12. How much time should I still be putting into Python practice now that we're on libraries?

Python doesn't stop being important after Week 2—in fact, from Week 3 onwards, it becomes more critical, especially as we move towards agentic systems.

Concrete practice suggestions for this week:

- Rebuild the **CLI calculator** yourself from scratch (without looking), then compare.
- Write tiny scripts that:
    - call a public API with requests,
    - parse a JSON response and print a few fields,
    - load a small CSV into pandas and filter by a condition,
    - plot a quick chart of that filtered data.
- Use ChatGPT to:
    - explain any error messages you don't understand,
    - suggest small extensions (e.g., "add a new operation to the calculator" or "add one more plot type").

The more you practise now, the more natural it will feel later when agents are orchestrating these same kinds of functions.

---

13. How will we use ChatGPT or other AI tools in Week 3 if the focus is on libraries and data?

AI tools show up as **assistants, not as the main subject**. They are mainly used to:

- Explain error messages or confusing stack traces.
- Suggest ways to refactor repetitive code into helper functions or modules.
- Help generate small example snippets (e.g., a basic requests call, a simple pandas filter) that you then run and tweak.

The session keeps attention on the Python libraries and project structure, with AI tools helping you clear roadblocks faster.

---

14. What will not be covered in Week 3, even though it sounds related?

To keep Week 3 focused and manageable, the following are **out of scope**:

- Advanced API topics: OAuth, complex auth flows, webhooks, or streaming APIs.
- Large-scale data engineering (big data frameworks, distributed systems).
- Advanced statistics or machine-learning models in NumPy/pandas.
- Interactive or web-based dashboards.
- Full packaging, publishing, or dependency-management workflows.

Those require more infrastructure and context than makes sense at this stage. Week 3 is about core building blocks you'll keep using later.

---

15. If I feel lost during Week 3, what should I prioritise understanding?

If the content feels heavy, narrow your focus to the **highest-leverage basics**:

- Being comfortable reading and writing import statements.
- Understanding what each of these does in simple examples:
    - math, random, datetime, json, requests.
    - numpy arrays and basic operations.
    - pandas DataFrames and simple filters.
- Following the flow of the **CLI calculator** code across multiple files.

If you can:

- Explain how the calculator works end-to-end, and
- Load a small dataset with pandas and produce one or two simple plots,

you’re in solid shape for what the programme expects from Week 3 and for the more agent-focused weeks that follow.