

Facilitator Guide – Week 3

Course: Agentic AI (Sync Course)

Topic: Working with Python Libraries & Error Handling

Duration: 6 Hours (360 minutes)

Session Type: Expert-led Lecture + Hands-On

Note: This session plan provides a recommended flow of activities and references to tools or libraries for the live session. The Programme Leader (PL) may modify or adapt these activities, tools, or demonstrations at their discretion based on learner needs and session dynamics.

Session Overview

This session introduces participants to Python libraries (standard and third-party), error handling, and data analysis with NumPy, Pandas, Matplotlib, and Seaborn. Participants will learn how to import and use libraries effectively, handle errors gracefully, and apply modular coding practices. The session culminates in building practical projects like a CLI calculator and visualizations.

Learning Objectives

By the end of this session, participants will be able to:

1. Understand why error handling is essential for robust programming.
2. Apply try, except, else, and finally blocks for structured error handling.
3. Differentiate between standard and third-party libraries.
4. Use common libraries such as math, random, requests, json, NumPy, and Pandas.
5. Apply data visualization with Matplotlib and Seaborn.
6. Build a CLI calculator project with modularized code.
7. Practice modularization by organizing functions and libraries effectively.

Pro Tips for Facilitation

- **Normalize Errors:** Emphasize that encountering errors is part of coding, not failure.

- **Live Debugging:** Demonstrate error handling with real-time mistakes for authenticity.
- **Show Quick Wins:** Use small library demos (random, datetime) to hook interest.
- **Modular Mindset:** Reinforce “don’t repeat yourself” (DRY) by showing modular code structure.
- **Visual Impact:** Use visualizations early to energize the room—graphs help learners “see” Python’s power.
- **Promote Collaboration:** Let learners extend the calculator together for peer-driven creativity.

Session Details

Topic	Facilitator Focus	What Learners Do	Timing
Opening	Context: Why libraries + error handling matter. Icebreaker: “What’s the most frustrating error you’ve faced?”	Share experiences.	20 min
Error Handling	Explain syntax vs runtime errors, common exceptions, structured error handling.	Predict/fix error scenarios.	60 min
Python Libraries – Introduction	Define libraries, differentiate standard vs third-party.	Poll: libraries used.	30 min
Standard Libraries	Introduce math, random, datetime, os.	Perform calculations, random sims, system checks.	40 min
Third-Party Libraries	Introduce requests, json, NumPy, Pandas. Demo API + data handling.	Parse JSON, use arrays/tables.	50 min
Data Visualization	Introduce Matplotlib + Seaborn. Compare usage.	Create line, bar, histogram, scatter plots.	60 min
Project – CLI Calculator	Build modular calculator (main.py, operations.py, utils.py).	Extend with new functions in groups.	70 min
Closing	Recap, reflection, preview next session.	Share takeaways.	30 min

Total: 360 minutes (6 hours)

Session Breakdown (Narrative Flow)

Opening Activities (20 min)

- **Context Setting (10 min):** Explain why error handling and libraries are critical for professional-grade programming.
- **Icebreaker (10 min):** Learners share the most frustrating coding error they've faced and how they dealt with it.

Topic 1: Error Handling (60 min)

- **Notes:** Explain syntax vs runtime errors. Introduce common exceptions: `ZeroDivisionError`, `ValueError`, `TypeError`, `IndexError`, `KeyError`, `FileNotFoundError`.
- **Demo:** Show a try-except-else-finally block.
- **Activity:** Learners predict outcomes of error scenarios (e.g., divide by zero, missing file).
- **Discussion:** Why structured handling improves debugging and stability.

Topic 2: Python Libraries – Introduction (30 min)

- **Notes:** Explain what libraries are and how they accelerate coding.
- **Discussion:** Standard vs third-party libraries.
- **Engagement:** Quick poll—Which libraries have you already used?

Topic 3: Standard Libraries (40 min)

- **Notes:** Cover math (formulas), random (simulations), datetime (time operations), os (system-level).
- **Hands-On:** Learners compute factorials, generate random numbers, log timestamps, and check system paths.
- **Discussion:** Everyday relevance of standard libraries.

Topic 4: Third-Party Libraries (50 min)

- **Notes:** Introduce installing/importing packages. Cover:
 - requests for API calls.
 - json for parsing.
 - NumPy for arrays.
 - Pandas for tabular data.
- **Demo:** API call with requests, parse JSON, basic NumPy array, Pandas DataFrame.
- **Hands-On:** Learners replicate tasks and explore dataset handling.

Topic 5: Data Visualization (60 min)

- **Notes:** Introduce Matplotlib for customizable plotting; Seaborn for high-level statistical plots.
- **Hands-On:** Learners create line, bar, histogram, and scatter plots.
- **Engagement:** Compare when Matplotlib vs Seaborn is best suited.
- **Discussion:** Why visualisation matters for AI/ML projects.

Topic 6: Project – CLI Calculator (70 min)

- **Notes:** Walk through modular design with separate files (main.py, operations.py, utils.py).
- **Demo:** Build basic calculator.
- **Hands-On:** Learners extend functionality (e.g., add percentage operation).
- **Engagement:** Groups present variations of their calculators.
- **Discussion:** Benefits of modularization (reusability, readability, debugging ease).

Closing Activities (30 min)

- **Recap (10 min):** Reinforce error handling, libraries, visualization, modularization.
- **Reflection (10 min):** Learners share one project where they can apply libraries and modular coding.
- **Preview (10 min):** Next session—**File Handling & Advanced Error Management.**

Engagement Strategies

- Predict-error exercises to demystify debugging.
- Quick polls to activate prior knowledge.
- Hands-on exploration of each library.
- Visualization tasks for immediate payoff.
- Mini-project (CLI calculator) to integrate multiple concepts.
- Peer sharing of modularization approaches.

Facilitation Tips

- Keep **40% time hands-on** to reinforce concepts.
- Encourage learners to explain error handling logic in simple terms.
- Watch out for **pitfalls**: wildcard imports (from x import *), overly broad exceptions (except Exception).
- Emphasize real-world impact: libraries save time and reduce errors.
- Promote group creativity during calculator extension.

Follow-Up & Reflection

- **Task:** Write down two libraries you want to explore further and how they could help in your work.
- **Practice:** Extend the CLI calculator with new features (e.g., memory, exponentiation).
- **Resources:** Share docs for Python Standard Library, NumPy, Pandas, Matplotlib, Seaborn.