



TEACHING GUIDE - 2026-2027

Programming with Python: Advanced

UGRA_015140

Departments	Data, Analytics, Technology and Artificial Intelligence (DATA), Dept. of Operations, Innovation & Data Sciences
Teaching Languages	English
ECTS	2
Teacher responsible	Carles Roger Riera Molina - carles.riera1@esade.edu

Course Goals

A Python notebook can produce the correct result and still be difficult for anyone else to run, understand or maintain. Files may depend on a particular execution order, paths may only work on one laptop and external services may make the project fragile.

This course shows students how to turn a working notebook or prototype into a small but reliable Python project. Students will learn how to execute code from the terminal, organise it into scripts and modules, configure inputs safely, produce useful errors and logs, and protect the most important behaviour with automated tests.

By the end of the course, students should be able to:

1. Navigate a Python project and execute scripts from the terminal.
2. Identify hidden notebook state and move reusable code into functions and modules.
3. Organise source code, tests, data, outputs and documentation in a clear project structure.
4. Accept input and output parameters through a simple command-line interface.
5. Keep paths, settings and secrets outside the main program logic.
6. Validate essential inputs and produce informative failures.
7. Use logging to make a program's execution understandable.
8. Write unit tests and a smoke test using pytest.
9. Isolate an external API or service so that the project can be tested without making real paid calls.
10. Document the environment and execution process so that another person can run the project.

The course is advanced because it focuses on the reliability and maintainability of the complete project, not because it introduces more complex algorithms.

Previous knowledge

Students are expected to be able to:

1. Read and modify Python code using variables, conditions, loops and functions.
2. Import and use external libraries.
3. Read and write simple files or work with tabular data.
4. Understand the general purpose of an input, a transformation and an output.

Previous experience with software engineering, testing, command-line interfaces or machine learning is not required. A starter project and short preparation material will be provided.

Prerequisites

Introductory Python programming experience is required.

Intermediate Python programming & a glimpse into AI applications, Algorithmic Thinking & Programming, or equivalent experience is recommended. Intermediate Python is helpful but is not a mandatory administrative prerequisite.

Teaching methodology

The course follows a project-based and highly practical approach. Students will begin with a notebook or small prototype that works, but is difficult to reproduce or maintain. Across five teaching sessions, they will progressively transform it into a structured Python project.

The course follows a single progression:

- notebook script project configurable application tested workflow

Each session combines:

1. A short explanation of the practical problem.
2. A live refactoring or debugging demonstration.
3. A guided modification of the starter project.
4. An applied challenge.
5. A checkpoint in which students run, inspect and discuss the resulting project.

All sessions will work on the same project. This avoids disconnected exercises and allows students to see how individual practices (functions, configuration, validation, logging and tests) contribute to a complete result.

The project may include data processing or an external AI service as its business case. The external service is treated as a dependency to be integrated reliably, not as a topic in machine-learning theory.

Lectures and in-class exercises will represent approximately 50% of the student workload. Session preparation, assignments and preparation for the

final examination will represent the remaining workload.

A prepared environment, starter repository and fallback data or responses will be provided so that class time can focus on programming rather than installation problems.

Course materials, source code, instructions and supporting resources will be available through Moodle.

Description

Course contribution to program

Business analytics and AI projects often begin as quick prototypes. Their value, however, depends on whether they can be repeated, shared and trusted.

This course develops the practical Python skills required to move from individual experimentation to collaborative and reusable work. It helps students understand how to:

- convert exploratory code into an executable workflow;
- reduce dependence on a particular notebook or laptop;
- make inputs, outputs and configuration explicit;
- detect invalid data or configuration early;
- understand what a program did through useful logs;
- protect critical behaviour with automated tests;
- integrate external services without making the whole project fragile;
- explain clearly how another person can run the project.

These skills complement courses in programming, data analysis and artificial intelligence.

The practices covered are relevant to analytics, consulting, operations, technology management and any role in which a prototype must become a dependable tool.

Short description

Many Python projects work only inside the notebook—and only on the laptop of the person who created them.

In this hands-on course, students will start from a working prototype and turn it into a small professional Python project.

They will move code into scripts and modules, run it from the terminal, add configuration and a command-line interface, produce useful logs and errors, and write automated tests.

By the end of the course, students will have a project that another person can install, run, test and understand. The focus is not on adding more

Bibliography

Okken, B., Python Testing with pytest, Pragmatic Bookshelf (Book)

Percival, H., & Gregory, B., Architecture Patterns with Python, O'Reilly Media, Inc (Book)

Content

#	Topic
1	Moving Beyond the Notebook. Students will learn to transform exploratory notebook code into reusable functions and executable Python scripts. They will use the terminal, work with file paths, and identify hidden dependencies that make notebooks difficult to reproduce.
2	Structuring a Maintainable Python Project. Students will learn to organise source code, tests, data, outputs, and documentation into a clear project structure. They will divide a workflow into modules and functions with explicit responsibilities, inputs, and outputs.
3	Building Reliable and Testable Python Applications. Students will learn to validate essential inputs, produce informative errors, and use logging to understand the execution of a program. They will write automated tests with pytest and isolate external services so that the project can be tested without relying on live or paid API calls.
4	Delivering a Reproducible End-to-End Workflow. Students will learn to integrate the course practices into a configurable Python application that can be executed from the terminal. They will document the environment, configuration, inputs, outputs, and limitations so that another person can prepare, run, test, and understand the project.

Assessment

Tool	Assessment tool	Category	Weight %
Attendance and punctuality	Class participation	Ordinary round	10.00%
Individual or team exercises	Sessions preparation	Ordinary round	10.00%
Individual or team exercises	Assignment 1	Ordinary round	20.00%
Individual or team exercises	Assignment 2	Ordinary round	25.00%
Written and/or oral exams	Final exam	Ordinary round	35.00%
Written and/or oral exams	Retake exam	Retake	100.00%

PROGRAMS

BBA20-Bachelor of Business Administration (BBA) (Undergraduates: Business)

BBA20 Year 4 (Optative)

BBA20 Year 3 (Optative)

BBA23-Bachelor of Business Administration (BBA) (Undergraduates: Business)

BBA23 Year 3 (Optative)

BBA23 Year 4 (Optative)

BBA25-Bachelor of Business Administration (BBA) (Undergraduates: Business)

BBA25 Year 1 (Optative)

BBA25 Year 3 (Optative)

BITLASI22-Bachelor in Transformational Leadership and Social Impact (Undergraduates: Business)

BITLASI22 Year 4 (Optative)

GBD20-Double Degree in Business Administration and Law (Undergraduates: Law)

GBD20 Year 4 (Optative)

GBD20 Year 2 (Optative)

GBD20 Year 3 (Optative)

GBD20 Year 1 (Optative)