

TP-2: Clean Coding & Modular

Exercise 1: Apply PEP8 and Docstrings

Objective

Practice clean code formatting and documentation using **PEP8** and **docstrings**.

Instruction

Refactor the messy code below to:

- Follow PEP8 (indentation, spacing, variable naming).
- Add docstrings to explain each function.

Exercise 2: Add Configuration and Constants

Objective

Use a **configuration module** to separate constant values (like file paths or thresholds) from main code.

Instruction

1. Create a new file named `config.py`.
2. Move constant values (like CSV path or threshold age) to that file.
3. Import and use them in your functions.

Exercise 3: Add Logging and Exception Handling

Objective

Implement **logging** and **try/except** blocks instead of print statements.

Instruction

1. Import and configure the `logging` module.
2. Replace `print()` with appropriate logging levels.
3. Handle `FileNotFoundError` using a `try/except` block.

Exercise 4: Modularize the Project

Objective

Split code into multiple [modules](#) inside a package for clean structure.

Instruction

1. Create the following structure:
2. preprocessing_package/
 3. |— __init__.py
 4. |— data_loader.py
 5. |— data_cleaner.py
 6. |— config.py
7. Move each function into its proper file.
8. Import and use them in `main.py`.

Exercise 5: Package Setup & Code Quality Check

Objective

Convert module into a Python package and check code style with [flake8](#) or [black](#).

Instruction

1. Create a `setup.py` file:
 2. `from setuptools import setup, find_packages`
 - 3.
 4. `setup(`
 5. `name="preprocessing_package",`
 6. `version="0.1",`
 7. `packages=find_packages(),`
 8. `install_requires=["pandas"],`
 9. `description="Simple preprocessing package for data science",`
 10. `author="Your Name",`
 11. `)`
12. Install locally:
 13. `pip install -e .`
14. Run:
 15. `black .` # to auto-format
 16. `flake8 .` # to check for style errors
17. Submit your formatted and working project.