Data-Driven Application Project

🎯 Project Overview

In groups of 2 or 3 persons, you will build a complete data application that processes, analyzes, and visualizes real-world data. Your application must be containerized with Docker and can be easily deploy locally. You can also deploy it in the cloud if you wish to.

🏷 Requirements

Core Technologies

- **Python 3.11+** for backend logic
- **Pandas** for data manipulation and analysis
- **Docker** for containerization
- **Streamlit** and/or **Dash** and/or **FastAPI** for user interface/API
- ... And any other libraries as needed (e.g., requests, matplotlib, seaborn, pytest, etc.)

Additional Technologies

- **Web Scraping** (BeautifulSoup, Scrapy, or Selenium)
- **External API Integration** (REST APIs, weather, finance, etc.)
- **Large Language Models** (OpenAI API, Hugging Face, local LLMs)
- **Database** (PostgreSQL, MongoDB, SQLite)
- **Machine Learning** (scikit-learn, XGBoost for predictions)
-

🏗 Project Structure

The structure of yout project will vary depending on your specific application, but here is a suggested layout:

```
your-project-name/
├── .gitignore
├── Dockerfile
├── compose.yml (optional)
├── requirements.txt
├── README.md
├── .env (optional)
├── config.py (optional)
├── config.yaml (optional)
├── src/
│   ├── __init__.py
│   ├── your_python_module_1.py # your module containg your functions
│   └── your_python_module_2.py # another module
├── app/
│   ├── streamlit_app.py        # If using Streamlit
│   └── fastapi_app.py          # If using FastAPI
```

```
├── data/
│   ├── raw/                    # Original data
│   └── processed/              # Cleaned/processed data
├── tests/
│   └── test_data_processing.py (optional)
└── notebooks/
    └── exploration.ipynb       # Data exploration or statistics (optional)
```

📊 Project Ideas

There are a lot of Open data sources and APIs available online. Why not build a sports analytics app, a real estate market analyzer, a crypto price tracker, a climate impact analysis app, an e-commerce price comparison tool, or a movie/music recommendation engine?

📅 Timeline

The source code must be handle in a compressed file and submit on Ametice before the **31st of January 2026 at 23:59**.

Inside the README.md file, you must provide a link to a public Git repository containing all the source code, Dockerfile, compose.yaml, requirements.txt, and any other necessary files to run the project.

Don't forget to write the names of all group members in the README.md file.

You can also provide a link to the image on Docker Hub if you have pushed it there.

If your data have a size greater than 4-5 Mo, do not include the data files in the repository and provide scripts to download and preprocess the data.

💡 Best Practices

Code Quality

- Use type hints in functions
- Write docstrings for all functions
- Follow PEP 8 style guide
- Use logging instead of print()

Data Handling

- Never commit raw data files > 4-5MB
- Use `.gitignore` for sensitive data and large files
- Handle API rate limits gracefully
- Implement error handling and retries

Docker

- Use `.dockerignore` to exclude unnecessary files
- Keep image size small (use slim base images)

- Use environment variables for configuration
- Document how to run the container

API Integration

- Use environment variables for API keys
- Implement rate limiting awareness
- Add request timeouts
- Log API responses for debugging
- Handle API errors gracefully

📧 Questions?

Reach me at virgile.pesce@univ-amu.fr

Good luck! 🚀