Executive summary — ecommerce-ETL (Books demo)

Project: End-to-end ETL pipeline — scrape → clean → validate → load
Author / Repo: EthoKikon /ecommerce-ETL
Date: 16-09-2025

Objective

Build a reproducible ETL pipeline that extracts product metadata from a public e-commerce demo site, cleans and validates the data, and stores it in a Postgres database for downstream analytics.

Approach

- **Extract:** Web scraping using `requests` + `BeautifulSoup` to collect product metadata and save `raw_books.csv`.
- **Transform:** Data cleaning with pandas; robust price parsing to handle encoding artefacts; schema validation via `pandera`. Cleaned data saved as parquet `outputs/books_clean.parquet`.
- **Load:** Load cleaned data into a Postgres `books` table using SQLAlchemy.
- **Orchestration:** Prefect flow (`src/flow.py`) orchestrates extract → transform → load.
- **Testing & Quality:** Unit tests (`pytest`) and pre-commit hooks (`black, isort, flake8`) enforce quality.

Key results (example run)

- **Records processed:** 20 product records (books.toscrape.com).
- **Data outputs:** `outputs/books_clean.parquet` (cleaned snapshot).
- **Loaded rows:** 20 rows in Postgres `books` table (verified via SQL).
- **Tests:** 2 unit tests passed locally and in CI.
- **Reproducibility:** `python -m src.cli --pages 1` or `python -m src.flow` reproduce the run.

Deliverables

- Production-ready code for extract/transform/load: `src/`
- Prefect flow for orchestration: `src/flow.py`
- Docker Compose for local Postgres: `docker-compose.yml`
- Notebook demo + visualizations: `docs/notebook.html` (or `notebooks/dev_playground.ipynb`)
- Short demo screenshots in `docs/images/` (terminal logs, DB verification, chart).

How to evaluate (quick)

1. Run the pipeline locally (see README quickstart).
2. Confirm the Postgres table has expected rows:

```sql
SELECT COUNT(*) FROM books;
SELECT * FROM books LIMIT 5;
```