# 1. Technical Architecture Document

This project is a News Aggregation and Analysis System developed using Python. It collects news articles from multiple sources, saves them in a PostgreSQL database, and allows exporting data into CSV format. The project is structured in a modular way to separate configuration, scraping, data handling, and reporting.  
  
The core of the system includes the following modules:  
  
1. Scrapers:  
These are Python scripts using libraries such as BeautifulSoup and Requests for regular scraping, and Scrapy for more complex crawling. The scrapers fetch news from various sources like BBC, CNN, PromptCloud blog, Al Jazeera Sports, and an RSS feed.  
  
2. Database Module:  
This module contains SQLAlchemy models and logic to connect to a PostgreSQL database. All scraped articles are stored with information such as title, link, summary, author, source, and published date.  
  
3. Configuration:  
Settings for scrapers are stored in YAML format under `config/scrapers.yaml`. This allows easy editing of the scraping targets and settings without changing the code.  
  
4. Report:  
This module contains logic to export articles from the database into CSV files. The exported CSV can be used for analysis or sharing.  
  
5. Tests:  
Unit tests are written to check that models work correctly and scrapers insert data as expected. Tests are placed in the `tests/` folder.  
  
6. Scrapy Spider:  
A separate Scrapy spider (`sports\_spider`) is implemented under the `scrapy\_crawler` folder. It uses Scrapy’s item pipeline to save sports articles into the database.  
  
7. Combined Runner:  
A `combined.py` script is included to run all scraper scripts at once.

The system uses a virtual environment and has all dependencies listed in `requirements.txt`.

# 2. User Guide with Examples

To use this project, make sure you are in the virtual environment and have PostgreSQL running.  
  
Example: Run the BBC scraper  
  
> python src/scrapers/bbc\_scraper.py  
  
Example: Run the CNN scraper  
  
> python src/scrapers/cnn\_scraper.py  
  
Example: Run all scrapers together :  
  
> python combined.py  
  
Example: Export all articles from the database to CSV:  
  
> python src/report/export\_csv.py  
  
Example: Run the Al Jazeera sports Scrapy spider:  
  
> $env:PYTHONPATH=(Get-Location).Path; cd src/scrapers/scrapy\_crawler; scrapy crawl sports  
  
Example: Run tests:  
  
> python -m unittest tests.test\_scrapers.py

# 3. Setup and Installation Instructions

1. Clone the project from GitHub or download the folder.  
2. Navigate to the root folder: PY\_FINAL\_PROJECT  
3. Create and activate a virtual environment:  
  
> python -m venv venv  
> venv\Scripts\activate  
  
4. Install the required dependencies:  
  
> pip install -r requirements.txt  
  
5. Set up PostgreSQL and create a database named `final\_project`.  
6. Make sure `settings.yaml` has the correct database URL.  
7. Adjust `scrapers.yaml` if needed.  
8. Run any scraper or report script using the examples from Section 2.

# 4. API Reference for Modules

- src/config/settings.yaml: Contains global configuration (like database URL).  
- src/config/scrapers.yaml: Contains configuration for each scraper.  
- src/data/database.py: Establishes connection to the database.  
- src/data/models.py: SQLAlchemy model for NewsArticle.  
- src/scrapers/[source]\_scraper.py: Scraper for each news source.  
- src/scrapers/drop\_table.py: Deletes all entries in the database.  
- src/scrapers/main.py: Optional main script to coordinate execution.  
- src/report/export\_csv.py: Exports all articles to `articles\_export.csv`.  
- combined.py: Runs all scrapers.  
- scrapy\_crawler/reuters/spiders/sports\_spider.py: Scrapy spider for Al Jazeera Sports.  
- scrapy\_crawler/reuters/pipelines.py: Saves Scrapy spider output to database.  
- tests/test\_scrapers.py: Checks if scraping and saving to DB works.  
- tests/test\_models.py: Checks data model behavior.