
Assignment 1: CSV to Summary DAG

Objective: Create a DAG that reads a CSV file, counts rows, and logs the result.

- Step 1: Write a task that checks if `data/customers.csv` exists.
- Step 2: Read the CSV file and count total rows.
- Step 3: Log the row count.
- Bonus: Send a message (use BashOperator) if the count is greater than 100.

Assignment 2: Daily Sales Report

Objective: Automate a daily sales summarization process.

- Schedule: Run every day at 6 AM.
- Step 1: Read `sales.csv` from a predefined folder.
- Step 2: Group by category and calculate total sales amount.
- Step 3: Save the summary as a new CSV file.
- Step 4: Archive the original file to another folder.
- Bonus: Fail the DAG if any step takes more than 5 minutes.

Assignment 3: Retry with Alerts

Objective: Simulate a flaky API call with retries and alert on final failure.

- Step 1: Add a task that randomly fails.
- Step 2: Configure retries and exponential backoff.
- Step 3: After all retries, send an alert/log a message.
- Bonus: Add a task that only runs **if** the API succeeded.

Assignment 4: Branching Logic

Objective: Create a DAG with conditional logic based on file size.

- Step 1: Check file size of `data/inventory.csv`.
- Step 2: If file is small (<500KB), run Task A (light summary).
- Step 3: If large, run Task B (detailed processing).
- Step 4: Join back for a common cleanup task.
- Bonus: Implement using BranchPythonOperator.

Assignment 5: Dynamic DAG with Multiple Files

Objective: Process any number of new `.csv` files in a folder dynamically.

- Step 1: Detect how many `.csv` files are in `/data/input/`.
- Step 2: For each file, generate a task that:
  - Validates headers
  - Calculates row count
  - Writes a summary
- Step 3: Merge all summaries into a single output file.
- Bonus: Use TaskFlow API or dynamic task mapping.
