
Assignment 1: File Sensor Pipeline

Objective: Build a DAG that waits for a file to arrive before processing.

- Step 1: Use a FileSensor to wait for the presence of `data/incoming/report.csv`.
- Step 2: Once detected, read and process the file.
- Step 3: Move the processed file to an archive folder.
- Bonus: Set the sensor to timeout after 10 minutes and fail gracefully.

Assignment 2: Data Quality Validation

Objective: Validate incoming data before further processing.

- Step 1: Read `data/orders.csv`.
- Step 2: Validate that all required columns are present.
- Step 3: Check for nulls in mandatory fields.
- Step 4: If validation passes, proceed to summarization.
- Step 5: If validation fails, stop the DAG and log an error.

Assignment 3: Trigger Another DAG

Objective: Chain two DAGs together using an external trigger.

- Step 1: Create `parent_dag` that performs a simple task.
- Step 2: At the end of `parent_dag`, trigger `child_dag`.
- Step 3: `child_dag` should run only when triggered by `parent_dag`.
- Bonus: Pass metadata (like date) from `parent_dag` to `child_dag`.

Assignment 4: Retry and Timeout Handling

Objective: Build a DAG with strict retry and timeout behavior.

- Step 1: Create a task that simulates long-running work (e.g., `sleep`).
- Step 2: Configure a hard timeout for the task.
- Step 3: Add retries with exponential backoff.
- Step 4: Log clear messages for success or failure scenarios.

Assignment 5: Time-Based Conditional Tasks

Objective: Run different tasks based on the time of day.

- Step 1: Use a Python task to check the current system time.
- Step 2: If it's morning, run Task A.
- Step 3: If it's afternoon, run Task B.
- Step 4: All branches should lead to a final cleanup task.
- Bonus: Add a condition for weekends to skip the DAG entirely.

Assignment 6: Email Notification Workflow

Objective: Send email alerts based on task outcomes.

- Step 1: Create a simple DAG with two tasks.
- Step 2: Configure an email alert for task failures.
- Step 3: Send a success email only after all tasks complete successfully.

- Bonus: Use Airflow Variables for dynamic recipient configuration.

Assignment 7: External API Interaction

Objective: Build a DAG that interacts with a public API.

- Step 1: Use an HTTP operator to query a public API (e.g., weather, crypto price).
- Step 2: Parse the API response and extract relevant data.
- Step 3: Log the data or save it to a file.
- Step 4: Fail the DAG if the API response status is not 200.
