Name: Sebastian James Sampao Section: 3DSA

*Formative Lab Exercise #5:*

**Workflow Orchestration with Kestra**

# **Kestra Overview**

**A diagram of a computer

AI-generated content may be incorrect.**

Sources:

<https://kestra.io/docs/architecture>

1. Kestra Key Features

Kestra is an open-source orchestration platform that:

* Lets you define workflows declaratively in YAML
* Allows non-developers to automate tasks with a no-code interface
* Keeps everything versioned and governed, so it stays secure and auditable
* Extends easily for custom use cases through plugins and custom scripts.

Kestra allows you to combine the flexibility of code-based orchestration with a no-code interface that anyone can learn in minutes.

Architecture

1. JDBC Backend

* This is similar to the Apache Airflow Metadata Database, this contains the metadata for the orchestration based on either mySQL or PostgreSQL

1. Server

* The central part of the system contains:
  1. Scheduler
  + The component that is responsible for scheduling jobs, sending tasks to the executor, and managing the flow of events in the workflow.
  1. Executor
  + Responsible for the orchestration logic including flow triggers.
  1. Worker
  + These are the processes that carry out the computation of runnable tasks or jobs
  1. Webserver for the
  + Serves both the API and the User Interface

Sources:

<https://kestra.io/docs/why-kestra>

<https://kestra.io/docs/architecture>

1. Differences with Airflow

* YAML based instead of the code based
* The UI allows you to edit directly the workflow
* Less verbose than Airflow
* Does not require Python knowledge to create complex pipelines
* Execution is event driven unlike Airflow that rely on third-party executioners like Celery

# **Lab Part 0: Getting Started with Kestra**

### **Task 0.1: Setting up Kestra using Docker Compose**

*Bash*

curl -o docker-compose.yml \

https://raw.githubusercontent.com/kestra-io/kestra/develop/docker-compose.yml

### **Task 0.2: Accessing the Kestra UI**

*Bash*

docker-compose build

docker-compose up

*Web Browser*

Access: localhost:8080  
Username: [admin@kestra.io](mailto:admin@kestra.io)  
Password: Admin1234

📸 Screenshot Submission 0.3:

Show that Kestra is running on Docker and the UI is accessible

A black screen with colorful text

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

🧠 Checkpoint Question Submission 0.4:

Checkpoint: What are the main services that Kestra launches when run with Docker Compose?

🧠 Checkpoint Question Submission 0.5:

Checkpoint: How does the Kestra UI help you understand what’s happening in the background?

# **Lab Part 1: Writing Your First DAG on Kestra**

### **Task 1.1: Create a simple sequential DAG in YAML**

*YAML*

id: hello-world

namespace: simple\_dag

tasks:

  - id: step1

    type: io.kestra.plugin.core.log.Log

    message: "Hello! Running first step"

  - id: step2

    type: io.kestra.plugin.core.log.Log

    message: "Goodbye! Running 2nd and last step"

📸 Screenshot Submission 1.2:

YAML file for your first DAG on Kestra

A screenshot of a computer

AI-generated content may be incorrect.

📸 Screenshot Submission 1.3:

Show your simple Kestra flow loaded and a successful run.

A screenshot of a computer

AI-generated content may be incorrect.

🧠 Checkpoint Question Submission 1.4:

In your YAML, what structure is used to define a sequence of tasks?

🧠 Checkpoint Question Submission 1.5:

Compared to Airflow’s BashOperator, how does Kestra’s equivalent task behave?

# **Lab Part 2: Writing Your First DAG on Kestra**

### **Task 2.1: Use a download task to fetch NYC Yellow Taxi data**

📸 Screenshot Submission 2.2:

YAML file for your Ingestion DAG

A screen shot of a computer

AI-generated content may be incorrect.

📸 Screenshot Submission 2.2:

Show your ingestion DAG and log output of data download.

A screenshot of a computer

AI-generated content may be incorrect.

🧠 Checkpoint Question Submission 1.4:

Which Kestra task did you use to download the taxi dataset?

🧠 Checkpoint Question Submission 1.4:

How did you make sure the file was saved to the correct location?

# **Lab Part 3: Executing and Monitoring DAG on Kestra**

📸 Screenshot Submission 3.3:

Show the execution status and the run view (graph view).

📸 Screenshot Submission 3.4:

Show the execution status and the run view (timeline view).

🧠 Checkpoint Question Submission 3.5:

What’s the difference between a failed flow run and a failed task?

🧠 Checkpoint Question Submission 3.5:

What features does Kestra provide to help you debug failed flows?

# **Lab Part 4: Executing and Monitoring DAG on Kestra**

📸 Screenshot Submission 4.2:

Show your flow using a dynamic date-based filename and execution logs.

🧠 Checkpoint Question Submission 4.3:

What features does Kestra provide to help you debug failed flows?

🧠 Checkpoint Question Submission 4.4:

Why is it powerful to pass values into a DAG at runtime?

**Appendix 1: docker-compose.yaml**

Appendices

* 1. Include your customized docker-compose.yml, sample YAML flows, and any configs used