**Lab Exercise 17- Built-in Tagging feature in Metaflow**

Here is the corrected version of the tagging\_flow.py file using current.run.add\_tag() instead of current.step.add\_tag():

**Corrected File: tagging\_flow.py**



from metaflow import FlowSpec, step, current

class TaggingFlow(FlowSpec):

@step

def start(self):



# Add a tag to the current run (flow run, not a specific step)

current.run.add\_tag('start\_step\_run')



self.message = "This is a demo of tagging and metadata."



print(self.message)



self.next(self.process\_data)

@step

def process\_data(self):

# Simulate some processing

current.run.add\_tag('processing\_step\_run') # Add a tag to this run



self.processed\_data = self.message.upper()



print(f"Processed Data: {self.processed\_data}")

self.next(self.end)

@step



def end(self):

# Final step of the flow

current.run.add\_tag('end\_step\_run')



print("Flow finished successfully!")

if \_\_name\_\_ == '\_\_main\_\_':

TaggingFlow()

**Key Changes:**

* I replaced current.step.add\_tag() with current.run.add\_tag() because tagging individual steps is not supported in Metaflow.

You can still inspect the tags at the run level, and all tags will apply to the entire run rather than specific steps.

**Running the Flow:**



python tagging\_flow.py run --tag my\_experiment --tag version1

This will add the my\_experiment and version1 tags globally to the run, and the additional tags (start\_step\_run, processing\_step\_run, end\_step\_run) will be added during execution inside each step. You can inspect the tags with the Metaflow client, as shown earlier.

To inspect the tags added during the run, you can use the Metaflow Client API. Here’s the code to check and inspect the tags for a specific run of your TaggingFlow. This script will allow you to see the tags that were added both when running the flow (my\_experiment, version1) and dynamically during the steps (start\_step\_run, processing\_step\_run, end\_step\_run).

**File: inspect\_tags.py**

# inspect\_tags.py

from metaflow import Flow

def inspect\_tagging\_flow():

# Fetch the latest run of the TaggingFlow

run = Flow('TaggingFlow').latest\_run

# Print the run ID and tags associated with the entire run

print(f"Run ID: {run.id}")

print(f"Run Tags: {run.tags}")

# Iterate through each step in the flow and print the step name

for step in run:

print(f"Step: {step.id}")

print(f"Step Tags: {step.tags}")

if \_\_name\_\_ == '\_\_main\_\_':

inspect\_tagging\_flow()

**How the Code Works:**

1. **Fetching the Latest Run:**  
   This script fetches the latest run of your TaggingFlow using Flow('TaggingFlow').latest\_run.
2. **Printing Tags for the Entire Run:**  
   It prints out the tags associated with the entire run (my\_experiment, version1, start\_step\_run, etc.) using run.tags.
3. **Inspecting Each Step:**  
   It iterates through each step in the flow (start, process\_data, and end), printing the step name (step.id) and any tags associated with that step. Since tags are applied at the run level, the tags shown will be those associated with the entire run.

**How to Run the Inspection Code:**

1. Run your tagging flow first to ensure the tags are applied:

bash

Copy code

python tagging\_flow.py run --tag my\_experiment --tag version1

1. After the flow completes, run the inspection script:

bash

Copy code

python inspect\_tags.py

**Expected Output:**

bash

Copy code

Run ID: 5

Run Tags: {'my\_experiment', 'version1', 'start\_step\_run', 'processing\_step\_run', 'end\_step\_run'}

Step: start

Step Tags: {'my\_experiment', 'version1', 'start\_step\_run'}

Step: process\_data

Step Tags: {'my\_experiment', 'version1', 'processing\_step\_run'}

Step: end

Step Tags: {'my\_experiment', 'version1', 'end\_step\_run'}

**Explanation of Output:**

* **Run ID:** The unique identifier for the specific flow run (e.g., 5).
* **Run Tags:** These are the global tags applied to the run, including the ones added when the run was executed (my\_experiment, version1), as well as those added inside the steps (start\_step\_run, processing\_step\_run, end\_step\_run).
* **Step Tags:** While tags are applied globally at the run level, this script also shows the run-level tags for each step, reflecting the overall flow context.

To inspect the tags added during the run, you can use the Metaflow Client API. Here’s the code to check and inspect the tags for a specific run of your TaggingFlow. This script will allow you to see the tags that were added both when running the flow (my\_experiment, version1) and dynamically during the steps (start\_step\_run, processing\_step\_run, end\_step\_run).

**File: inspect\_tags.py**

# inspect\_tags.py

from metaflow import Flow

def inspect\_tagging\_flow():

# Fetch the latest run of the TaggingFlow

run = Flow('TaggingFlow').latest\_run

# Print the run ID and tags associated with the entire run

print(f"Run ID: {run.id}")

print(f"Run Tags: {run.tags}")

# Iterate through each step in the flow and print the step name

for step in run:

print(f"Step: {step.id}")

print(f"Step Tags: {step.tags}")

if \_\_name\_\_ == '\_\_main\_\_':

inspect\_tagging\_flow()

**How the Code Works:**

1. **Fetching the Latest Run:**  
   This script fetches the latest run of your TaggingFlow using Flow('TaggingFlow').latest\_run.
2. **Printing Tags for the Entire Run:**  
   It prints out the tags associated with the entire run (my\_experiment, version1, start\_step\_run, etc.) using run.tags.
3. **Inspecting Each Step:**  
   It iterates through each step in the flow (start, process\_data, and end), printing the step name (step.id) and any tags associated with that step. Since tags are applied at the run level, the tags shown will be those associated with the entire run.

**How to Run the Inspection Code:**

1. Run your tagging flow first to ensure the tags are applied:

python tagging\_flow.py run --tag my\_experiment --tag version1

1. After the flow completes, run the inspection script:

python inspect\_tags.py

**Expected Output:**

Run ID: 5

Run Tags: {'my\_experiment', 'version1', 'start\_step\_run', 'processing\_step\_run', 'end\_step\_run'}

Step: start

Step Tags: {'my\_experiment', 'version1', 'start\_step\_run'}

Step: process\_data

Step Tags: {'my\_experiment', 'version1', 'processing\_step\_run'}

Step: end

Step Tags: {'my\_experiment', 'version1', 'end\_step\_run'}

**Explanation of Output:**

* **Run ID:** The unique identifier for the specific flow run (e.g., 5).
* **Run Tags:** These are the global tags applied to the run, including the ones added when the run was executed (my\_experiment, version1), as well as those added inside the steps (start\_step\_run, processing\_step\_run, end\_step\_run).
* **Step Tags:** While tags are applied globally at the run level, this script also shows the run-level tags for each step, reflecting the overall flow context.