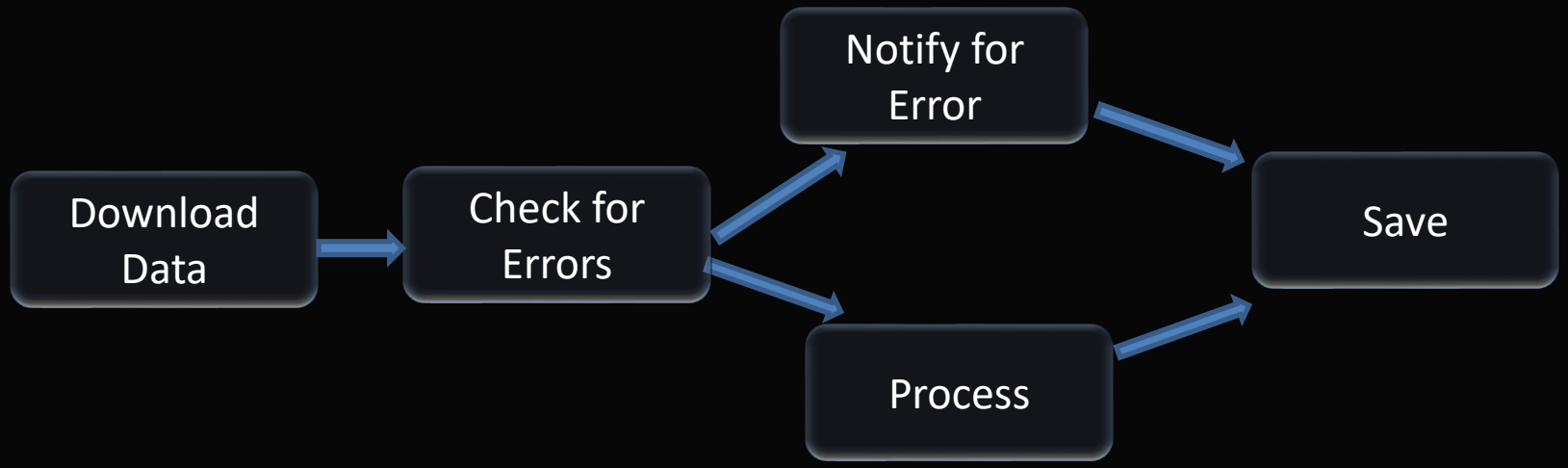
Assignment 1

First DAG

Duration – 15 mins

Create a DAG with name “Report_Analysis” with below details

- Start_date - yesterday’s date 1 pm
- Schedule interval - at every 30 min such as 1:30,2:30,3:30 so on
- Tag - assignment
- catchup - true
- Task - Add five tasks with dummy operator as shown below



Please note: Restart your webserver container to visualize newly added DAG on UI (it wil take 1-2 mins to restart)
docker restart airflow-docker_airflow-webserver_1
docker ps (make sure that your webserver container is healthy before you check in browser)
Also make sure you unpause your dag

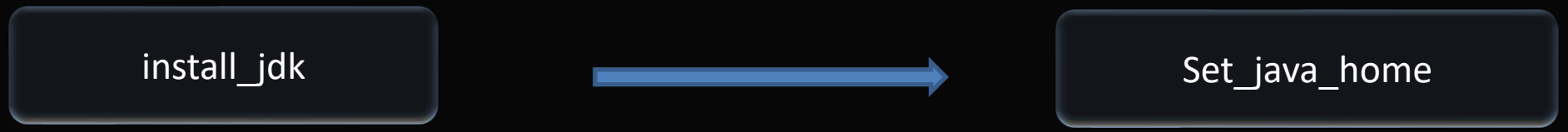
Assignment 2

BashOperator

Duration – 15 mins

Create a DAG with name “bash_operator_assignment” with below details

- Start_date - yesterday’s date
- Schedule interval - @daily
- Tag - assignment
- Task – Add two Bash Operator tasks as shown below



bash_command =
“cd /opt/airflow &&
curl -L -C - -b ‘oraclelicense=accept-securebackup-cookie’ -O
https://download.oracle.com/java/17/latest/jdk-17_linux-x64_bin.tar.gz &&
tar –xvf linux-x64_bin.tar.gz && rm –rf jdk-17_linux-x64_bin.tar.gz”

bash_command="echo 'export JAVA_HOME=opt/airflow/jdk-17.0.6' >> ~/.bashrc && source ~/.bashrc"

1. Restart your webserver to see this newly added DAG on UI docker restart airflow-docker_airflow-webserver
2. Now go to your worker container
   docker exec -it airflow-docker_airflow-worker_1 /bin/bash
3. Run this command
   echo $JAVA_HOME (Did you see JAVA_HOME set to /opt/airflow/jdk-17.0.6)

Assignment 3

Python Operator

Duration – 15 mins

Create a DAG with name "python_operator_assignment" with below details

- Start_date - yesterday's date
- Schedule interval - @daily
- Tag - assignment
- Task – Add two python Operator tasks as shown below

Use PythonOperator

Create a function that accepts two argument
And return the added value

Use TaskFlow API

Create a function that accepts two argument and returns multiplication as an output

1. Restart your webserver to see this newly added DAG on UI docker restart airflow-docker_airflow-webserver
2. Check out the logs of the above task to see the returned values of the function

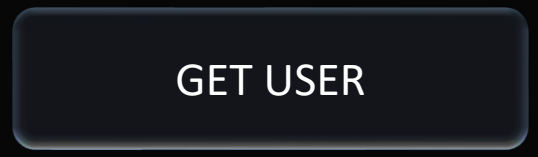
Assignment 4

HTTP Operator

Duration – 15 mins

Create a DAG with name "http_operator_assignment" with below details

- Start_date - yesterday's date
- Schedule interval - @daily
- Tag - assignment
- Task – Add task as shown below



Use SimpleHTTPOperator
Endpoint = api/users/2
Connection = reqres (Make sure you create this connection in Airflow UI with this name and host as https://reqres.in and connection type as "HTTP")

1. Restart your webserver to see this newly added DAG on UI docker restart airflow-docker_airflow-webserver
2. Check out the logs of the above task to see the returned values of the function
3. Go to postgres and check out connection details in connection table maintained by airflow

```
docker exec -it airflow-docker_postgres_1 /bin/bash
psql -U airflow
\dt
select * from connection;
```

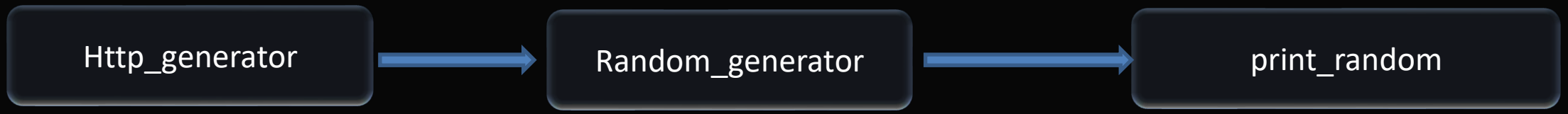
Assignment 5

Xcom

Duration – 20 mins

Create a DAG with name “xcom_assignment” with below details

- Start_date - yesterday’s date
- Schedule interval - @daily
- Tag - assignment
- Task – Add Operator tasks as shown below



Use SimpleHTTPOperator
Endpoint = api/users/2
Connection = reqres

(Make sure you create this connection in Airflow UI with this name and host as https://reqres.in and connection type as “HTTP”)

Use PythonOperator
Create a function that generates a random number
And push that random number as xcom variable with key as “random”
It also pulls json produced by last task and print it

Use PythonOperator
to pull xcom named “random” generated by last task
And print the value

1. Restart your webserver to see this newly added DAG on UI docker restart airflow-docker_airflow-webserver
2. Check out the xcom of “random_generator” task to see the xcom produced and logs to see printed json.
3. Check out the logs of “print_random” task.

Assignment 6

Variables

Duration – 15 mins

Create a DAG with name “variable_assignment” with below details

- Start_date - yesterday’s date
- Schedule interval - @daily
- Tag - assignment
- Task – Add two postgres Operator tasks as shown below

Use PostgresOperator

Create a table in postgres database with below sql.
“Create table if not exists employees
(name varchar(25), department varchar(25)),
created_at varchar(25)”
Make sure to insert this sql as a variable from outside.

Create CustomPostgresOperator

to populate employees table.
Use parameters to dynamically populate values.
Create variables for name and department.
Insert “ds” predefined variable inside created_at.
Remember to use ‘{{ds}}’ syntax

1. Restart your webserver to see this newly added DAG on UI
   docker restart airflow-docker_airflow-webserver
1. Check out the rendered tab of above tasks to see rendered values.
2. Check out the postgres and see what is inserted.
   docker exec -it airflow-docker_postgres_1 /bin/bash
   psql -U airflow
   \dt
   select * from employees;

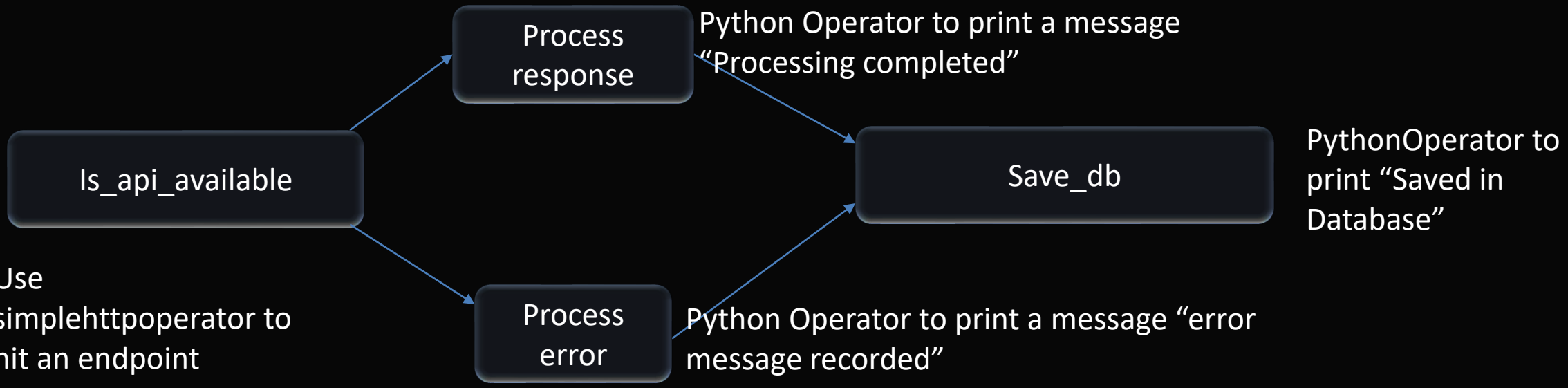
Assignment 7

Branching Operator

Duration – 15 mins

Create a DAG with name “branch_operator_assignment” with below details

- Start_date - yesterday’s date
- Schedule interval - @daily
- Tag - assignment
- Task – Add two python Operator tasks as shown below



Use simplehttpoperator to hit an endpoint

Use connection and endpoint name from a variable so that you can try with different values

Assignment 8

Full Pipeline

Assignment

Create a Data pipeline

- Dag_id – my_datapipeline
- Tags – “assignments”
- Set the schedule interval to hourly
- Add the start date as 1 day ago
- Set the catchup parameter as True
- Set the owner as ‘airflow’

Get Single User from below
API
https://reqres.in/api/users/2
response_filter=lambda response:
json.loads(response.text)
Pass the response as xcom variable

Fetch the xcom response
From last task,
Process Json to fetch users id,
email,first_name,last_name

Push id,email,first_name,last_name
as xcom variables

Create a customers tables

Pull Xcom from last task
to dynamically in insert statement