# Princess Sumaya University for Technology

# Data Engineering Course

# Assignment 1

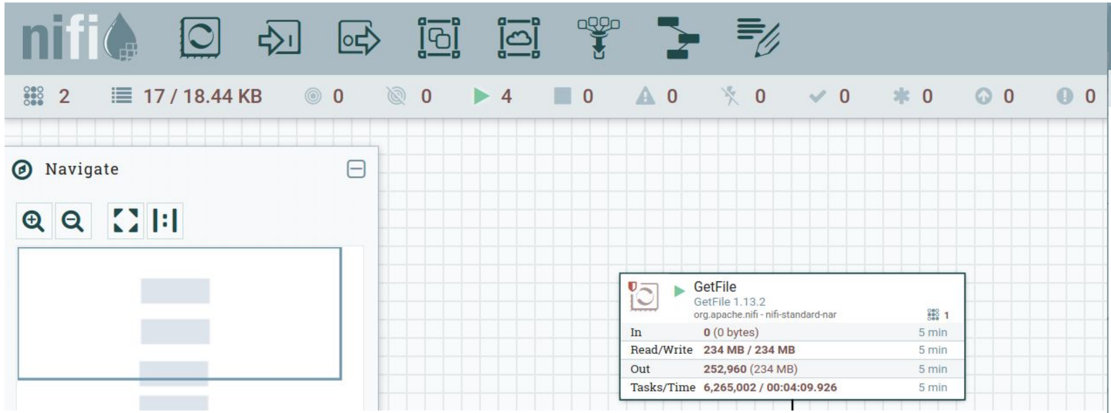
# Dr Ibrahim Abu Alhaol

By Waed Alsawarieh , 20208020

## Question 1

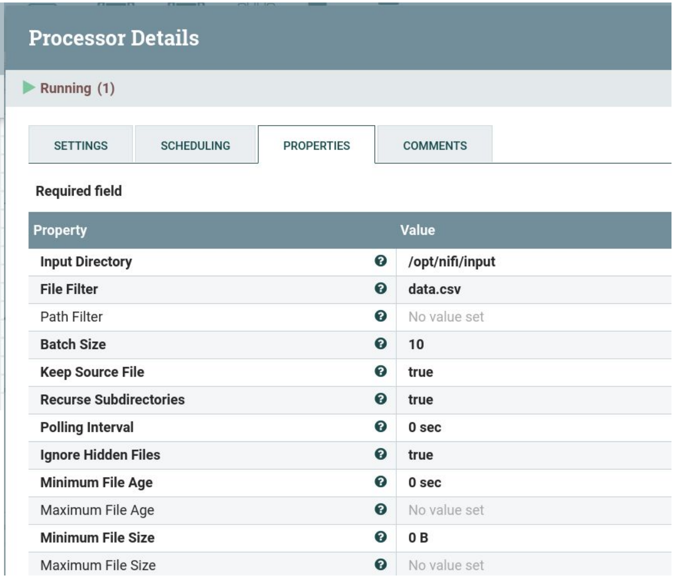
Q1: Provide similar to AirFlow implementation but using NiFi and provide the GitHub repo with all dependencies and detailed REAME.MD and PPT presentation on how to run your workflow.

# 1.1 Get Data CSV From Input Directory in NIFI Container



GetFile
GetFile 1.13.2
org.apache.nifi - nifi-standard-nar
1

| | | |
|---|---|---|
| In | 0 (0 bytes) | 5 min |
| Read/Write | 234 MB / 234 MB | 5 min |
| Out | 252,960 (234 MB) | 5 min |
| Tasks/Time | 6,265,002 / 00:04:09.926 | 5 min |



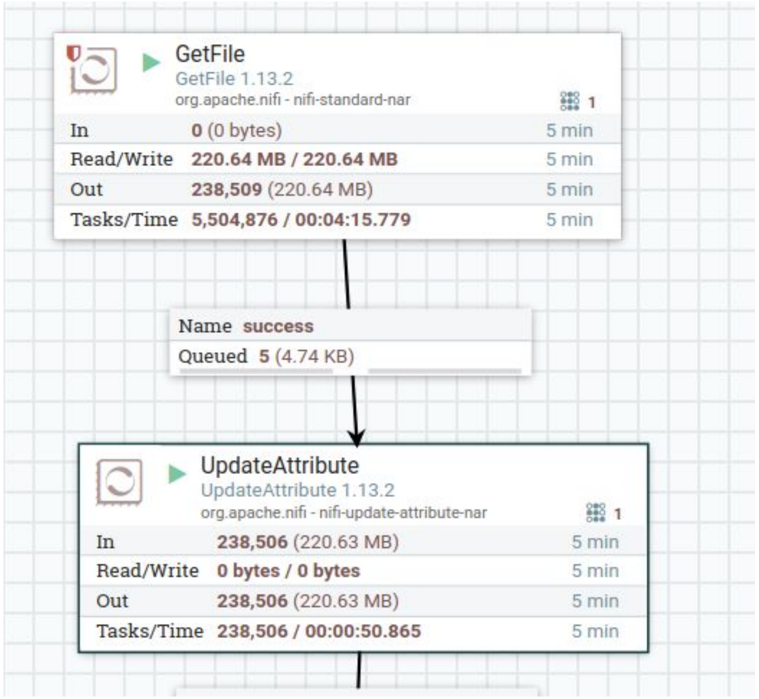
Processor Details

Running (1)

Required field

| Property | Value |
|---|---|
| Input Directory | /opt/nifi/input |
| File Filter | data.csv |
| Path Filter | No value set |
| Batch Size | 10 |
| Keep Source File | true |
| Recurse Subdirectories | true |
| Polling Interval | 0 sec |
| Ignore Hidden Files | true |
| Minimum File Age | 0 sec |
| Maximum File Age | No value set |
| Minimum File Size | 0 B |
| Maximum File Size | No value set |

# 1.2 UpdateAttribute Processor to update attribute from data.csv to data.json



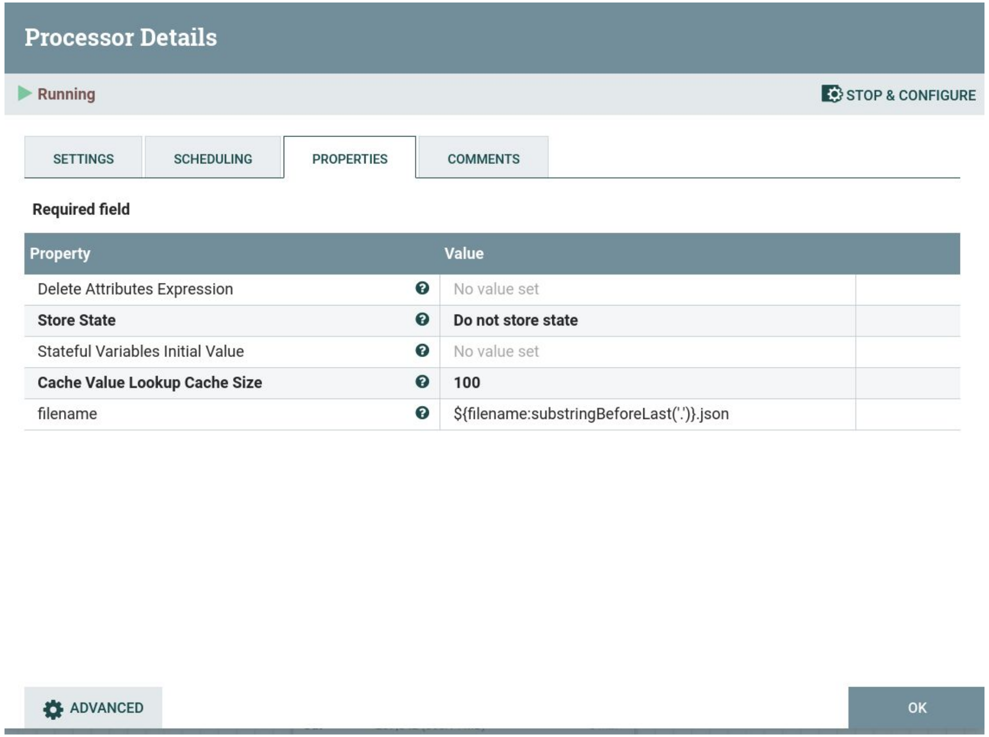
GetFile
GetFile 1.13.2
org.apache.nifi - nifi-standard-nar

| | | |
|---|---|---|
| In | 0 (0 bytes) | 5 min |
| Read/Write | 220.64 MB / 220.64 MB | 5 min |
| Out | 238,509 (220.64 MB) | 5 min |
| Tasks/Time | 5,504,876 / 00:04:15.779 | 5 min |

Name success
Queued 5 (4.74 KB)

UpdateAttribute
UpdateAttribute 1.13.2
org.apache.nifi - nifi-update-attribute-nar

| | | |
|---|---|---|
| In | 238,506 (220.63 MB) | 5 min |
| Read/Write | 0 bytes / 0 bytes | 5 min |
| Out | 238,506 (220.63 MB) | 5 min |
| Tasks/Time | 238,506 / 00:00:50.865 | 5 min |

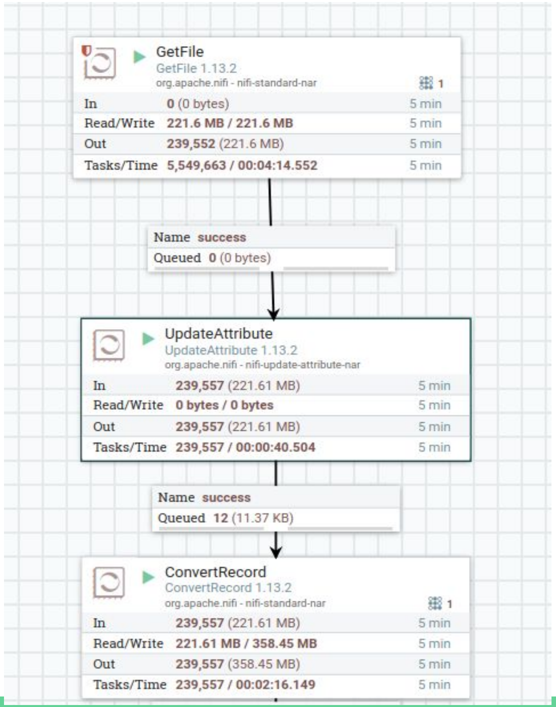


Required field

| Property | Value |
|---|---|
| Delete Attributes Expression | No value set |
| Store State | Do not store state |
| Stateful Variables Initial Value | No value set |
| Cache Value Lookup Cache Size | 100 |
| filename | ${filename:substringBeforeLast('.')}.json |

## 1.3 ConvertRecord Processor

Converts records from one data format to another using configured Record Reader and Record Write Controller Services.



GetFile
GetFile 1.13.2
org.apache.nifi - nifi-standard-nar
1

| | | |
|---|---|---|
| In | 0 (0 bytes) | 5 min |
| Read/Write | 221.6 MB / 221.6 MB | 5 min |
| Out | 239,552 (221.6 MB) | 5 min |
| Tasks/Time | 5,549,663 / 00:04:14.552 | 5 min |

Name success
Queued 0 (0 bytes)

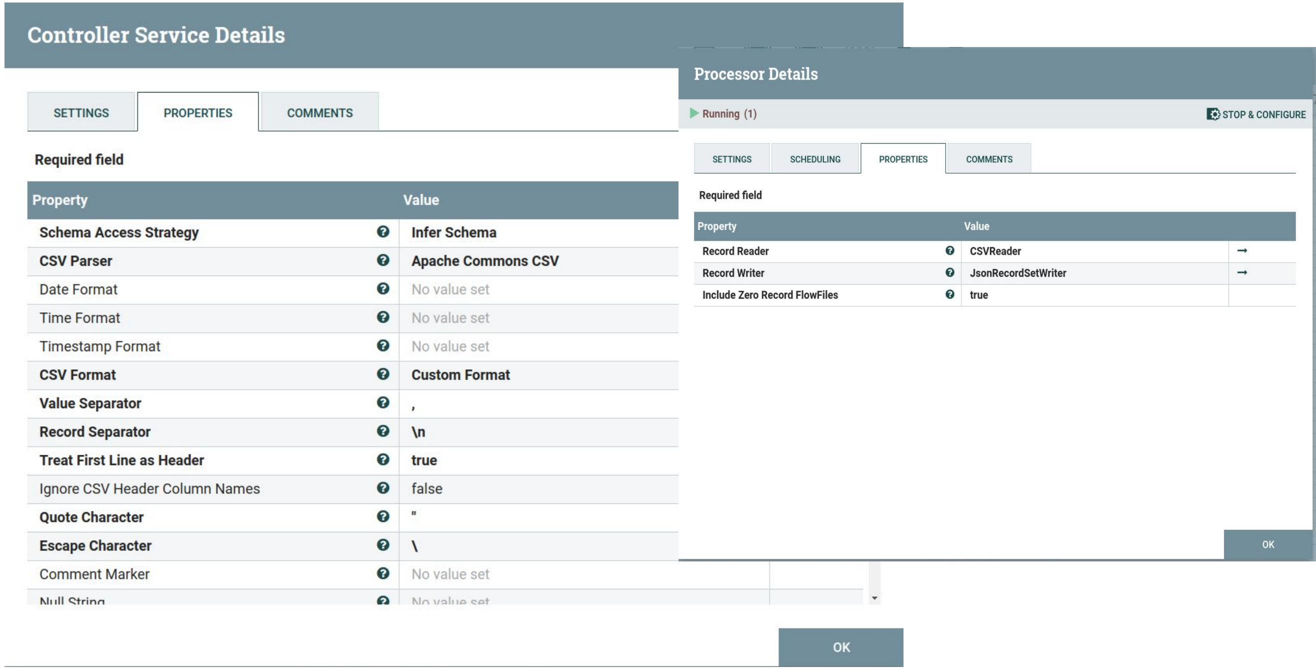
UpdateAttribute
UpdateAttribute 1.13.2
org.apache.nifi - nifi-update-attribute-nar

| | | |
|---|---|---|
| In | 239,557 (221.61 MB) | 5 min |
| Read/Write | 0 bytes / 0 bytes | 5 min |
| Out | 239,557 (221.61 MB) | 5 min |
| Tasks/Time | 239,557 / 00:00:40.504 | 5 min |

Name success
Queued 12 (11.37 KB)

ConvertRecord
ConvertRecord 1.13.2
org.apache.nifi - nifi-standard-nar
1

| | | |
|---|---|---|
| In | 239,557 (221.61 MB) | 5 min |
| Read/Write | 221.61 MB / 358.45 MB | 5 min |
| Out | 239,557 (358.45 MB) | 5 min |
| Tasks/Time | 239,557 / 00:02:16.149 | 5 min |



Controller Service Details

Required field

| Property | Value |
|---|---|
| Schema Access Strategy | Infer Schema |
| CSV Parser | Apache Commons CSV |
| Date Format | No value set |
| Time Format | No value set |
| Timestamp Format | No value set |
| CSV Format | Custom Format |
| Value Separator | , |
| Record Separator | \n |
| Treat First Line as Header | true |
| Ignore CSV Header Column Names | false |
| Quote Character | " |
| Escape Character | \ |
| Comment Marker | No value set |
| Null String | No value set |

Processor Details

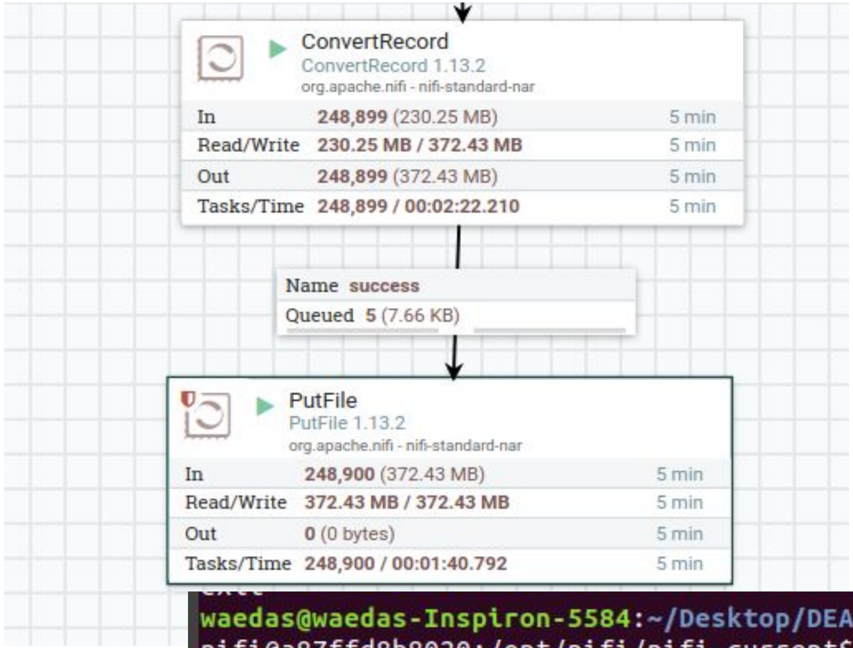
Running (1)

Required field

| Property | Value |
|---|---|
| Record Reader | CSVReader |
| Record Writer | JsonRecordSetWriter |
| Include Zero Record FlowFiles | true |

## 1.4 PutFile Processor

Writes the contents of a FlowFile to the local file system



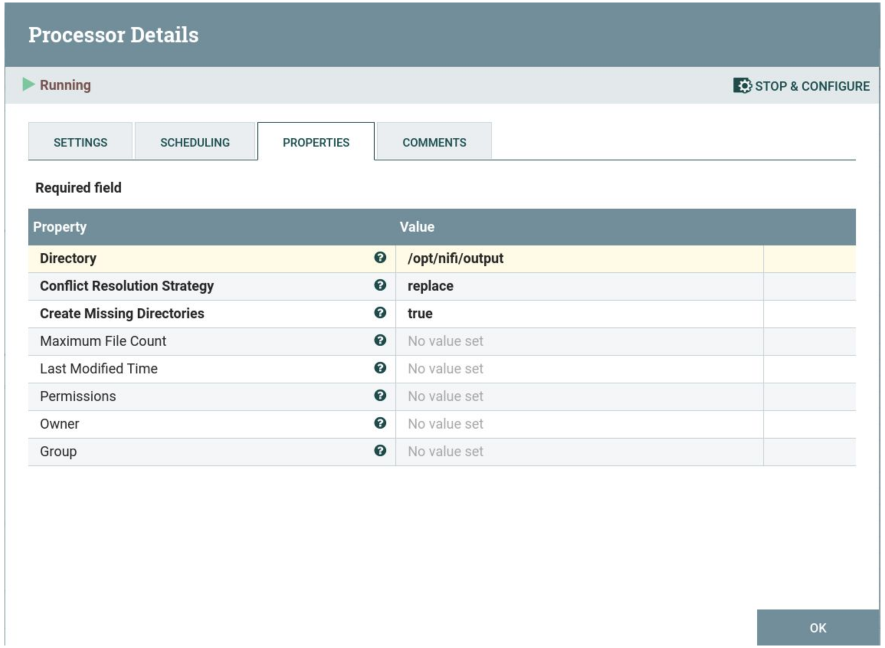
ConvertRecord
ConvertRecord 1.13.2
org.apache.nifi - nifi-standard-nar

| | | |
|---|---|---|
| In | 248,899 (230.25 MB) | 5 min |
| Read/Write | 230.25 MB / 372.43 MB | 5 min |
| Out | 248,899 (372.43 MB) | 5 min |
| Tasks/Time | 248,899 / 00:02:22.210 | 5 min |

Name success
Queued 5 (7.66 KB)

PutFile
PutFile 1.13.2
org.apache.nifi - nifi-standard-nar

| | | |
|---|---|---|
| In | 248,900 (372.43 MB) | 5 min |
| Read/Write | 372.43 MB / 372.43 MB | 5 min |
| Out | 0 (0 bytes) | 5 min |
| Tasks/Time | 248,900 / 00:01:40.792 | 5 min |

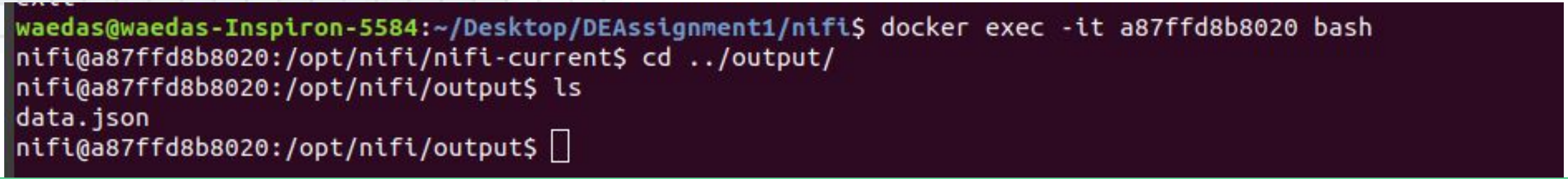


Processor Details

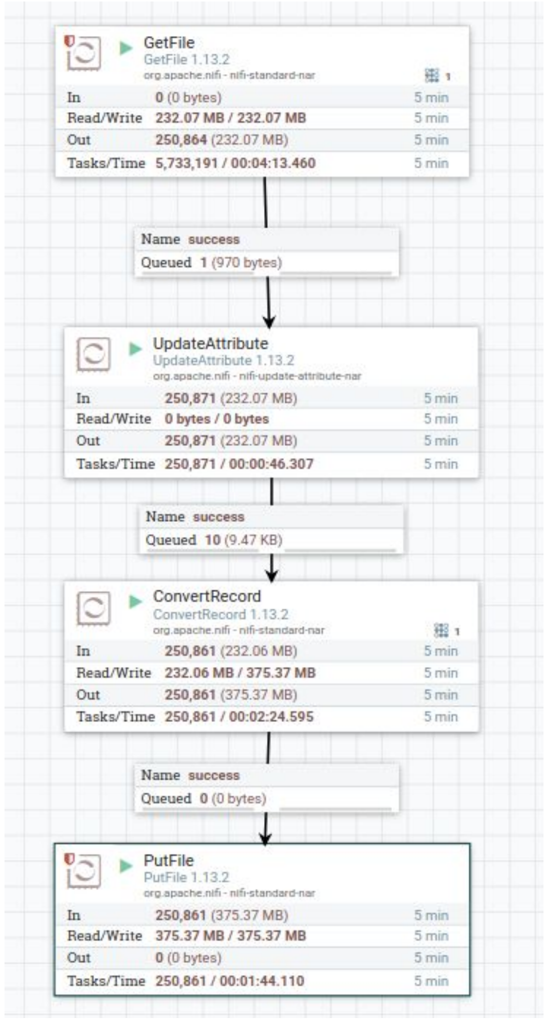
Running

Required field

| Property | Value |
|---|---|
| Directory | /opt/nifi/output |
| Conflict Resolution Strategy | replace |
| Create Missing Directories | true |
| Maximum File Count | No value set |
| Last Modified Time | No value set |
| Permissions | No value set |
| Owner | No value set |
| Group | No value set |

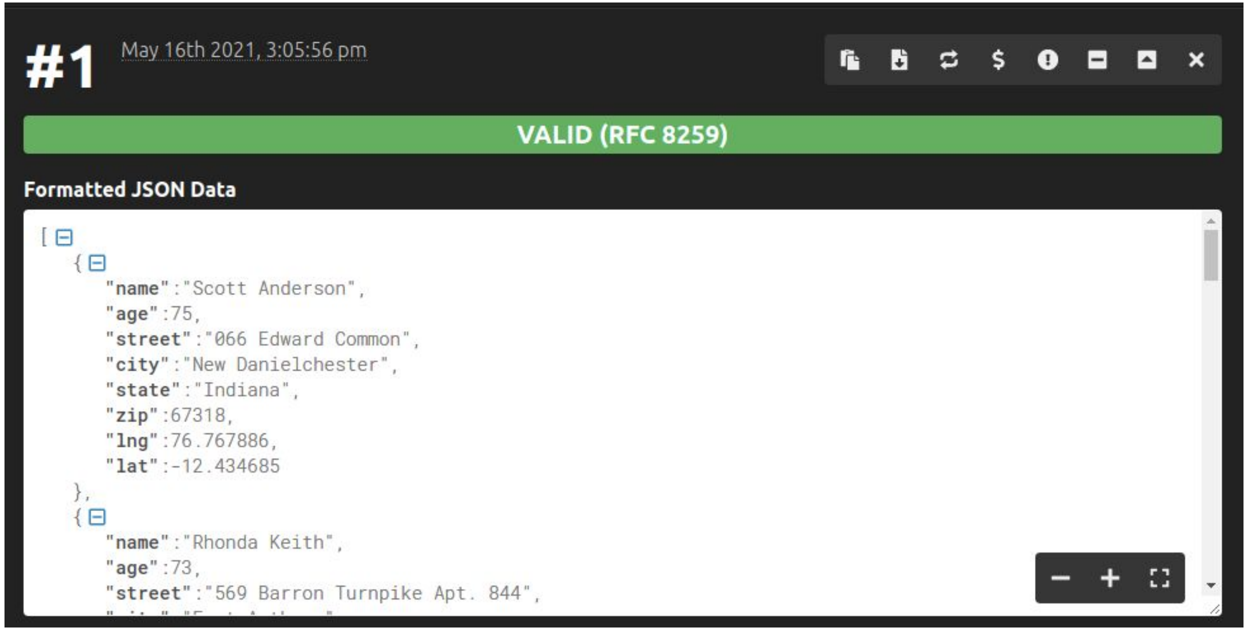


```
waedas@waedas-Inspiron-5584:~/Desktop/DEAssignment1/nifi$ docker exec -it a87ffd8b8020 bash
nifi@a87ffd8b8020:/opt/nifi/nifi-current$ cd ../output/
nifi@a87ffd8b8020:/opt/nifi/output$ ls
data.json
nifi@a87ffd8b8020:/opt/nifi/output$ 
```



```
GetFile
GetFile 1.13.2
org.apache.nifi - nifi-standard-nar                  1
In          0 (0 bytes)                         5 min
Read/Write  232.07 MB / 232.07 MB               5 min
Out         250,864 (232.07 MB)                 5 min
Tasks/Time  5,733,191 / 00:04:13.460            5 min

Name    success
Queued  1 (970 bytes)

UpdateAttribute
UpdateAttribute 1.13.2
org.apache.nifi - nifi-update-attribute-nar
In          250,871 (232.07 MB)                 5 min
Read/Write  0 bytes / 0 bytes                   5 min
Out         250,871 (232.07 MB)                 5 min
Tasks/Time  250,871 / 00:00:46.307              5 min

Name    success
Queued  10 (9.47 KB)

ConvertRecord
ConvertRecord 1.13.2
org.apache.nifi - nifi-standard-nar                  1
In          250,861 (232.06 MB)                 5 min
Read/Write  232.06 MB / 375.37 MB               5 min
Out         250,861 (375.37 MB)                 5 min
Tasks/Time  250,861 / 00:02:24.595              5 min

Name    success
Queued  0 (0 bytes)

PutFile
PutFile 1.13.2
org.apache.nifi - nifi-standard-nar
In          250,861 (375.37 MB)                 5 min
Read/Write  375.37 MB / 375.37 MB               5 min
Out         0 (0 bytes)                         5 min
Tasks/Time  250,861 / 00:01:44.110              5 min
```

## Validate data.json



**#1** May 16th 2021, 3:05:56 pm

**VALID (RFC 8259)**

**Formatted JSON Data**

```
[
   {
      "name":"Scott Anderson",
      "age":75,
      "street":"066 Edward Common",
      "city":"New Danielchester",
      "state":"Indiana",
      "zip":67318,
      "lng":76.767886,
      "lat":-12.434685
   },
   {
      "name":"Rhonda Keith",
      "age":73,
      "street":"569 Barron Turnpike Apt. 844",
```

## Question 2

Q2: Provide Similar to Airflow implementation but with csv file is extracted from Postgresql table and the produced json file is pushed to MongoDB database. Provide Github repo with all dependencies and detailed REAME.MD and PPT presentation how to run your workflow.

## Docker Compose File- Services

1. **Apache Airflow**

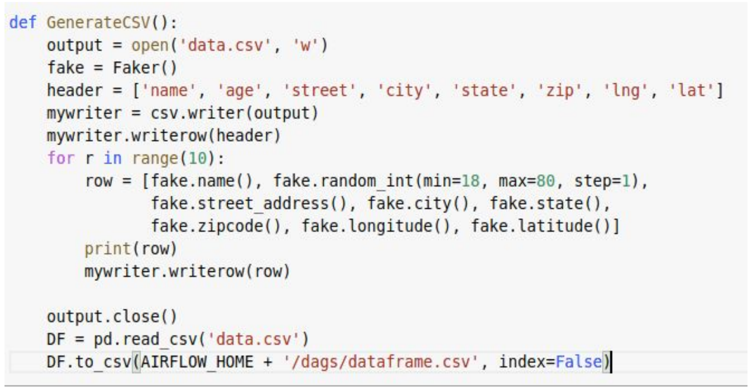
2. **Postgresql**

3. **pgAdmin**

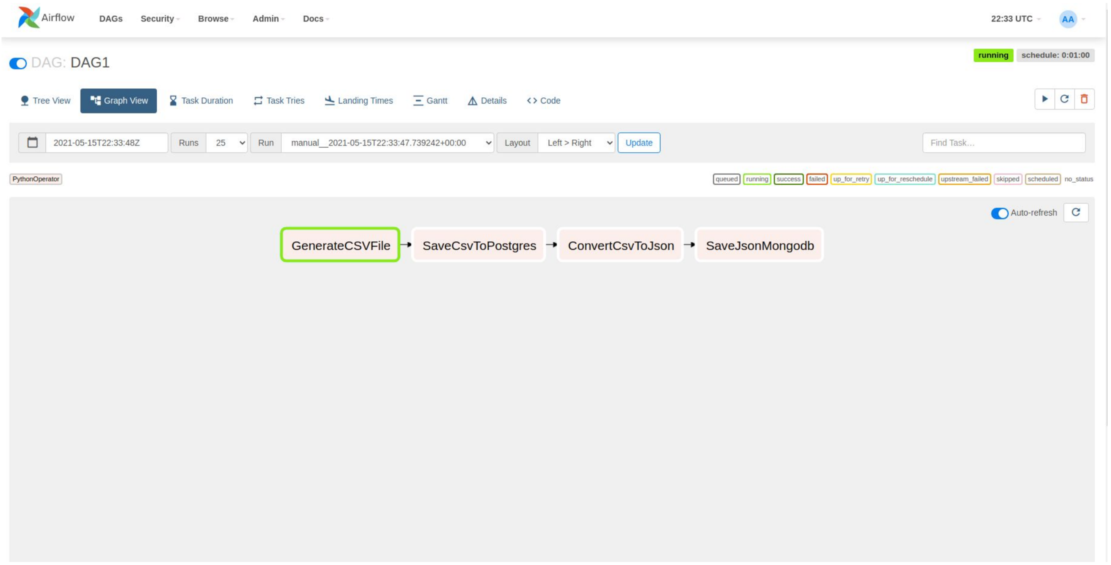
4. **mongo-express**: web-based **MongoDB** admin interface.

5. **mongo** : MongoDB document database.

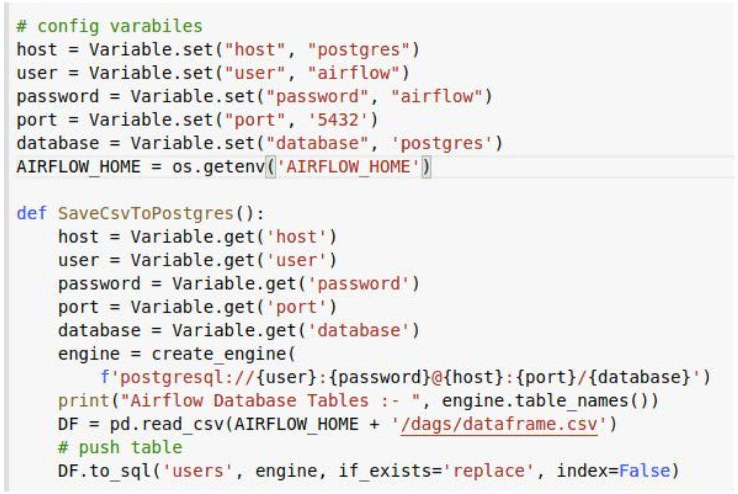
# 2.1 Generate CSV file from faker



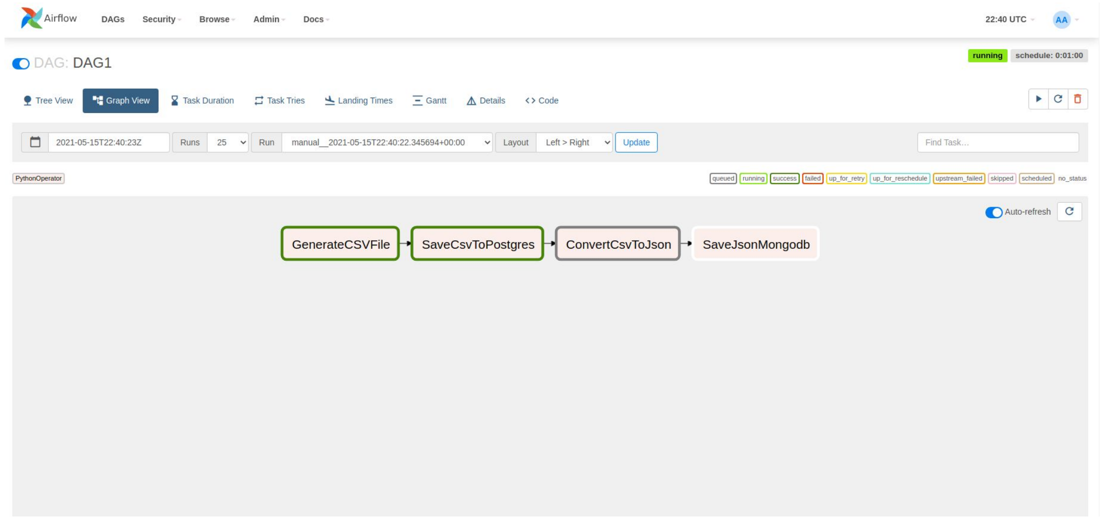
```python
def GenerateCSV():
    output = open('data.csv', 'w')
    fake = Faker()
    header = ['name', 'age', 'street', 'city', 'state', 'zip', 'lng', 'lat']
    mywriter = csv.writer(output)
    mywriter.writerow(header)
    for r in range(10):
        row = [fake.name(), fake.random_int(min=18, max=80, step=1),
               fake.street_address(), fake.city(), fake.state(),
               fake.zipcode(), fake.longitude(), fake.latitude()]
        print(row)
        mywriter.writerow(row)

    output.close()
    DF = pd.read_csv('data.csv')
    DF.to_csv(AIRFLOW_HOME + '/dags/dataframe.csv', index=False)
```



## 2.2 Save CSV in Postgresql Database



```
# config varabiles
host = Variable.set("host", "postgres")
user = Variable.set("user", "airflow")
password = Variable.set("password", "airflow")
port = Variable.set("port", '5432')
database = Variable.set("database", 'postgres')
AIRFLOW_HOME = os.getenv('AIRFLOW_HOME')

def SaveCsvToPostgres():
    host = Variable.get('host')
    user = Variable.get('user')
    password = Variable.get('password')
    port = Variable.get('port')
    database = Variable.get('database')
    engine = create_engine(
        f'postgresql://{user}:{password}@{host}:{port}/{database}')
    print("Airflow Database Tables :- ", engine.table_names())
    DF = pd.read_csv(AIRFLOW_HOME + '/dags/dataframe.csv')
    # push table
    DF.to_sql('users', engine, if_exists='replace', index=False)
```



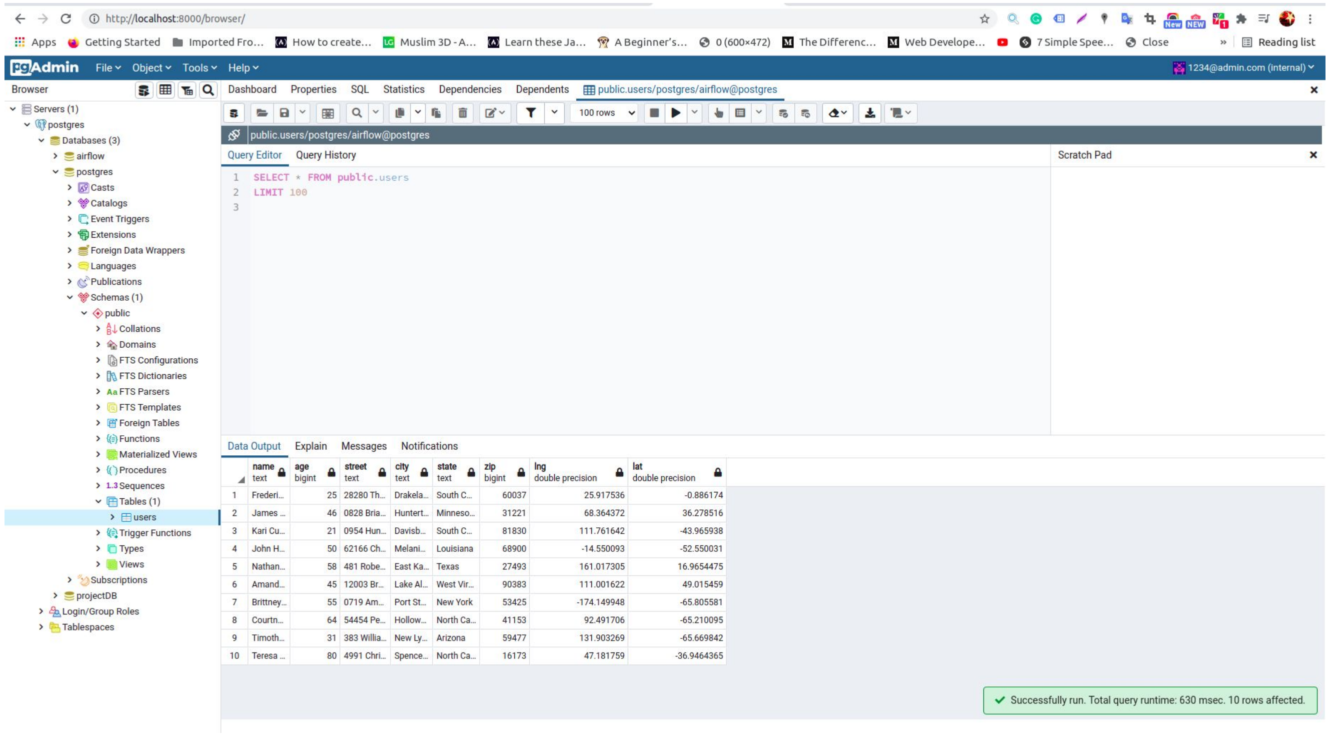
DAG: DAG1

running  schedule: 0:01:00

2021-05-15T22:40:23Z  Runs 25  Run manual__2021-05-15T22:40:22.345694+00:00  Layout Left > Right

GenerateCSVFile → SaveCsvToPostgres → ConvertCsvToJson → SaveJsonMongodb

# 2.3 Save CSV in Postgresql Database

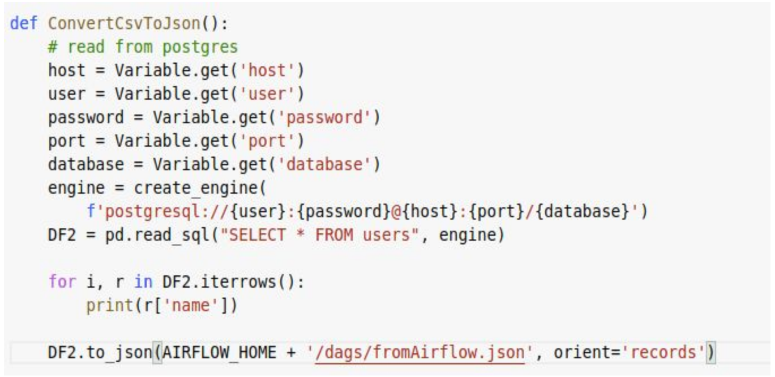


```
SELECT * FROM public.users
LIMIT 100
```

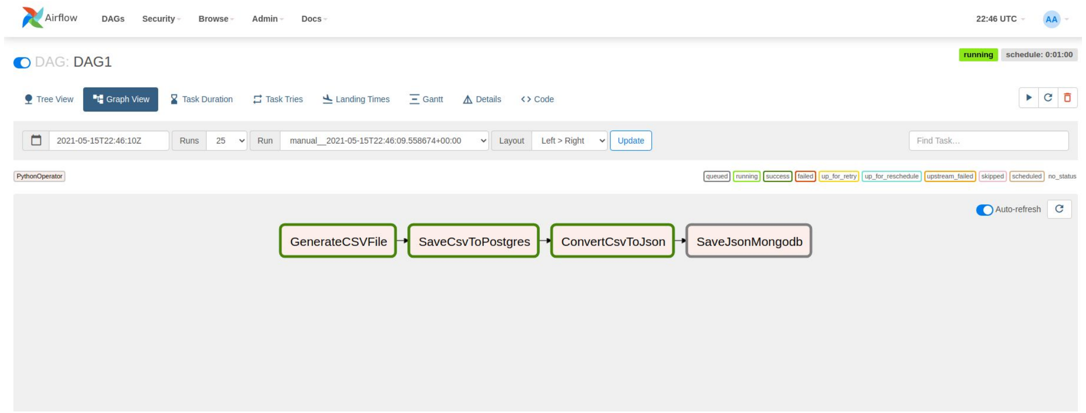
|  | name<br>text | age<br>bigint | street<br>text | city<br>text | state<br>text | zip<br>bigint | lng<br>double precision | lat<br>double precision |
|---|---|---|---|---|---|---|---|---|
| 1 | Frederi... | 25 | 28280 Th... | Drakela... | South C... | 60037 | 25.917536 | -0.886174 |
| 2 | James ... | 46 | 0828 Bria... | Huntert... | Minneso... | 31221 | 68.364372 | 36.278516 |
| 3 | Kari Cu... | 21 | 0954 Hun... | Davisb... | South C... | 81830 | 111.761642 | -43.965938 |
| 4 | John H... | 50 | 62166 Ch... | Melani... | Louisiana | 68900 | -14.550093 | -52.550031 |
| 5 | Nathan... | 58 | 481 Robe... | East Ka... | Texas | 27493 | 161.017305 | 16.9654475 |
| 6 | Amand... | 45 | 12003 Br... | Lake Al... | West Vir... | 90383 | 111.001622 | 49.015459 |
| 7 | Brittney... | 55 | 0719 Am... | Port St... | New York | 53425 | -174.149948 | -65.805581 |
| 8 | Courtn... | 64 | 54454 Pe... | Hollow... | North Ca... | 41153 | 92.491706 | -65.210095 |
| 9 | Timoth... | 31 | 383 Willia... | New Ly... | Arizona | 59477 | 131.903269 | -65.669842 |
| 10 | Teresa ... | 80 | 4991 Chri... | Spence... | North Ca... | 16173 | 47.181759 | -36.9464365 |

✓ Successfully run. Total query runtime: 630 msec. 10 rows affected.

## 2.4 Convert To JSON



```python
def ConvertCsvToJson():
    # read from postgres
    host = Variable.get('host')
    user = Variable.get('user')
    password = Variable.get('password')
    port = Variable.get('port')
    database = Variable.get('database')
    engine = create_engine(
        f'postgresql://{user}:{password}@{host}:{port}/{database}')
    DF2 = pd.read_sql("SELECT * FROM users", engine)

    for i, r in DF2.iterrows():
        print(r['name'])

    DF2.to_json(AIRFLOW_HOME + '/dags/fromAirflow.json', orient='records')
```



Airflow DAGs Security Browse Admin Docs 22:46 UTC AA

DAG: DAG1 running schedule: 0:01:00

Tree View | Graph View | Task Duration | Task Tries | Landing Times | Gantt | Details | Code

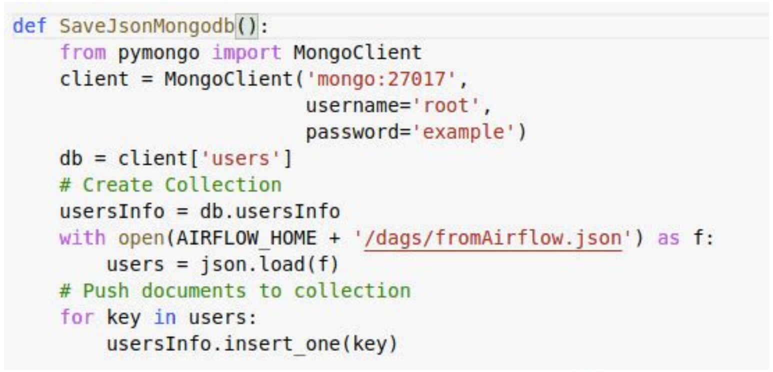
2021-05-15T22:46:10Z Runs 25 Run manual__2021-05-15T22:46:09.558674+00:00 Layout Left > Right Update Find Task...

PythonOperator queued running success failed up_for_retry up_for_reschedule upstream_failed skipped scheduled no_status

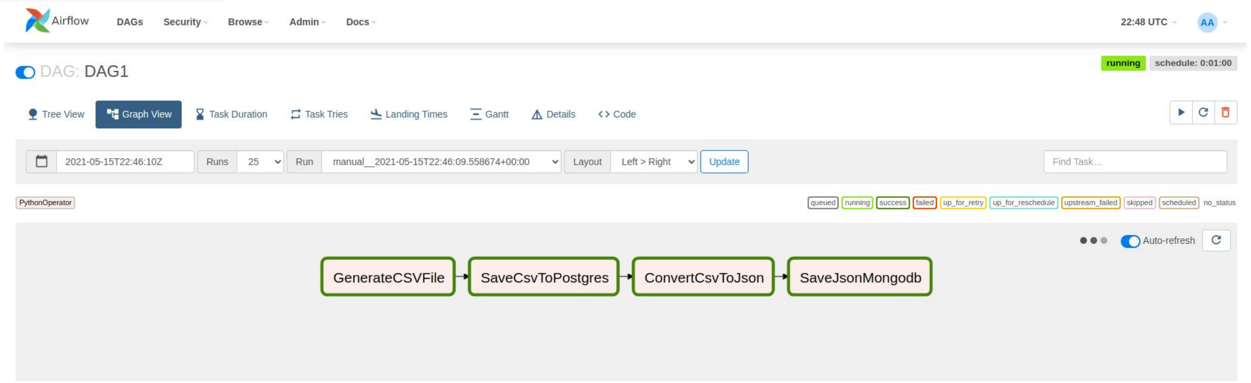
Auto-refresh

GenerateCSVFile → SaveCsvToPostgres → ConvertCsvToJson → SaveJsonMongodb

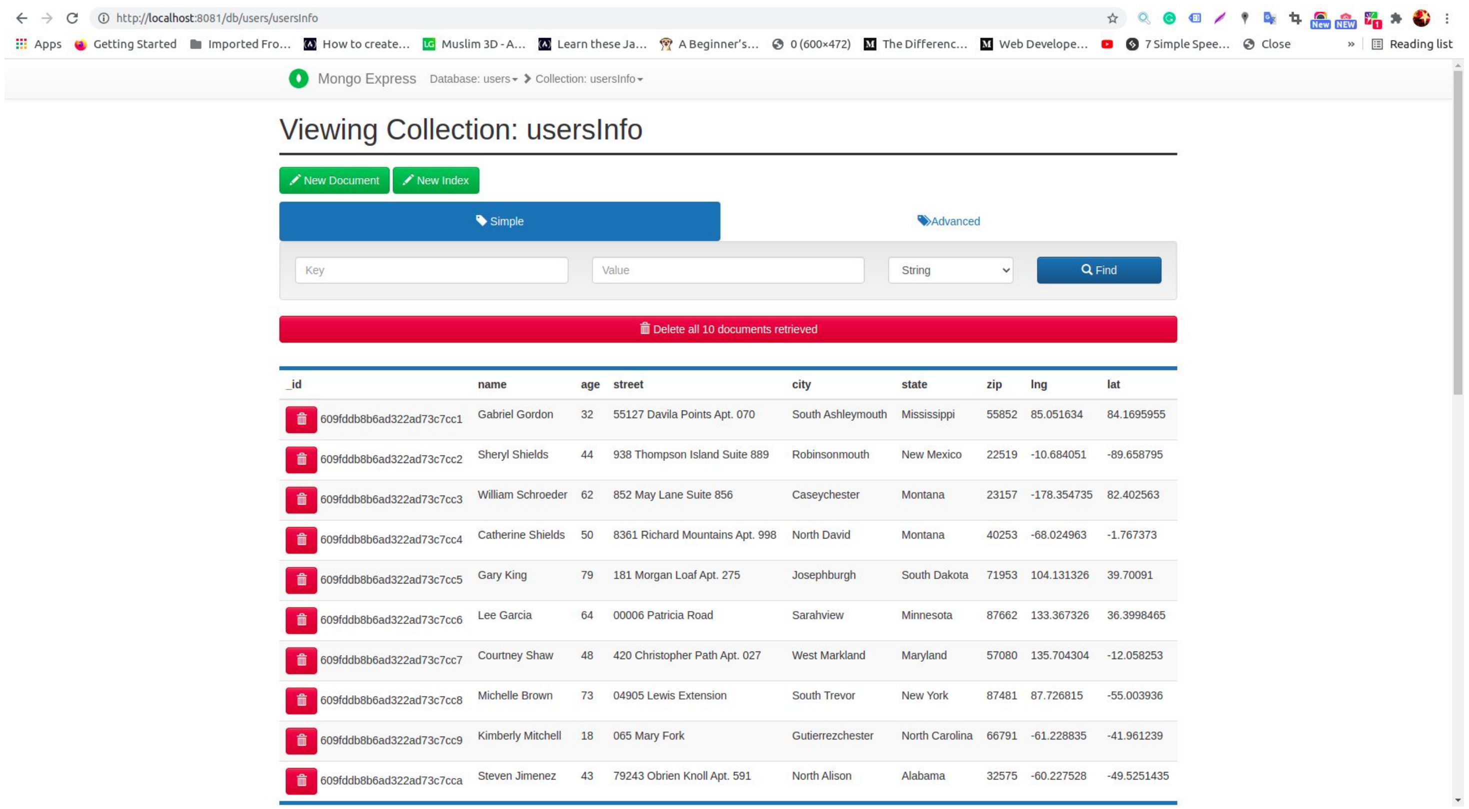
## 2.4 Save JSON file in MongoDB



```python
def SaveJsonMongodb():
    from pymongo import MongoClient
    client = MongoClient('mongo:27017',
                         username='root',
                         password='example')
    db = client['users']
    # Create Collection
    usersInfo = db.usersInfo
    with open(AIRFLOW_HOME + '/dags/fromAirflow.json') as f:
        users = json.load(f)
    # Push documents to collection
    for key in users:
        usersInfo.insert_one(key)
```



# 2.4 Save Json file in MongoDB



Viewing Collection: usersInfo

Delete all 10 documents retrieved

| _id | name | age | street | city | state | zip | lng | lat |
|---|---|---|---|---|---|---|---|---|
| 609fddb8b6ad322ad73c7cc1 | Gabriel Gordon | 32 | 55127 Davila Points Apt. 070 | South Ashleymouth | Mississippi | 55852 | 85.051634 | 84.1695955 |
| 609fddb8b6ad322ad73c7cc2 | Sheryl Shields | 44 | 938 Thompson Island Suite 889 | Robinsonmouth | New Mexico | 22519 | -10.684051 | -89.658795 |
| 609fddb8b6ad322ad73c7cc3 | William Schroeder | 62 | 852 May Lane Suite 856 | Caseychester | Montana | 23157 | -178.354735 | 82.402563 |
| 609fddb8b6ad322ad73c7cc4 | Catherine Shields | 50 | 8361 Richard Mountains Apt. 998 | North David | Montana | 40253 | -68.024963 | -1.767373 |
| 609fddb8b6ad322ad73c7cc5 | Gary King | 79 | 181 Morgan Loaf Apt. 275 | Josephburgh | South Dakota | 71953 | 104.131326 | 39.70091 |
| 609fddb8b6ad322ad73c7cc6 | Lee Garcia | 64 | 00006 Patricia Road | Sarahview | Minnesota | 87662 | 133.367326 | 36.3998465 |
| 609fddb8b6ad322ad73c7cc7 | Courtney Shaw | 48 | 420 Christopher Path Apt. 027 | West Markland | Maryland | 57080 | 135.704304 | -12.058253 |
| 609fddb8b6ad322ad73c7cc8 | Michelle Brown | 73 | 04905 Lewis Extension | South Trevor | New York | 87481 | 87.726815 | -55.003936 |
| 609fddb8b6ad322ad73c7cc9 | Kimberly Mitchell | 18 | 065 Mary Fork | Gutierrezchester | North Carolina | 66791 | -61.228835 | -41.961239 |
| 609fddb8b6ad322ad73c7cca | Steven Jimenez | 43 | 79243 Obrien Knoll Apt. 591 | North Alison | Alabama | 32575 | -60.227528 | -49.5251435 |

## Research

Explain all of the research you’ve done about this issue/challenge.

What was the goal of your research? Be sure to explain how you found it and anyone who might have helped you!