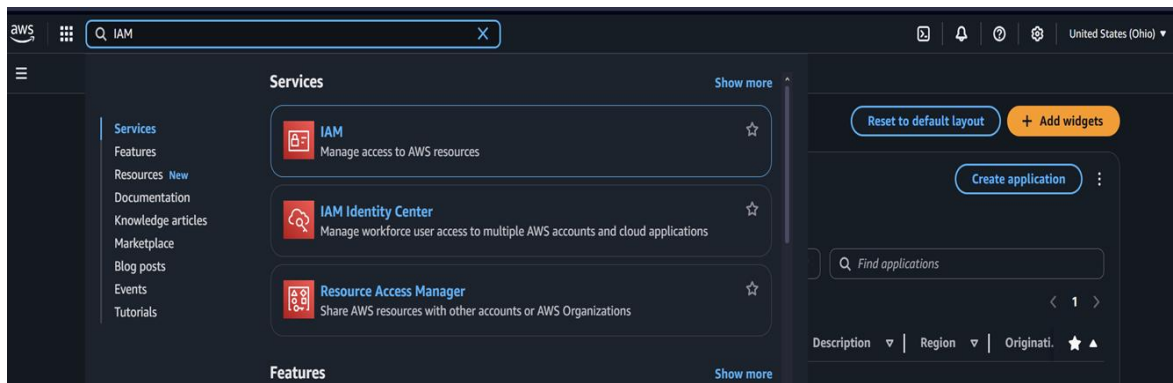


## PROJECT – END TO END ML BUSINESS PIPELINE

Create an AWS Account and set up IAM Roles.

Go to <https://aws.amazon.com> and sign up.

### Set up IAM Roles



The IAM Roles helps in granting permission to entities that interact with AWS resources. Before creating the roles, a user profile must be created, with access key and secret access key. Policies are created for each user or group.

Install AWS-CLI on windows or Linux to interact with AWS services using commands in your command-line shell. This can be done using: <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

After downloading, open the CMD or PowerShell:

```
aws onfigure
```

and enter the access key ID, secret access key, region and output format.

S3 bucket was created using

```
Aws s3 mb s3://folder-name
```

Extra folders were created: raw, scripts, temp, production, transformation, athena-results.

Some Python packages are installed using CMD.

The python files ran in the CMD: ingest\_fmp.py and ingest\_mysql.py is ran and the data saved at their respective folders in the S3 bucket.

An IAM Role is created for Glue which is used to manage data integration. It is used to automate modern data pipelines with built in ETL processes.

The screenshot shows the 'Select trusted entity' step in the AWS IAM console. On the left, a sidebar lists the steps: Step 1 (selected), Step 2, Step 3, and Step 4. The main area is titled 'Select trusted entity' and contains two sections. The 'Trusted entity type' section has five radio buttons: 'AWS service' (selected), 'AWS account', 'Web identity', 'SAML 2.0 federation', and 'Custom trust policy'. The 'Use case' section has a dropdown menu for 'Service or use case' with 'Glue' selected, and a 'Use case' dropdown with 'Glue' selected. At the bottom right are 'Cancel' and 'Next' buttons.

The role is given AmazonS3FullAccess and AWSGlueServices permissions.

After that a Crawler and a database are created.

The screenshot shows the 'Set crawler properties' step in the AWS Glue console. On the left, a sidebar lists the steps: Step 1 (selected), Step 2, Step 3, Step 4, Step 5, and Step 6. The main area is titled 'Set crawler properties' and contains a 'Crawler details' section. This section has a 'Name' field with a placeholder 'Enter a unique crawler name' and a 'Description - optional' field with a placeholder 'Enter a description'. Below these is a 'Tags - optional' section with a placeholder 'Use tags to organize and identify your resources.' At the bottom right are 'Cancel' and 'Next' buttons.

The database can be created using the Crawler or with Athena

The screenshot shows the 'Set output and scheduling' step in the AWS Glue console. On the left, a sidebar lists the steps: Step 1, Step 2, Step 3, Step 4, Step 5 (selected), and Step 6. The main area is titled 'Set output and scheduling' and contains an 'Output configuration' section. This section has a 'Target database' dropdown menu with a placeholder 'Choose a database' and a 'Clear selection' button. Below this is a 'Table name prefix - optional' field with a placeholder 'Type a prefix added to table names'. At the bottom right are 'Cancel' and 'Next' buttons.

Clicking the add database goes straight to creating a database

The screenshot shows the AWS Glue console's 'Create a database' page. The left sidebar contains the AWS Glue navigation menu with categories like 'Getting started', 'Data Catalog', and 'Data Integration and ETL'. The main panel is titled 'Create a database' and includes a sub-header 'Create a database in the AWS Glue Data Catalog.' Below this, there are two main sections: 'Database details' and 'Database settings'. The 'Database details' section has a 'Name' field with a placeholder 'Enter a unique database name' and a note that the name must be unique, lowercase, and under 255 characters. It also has an optional 'Description' field with a placeholder 'Enter text' and a note that descriptions can be up to 2048 characters long. The 'Database settings' section has an optional 'Location' field with a placeholder 'Set the URI location for use by clients of the Data Catalog.' and a note that an S3 location is required for managed tables and Zero-ETL integrations. At the bottom right, there are 'Cancel' and 'Create database' buttons.

OR

The screenshot shows the Athena editor interface. At the top, there are tabs for 'Editor', 'Recent queries', and 'Saved queries'. The 'Editor' tab is active. Below the tabs, there's a 'Data' section with a refresh icon and a back arrow. Under 'Data source', 'AwsDataCatalog' is selected. Under 'Catalog', 'None' is selected. Under 'Database', 'financials\_db' is selected. Below these, there's a 'Tables and views' section with a 'Create' button and a settings icon. A search bar with the placeholder 'Filter tables and views' is present. Below the search bar, there's a list of tables under the heading 'Tables (78)'. The first three tables are: '001b13b7\_aae5\_4276\_88a6\_97d62e3\_a5924\_csv', '001b13b7\_aae5\_4276\_88a6\_97d62e3\_a5924\_csv\_metadata', and '019126f8\_da82\_4b61\_a65e\_e34ce42\_63406\_csv'.

In Athena, go to the Athena editor and type

Create DATABASE

Files: transform\_financial\_data.py, join\_financial\_summary.py and data\_wrangler.py are run via glue jobs which normalizes the raw fmp data saving it as a parquet file and joins transformed income statements of stock prices. The data\_wrangler preprocesses and enriches the income data by adding new KPIs. The data is saved as a csv file, which is used by PowerBI for visualization and for modelling using machine learning.

Connect to Power BI to Athena

Athena is connected to PowerBI using odbc driver. The KPIs created were used to create visualizations.

company	year	revenue_old	netincome_old	netprofitmargin	grossprofit	revenue	netincome	costandexpenses	costofgoodsandservices	netprofit
NFLX	2023	\$567.81	\$56.54	99.58	\$55,565,114.00	\$56,781,000	\$56,540,000	\$2,026,476.67	\$1,215,886.00	\$45,382,572.
TSLA	2023	\$305.68	\$78.39	256.44	\$30,127,208.50	\$30,568,000	\$78,390,000	\$734,652.50	\$440,791.50	\$24,941,191.
NVDA	2023	\$235.02	\$40.79	173.56	\$23,298,867.51	\$23,502,000	\$40,790,000	\$338,554.14	\$203,132.49	\$19,491,129.
AMZN	2023	\$318.72	\$76.35	239.55	\$31,468,709.19	\$31,872,000	\$76,350,000	\$672,151.36	\$403,290.81	\$26,123,766.
NVDA	2023	\$296.5	\$16.1	54.30	\$29,468,966.11	\$29,650,000	\$16,100,000	\$301,723.16	\$181,033.89	\$24,749,512.
NVDA	2023	\$292.26	\$28.08	96.08	\$28,867,250.08	\$29,226,000	\$28,080,000	\$597,916.54	\$358,749.92	\$24,018,215.
GOOG	2023	\$320.46	\$62.25	194.25	\$31,172,338.91	\$32,046,000	\$62,250,000	\$1,456,101.82	\$873,661.09	\$25,204,114.
NFLX	2023	\$429.15	\$71.87	167.47	\$42,664,467.57	\$42,915,000	\$71,870,000	\$417,554.05	\$250,532.43	\$35,905,503.
TSLA	2023	\$118.39	\$26.3	222.15	\$11,529,260.14	\$11,839,000	\$26,300,000	\$516,233.10	\$309,739.86	\$9,344,164.
BABA	2023	\$553.84	\$52.82	95.37	\$55,125,859.59	\$55,384,000	\$52,820,000	\$430,234.01	\$258,140.41	\$46,407,515.
ORCL	2023	\$399.13	\$69.02	172.93	\$39,391,463.75	\$39,913,000	\$69,020,000	\$869,227.08	\$521,536.25	\$32,684,258.
GOOG	2023	\$241.38	\$32.18	133.32	\$23,563,472.09	\$24,138,000	\$32,180,000	\$957,546.52	\$574,527.91	\$19,211,975.
TSLA	2023	\$497.46	\$93.29	187.53	\$49,156,901.26	\$49,746,000	\$93,290,000	\$981,831.23	\$589,098.74	\$40,867,999.
NVDA	2023	\$467.99	\$47.22	100.90	\$45,448,044.56	\$46,799,000	\$47,220,000	\$2,251,592.41	\$1,350,955.44	\$36,638,070.
GOOG	2023	\$570.88	\$120.99	211.94	\$56,427,075.26	\$57,088,000	\$120,990,000	\$1,101,541.23	\$660,924.74	\$46,999,308.
MSFT	2023	\$562.96	\$60.22	122.96	\$55,080,878.21	\$56,206,000	\$60,220,000	\$2,025,202.89	\$1,215,121.79	\$44,973,705.

See the PowerBI Business Report file.

Airflow and Automation

An environment was created for the airflow and the ran on the webserver

### ■ Installing Apache Airflow on Windows with WSL

- 1: Activate Ubuntu or Other Linux System
- 2: Create a Virtual Environment and Activate It

```
#Create virtual environment
python3 -m venv airflow-env
#Activate the virtual environment
source airflow-env/bin/activate
```
- 3: Set the \$AIRFLOW\_HOME Parameter

```
nano ~/.bashrc
#Type the following
AIRFLOW_HOME=/c/Users/sriw/airflow
#Press Ctrl+S and Ctrl+X to exit the editor
```
- 4: Install Apache Airflow

```
pip install apache-airflow
```
- 5: Initialize the Database

```
airflow db init
```
- 6: Create an Admin User

```
airflow users create \
--username admin \
--password admin \
--firstname <YourFirstName> \
--lastname <YourLastName> \
--role Admin \
--email admin@example.com
```
- 7: Run the Web Server and Scheduler

```
airflow webserver --port 8080 -D
airflow scheduler -D
```

The Airflow web server will be accessible at  
http://localhost:8080 in your web browser and log in using  
the above-created User (admin & admin)

A folder called dag is created, which has files: api\_ingestion\_dag.py, financial\_pipeline\_dag.py

The dag files contain specific tasks that must be scheduled.

## ML Training

The data saved as a csv file is modelled using a machine learning technique. AWS Sagemaker was used for this.

```
1 from sagemaker.sklearn.estimator import SKLearn
2
3 sklearn_est = SKLearn(
4     entry_point='ML_model_script.py',
5     role='arn:aws:iam::627206123828:role/service-role/AmazonSageMaker-ExecutionRole-20250809T175141',
6     instance_type='ml.m5.xlarge',
7     instance_count=1,
8     framework_version='1.0-1',
9     base_job_name='Financial-ml-model',
10    py_version='py3',
11    hyperparameters={
12        'n_estimators': 100,
13        'max_depth': 10,
14        'min_samples_split': 2,
15        'min_samples_leaf': 1,
16        'random_state': 42
17    },
18    use_spot_instances=True,
19    max_wait = 7200,
20    max_run = 3600,
21 )
```