



University of New Haven

TAGLIATELA COLLEGE OF ENGINEERING

Electrical & Computer Engineering and Computer Science

## DATA SCIENCE

# TECHNICAL REPORT



2<sup>nd</sup> SEMESTER

# CONTENTS

**Project Name .....2**

**Executive Summary .....2**

**Technical Report .....3**

**Highlights of Project .....3**

**Submitted on: .....3**

**Abstract .....4**

**Methodology .....5**

**Data Pipeline .....7**

**Execution .....8**

**Results Section .....13**

**Conclusion .....21**

**Contributions/References .....22**

## Executive Summary



### Team Members:

- SAI TEJA THOTA
- VEDANTH REDDY DODDANNAGARI
- MANICHANDRA DOMALA
- SIRI H G

### Highlights of Project

- Uploading data to EC2 Instance
- Transferring data from EC2 to S3 using Boto3
- Create Database using AWS Glue
- Set up AWS Crawler to crawl S3 bucket
- Perform transformation using Athena
- Connect the Data to Power BI using Open Database Connectivity (ODBC)
- Visualize using Power BI



## ABSTRACT

The project's main goal is to integrate Power BI and Amazon Web Services (AWS) to streamline and optimize data administration and visualization procedures. Data is first uploaded to an Amazon Simple Storage Service (S3) bucket, then moved to an Elastic Compute Cloud (EC2) instance. A database is made using AWS Glue to arrange and structure the data. Next, to make sure the data is current, an AWS Crawler is configured to automatically crawl the S3 bucket. Athena effectively handles data transformation, allowing for smooth querying and analysis. Lastly, Open Database Connectivity (ODBC) is used to connect the data to Power BI, enabling intelligent analysis and visualization. This project shows how cloud-based services may be integrated to improve data management, transformation, and visualization capabilities, facilitating actionable insights and decision-making.

## CRISP-DM METHODOLOGY

The CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology is a widely used framework for guiding data mining and analytics projects.

CRISP-DM phases:

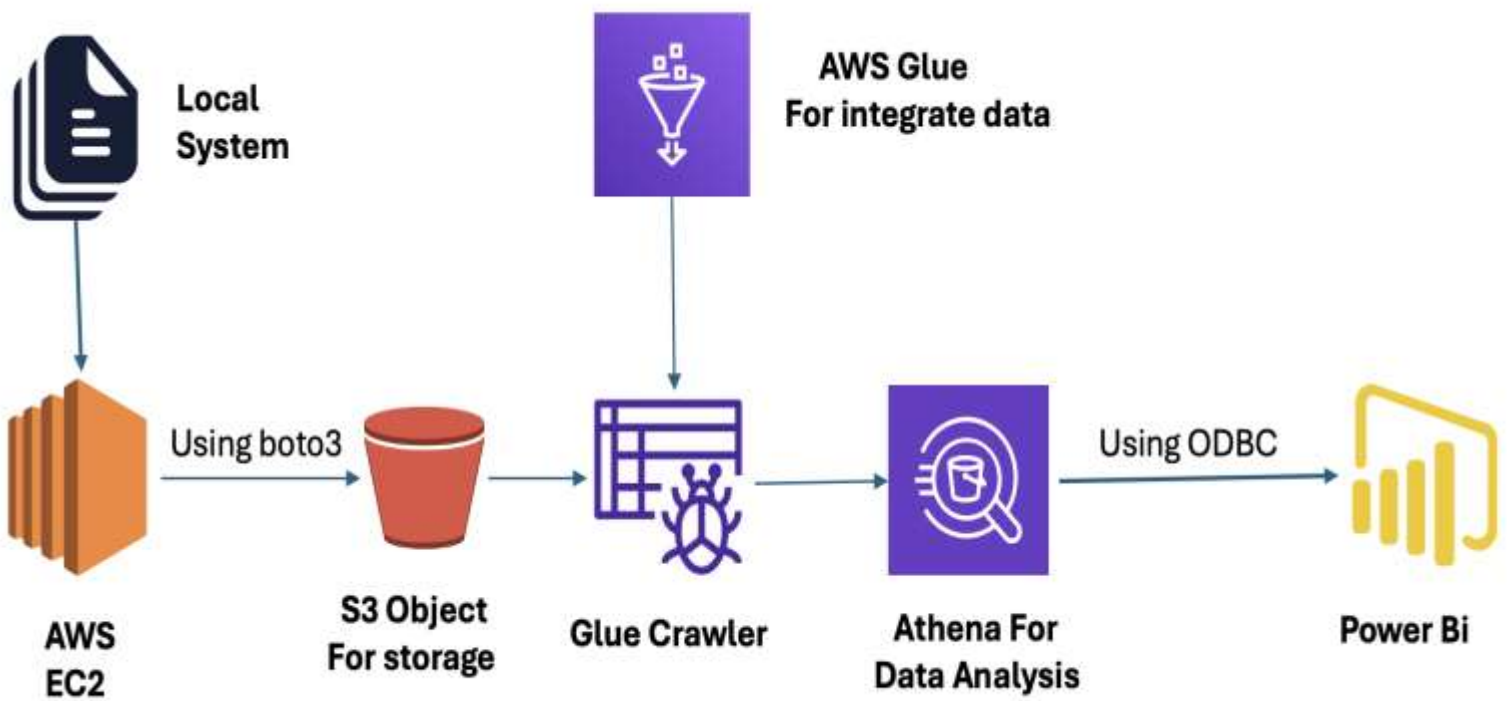
- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation

1. **Business Understanding:** Understanding the project's goals and requirements from a business standpoint is the main goal of this first stage. It includes establishing the project's objectives, the business issue that must be resolved, and the metrics by which success will be judged.
2. **Understanding Data:** During this stage, pertinent project data is gathered, examined, and evaluated to obtain understanding of its composition, relationships, and quality. This entails gathering data, exploring data, and evaluating the preliminary quality of data.
3. **Data Preparation:** After the data has been comprehended, it must be ready for analysis. In this stage, the data must be cleaned, missing values must be handled, variables must be transformed, and derived variables must be created as needed. The objective is to produce a tidy, organized dataset suitable for modeling.

4. **Modeling**: To create descriptive or predictive models, a variety of modeling techniques are chosen and used to the prepared dataset during this step. This entails deciding on suitable algorithms, creating models, and utilizing validation methods to evaluate each model's performance.
5. **Evaluation**: To ascertain how well the models created in the preceding phase satisfy the project's goals, an evaluation is conducted. This entails evaluating model performance in relation to the first phase's established business criteria and making any necessary model refinements.
6. **Deployment**: Putting the models into use in the operational environment is the last stage. This could entail putting model monitoring protocols in place, integrating the models into already-existing systems, and offering end users documentation and training.



## DATA PIPELINE





## DATA ENGINEERING PIPELINE SCHEMA

- **Data Ingestion:** Boto3  
Boto3 is the Python SDK for Amazon Web Services (AWS). It enables Python programmatic interaction between developers and different AWS services. Python code is all that is needed to create, set up, and maintain AWS resources including EC2 instances, S3 buckets, DynamoDB tables, and more using Boto3.
- **Data Storage:** AWS S3
- **Data Processing:** Glue and Crawler
- **Data Consumption:** AWS Athena and Power BI
  - **Data Visualization:** Showcase the results through comprehensive visualizations.

### Uploading data to EC2 Instance

In your system (cmd):

1. Transfer log files from local system to Linux 2 instance
2. Syntax: `scp -i "/keyPath fileToBeUploadedPath/keypair.pem"`  
"path of data      file/file\_name.csv"  
username@ip:pathInRemoteSys
3. Example:  
`pscp -i " /Downloads/project.pem"`  
"/Downloads/data/ride\_share\_data.csv"  
ec2user@32.153.269.114:/home/ec2-user/
4. CSV file uploads to EC2 instance.

**Note: To confirm that files have been transferred; you can use 'ls'.**

```
(base) manichandrdomala@Manichandras-MacBook-Air Flask % scp -i "/Users/manichandrdomala/Desktop/Data_Enng/Flask/analytics.pem" "/Users/manichandrdomala/Desktop/Data_Enng/Mid_term_Project/rideshare_kaggle.csv" ec2-user@ec2-44-202-106-232.compute-1.amazonaws.com:/home/ec2-user/
The authenticity of host 'ec2-44-202-106-232.compute-1.amazonaws.com (44.202.106.232)' can't be established.
ED25519 key fingerprint is SHA256:EnCpkcOT3x04m8pX8SkZDhwjcYG1QgafvRwMmlX/MiE.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:50: 54.172.121.45
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-202-106-232.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
rideshare_kaggle.csv
100% 350MB 2.7MB/s 02:09
(base) manichandrdomala@Manichandras-MacBook-Air Flask %
```

## Transferring data from EC2 to S3 using Boto3

In your system (cmd):

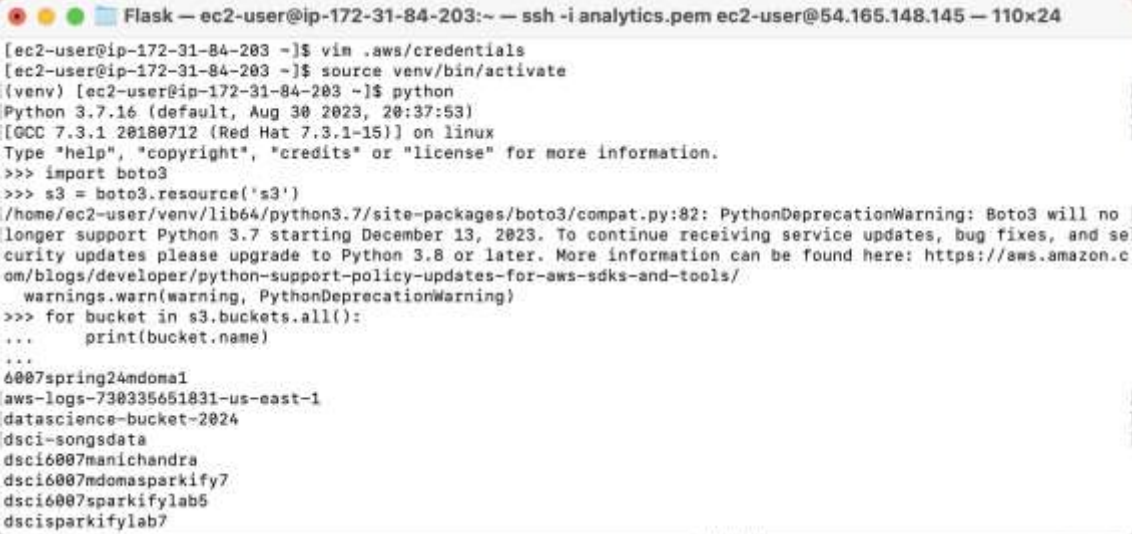
1. Create Amazon Linux Instance
2. SSH into Instance
3. In SSH, go to home (cd ~)
4. Create directory.  
    mkdir .aws
5. Create new file.  
    vim .aws/credentials
6. Paste entire AWS CLI content from AWS details.
7. Create a virtual environment.  
    python3 -m venv venv  
    source venv/bin/activate

## 8. Install Boto3

`pip install boto3`

## 9. to check if installation successful, run python [type python3] then follow these steps:

- `import boto3`
- `s3 = boto3.resource('s3')`
- `for bucket in s3.buckets.all():`  
    `print(bucket.name)`
- you should be able to see all the buckets in your s3 as below.



```
Flask — ec2-user@ip-172-31-84-203:~ — ssh -i analytics.pem ec2-user@54.165.148.145 — 110x24
[ec2-user@ip-172-31-84-203 ~]$ vim .aws/credentials
[ec2-user@ip-172-31-84-203 ~]$ source venv/bin/activate
(venv) [ec2-user@ip-172-31-84-203 ~]$ python
Python 3.7.16 (default, Aug 30 2023, 20:37:53)
[GCC 7.3.1 20180712 (Red Hat 7.3.1-15)] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> import boto3
>>> s3 = boto3.resource('s3')
/home/ec2-user/venv/lib64/python3.7/site-packages/boto3/compat.py:82: PythonDeprecationWarning: Boto3 will no
longer support Python 3.7 starting December 13, 2023. To continue receiving service updates, bug fixes, and se
curity updates please upgrade to Python 3.8 or later. More information can be found here: https://aws.amazon.c
om/blogs/developer/python-support-policy-updates-for-aws-sdks-and-tools/
  warnings.warn(warning, PythonDeprecationWarning)
>>> for bucket in s3.buckets.all():
...     print(bucket.name)
...
6007spring24mdoma1
aws-logs-730335651831-us-east-1
datascience-bucket-2024
dsci-songsdata
dsci6007manichandra
dsci6007mdomasparkify7
dsci6007sparkifylab5
dscisparkifylab7
```

## 10. To be able to upload the data to S3; make sure the data file actually exist in your instance

## 11. Create python file.

`touch uploadtos3.py`

`vim uploadtos3.py`

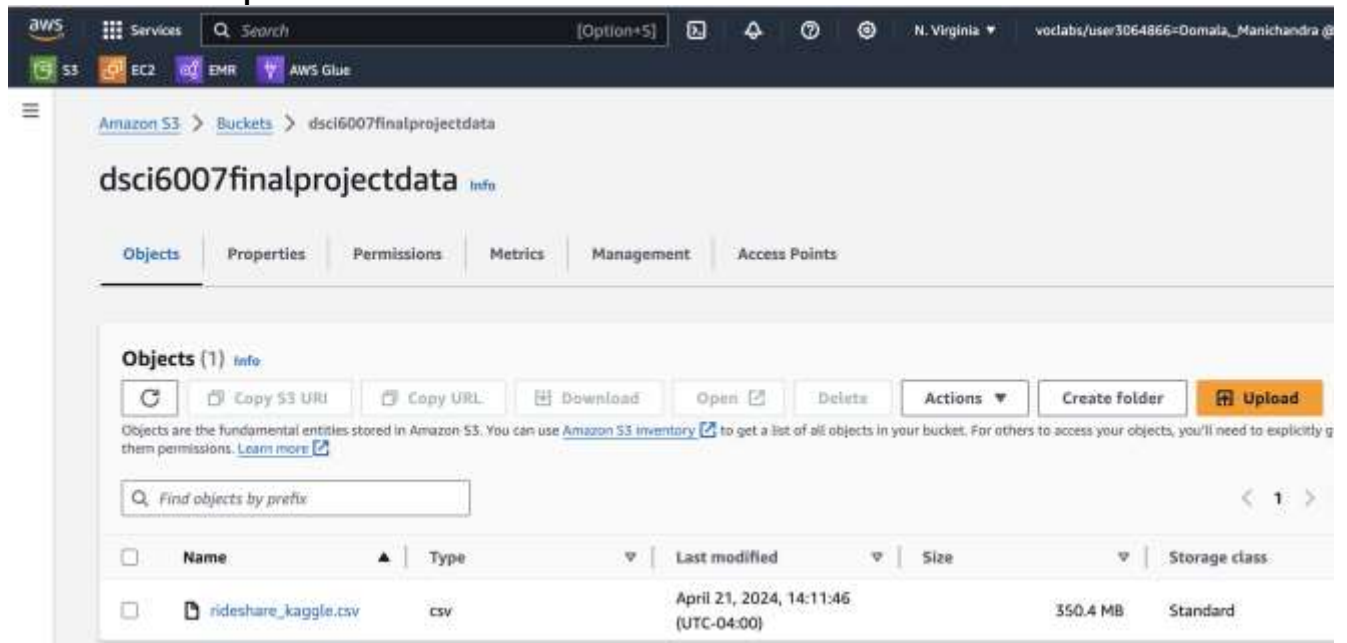
## 12. Paste the entire code given below.

## 13. Save and quit (:wq)

## 14. Run the file.

`Python uploadtos3.py`

15. Go to S3[Choose the bucket created by code] and confirm that files has been uploaded.



## Uploadtos3.py

```
import os
import boto3
import logging
from botocore.exceptions import ClientError

def create_bucket(bucket_name, region=None):
    """Create an S3 bucket in a specified region
    If a region is not specified, the bucket is created in the S3 default
    region (us-east-1).
    :param bucket_name: Bucket to create
    :param region: String region to create bucket in, e.g., 'us-west-2'
    :return: True if bucket created, else False
    """
    # Create bucket
    try:
        if region is None:
            s3_client = boto3.client("s3")
            s3_client.create_bucket(Bucket=bucket_name)
```

```

else:
    s3_client = boto3.client("s3", region_name=region)
    location = {"LocationConstraint": region}
    s3_client.create_bucket(
        Bucket=bucket_name, CreateBucketConfiguration=location
    )
except ClientError as e:
    logging.error(e)
    return False
return True

def upload_csv_to_s3(csv_file_path, bucket_name, object_key):
    """Upload a CSV file to an S3 bucket
    :param csv_file_path: Path to the CSV file
    :param bucket_name: Name of the S3 bucket
    :param object_key: Key (path) of the object in the bucket
    :return: True if successful, False otherwise """
    s3_client = boto3.client("s3")
    try:
        s3_client.upload_file(csv_file_path, bucket_name, object_key)
    except ClientError as e:
        logging.error(e)
        return False
    return True

# Make sure to give a unique (non-existing) bucket name
bucket_name = "dsci6007finalprojectdata"
create_bucket(bucket_name)

# Path to the CSV file you want to upload
csv_file_path = "/home/ec2-user/rideshare_kaggle.csv"

# Key (path) of the object in the bucket
object_key = "rideshare_kaggle.csv"
# Upload the CSV file to S3
upload_csv_to_s3(csv_file_path, bucket_name, object_key)

```

## RESULTS

### Data Analysis using Athena

Count the number of rides by hour:

```
SELECT hour, COUNT(*) AS total_rides
FROM "rideshare_dataset"."rideshare_dataset_raw_dataset"
GROUP BY hour
ORDER BY total_rides DESC;
```

The screenshot shows the AWS Athena Query Editor interface. The query editor displays the following SQL query:

```
SELECT hour, COUNT(*) AS total_rides
FROM "rideshare_dataset"."rideshare_dataset_raw_dataset"
GROUP BY hour
ORDER BY total_rides DESC;
```

The query has been executed successfully, and the results are displayed in a table with 24 rows. The results are ordered by total\_rides in descending order.

hour	total_rides
0	22472
1	19501
2	18388
3	18384
4	18384
5	18384
6	18384
7	18384
8	18384
9	18384
10	18384
11	18384
12	18384
13	18384
14	18384
15	18384
16	18384
17	18384
18	18384
19	10384

## Frequently Travelled source & destination

SELECT source, destination, COUNT(\*) AS route\_count  
 FROM rideshare\_dataset.rideshare\_dataset\_raw\_dataset  
 GROUP BY source, destination  
 ORDER BY route\_count DESC  
 LIMIT 10;

The screenshot shows the AWS Athena console interface. The query editor displays the following SQL query:

```
1 SELECT source, destination, COUNT(*) AS route_count
2 FROM rideshare_dataset.rideshare_dataset_raw_dataset
3 GROUP BY source, destination
4 ORDER BY route_count DESC
5 LIMIT 10;
```

The query has been executed successfully. The results are displayed in a table with 10 rows, showing the source, destination, and route count for the most frequent travel routes.

#	source	destination	route_count
1	"South Station"	"Financial District"	18952
2	"Financial District"	"South Station"	18952
3	"North End"	"Back Bay"	18225
4	"Back Bay"	"North End"	18225
5	"West End"	"Finchway"	12165
6	"Finchway"	"West End"	12165
7	"Haymarket Square"	"Financial District"	10754
8	"Financial District"	"Haymarket Square"	10754
9	"North End"	"Beacon Hill"	10039
10	"Beacon Hill"	"North End"	10039



Count the number of rides by cab type:

```
SELECT cab_type, COUNT(*) AS ride_count  
FROM rideshare_dataset.rideshare_dataset_raw_dataset  
GROUP BY cab_type;
```

The screenshot displays the AWS Athena console interface. On the left, the 'Data' sidebar shows the 'Data source' as 'AwsDataCatalog', the 'Database' as 'rideshare\_dataset', and a list of tables including 'rideshare\_dataset\_raw\_dataset'. The main panel shows a SQL query editor with the following query:

```
1 SELECT cab_type, COUNT(*) AS ride_count  
2 FROM rideshare_dataset.rideshare_dataset_raw_dataset  
3 GROUP BY cab_type;
```

Below the query editor, the 'Query results' tab is active, showing a 'Completed' status. The query execution details are: Time in queue: 106 ms, Run time: 849 ms, Data scanned: 350.36 MB. The results are displayed in a table with 2 columns: 'cab\_type' and 'ride\_count'.

#	cab_type	ride_count
1	"Uber"	385663
2	"Lyft"	507408

## By weather

SELECT short\_summary ,count(short\_summary) as total  
FROM rideshare\_dataset.rideshare\_dataset\_raw\_dataset  
group by short\_summary order by total desc;

The screenshot shows the AWS Athena console interface. The query editor displays the following SQL query:

```
1 SELECT
2   short_summary ,count(short_summary) as total
3 FROM rideshare_dataset.rideshare_dataset_raw_dataset
4 group by short_summary order by total desc;
```

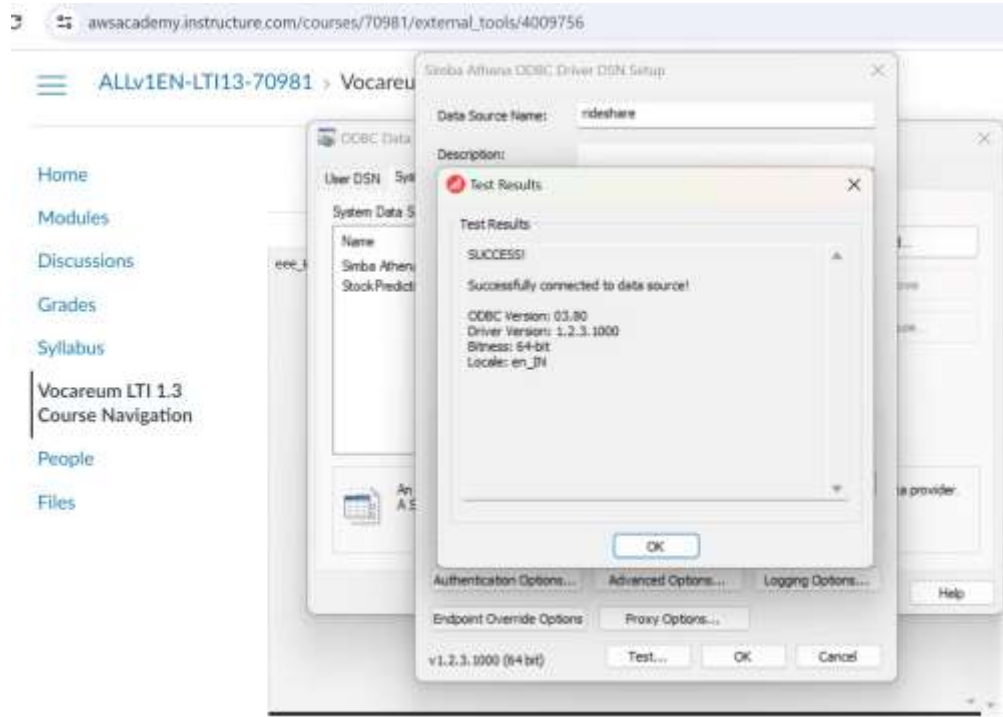
The query has been executed successfully, and the results are shown in a table with 9 rows. The table has two columns: short\_summary and total.

#	short_summary	total
1	"Overcast"	218805
2	"Mostly Cloudy"	146210
3	"Partly Cloudy"	127234
4	"Clear"	87126
5	"Light Rain"	54912
6	"Rain"	23712
7	"Possible Drizzle"	18838
8	"Foggy"	9060
9	"Drizzle"	7286

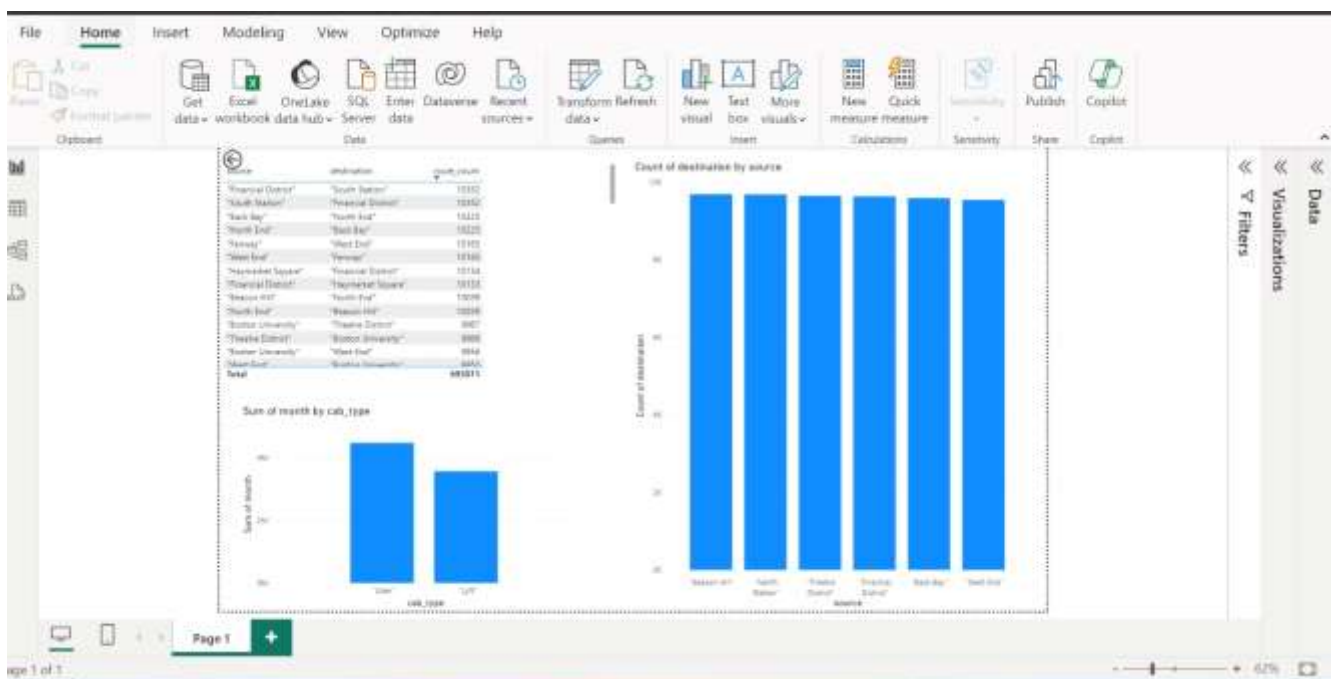
## Connecting AWS Athena to Power BI Using ODBC

To connect AWS Athena to Power BI using ODBC (Open Database Connectivity), you'll need to follow these general steps:

1. Set up AWS Athena: Make sure you have an AWS account and Athena is properly configured with the necessary permissions to access your data stored in S3.
2. Install Athena ODBC Driver: Download and install the ODBC driver for Athena. AWS provides an official ODBC driver that you can download from the AWS website.
3. Configure ODBC Data Source: Set up an ODBC data source for Athena on your machine. This involves specifying the connection details such as AWS region, S3 bucket, and access credentials.
4. Connect Power BI to Athena: Open Power BI and use the ODBC data source you configured to establish a connection to Athena. You'll need to provide the same connection details in Power BI.

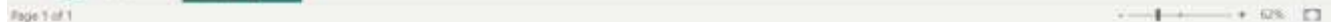


## Data Visualization using Power BI

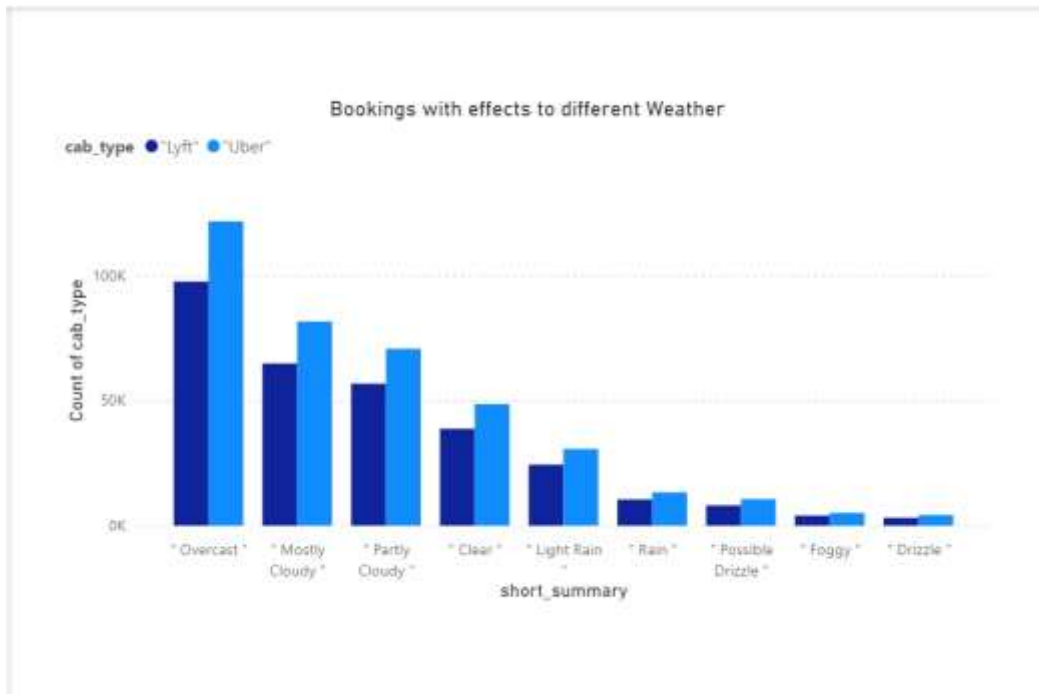


Above visualization on right shows mostly travelled source to northeastern university destination.

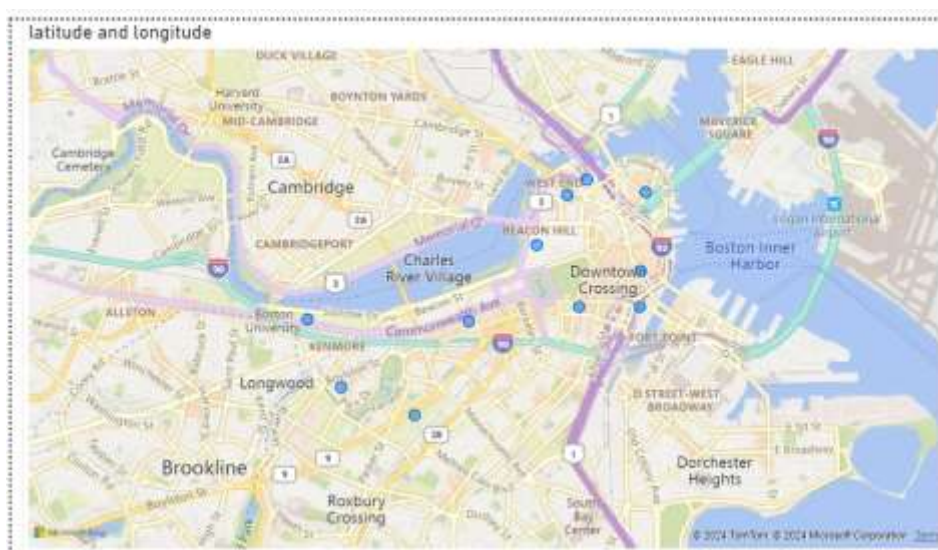
By using this visualization northeastern university can run shuttle to those source location to help student to travel Hassel freely to university.



And, In which show number of uber and lyft ride. It shows uber was leading at those location and by using this visualization lyft can increase their available at those location.



Bar Graph shows bookings with effects to different weather, their count of cab\_type by cab\_type. Weather was mostly Overcast and Cloudy.



This visualization shows Locations by latitude and longitude.

## CONCLUSION

To sum up, the amalgamation of Power BI with Amazon Web Services (AWS) offers an all-encompassing approach to augmenting data management, conversion, and visualization procedures. Utilizing AWS services like Glue, Crawler, Athena, S3, and EC2, data is organized, stored, and made easily accessible for analysis. Data freshness is guaranteed by the automated nature of the AWS Crawler, while smooth data processing and querying are made possible by Athena.

Decision-makers are equipped with actionable insights when Power BI is connected to the AWS environment through the use of Open Database Connectivity (ODBC), which enables intuitive and intelligent analysis. This project serves as an excellent example of how cloud-based services can streamline and optimize data management procedures, allowing organizations to make decisions that are well-informed and based on current, reliable data. Businesses can fully utilize their data through the smooth integration of Power BI and AWS, fostering innovation and giving them a competitive edge in the ever-changing business environment of today.



## CONTRIBUTIONS/REFERENCES

- <https://www.kaggle.com/datasets/brllrb/uber-and-lyft-dataset-boston-ma>
- <https://docs.aws.amazon.com/>
- <https://www.linkedin.com/learning/sql-essential-training-2019/understanding-sql?u=2359714>
- <https://www.linkedin.com/learning/amazon-redshift-essentials/turn-data-into-information-with-redshift?u=2359714>
- <https://www.youtube.com/live/77jlzgvCIYY?si=uJlPFw4GrRnEVEAs>