



RMIT University Vietnam
School of Science, Engineering, and Technology

EEET2574
Big Data for Engineering

Assignment 3: Big Data Project

Final Report

Mitigating Flight Delays through Weather Analytics and Big Data

Group: Team 1

Students:

Phan Nhat Minh (s3978598)
Tran Manh Cuong (s3974735)
Nguyen Vinh Gia Bao (s3986287)

Lecturer: Dr. Arthur Tang

Submission Date: 13th January, 2025

"We declare that in submitting all work for this assessment I have read, understood and agree to the content and expectations of the [Assessment Declaration](#) "

Table of Contents

1. Overview	3
1.1. Statement of the Problem	3
1.2. Role of Big Data	3
1.3. Scope of Work.....	4
1.4. Roles and Responsibilities.....	5
2. Solution Design	6
2.1. Proposed Solution	6
2.2. Data Sources.....	6
2.2.1. Historical Dataset 1: 2023 U.S. Civil Flights	6
2.2.2. Historical Dataset 2: 2023 Weather Meteo by Airport	8
2.2.3. Historical Dataset 3: Airport Geolocation	9
2.2.4. Streaming Data: OpenWeatherMap API.....	9
2.3. Data Pipelines	10
2.3.1. Streaming Data ETL Pipeline	10
2.3.2. Historical Data & Testing Data ETL Pipeline	11
2.3.3. Prediction Model Data Pipeline.....	13
2.3.4. Visualization Data Pipeline	14
2.4. Technology and System Architecture.....	15
2.5. Model Selection.....	17
2.6. Cost Estimation.....	18
2.7. Preliminary Results.....	18
2.7.1. Data Prediction Model.....	18
2.7.2. Data Visualization Dashboards.....	20
3. Conclusion.....	21
3.1. Summary of Work.....	21
3.2. Challenges and Limitations	22
3.3. Future Work.....	22
4. References	23
Appendices.....	24
Appendix A. Declaration of Project Team Member	24
Declaration of Project Team Member	24
Appendix B. Cost Estimation	26
Appendix C. Visualization Dashboards.....	27
Appendix D. Streaming Data Pipeline Process	33

1. Overview

1.1. Statement of the Problem

Flight delays and cancellations remain a persistent and costly issue in the aviation industry, with significant impacts on passengers, airlines, and the broader economy. The Government Accountability Office (GAO) reported that in 2007, approximately one in four passengers experienced delays averaging 1 hour and 54 minutes, with 24% of flights delayed and 2% canceled entirely. These disruptions accounted for 320 million hours of passenger delays and an estimated economic loss of up to \$41 billion [1]. Even as recently as 2023, nearly 20% of flights were delayed, demonstrating the ongoing prevalence of this issue [2].

The economic burden of flight delays on U.S. airlines is equally significant, with annual costs estimated at \$22 billion, driven by increased fuel consumption, crew expenses, and passenger compensation [3]. In 2022 alone, delayed and canceled flights generated an economic impact of \$30–34 billion, reflecting the costs of extended airline operations and the value of lost passenger time [4]. Passengers face additional hardships, including financial losses from rescheduled accommodations, decreased productivity, physical discomfort, and a decline in trust and loyalty toward airlines [5].

Given that weather is consistently identified as a primary cause of flight cancellations, including in the years preceding the COVID-19 pandemic [1], addressing weather-related disruptions offers a critical opportunity to reduce delays. Leveraging big data and machine learning models to predict flight delays based on weather forecasts can provide a proactive solution. By mitigating weather-related delays, airlines can minimize disruptions, reduce costs, and improve the passenger experience, highlighting the urgency for innovative strategies in this domain.

1.2. Role of Big Data

Big data enables real-time analysis of complex variables that contribute to flight disruptions, such as weather conditions, traffic congestion in airspace, and airport capacity constraints. Predictive models built on big data can forecast potential delays before they occur, allowing airlines to make proactive adjustments. For instance, analyzing weather forecasts in conjunction with historical delay patterns can help airlines reschedule or reroute flights more effectively, reducing disruptions for passengers and minimizing operational costs.

By processing vast amounts of data, big data analytics can uncover patterns and trends that would otherwise remain hidden. These insights can lead to:

- **Predictive Delay Management:** Identifying flights most likely to be delayed due to specific weather conditions, enabling preemptive scheduling changes.
- **Resource Optimization:** Improving the allocation of resources, such as crew and ground staff, based on forecasted delays.
- **Dynamic Rebooking Systems:** Using real-time data to optimize passenger rebooking strategies during disruptions, enhancing the travel experience and reducing financial losses.
- **Weather Forecast Integration:** Correlating weather data with flight performance metrics to create robust models for disruption mitigation.

A notable example of big data's successful application is the Federal Aviation Administration's (FAA) Traffic Flow Management System (TFMS). This system processes real-time air traffic and weather data from various sources, including flight plans and radar detections, to manage air traffic flow more efficiently. By mitigating congestion and minimizing delays, TFMS demonstrates how big data can transform operational efficiency in aviation [6].

1.3. Scope of Work

The table below provides a high-level overview of the project in-scope:

Table 1. Scope of Work

Priority	Phase	Description	Milestone
High	Conceptualization	Identify the problem	<ul style="list-style-type: none"> Conceptualized the product from the identified problem and data sources for the project. Clear articulation of the problem statement. List of potential data sources.
High		Draft the solution	<ul style="list-style-type: none"> Drafted a preliminary solution design. Completion of draft by deadline. Feedback received and incorporated into the draft.
High		Find data sources	<ul style="list-style-type: none"> Identified reliable and diverse data sources for the project. Minimum of 3 viable data sources identified. Data relevance and reliability verified.
Medium	Planning	Define roles and responsibilities of team members	<ul style="list-style-type: none"> Role distribution documented and shared with the team. All roles assigned to members. Team alignment on responsibilities.
Medium		Develop detailed project timeline and deliverables	<ul style="list-style-type: none"> Timeline finalized and deliverables outlined. Approved project plan with deadlines. Team adherence to planned milestones.
High	Execution	Build the data pipeline	<ul style="list-style-type: none"> Data pipeline developed and tested. Pipeline handles expected data volume. Successful initial tests with sample data.
High		Develop predictive model or analytical solution	<ul style="list-style-type: none"> Model built and validated. Model achieves a predefined accuracy metric. Successful validation on test datasets.
High		Fetch and process streaming data	<ul style="list-style-type: none"> Streaming data successfully fetched and processed. Real-time data ingestion pipeline operational. Latency within acceptable thresholds.

High		Build visualization dashboards	<ul style="list-style-type: none"> Dashboards created to display project insights. Dashboards are user-friendly, responsive, and updated in real-time. Stakeholder approval of visualizations.
Medium		Analyze results and refine the solution	<ul style="list-style-type: none"> Results analyzed and improvements suggested. Insights generated within deadline. Actionable recommendations documented.
Low	Deployment	Deploy the solution to production	<ul style="list-style-type: none"> Solution deployed for live data analysis. Solution operational without major issues. Real-time data processing capability.
Low	Post-Deployment	Monitor solution performance and make necessary adjustments	<ul style="list-style-type: none"> Performance reports generated and refinements implemented. Monthly performance report. Issues addressed within SLA timelines.

1.4. Roles and Responsibilities

Table 2. Roles and Responsibilities

Member	Role	Responsibilities
Phan Nhat Minh (s3978598)	Project Manager, Data Scientist	Minh is responsible for planning the project by outlining milestones and overseeing the team's progress to ensure alignment with objectives. Key tasks involve managing documentation by structuring reports and contributing to their content. Additionally, Minh is responsible for building pipelines for data visualization, designing user-friendly dashboards, and supporting team members in debugging and development. These combined efforts ensure efficient project execution and successful delivery of results.
Tran Manh Cuong (s3974735)	Data Scientist, Data Engineer	Cuong is responsible for designing the product's infrastructure and managing cloud services to ensure scalability and reliability. Key tasks involve processing streaming data, building pipelines to efficiently fetch and load it, and contributing to the project report by writing specific sections. Additionally, Cuong involves supporting team members in debugging and development, ensuring smooth progress and collaboration throughout the project.
Nguyen Vinh Gia Bao (s3986287)	Data Scientist, Data Engineer	Bao is responsible for designing and building the ETL pipeline for raw historical data, ensuring efficient data extraction, transformation, and loading processes. Additional tasks involve creating a pipeline for processing data used in model training,

		overseeing the training process, and taking accountability for the model's performance and outcomes. Bao also includes contributing to the project report by writing specific sections and providing support to team members in debugging and development to maintain smooth project progress.
--	--	--

2. Solution Design

2.1. Proposed Solution

The project aims to develop a data-driven solution that leverages historical and real-time data to predict flight delays with high accuracy. This predictive model will be the centerpiece of a broader system designed to enhance operational efficiency and improve the passenger experience. The solution encompasses the following key features:

- **Predictive Model:** The system will utilize a machine learning model trained on historical flight and weather datasets to forecast potential delays. As real-time data becomes available, such as current weather conditions and flight schedules, the model will continuously update its predictions, ensuring accuracy and adaptability.
- **Real-Time Delay Prediction:** By integrating weather data, flight schedules, and air traffic patterns, the system will identify potential disruptions proactively. This allows airlines to take preemptive measures to mitigate delays, ensuring smoother operations and reduced cascading disruptions.
- **Visualization Dashboards:** Intuitive dashboards tailored for Air Traffic Controllers (ATC) and airline managers will present real-time insights. These dashboards will allow stakeholders to monitor flight statuses, make informed decisions, and communicate updates to passengers promptly.

The entire solution will be built and deployed on AWS cloud platforms, leveraging their scalability, reliability, and cost-efficiency. This approach was chosen because cloud services eliminate the need for expensive on-premise infrastructure while providing the flexibility and speed required for handling dynamic and large-scale aviation data. The cloud's robust security features also ensure that sensitive flight data remains protected.

2.2. Data Sources

2.2.1. *Historical Dataset 1: 2023 U.S. Civil Flights*

The main dataset containing the characteristics of each civil flight in the United States in 2023. Delays are given in duration and classified as a category to facilitate analysis. Each flight contains information on the airline, manufacturer, model and age of the aircraft. This dataset will be merged with the weather report at each airport by the date of 2023.

This dataset, sourced from the Bureau of Transportation Statistics (BTS) website, is available on Kaggle. Please find the original dataset [here](#).

The following table will present all columns that are in the dataset:

Table 3. 2023 U.S. Civil Flights Dataset Description

Fields	Description
flightdate	The date of the flight in YYYY-MM-DD format
day_of_week	Numerical representation of the day of the week the flight occurred (e.g., 1 (Monday), 2 (Tuesday))
airline	The airline operating the flight
tail_number	The unique identifier for the aircraft
dep_airport	The IATA code of the departure airport
dep_cityname	The name of the city where the departure airport is located
deptime_label	The time label of the flight's departure (e.g., Morning, Afternoon)
dep_delay	Delay time (in minutes) for the departure
dep_delay_tag	Whether the departure flight was delayed or not (labeled “1” when “dep_delay” is more than 5)
dep_delay_type	The severity of the delay time (e.g., low, medium, high)
arr_airport	The IATA code of the arrival airport
arr_cityname	The name of the city where the arrival airport is located
arr_delay	Delay time (in minutes) for the arrival
arr_delay_tag	Whether the arrival flight was delayed or not (labeled “1” when “arr_delay” is more than 15)
arr_delay_type	The severity of the delay time (e.g., low, medium, high)
flight_duration	The total duration of the flight in minutes
distance_type	Classification of the flight's distance (e.g., short-haul, long-haul)
delay_carrier	Delay time (in minutes) attributed to the carrier
delay_weather	Delay time (in minutes) caused by weather conditions
delay_nas	Delay time (in minutes) due to National Airspace System issues

delay_security	Delay time (in minutes) due to security-related issues
delay_lastaircraft	Delay time (in minutes) attributed to the previous aircraft
manufacturer	The manufacturer of the aircraft (e.g., Boeing, Airbus)
model	The specific model of the aircraft (e.g., 737-800, A320)
aircraft_age	The age of the aircraft in years

2.2.2. Historical Dataset 2: 2023 Weather Meteo by Airport

This dataset contains daily weather information for various airports throughout 2023. Each entry includes key meteorological data such as temperature, precipitation, wind speed, and atmospheric pressure. This dataset will be merged with the U.S. civil flight data in 2023.

This dataset, sourced from the [Meteostat Python library](#), is available on Kaggle. Please find the original dataset [here](#).

The following table will present all columns that are in the dataset:

Table 4. 2023 Weather Meteo by Airport Dataset Description

Fields	Description
time	The date when the weather was recorded at the departure airport
tavg	The average temperature (°C) on the flight date
tmin	The minimum temperature (°C) on the flight date
tmax	The maximum temperature (°C) on the flight date
prcp	The amount of precipitation (mm) on the flight date
snow	The amount of snowfall precipitation (mm) on the flight date
wdir	The wind direction (degrees) on the flight date
wspd	The wind speed (km/h) on the flight date
pres	The atmospheric pressure (hPa) on the flight date
airport_id	The IATA code of the departure airport

2.2.3. *Historical Dataset 3: Airport Geolocation*

This dataset provides geolocation and identification details for U.S. airports in 2023. It includes the IATA codes, airport names, cities, states, countries, and precise latitude and longitude coordinates.

This dataset, sourced from Gwen on Kaggle. Please find the original dataset [here](#).

The following table will present all columns that are in the dataset:

Table 5. Airport Geolocation Dataset Description

Fields	Description
IATA_CODE	The unique three-letter code assigned to the airport by the International Air Transport Association (IATA)
AIRPORT	The full name of the airport
CITY	The city where the airport is located
STATE	The state where the airport is situated
COUNTRY	The country where the airport is located
LATITUDE	The geographical latitude (decimal degree) of the airport
LONGITUDE	The geographical longitude (decimal degree) of the airport

2.2.4. *Streaming Data: OpenWeatherMap API*

The OpenWeatherMap API offers comprehensive weather insights, including real-time data, multi-day forecasts, and historical information for millions of locations globally. Its features encompass current weather conditions, detailed daily forecasts, air quality metrics, and weather alerts, all delivered in an easy-to-parse JSON format. In this project, the API was leveraged to retrieve daily weather data for a specific location, providing up to eight days of accurate and detailed forecasting, enabling precise and actionable weather analysis.

The table below highlights the specific fields extracted and selected from each API call, showcasing the key data points utilized for analysis:

Table 6. Real-time Weather Dataset extracted from OpenWeatherMap API Description

Fields	Description
lat	The latitude of the location.
long	The longitude of the location.

daily.dt	The date of the forecast.
daily.temp.min	The minimum temperature (°C) for the day.
daily.temp.max	The maximum temperature (°C) for the day.
daily.pressure	The atmospheric pressure (hPa) for the day.
daily.wind_speed	The wind speed (m/s) for the day.
daily.wind_deg	The wind direction (°) for the day.
daily.rain	The rain volume (mm) for the day.
daily.snow	The snow volume (mm) for the day.

2.3. Data Pipelines

2.3.1. Streaming Data ETL Pipeline

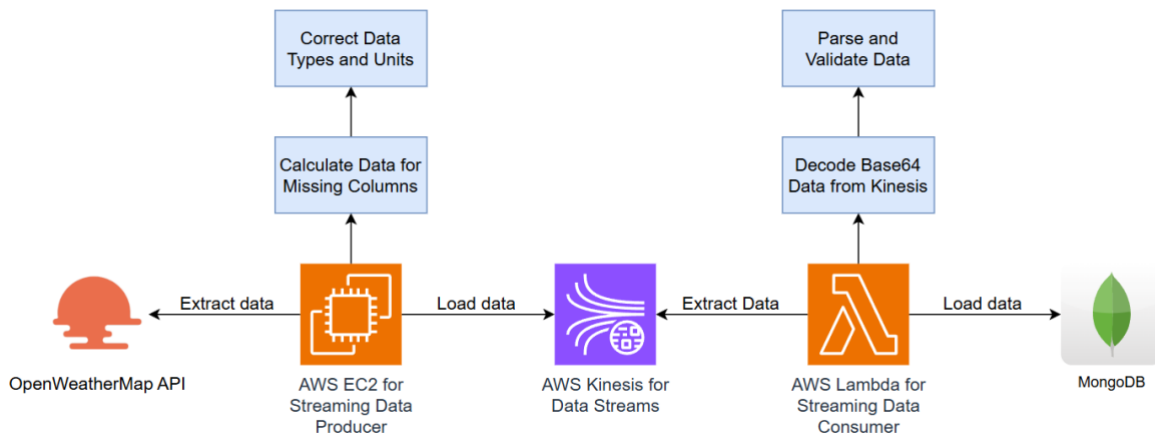


Figure 1. Streaming Data ETL Pipeline Diagram

This pipeline is designed to process and store weather data by integrating real-time data from the OpenWeatherMap API into MongoDB. It utilizes AWS services like EC2, Kinesis, and Lambda to extract, transform, and load the data efficiently. Below is a detailed description of each step in the pipeline:

Table 7. Pipeline Steps for Streaming Data

Dataset(s)	Step	Description
OpenWeatherMap API	1	Weather data is fetched from the OpenWeatherMap API hourly using an AWS EC2 instance. This includes details like temperature, wind speed, precipitation, and more for specified locations.

	2	Missing or incomplete data is handled by calculating derived metrics (e.g., average temperature) and filling gaps to ensure all required fields are populated.
	3	The raw data is transformed to ensure consistency with historical data, including converting units (e.g., wind speed to km/h) and correcting data types for seamless downstream processing.
	4	The transformed data is sent to an AWS Kinesis Data Stream hourly, enabling real-time streaming for subsequent processing stages.
	5	An AWS Lambda trigger is configured to automatically extract streaming data from Kinesis whenever new records are loaded.
	6	In AWS Lambda, the Base64-encoded data from Kinesis is decoded into a readable format for further validation and processing.
	7	The decoded data is parsed into JSON format, and each record is validated to ensure the presence and correctness of critical fields like timestamps, location IDs, and weather metrics.
	8	Validated data is updated or inserted into the MongoDB database.

2.3.2. Historical Data & Testing Data ETL Pipeline

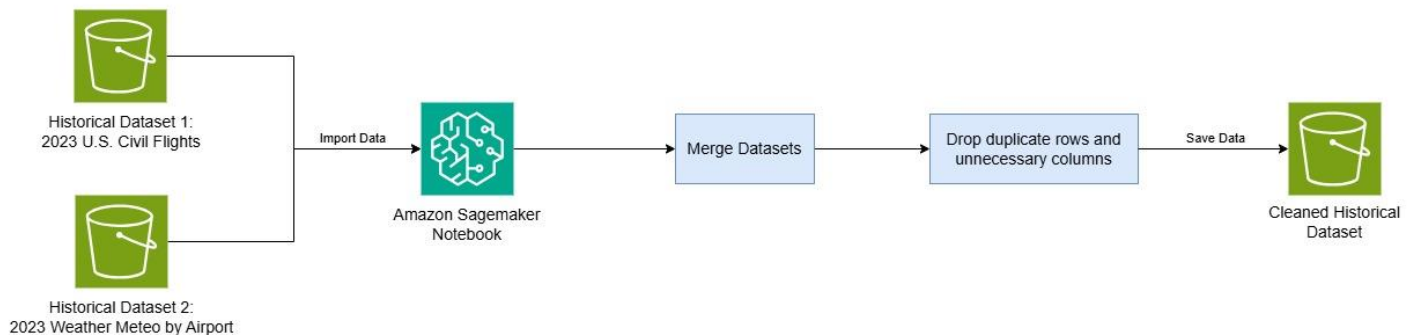


Figure 2. Cleaned Historical Dataset ETL Pipeline Diagram

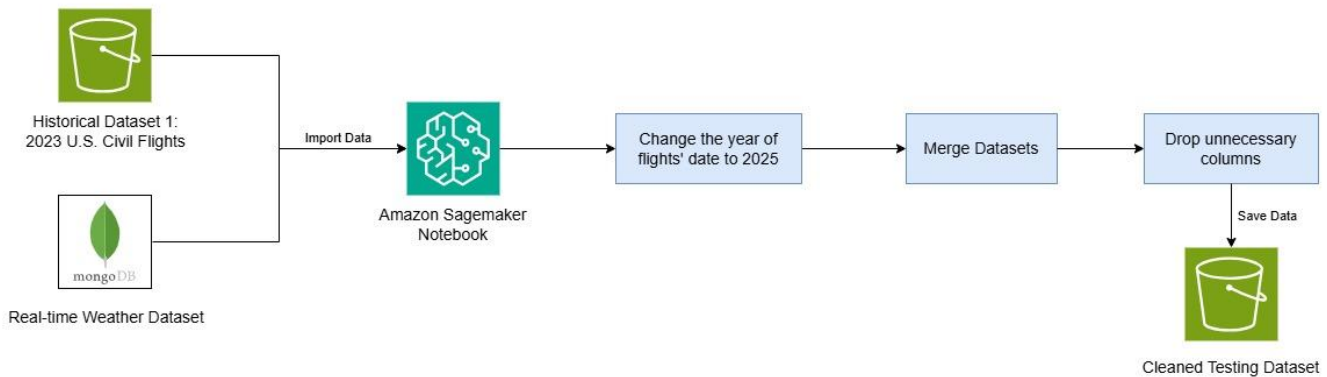


Figure 3. Cleaned Testing Dataset ETL Pipeline Diagram

The pipelines were executed in a single notebook. In the first pipeline, historical datasets were uploaded to S3, imported into Amazon SageMaker, and processed into a cleaned historical dataset. The second pipeline involved importing the real-time weather dataset into SageMaker, merging it with the modified flight dataset, and generating a cleaned testing dataset. The detailed steps of the pipeline are outlined below:

Table 8. Pipeline Steps for Historical and Testing Data

Dataset(s)	Step	Description
2023 U.S. Civil Flights' Dataset + 2023 Daily Weather Data by Airport	1	Import the .csv files in the “raw-data” folder from S3 into Sagemaker.
	2	Merge two .csv files into one single dataframe.
	3	Lowercase the columns in the dataframe.
	4	Drop duplicate rows and unnecessary columns.
	5	Check value counts and unique values for each column.
	6	Save the dataframe into one .csv file and upload it to S3.
2023 U.S. Civil Flights' Dataset + Real-time Weather Dataset	7	Import the streaming weather data from MongoDB and put it into a dataframe.
	8	Modify the year of the flight date in the flights' dataset to 2025 to accommodate with the streaming data.
	9	Merge two .csv files into one single dataframe.
	10	Drop unnecessary columns.
	11	Save the dataframe into one .csv file and upload it to S3.

2.3.3. Prediction Model Data Pipeline

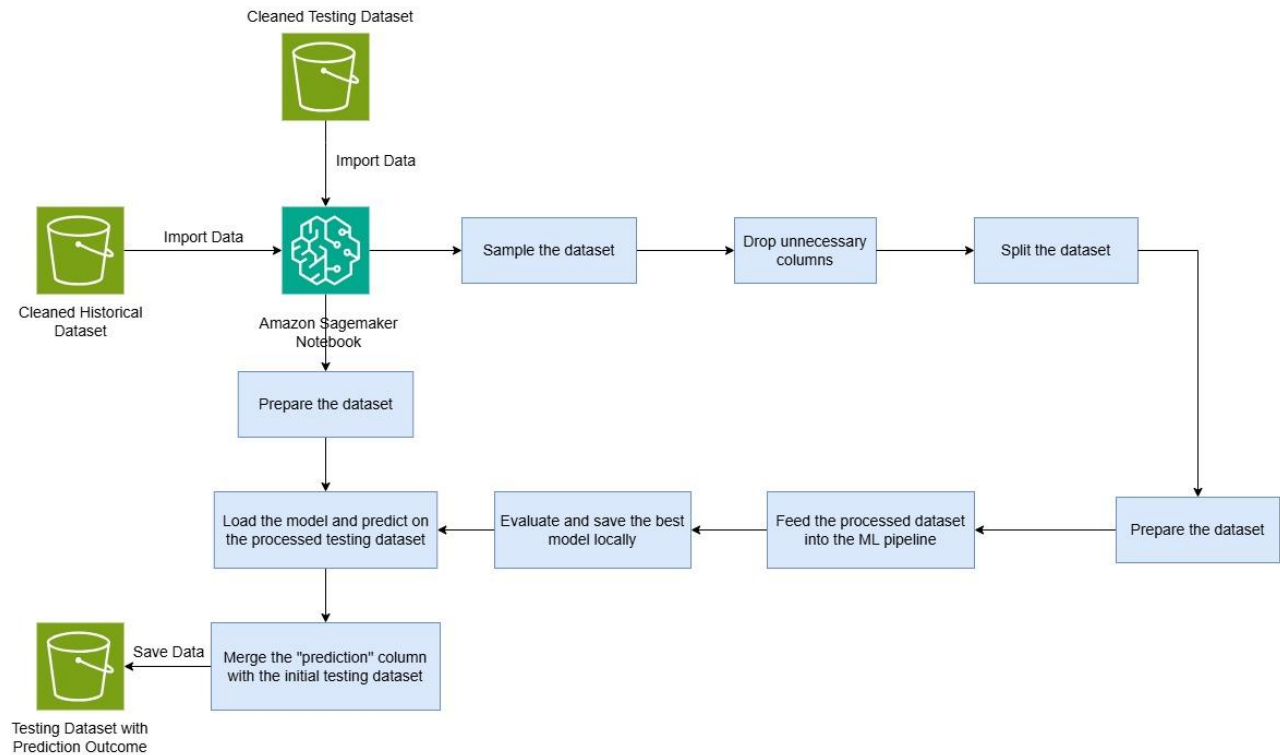


Figure 4. Model Training Data Pipeline Diagram

The pipeline is designed to transform historical and real-time datasets into actionable predictions. It begins with importing, cleaning, and preparing data in Amazon SageMaker. The datasets are then sampled, processed, and passed through an ML pipeline to train and evaluate models. Once the best model is selected, it is used to predict outcomes on the testing dataset, and the results are saved for further analysis. Detailed steps of the pipeline are as follows:

Table 9. Pipeline Steps for Model Training Data

Dataset(s)	Step	Description
Historical Dataset	1	Import the .csv file in the “cleaned-data” folder from S3 into the notebook.
	2	Randomly sample 10 departure flights per airport per date.
	3	Convert the Pandas DataFrame to Spark DataFrame.
	4	Drop irrelevant and high-cardinality columns.
	5	Split the dataset into training and testing data.
	6	One-hot encode categorical features and combine them with numerical features into a feature vector.

	7	Feed data into an ML pipeline with various scaling methods (no scaling, standard, min-max, robust).
	8	Evaluate model performance across scaling methods and save results in a table.
	9	Save the best model locally.
Testing Dataset	10	Import the .csv file in the “cleaned-data” folder from S3 into the notebook.
	11	Preprocess the testing dataset using the pipeline outlined in steps 3, 4 and 6 for the historical data.
	12	Load and use the best-performing model to predict delays on the transformed testing dataset.
	13	Merge the “prediction” column from the transformed testing dataset with the initial dataset.
	14	Save the dataframe into one .csv file and upload it to S3.

2.3.4. Visualization Data Pipeline

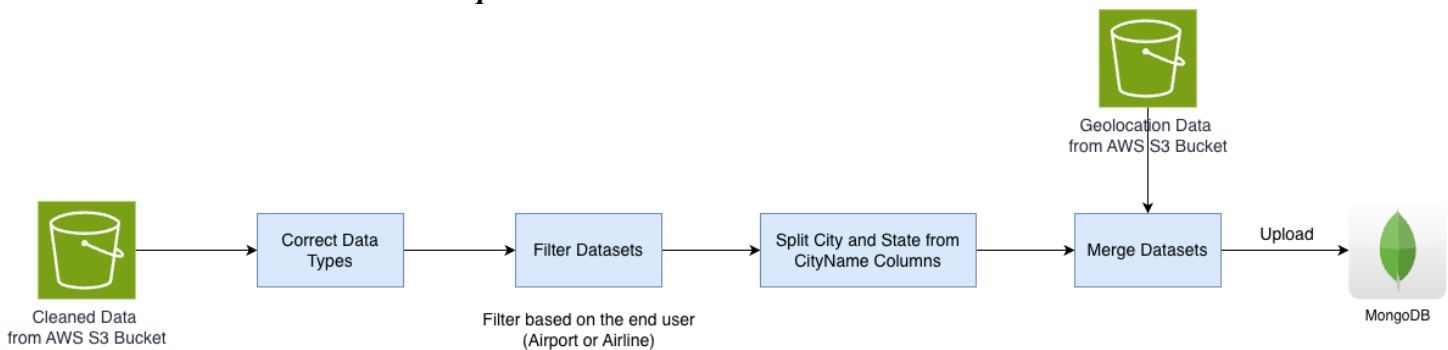


Figure 5. Visualization Data Pipeline Diagram

This pipeline is designed to process and integrate cleaned and geolocation data from AWS S3 storage to produce a dataset suitable for building visualization dashboards. The steps in the pipeline are as follows:

Table 10. Pipeline Steps for Visualization Data

Dataset(s)	Step	Description
Visualization Data	1	Cleaned data is retrieved from AWS S3 bucket.

	2	The initial dataset undergoes type correction to ensure consistency and alignment of data formats across all columns.
	3	Irrelevant or unnecessary data is filtered out, retaining only the records and fields pertinent to the visualization requirements.
	4	The CityName column for both Departure Airport and Arrival Airport is split into separate City and State columns to enable easier aggregation and visualization based on location.
	5	The cleaned data and geolocation data are merged based on common identifiers, creating a comprehensive dataset enriched with geospatial attributes.
	6	The final merged dataset is stored in a MongoDB database and ready for building dashboards with MongoDB Charts.

2.4. Technology and System Architecture

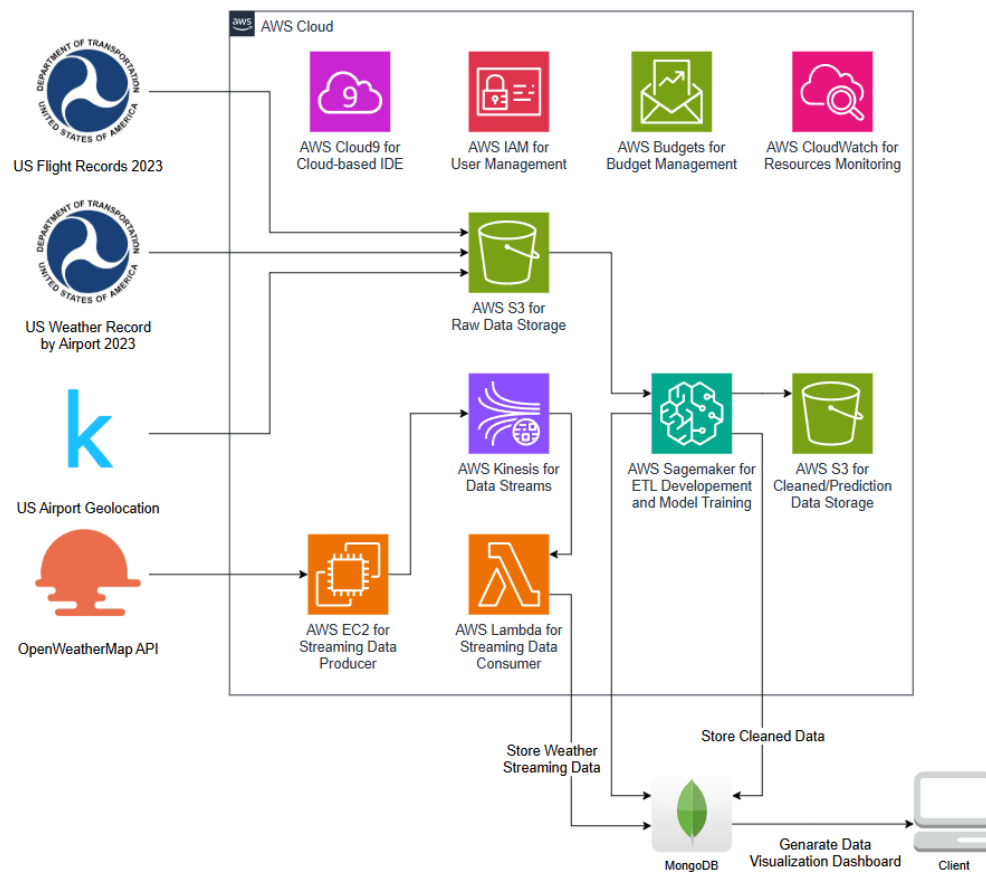


Figure 6. System Architecture Diagram

This system architecture is designed to efficiently process, store, and analyze both historical and streaming data using a combination of AWS services and MongoDB. It integrates multiple data sources, including historical data from US Flight Records 2023, US Weather Record by Airport 2023, and US Airport Geolocation, as well as real-time streaming data from the OpenWeatherMap API. These datasets flow through the system, enabling robust data processing and insightful visualization through dashboards.

Historical raw data is directly uploaded to **AWS S3**, which serves as the primary storage solution for the pipeline. The S3 bucket organizes raw historical data, cleaned data, and prediction data generated by machine learning models. S3's scalability, durability, and cost-effectiveness make it an ideal choice for managing large datasets while ensuring seamless integration with other AWS services like SageMaker. On the other hand, streaming data from the OpenWeatherMap API bypasses initial storage in S3. Instead, it is ingested hourly through a producer application hosted on **AWS EC2** and preprocessed before being streamed to **AWS Kinesis**. EC2 provides customizable computing resources, ensuring the producer operates reliably in the cloud.

To ensure the producer runs uninterrupted, **AWS Cloud9** is used as a cloud-based development environment. Cloud9 allows the producer to remain operational on EC2 even when local machines are offline. The preprocessed data from EC2 is streamed into **AWS Kinesis**, a scalable and low-latency streaming service that acts as the backbone for real-time data ingestion.

Once the streaming data reaches Kinesis, **AWS Lambda** is triggered to process it. Lambda decodes, parses, and validates the data, performing essential transformations before loading it into **MongoDB**. Using Lambda instead of EC2 for this step ensures cost efficiency, as Lambda follows a pay-as-you-go model and executes only when triggered by new data in Kinesis. This event-driven design reduces operational overhead while maintaining scalability.

Processed and cleaned streaming data is stored in **MongoDB**, which also serves as a central repository for cleaned historical data and preprocessed data prepared for visualization. MongoDB's document-based structure is highly suitable for storing hierarchical and JSON-like data, ensuring flexibility and fast retrieval for analysis. For advanced analytics, **AWS SageMaker** is utilized to develop three key data pipelines. The first is an ETL pipeline, which extracts raw historical data from S3 and streaming data from **MongoDB**, cleans, transforms, and merges the datasets to ensure consistency, and stores the cleaned in a S3 bucket for further processing. The second pipeline focuses on model training, where SageMaker preprocesses the cleaned data and trains predictive models, with prediction data stored back in S3. The third pipeline prepares the data for visualization, ensuring the data is optimized and stored in **MongoDB** for dashboard creation.

Visualization and analysis are powered by **MongoDB Charts**, which provides a user-friendly interface for creating interactive dashboards directly from MongoDB data. These dashboards enable stakeholders to gain actionable insights from the processed weather and flight data.

Supporting services such as **AWS IAM** ensure secure access management across all AWS services, while **AWS Budgets** helps monitor and control project costs. **AWS CloudWatch** provides real-time monitoring of system performance and resource usage, ensuring optimal operation and allowing for proactive troubleshooting.

This architecture is highly optimized for handling both historical and real-time data workflows. The combination of S3 for historical data storage, EC2, Kinesis, and Lambda for streaming data ingestion and processing, and SageMaker for advanced analytics and model training ensures scalability, reliability, and cost-efficiency. MongoDB's flexibility further enhances the system by enabling seamless integration with MongoDB Charts for visualization, making this architecture robust and well-suited for weather and flight data analysis.

2.5. Model Selection

For the classification models used in this project, we selected four approaches:

- Logistic Regression
- Gradient Boosting
- Random Forest
- Decision Tree

These models were chosen based on their suitability for the characteristics of the training data. The dataset includes categorical and numerical features, skewed data points and outliers. Tree-based models, such as Random Forest and Decision Tree, are well-suited for handling outliers while capturing non-linear relationships in the data. Gradient Boosting further refines predictions by sequentially improving weak learners, making it suitable for identifying complex patterns in the dataset. Logistic Regression, despite being a linear model, serves as a baseline and is effective in situations where the relationship between features and the target is more straightforward. This diverse set of models ensures that both simple and complex patterns in the data are thoroughly captured, allowing for a robust evaluation of classification performance.

Below are the parameters that specified for each classification model:

```
classification_models = [
    LogisticRegression(featuresCol="features", labelCol="label_indexed", maxIter=10, regParam=0.01),
    RandomForestClassifier(featuresCol="features", labelCol="label_indexed", numTrees=50, seed=42),
    GBTClassifier(featuresCol="features", labelCol="label_indexed", maxIter=10, seed=42),
    DecisionTreeClassifier(featuresCol="features", labelCol="label_indexed", maxDepth=5, seed=42)
]
```

Figure 7. Specified parameters for each classification model

For each model, a different type of scaling will also be used to have a more comprehensive view of the model's performance, which can be seen in the table below representing the best outcome for each model:

Table 11. Classification Model Evaluation Table

Model Name	Accuracy	Precision	Recall	F1 Score	Scaling
Gradient Boosting	0.884378	0.888864	0.883478	0.876245	MinMax Scaler
Decision Tree	0.881242	0.887953	0.881242	0.873269	No Scaling
Random Forest	0.873855	0.884591	0.873855	0.863551	No Scaling
Logistic Regression	0.873214	0.883926	0.873214	0.862819	No Scaling

2.6. Cost Estimation

Below is the estimated monthly running cost of the system:

Table 12. Cost Estimation

Service(s)	Configurations	Monthly Cost
AWS EC2	t2.micro, 1vCPU, 1 GiB Memory	4.23 USD
AWS Kinesis Data Streams	48 records per hour, 1 consumer application, 1 day for data retention	29.22 USD
AWS SageMaker	ml.m5.xlarge, 2 data scientists, 2 notebooks per data scientist, 8 hours per day, 30 days per month	97.92 USD
AWS S3	4 GB storage per month	0.10 USD
AWS Lambda	48 requests per hour, 128GB of memory allocated	0.01 USD
MongoDB	Free tier	0 USD
Total		131.48 USD

The system leverages the MongoDB free tier, ensuring cost-effective storage for processed and prepared data without compromising functionality. For AWS services, the costs are meticulously estimated using the AWS Pricing Calculator, ensuring accurate budgeting and resource allocation.

The detailed cost breakdown of the system can be explored using the following link to the AWS Pricing Calculator: [AWS Pricing Calculator Report Link](#)

2.7. Preliminary Results

2.7.1. Data Prediction Model

In critical decision-making tasks such as predicting flight delays due to weather conditions, evaluating model performance is vital to ensure reliable and actionable outcomes. To assess our binary classification models, we utilized several metrics: **accuracy**, **precision**, **recall**, and **F1 score**. These metrics were chosen to provide a comprehensive evaluation of the models' ability to distinguish between delayed and on-time flights effectively.

Accuracy, which measures the proportion of correctly classified samples to the total number of samples, is defined as [7]:

$$accuracy = \frac{true\ positives + true\ negatives}{true\ positives + false\ positives + true\ negatives + false\ negatives}$$

Precision evaluates the proportion of true positives (correctly classified as “delayed”) among all positive predictions and is given by:

$$precision = \frac{true\ positives}{true\ positives + false\ positives}$$

Recall, also known as sensitivity or true positive rate, assesses how well the model identifies actual positives. It is calculated as:

$$recall = \frac{true\ positives}{true\ positives + false\ negatives}$$

F1 score combines precision and recall into a single metric to address their trade-off, particularly in cases of class imbalance. It is the harmonic mean of precision and recall and is defined as:

$$F1 = 2 \times \frac{precision \times recall}{precision + recall}$$

From the evaluation table (Table 10), the team decided to use the Gradient Boosting model to generate predictions on the processed testing dataset, due to its high accuracy and recall score. Below are predictions for 10 departure flights whether they are delayed or not:

delay_weather	delay_nas	delay_security	delay_lastaircraft	manufacturer	model	aircraft_age	tavg	tmin	tmax	prcp	snow	wdir	wspd	pres	prediction
0	0	0	0	EMBRAER	170/175	10	-5.73	-9.72	-1.74	0.0	1.05	296.0	23.40	1024.0	0.0
0	0	0	0	CANADAI REGIONAL JET	CRJ	22	-5.73	-9.72	-1.74	0.0	1.05	296.0	23.40	1024.0	0.0
0	0	0	0	AIRBUS	A319	26	-5.73	-9.72	-1.74	0.0	1.05	296.0	23.40	1024.0	1.0
0	0	0	0	EMBRAER	170/175	7	-5.27	-9.38	-1.16	0.0	0.00	217.0	15.84	1031.0	1.0
0	0	0	0	EMBRAER	135/145	26	-5.27	-9.38	-1.16	0.0	0.00	217.0	15.84	1031.0	1.0
0	0	0	0	BOEING	737 NG	25	-5.27	-9.38	-1.16	0.0	0.00	217.0	15.84	1031.0	1.0
0	0	0	0	AIRBUS	A319	21	-5.27	-9.38	-1.16	0.0	0.00	217.0	15.84	1031.0	1.0
0	0	0	0	AIRBUS	A320	2	-9.12	-11.29	-6.96	0.0	0.00	280.0	18.25	1035.0	1.0
0	0	0	0	EMBRAER	170/175	5	-9.12	-11.29	-6.96	0.0	0.00	280.0	18.25	1035.0	0.0
0	0	0	0	AIRBUS	A319	27	-9.12	-11.29	-6.96	0.0	0.00	280.0	18.25	1035.0	1.0

Figure 8. Prediction Outcome for 10 Departure Flights

2.7.2. Data Visualization Dashboards

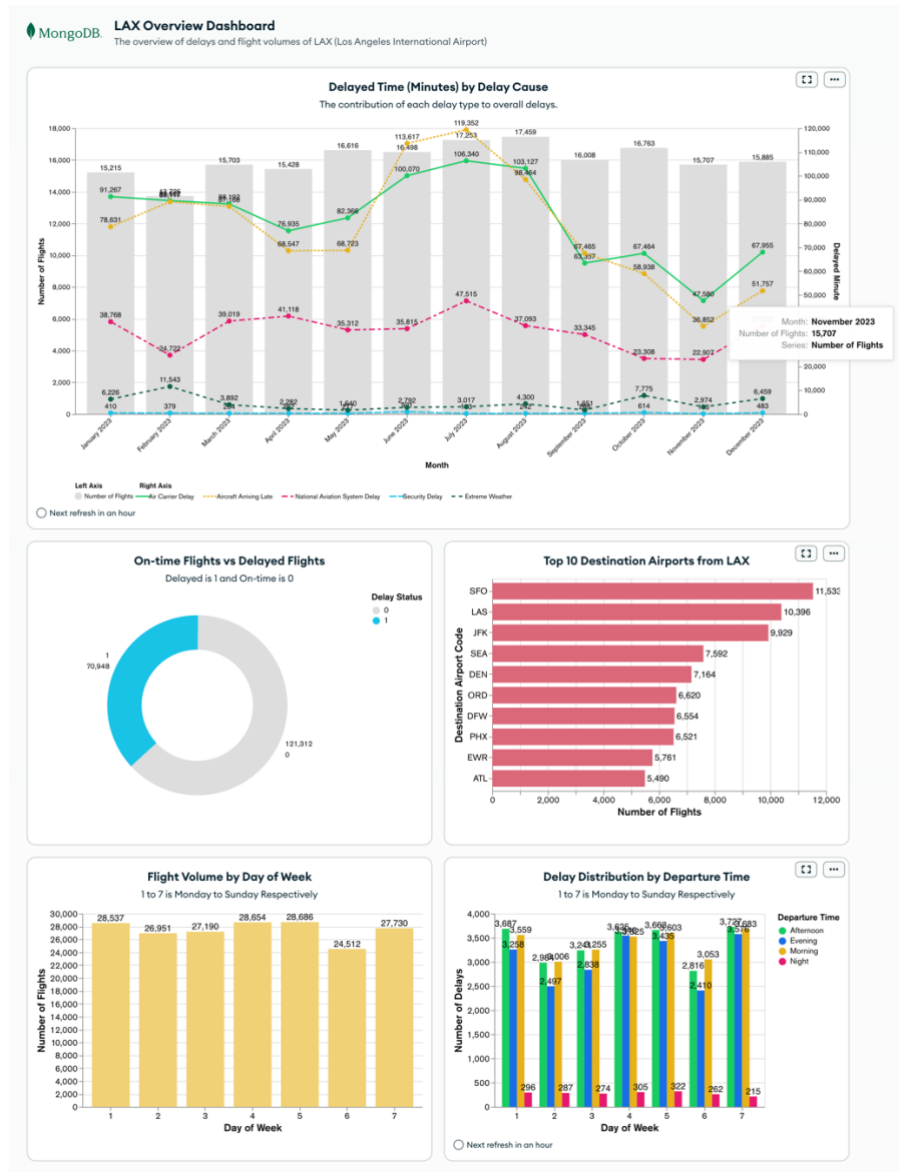


Figure 9. Example View of Overview Dashboard for Los Angeles International Airport (LAX)

The project contains 6 dashboards in total, with 1 main dashboard for the overview of the delay, and 5 additional dashboards for capturing the relationships between other factors to delay.

Table 13. Visualization Dashboards Description

Dashboard	Charts	Purpose	Demo Link
Overview Dashboard	<ul style="list-style-type: none"> Delay Time by Delay Cause On-time Flights vs Delayed Flights Top 10 Destination Airports 	To provide a high-level summary of flight performance, highlighting	LAX Overview Dashboard

	<ul style="list-style-type: none"> Flight Volume by Day of Week Delay Distribution by Departure Time 	delays, flight volumes, and key destinations.	
Weather Impact Dashboard	<ul style="list-style-type: none"> Average Temperature vs. Delayed Flights Precipitation vs. Delayed Flights Windspeed vs. Delayed Flights 	To analyze the impact of weather factors (temperature, precipitation, windspeed) on flight delays.	LAX Weather Impact Dashboard
Time Series Dashboard	<ul style="list-style-type: none"> Delayed Flights by Months Flight Volume for Delays Temporal Trends for Temperature Temporal Trends of Precipitation 	To identify temporal trends in flight delays and weather conditions over time.	LAX Time Series Dashboard
Performance Benchmark	<ul style="list-style-type: none"> Airline Performance On-time and Delay Rates by Airline 	To evaluate and compare airlines based on their on-time performance and delay rates.	LAX Performance Benchmark
Geographical Insights	<ul style="list-style-type: none"> Flights Volume by Destination Delays by States 	To provide a spatial analysis of flight volumes and delays by destination and state.	LAX Geographical Insight
Aircraft Analysis Dashboard	<ul style="list-style-type: none"> Average Delay by Aircraft Manufacturer Aircraft Age vs. Delays 	To assess the relationship between aircraft characteristics (manufacturer and age) and delay patterns.	LAX Aircraft Analysis Dashboard

3. Conclusion

3.1. Summary of Work

This project focused on mitigating flight delays by leveraging big data analytics and weather forecasting. The team developed a comprehensive solution to predict flight delays using historical and real-time data, with the aim of improving operational efficiency and passenger experiences.

Key achievements include the integration and preprocessing of datasets such as historical flight delays, weather reports, and geolocation data to build a unified dataset. A real-time data pipeline was implemented to ingest and process weather data from the OpenWeatherMap API using AWS services like EC2, Kinesis, and Lambda. Machine learning models, including Logistic Regression, Decision Tree, Random Forest, and Gradient Boosting, were trained and evaluated for predictive accuracy, with Gradient Boosting achieving the best results at 88.4% accuracy.

The project also developed six interactive dashboards using MongoDB Charts to visualize critical insights, such as delay causes, weather impacts, and airline performance. The solution was deployed on a scalable AWS cloud

architecture, ensuring robust and efficient data processing. This work highlights the successful application of big data analytics and machine learning to address persistent challenges in the aviation industry.

3.2. Challenges and Limitations

While the project benefited from comprehensive datasets, there were still challenges to address. One notable issue was ensuring the compatibility of various datasets. Historical flight data, weather data, and geolocation data came from different sources, requiring meticulous preprocessing to align formats, merge datasets, and ensure consistency. This step was critical but time-intensive.

Although the datasets were largely complete, the predictive models had to contend with class imbalance, as the number of on-time flights significantly exceeded delayed flights. This imbalance impacted certain evaluation metrics, such as recall, which could be improved further with advanced balancing techniques.

Another challenge was the system's real-time data processing. Despite functioning efficiently for most operations, occasional latency in the data streaming pipeline posed a challenge during peak processing periods. This issue highlights an opportunity to optimize the data ingestion process further.

Lastly, the scope of real-time data was somewhat constrained by reliance on the OpenWeatherMap API. While the API provided valuable weather metrics, it lacked detailed information on air traffic and airport capacity, which are critical variables in flight delay predictions. Addressing these gaps would require integrating additional real-time data sources in the future.

3.3. Future Work

Building on the success of this project, several opportunities for future development have been identified. Enhancing model accuracy remains a priority, with plans to experiment with advanced algorithms like XGBoost or neural networks to capture complex patterns and improve predictive performance. Addressing class imbalance using techniques like SMOTE could also refine model outputs.

Incorporating additional data sources, such as air traffic data, airport capacity metrics, and airline-specific operations, would further enhance prediction reliability. Optimizing real-time pipelines to reduce latency, potentially by exploring tools like Apache Kafka, is another avenue for improvement.

Expanding visualization capabilities to include more granular insights, such as passenger-specific metrics or predictive tools for airport operations, would also provide added value. In terms of scalability, transitioning the prototype into a production-ready system capable of handling global flight data is a critical next step, alongside ensuring compliance with aviation data privacy regulations.

Finally, integrating advanced weather analytics, such as long-term climate patterns and extreme weather forecasts, would further solidify the system's ability to mitigate flight delays. These enhancements will ensure the solution remains relevant and impactful in addressing the challenges of modern aviation.

4. References

- [1] S. Fleming, "National Airspace System: DOT and FAA Actions Will Likely Have a Limited Effect on Reducing Delays during Summer 2008 Travel Season," U.S. Government Accountability Office, Testimony before the Subcommittee on Aviation Operations, Safety, and Security, Committee on Commerce, Science, and Transportation, U.S. Senate, Jul. 15, 2008.
- [2] U.S. Bureau of Transportation Statistics, "TranStats: Home Drill Chart," [Online]. Available: https://www.transtats.bts.gov/homedrillchart.asp?utm_source=aaamwg&utm_medium=text&utm_content=us_bureau_of_transportation. [Accessed: Jan. 11, 2025].
- [3] J. Rapajic, Beyond Airline Disruptions. Aldershot, UK: Ashgate Publishing, 2009, p. 16, ISBN 9780754674405.
- [4] AirHelp, "The Economic Cost of Air Travel Disruptions," [Online]. Available: https://img.airhelp.com/Documents/AH_disruption_economic_cost.pdf?updatedAt=1695047330127. [Accessed: Jan. 11, 2025].
- [5] ClaimFlights, "Impact of Flight Delays," [Online]. Available: <https://claimflights.com/impact-of-flight-delays/>. [Accessed: Jan. 11, 2025].
- [6] Federal Aviation Administration, "Traffic Flow Management System (TFMS)," [Online]. Available: https://aspm.faa.gov/aspmhelp/index/Traffic_Flow_Management_System_%28TFMS%29.html. [Accessed: Jan. 11, 2025].
- [7] J. Czakon, "24 Evaluation Metrics for Binary Classification (And When to Use Them)," *neptune.ai*, Oct. 22, 2024. <https://neptune.ai/blog/evaluation-metrics-binary-classification> [Accessed: Jan. 11, 2025].

Appendices




Appendix A. Declaration of Project Team Member

Declaration of Project Team Member

Project Team 1

Full Name	Student ID	Contribution to the project (%)	Outline how this person contributes to the outcomes of the project
Phan Nhat Minh	S3978598	33.333%	Minh is responsible for planning the project by outlining milestones and overseeing the team's progress to ensure alignment with objectives. Key tasks involve managing documentation by structuring reports and contributing to their content. Additionally, Minh is responsible for building pipelines for data visualization, designing user-friendly dashboards, and supporting team members in debugging and development. These combined efforts ensure efficient project execution and successful delivery of results.
Tran Manh Cuong	s3974735	33.333%	Cuong is responsible for designing the product's infrastructure and managing cloud services to ensure scalability and reliability. Key tasks involve processing streaming data, building pipelines to efficiently fetch and load it, and contributing to the project report by writing specific sections. Additionally, Cuong involves supporting team members in debugging and development, ensuring smooth progress and collaboration throughout the project.
Nguyen Vinh Gia Bao	s3986287	33.333%	Bao is responsible for designing and building the ETL pipeline for raw historical data, ensuring efficient data extraction, transformation, and loading processes. Additional tasks involve creating a pipeline for processing data used in model training, overseeing the training process, and taking accountability for the model's performance and outcomes. Bao also includes contributing to the project report by writing specific sections and providing support to team members in debugging and development to maintain smooth project progress.

We state that all information provided in this form is true and correct.

Group member's name & signature		Date
Minh Phan		12 January 2025
Cuong Tran		12 January 2025
Bao Nguyen		12 January 2025

Appendix B. Cost Estimation

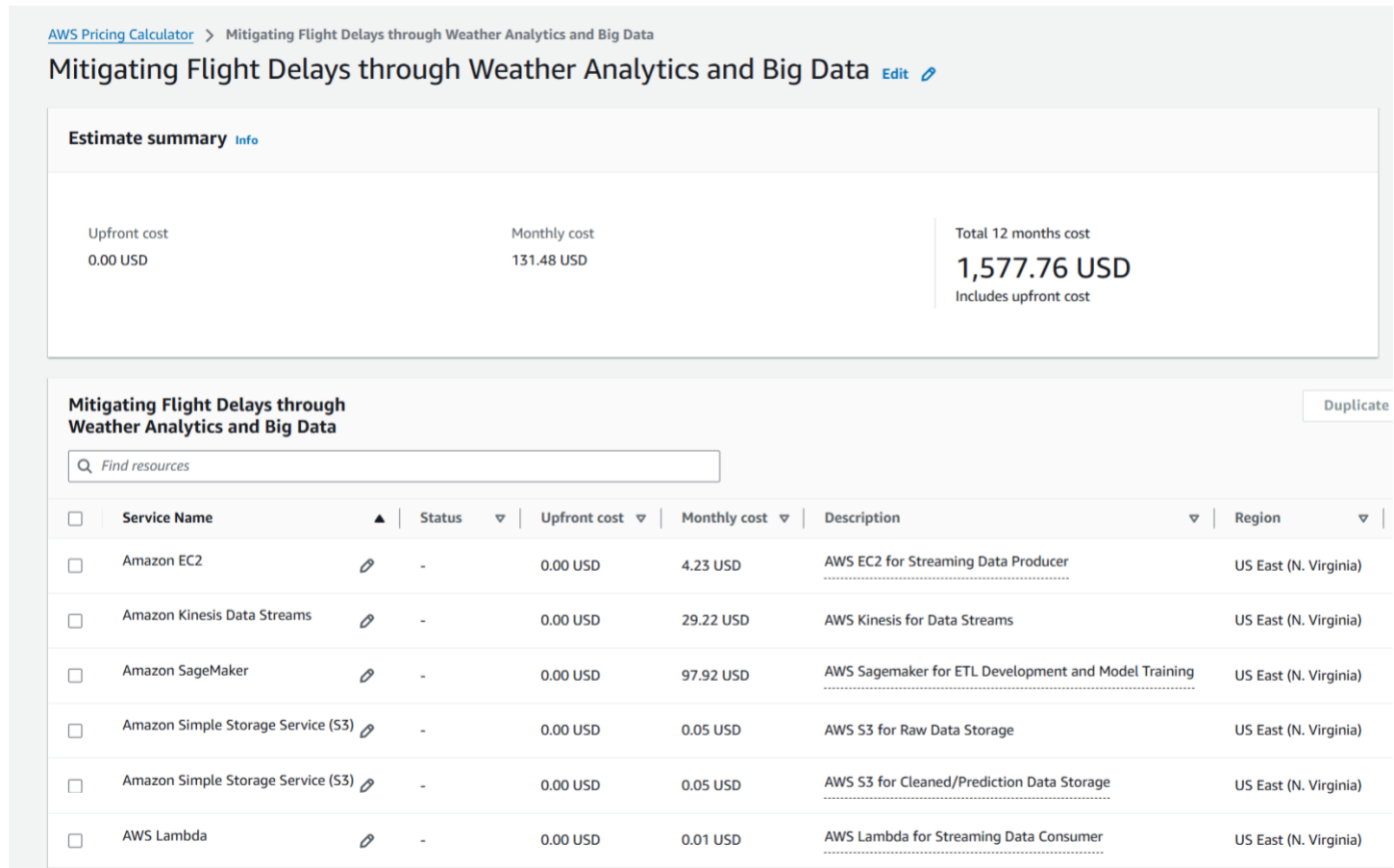


Figure B.1. Cost Estimation on AWS Pricing Calculator

Appendix C. Visualization Dashboards

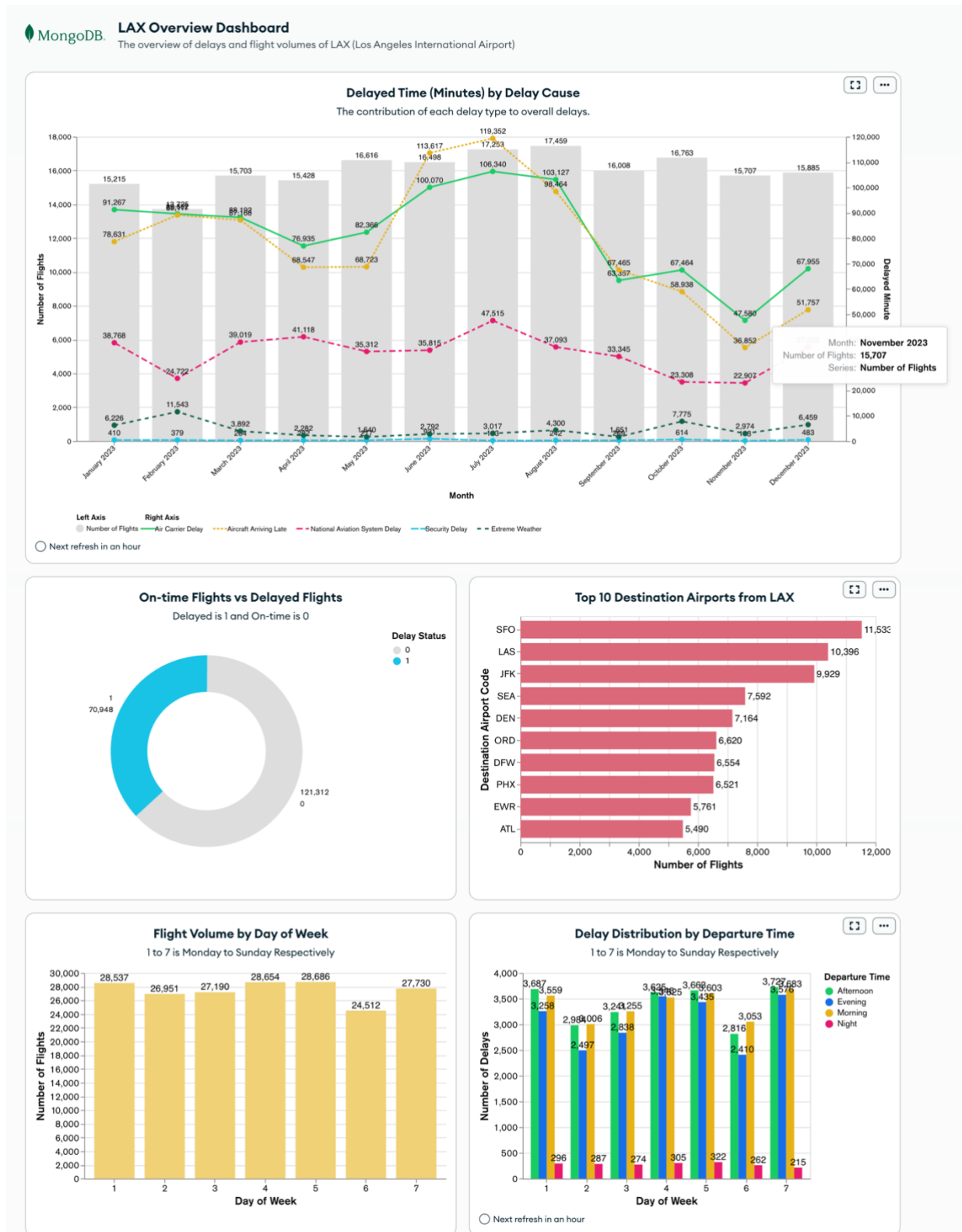


Figure C.1. Example View of Overview Dashboard for Los Angeles International Airport (LAX)

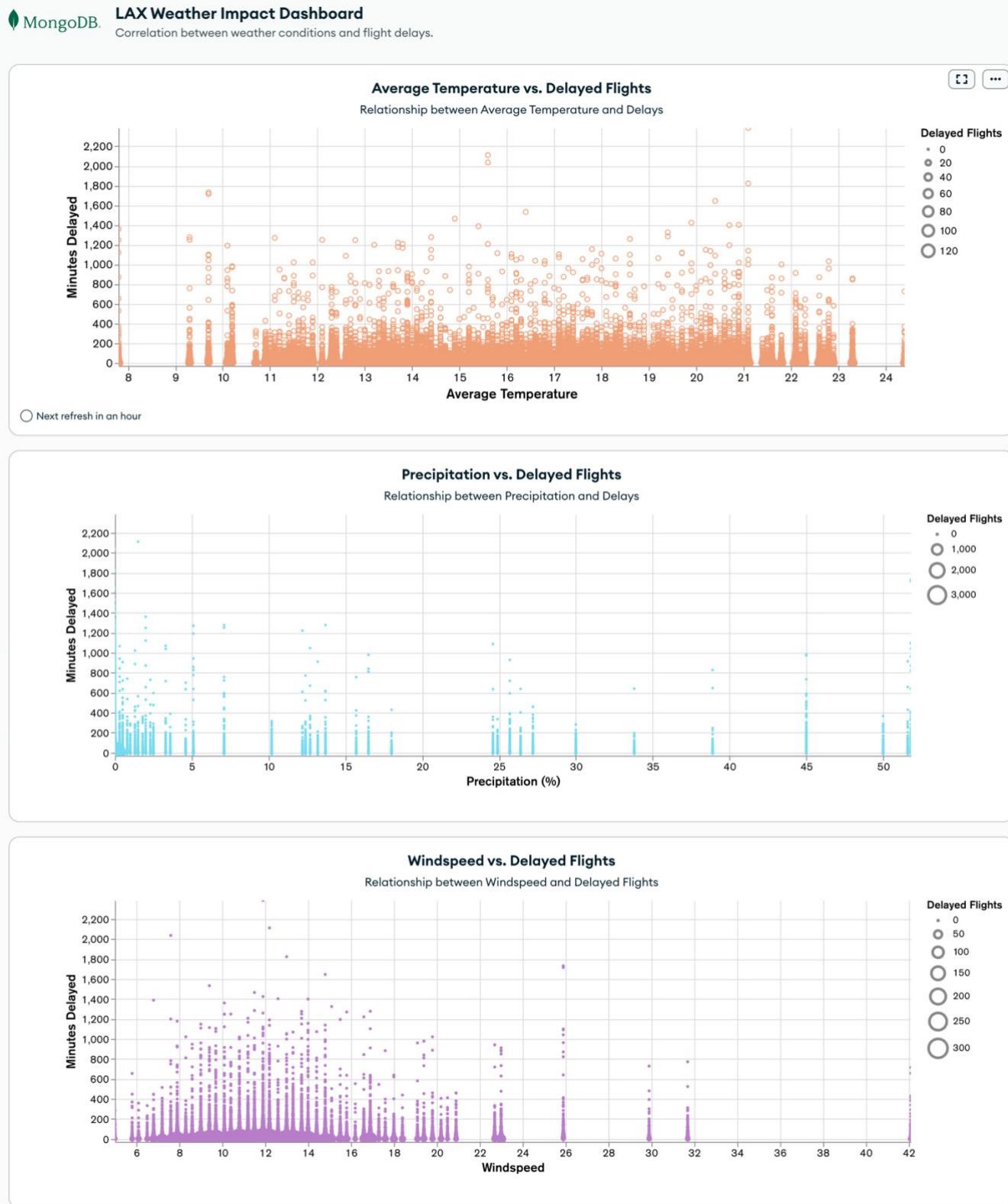


Figure C.2. Example View of Weather Impact Dashboard for Los Angeles International Airport (LAX)

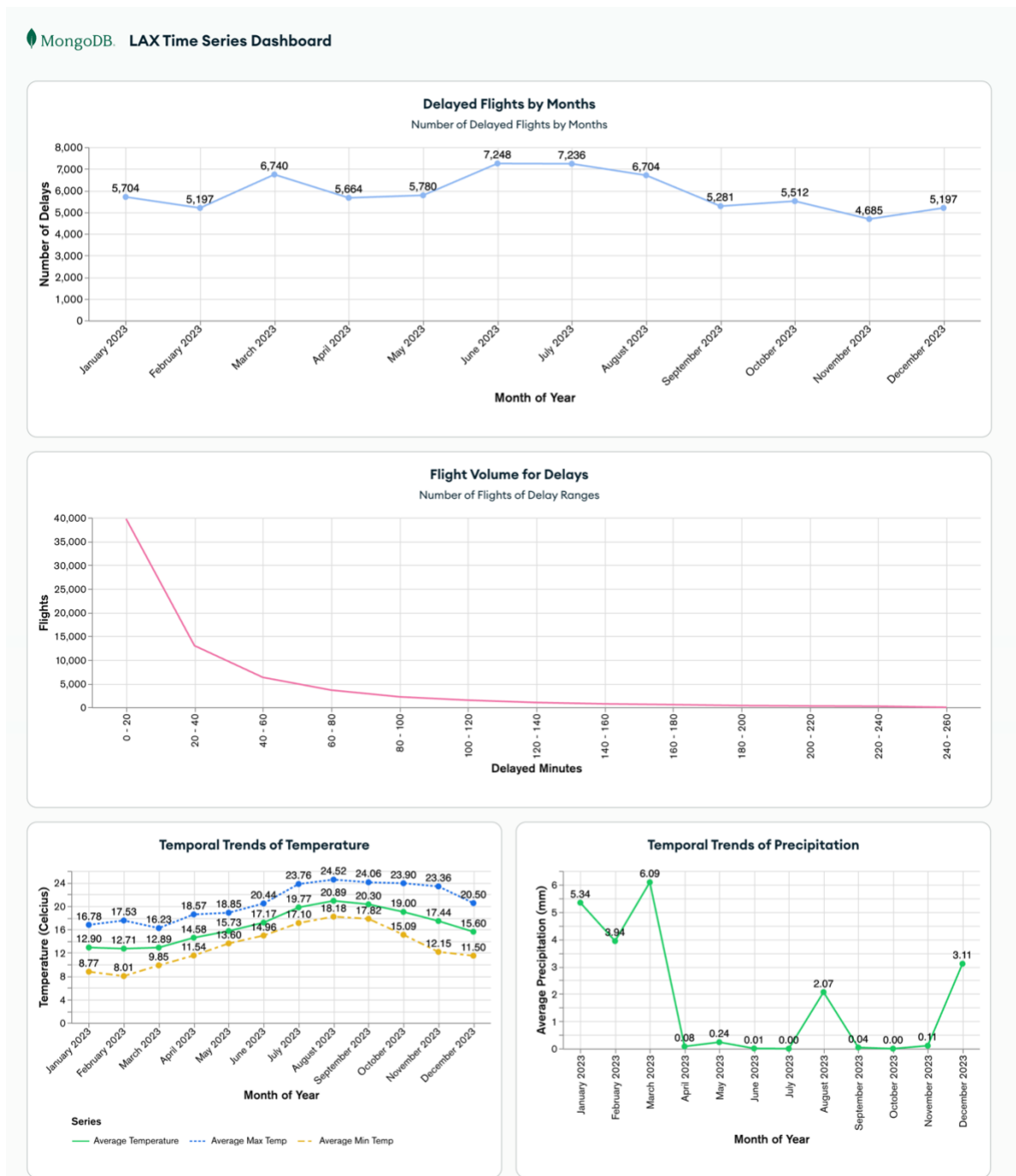


Figure C.3. Example View of Time Series Dashboard for Los Angeles International Airport (LAX)

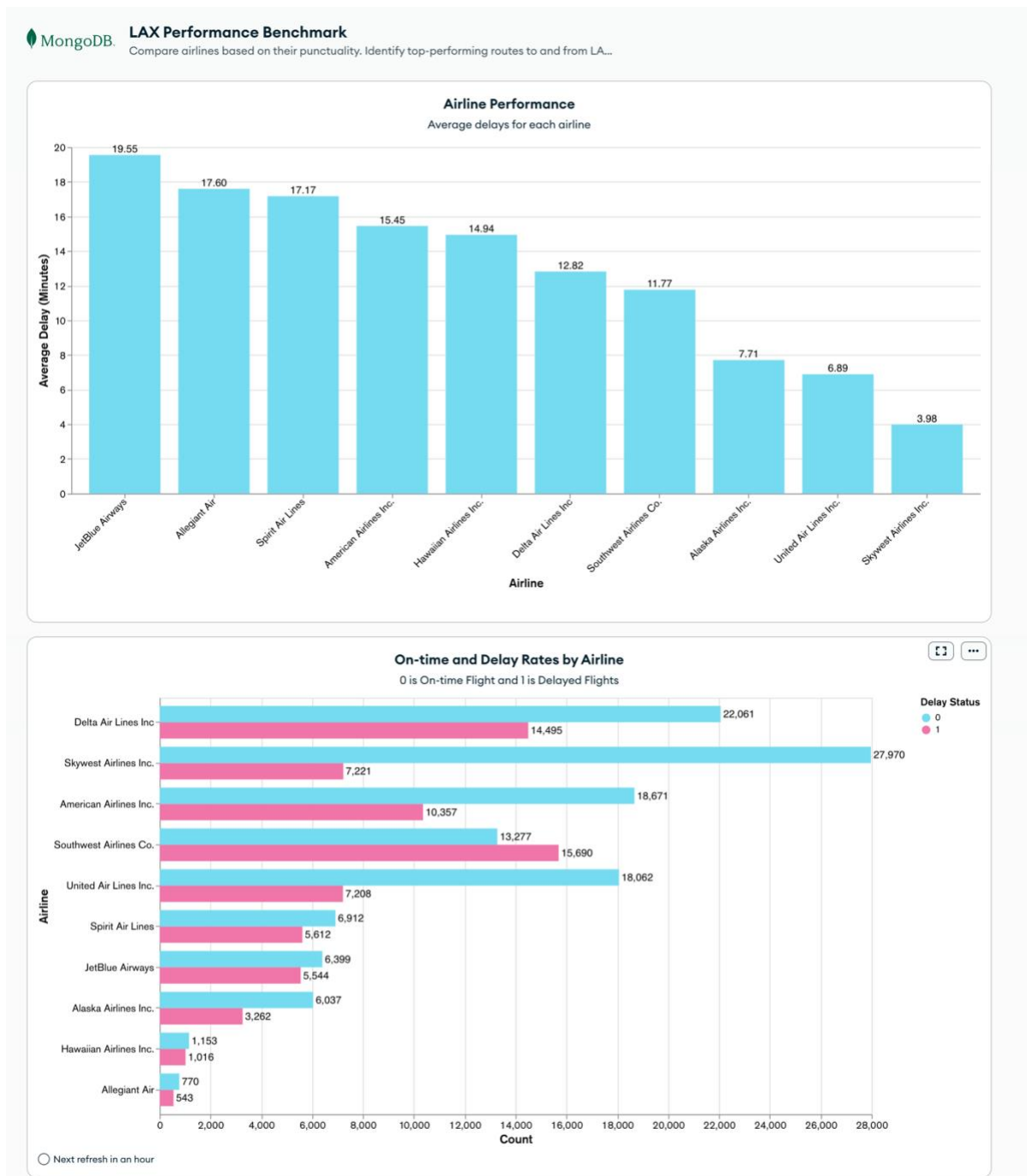


Figure C.4. Example View of Performance Benchmark for Los Angeles International Airport (LAX)

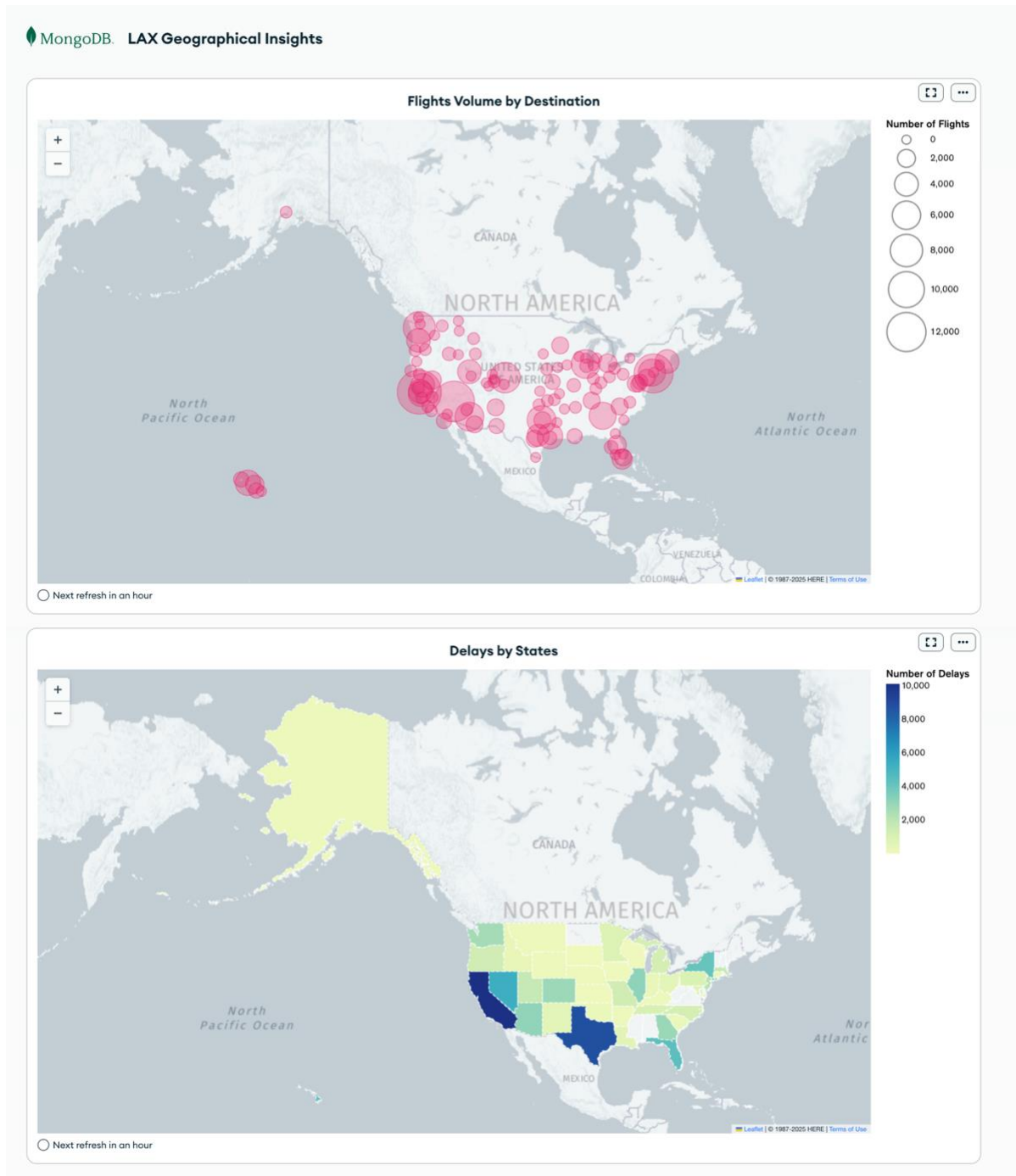


Figure C.5. Example View of Geographical Insights Dashboard for Los Angeles International Airport (LAX)

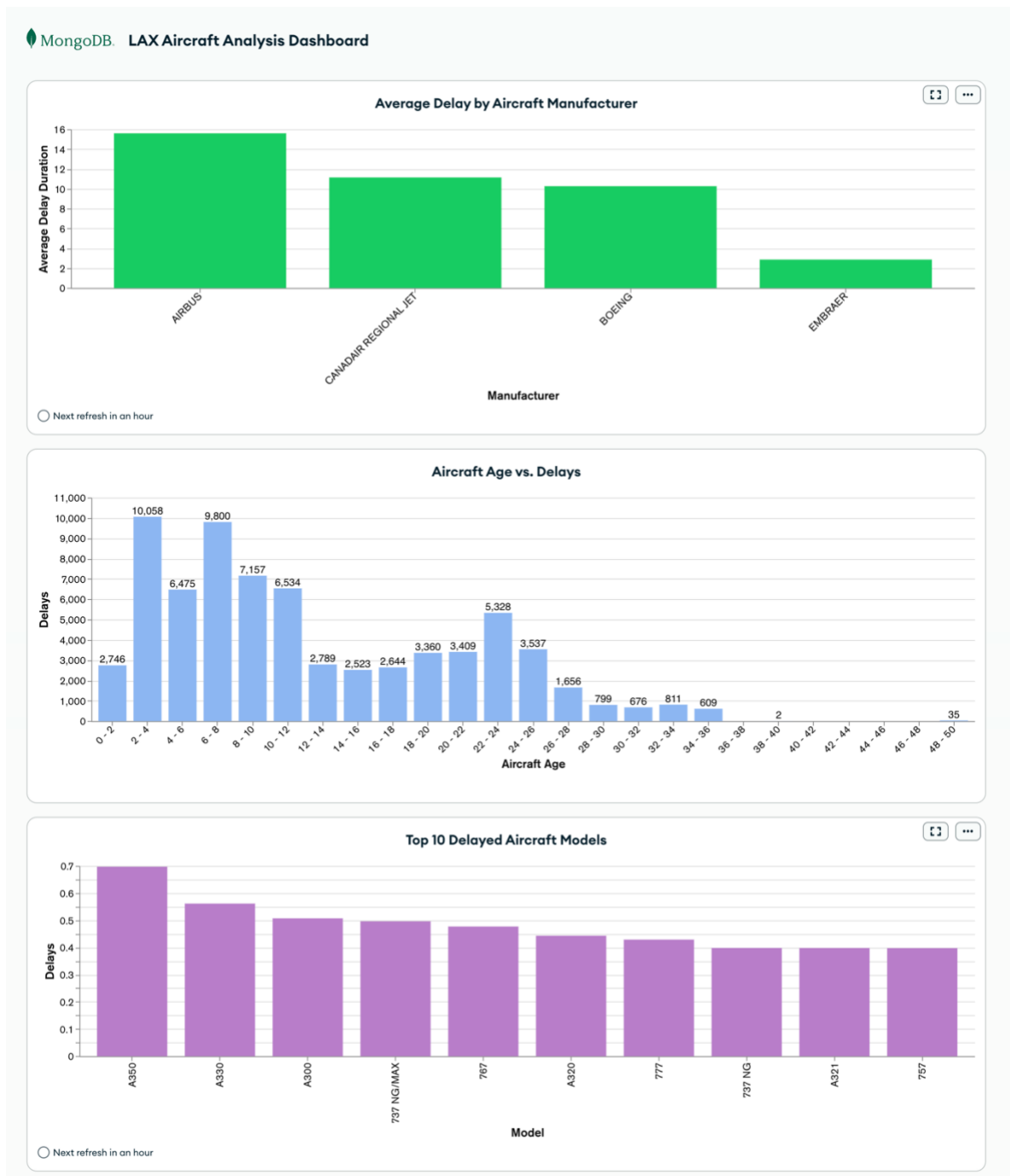


Figure C.6. Example View of Aircraft Analysis Dashboard for Los Angeles International Airport (LAX)

Appendix D. Streaming Data Pipeline Process

```

154 # Send data to Kinesis
155 def put_records(stream_name, records):
156     for record in records:
157         try:
158             response = kinesis_client.put_record(
159                 StreamName=stream_name,
160                 Data=json.dumps(record),
161                 PartitionKey=record["airport_id"]
162             )
163             log_message(f"Record sent to Kinesis. SequenceNumber: {response['SequenceNumber']}")
164             print(f"Record details: ")
165             print(json.dumps(record))
166         except Exception as e:
167             log_message(f"Failed to send record to Kinesis: {e}")
168
169 # Main logic
170 if __name__ == "__main__":
171     airports = ["ATL", "LAX", "JFK", "LGA", "MDW", "ORD"]
172     while True:
173         # Fetch and send daily weather data
174
175
python3 -ip-172-31-94-2x
[2025-01-12 21:45:28] Record sent to Kinesis. SequenceNumber: 49659360467528345188417944274882995888446087511324753922
Record details:
{"time": "2025-01-14", "tavg": -9.45, "tmin": -11.23, "tmax": -7.67, "prcp": 0, "snow": 0, "wdir": 300, "wspd": 21.92, "pres": 1028, "airport_id": "ORD"}
[2025-01-12 21:45:28] Record sent to Kinesis. SequenceNumber: 496593604675283451884179442748805413740085316769674166274
Record details:
{"time": "2025-01-15", "tavg": -7.03, "tmin": -9.34, "tmax": -4.71, "prcp": 0, "snow": 0, "wdir": 210, "wspd": 20.38, "pres": 1032, "airport_id": "ORD"}
[2025-01-12 21:45:28] Record sent to Kinesis. SequenceNumber: 496593604675283451884179442748807831591724546028023578626
Record details:
{"time": "2025-01-16", "tavg": -2.29, "tmin": -6.41, "tmax": 1.84, "prcp": 0, "snow": 0, "wdir": 251, "wspd": 25.81, "pres": 1014, "airport_id": "ORD"}
[2025-01-12 21:45:28] Record sent to Kinesis. SequenceNumber: 49659360467528345188417944274811458369183389915547697154
Record details:
{"time": "2025-01-17", "tavg": 0.88, "tmin": -2.06, "tmax": 3.82, "prcp": 0, "snow": 0, "wdir": 200, "wspd": 23.44, "pres": 1007, "airport_id": "ORD"}
[2025-01-12 21:45:28] Record sent to Kinesis. SequenceNumber: 49659360467528345188417944274813876220822619173897189506
Record details:
{"time": "2025-01-18", "tavg": -4.15, "tmin": -9.14, "tmax": 0.84, "prcp": 0, "snow": 0, "wdir": 329, "wspd": 27.07, "pres": 1017, "airport_id": "ORD"}
[2025-01-12 21:45:28] Record sent to Kinesis. SequenceNumber: 4965936046752834518841794427481992084920869231978648386
Record details:
{"time": "2025-01-19", "tavg": -12.05, "tmin": -14.21, "tmax": -9.88, "prcp": 0, "snow": 0, "wdir": 298, "wspd": 24.59, "pres": 1026, "airport_id": "ORD"}

```

Figure D.1. Code for Streaming Data Producer on AWS Cloud9

Partition key	Data	Sequence number
ATL	{"time": "2025-01-12", "tavg": -0.62, "tmin": -5.05...	49659360467528345188417944274666387270829634277144002562
ATL	{"time": "2025-01-13", "tavg": 4.64, "tmin": 0.86, "...	49659360467528345188417944274667596196649248906318708738
ATL	{"time": "2025-01-14", "tavg": 3.66, "tmin": -0.6, "...	49659360467528345188417944274671222974108092793842827266
ATL	{"time": "2025-01-15", "tavg": 4.03, "tmin": -0.11, ...	49659360467528345188417944274672431899927707423017533442
ATL	{"time": "2025-01-16", "tavg": 5.88, "tmin": 0.91, ...	49659360467528345188417944274676058677386551310541651970
ATL	{"time": "2025-01-17", "tavg": 8.41, "tmin": 3.05, ...	49659360467528345188417944274682103306484624456415182850
ATL	{"time": "2025-01-18", "tavg": 7.52, "tmin": 6.34, ...	49659360467528345188417944274683312232304239085589889026
ATL	{"time": "2025-01-19", "tavg": 3.97, "tmin": -1.61, ...	49659360467528345188417944274684521158123853714764595202
LAX	{"time": "2025-01-12", "tavg": 15.43, "tmin": 11.9...	49659360467528345188417944274686939009763082973114007554
LAX	{"time": "2025-01-13", "tavg": 14.72, "tmin": 10.7...	49659360467528345188417944274690565787221926860638126082
LAX	{"time": "2025-01-14", "tavg": 14.78, "tmin": 10.5...	49659360467528345188417944274691774713041541489812832258

Figure D.2. Data Viewer Tab on AWS Kinesis Streams

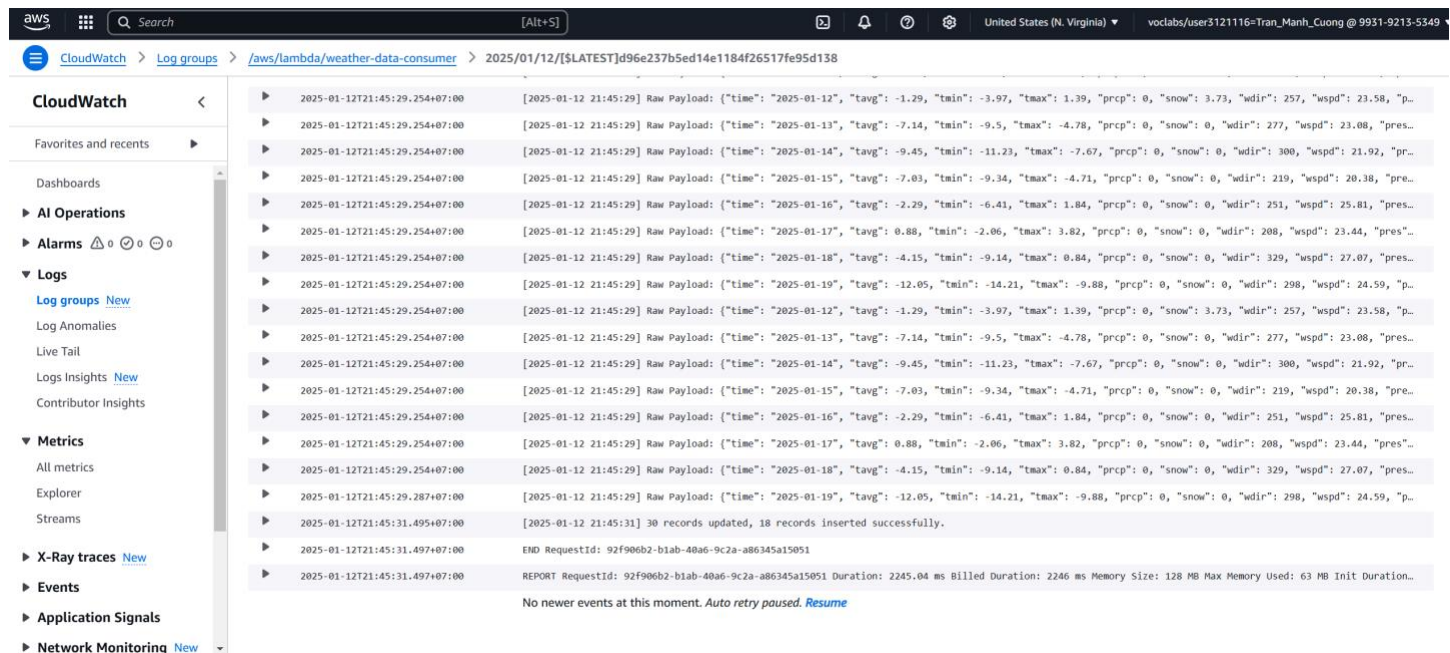


Figure D.3. Streaming Data Consumer Monitoring on AWS CloudWatch