National College of Ireland

Configuration Manual

MSc Research Project
MSc in Cloud Computing

Atharav Deshpande
Student ID: 23269065

School of Computing
National College of Ireland

Supervisor: Mr. Punit Gupta

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: …Atharav Jayant Deshpande….

Student ID: …23269065……

Programme: …MSc in Cloud Computing…… **Year:** ……2024-2025…..

Module: …Research Project……..……

Lecturer: …Mr. Punit Gupta……..……

Submission Due Date: …15/09/2025……..……

Project Title: …Real-Time Data Processing in Cloud Environments……..……

Word Count: …1536…… **Page Count:** …6……..……..……

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: …Atharav Jayant Deshpande……

Date: …15/09/2025……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Atharav Deshpande
Student ID: 23269065

1 Introduction

This manual provides step-by-step instructions for the deployment and configuration of the Real-Time Data Processing Pipeline on Amazon Web Services (AWS). The objective of this document is to guide a user with appropriate permissions through the process of setting up the entire architecture as described in the main research project.

The resulting system is an end-to-end, serverless data pipeline designed to ingest, process, store, and analyze a continuous stream of financial market data. The core AWS services utilized in this configuration include AWS IAM, S3, Kinesis Data Streams, Lambda, Glue, Athena, QuickSight, CloudWatch, and SNS.

Following the successful completion of all steps outlined in this guide, a fully functional and operational instance of the data processing pipeline will be deployed, complete with a custom monitoring dashboard for observing system health and performance.

2 Prerequisites

Before beginning the configuration process, ensure the following prerequisites are met:

- **AWS Account:** An active AWS account with administrative privileges to create and configure the necessary resources.
- **AWS Region:** All resources must be created in the same AWS region. This guide uses **eu-north-1 (Stockholm)**.
- **Source Code and Data:**
  - Access to the project's source code files:
    - combined_csv_script.py
    - data_simulation.py
    - lambda_function.py
    - app.py (for the monitoring dashboard)
    - dashboard.html (and other static assets)
  - The "NIFTY50 Stock Market Data" dataset downloaded from Kaggle.
- **Local Environment:**
  - Python 3.8 or higher installed.
  - The AWS Command Line Interface (CLI) installed and configured.
  - The following Python libraries installed (can be installed via pip):
    - pandas
    - boto3
    - flask
    - plotly
    -

3 Step-by-Step Configuration

This section details the sequential steps required to build the pipeline.

3.1 Security Setup: IAM Roles and Users

1. **Create IAM User for Data Producer:**
   - Navigate to the IAM console in AWS.

- Select "Users" and click "Add users".
- Set the User name to kinesisuser.
- Select "Access key - Programmatic access" as the credential type.
- Proceed to permissions. Select "Attach policies directly".
- Click "Create policy". In the JSON editor, paste a policy that grants access only to the Kinesis stream. For initial setup, AmazonKinesisFullAccess can be used.
- Complete the user creation process. **Crucially, download or copy the Access Key ID and Secret Access Key. This is the only time they will be visible.**
- Configure the local AWS CLI with these credentials using the command: aws configure --profile kinesisuser.

2. **Create IAM Role for AWS Services:**
   - In the IAM console, select "Roles" and click "Create role".
   - For the trusted entity type, select "AWS service". For the use case, select "Lambda" and proceed.
   - Attach the following AWS managed policies:
     - AmazonKinesisFullAccess
     - AmazonS3FullAccess
     - AmazonDynamoDBFullAccess (if using DynamoDB)
     - CloudWatchFullAccess
     - AWSGlueConsoleFullAccess
   - Name the role KinesisAnalyticsRole and create it.
   - Open the newly created role and navigate to the "Trust relationships" tab. Click "Edit trust policy" and update the JSON to allow multiple services to assume this role. The policy should include principals for Lambda, Glue, and QuickSight.

3.2 Data Preparation

1. Place all downloaded CSV files from the Kaggle dataset into a single directory named dataset/.
2. Execute the combined_csv_script.py script from the terminal: python combined_csv_script.py.
3. This will generate a file named consolidated_data.csv in the root directory.

3.3 Core Infrastructure Provisioning

1. **Create S3 Bucket:**
   - Navigate to the S3 console.
   - Click "Create bucket".
   - Set the Bucket name to financial-market-data-archive.
   - Ensure the region is set to eu-north-1.
   - Keep all other settings as default and create the bucket.
   - Inside the new bucket, create two folders: processed/ and athena-results/.
2. **Create Kinesis Data Stream:**
   - Navigate to the Kinesis console.
   - Click "Create data stream".
   - Set the Data stream name to financial-market-data-stream.
   - For "Data stream capacity", select **On-demand**.
   - Create the data stream.

3.4 Lambda Function for Real-Time Processing

1. Navigate to the AWS Lambda console.
2. Click "Create function".

3. Select "Author from scratch".
4. Set the Function name to ProcessFinancialStream.
5. Set the Runtime to **Python 3.12**.
6. Under "Permissions", choose "Use an existing role" and select the KinesisAnalyticsRole created earlier.
7. Create the function.
8. In the function's "Code source" editor, replace the default code with the contents of lambda_function.py.
9. In the "Configuration" tab, go to "Environment variables". Add a new variable:
    - Key: BUCKET_NAME
    - Value: financial-market-data-archive
10. Save the changes.
11. In the "Configuration" tab, select "Triggers" and click "Add trigger".
12. Choose "Kinesis" as the source. Select the financial-market-data-stream. Keep the other settings as default and add the trigger.

3.5 Pipeline Test: Running the Data Producer

1. Open the data_simulation.py script and verify that the stream_name and region_name variables match the configuration.
2. From the terminal, run the producer script: python data_simulation.py.
3. **Verification:**
    - Check the CloudWatch Logs for the ProcessFinancialStream Lambda function. New log entries should appear.
    - Navigate to the financial-market-data-archive S3 bucket. New JSON files should be appearing inside the processed/ folder.
4. Stop the script (Ctrl+C) after verification.

3.6 Analytics Layer: Glue and Athena

1. **Create Glue Database:**
    - Navigate to the AWS Glue console.
    - In the left menu, select "Databases" and click "Add database".
    - Set the Database name to financial_data_db and create it.
2. **Create Glue Crawler:**
    - In the left menu, select "Crawlers" and click "Create crawler".
    - Name the crawler FinancialDataCrawler.
    - For the data source, choose S3 and specify the path: s3://financial-market-data-archive/processed/.
    - For the IAM Role, select the existing KinesisAnalyticsRole.
    - For the target database, choose financial_data_db.
    - Complete the creation process.
    - Select the new crawler and click "Run crawler". Wait for it to complete. A new table named processed should now exist in the database.
3. **Configure and Test Athena:**
    - Navigate to the Athena console.
    - In the "Settings" tab, set the "Query result location" to the S3 path: s3://financial-market-data-archive/athena-results/.
    - In the "Editor" tab, select the financial_data_db database.
    - Execute a test query to verify functionality:

```
SELECT * FROM "processed" LIMIT 10;
```

    - The query should return 10 rows from the processed data.

3.7 Visualization Layer: QuickSight

1. Navigate to the QuickSight console and complete the sign-up process if necessary, ensuring the region is eu-north-1.
2. In QuickSight, manage datasets and click "New dataset".
3. Choose "Athena" as the data source. Name the data source FinancialMarketData and validate the connection.
4. Select the financial_data_db catalog and the processed table.
5. Choose to "Directly query your data" or import to "SPICE". Select SPICE for better performance.
6. Once the dataset is created, create a new analysis and build sample visualizations (e.g., a line chart of closeprice over timestamp).

3.8 Monitoring Dashboard Deployment

1. Open the app.py file and confirm the AWS resource names at the top match the configured names.
2. In the terminal, navigate to the project directory containing app.py and dashboard.html.
3. Run the Flask application: python app.py.
4. Open a web browser and navigate to http://127.0.0.1:5000.
5. The monitoring dashboard should load, displaying live metrics from the AWS pipeline.

4 Conclusion

Upon completing all the steps in this manual, a fully configured, end-to-end real-time data processing pipeline is deployed on AWS. The system is now capable of ingesting simulated financial data via Kinesis, processing it with Lambda, storing it durably in an S3 data lake, and making it available for analysis through Athena and QuickSight.

The custom monitoring dashboard provides a real-time, consolidated view of the pipeline's operational health. This deployed system serves as a functional artifact that can be used for further experimentation, development, or as a foundational blueprint for production-grade real-time analytics solutions.