CSCI 5410 Serverless Data Processing

Assignment 3 Report

Prepared by:
Vikram Venkatapathi - B00936916

Master of Applied Computer Science (Summer'23)
Faculty of Computer Science
Dalhousie University

GitLab Repo link : **https://git.cs.dal.ca/vikramv/csci5410-summer-23-b00936916/-/tree/A3**

Contents

Part I.

PART A

Explore & Build a Use Case

1. Introduction

As a former cashier at the Atlantic Superstore, I marveled at the complexities of inventory management. This inspired me to envision a scenario where AWS Kinesis revolutionizes the process, ensuring precise stock numbers and minimizing wastage. In this hypothetical scenario, real-time data streams from IoT devices and point-of-sale systems are leveraged to optimize inventory, prompt reordering, and deliver an exceptional shopping experience for customers.

2. Hypothetical Scenario and the Use Case

Hypothetical Scenario: Smart Inventory Management at Atlantic Superstore

In this hypothetical scenario, the Atlantic Superstore, a large retail chain, faces challenges with manual inventory management, leading to inefficiencies and occasional stockouts. The store envisions implementing an intelligent inventory management system using AWS Kinesis and other AWS services to address these issues.

Usecase:

The Atlantic Superstore decides to leverage AWS Kinesis for real-time data streaming to capture continuous updates from IoT devices and point-of-sale systems placed strategically across the store. These devices will transmit product movement, stock levels, and sales data in real-time to AWS Kinesis Data Streams. With AWS Kinesis Data Analytics, the store processes and analyzes the streaming data on the fly. Advanced analytics and machine learning algorithms are applied to forecast demand, identify trending products, and detect potential stock shortages. The system can instantly update inventory records and provide a comprehensive view of stock levels. To ensure efficient stock management, the Atlantic Superstore integrates AWS Lambda functions into the system. These functions automatically trigger reordering processes for low-stock items, streamlining inventory replenishment and preventing stockouts. Additionally, the store incorporates AWS Comprehend for sentiment analysis to understand customer preferences and feedback. By integrating customer sentiment with sales data, the store can optimize its inventory based on real-time demand and adapt to changing customer needs. The smart inventory management system using AWS Kinesis allows the Atlantic Superstore to optimize inventory levels, reduce wastage, and deliver a seamless shopping experience for its customers. With real-time insights and automated processes, the store can maintain accurate stock numbers, minimize manual efforts, and ensure products are readily available for customers, ultimately driving increased customer satisfaction and operational efficiency.

2.1. AWS Architecture diagram

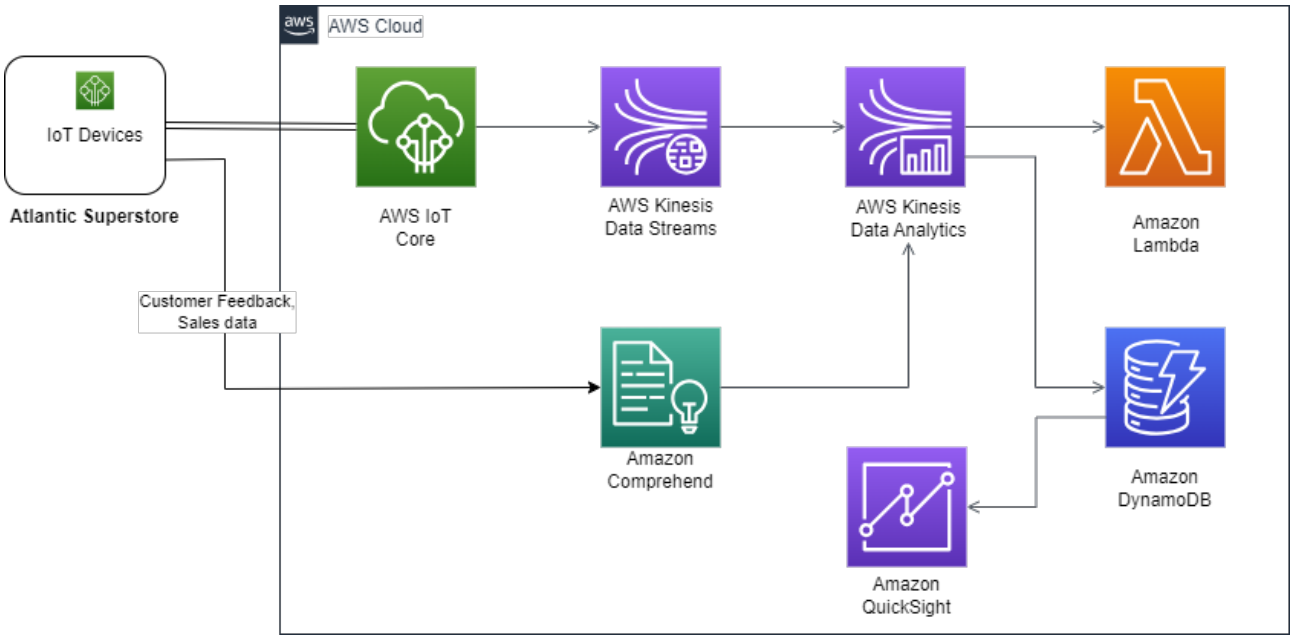

Figure 2.1.: ***Smart Inventory Management Architecture at Atlantic Superstore using AWS Services***

1. **IoT Devices**: These devices are strategically placed on shelves and equipped with sensors to detect product movement and measure stock levels.
2. **AWS IoT Core**: The IoT devices securely connect and transmit data to AWS Kinesis Data Streams, ensuring smooth communication with the cloud.
3. **AWS Kinesis Data Streams**: This service captures and streams real-time data from the IoT devices, including product IDs, timestamps, stock levels, and sales data.
4. **AWS Kinesis Data Analytics**: The streaming data is processed and analyzed in real-time to forecast demand, identify trending products, and detect potential stock shortages. It also triggers Amazon Lambda functions for automatic reordering and anomaly detection.
5. **Amazon Lambda**: Lambda functions automatically trigger reordering processes for low-stock items and flag any irregularities in stock movements.
6. **Amazon Comprehend**: This service is used for sentiment analysis, integrating customer feedback with sales data to understand customer preferences and optimize inventory accordingly.
7. **Amazon DynamoDB**: The analyzed data is stored in DynamoDB, a NoSQL database, offering fast and scalable access to historical inventory data.

8. **Amazon QuickSight**: The data in DynamoDB is visualized and analyzed through QuickSight, enabling the store to create interactive dashboards and gain real-time insights into stock levels, popular products, and inventory performance.

Part II.

PART B

Build an event-driven serverless application using AWS Lambda

3. Procedure followed for the given experiment

3.1. Flowchart

Part III.

PART C

Use AWS Lambda-SQS-SNS

4. Bot RideRequest

4.1. Actions taken to perform the bot creation operation

Part IV.

REFERENCES

References

[1] N. Naik, "Performance Evaluation of Distributed Systems in Multiple Clouds using Docker Swarm," 2021 *IEEE International Systems Conference (SysCon)*, Vancouver, BC, Canada, 2021, pp. 1-6, doi: 10.1109/SysCon48628.2021.9447123

[2] "Create standard repositories," *Google Cloud*. [Online]. Available: https://cloud.google.com/artifact-registry/docs/repositories/create-repos.[Accessed: 29 June 2023].

[3] "Deploying to cloud run," *Google Cloud*. [Online]. Available: https://cloud.google.com/run/docs/deploying. [Accessed: 29 June 2023].

[4] "Add data to cloud firestore," *Firebase*. [Online]. Available: https://firebase.google.com/docs/firestore/manage-data/add-data. [Accessed: 29 June 2023].

[5] "Get data with cloud firestore," *Firebase*. [Online]. Available: https://firebase.google.com/docs/firestore/query-data/get-data. [Accessed: 29 June 2023].

[6] "Containerize an application," *Docker Documentation*, 28-Jun-2023. [Online]. Available: https://docs.docker.com/get-started/02_our_app/. [Accessed: 29 June 2023].

[7] "Update the application," *Docker Documentation*, 28-Jun-2023. [Online]. Available: https://docs.docker.com/get-started/03_updating_app/. [Accessed: 29 June 2023].

[8] "Push and pull images," *Google Cloud*. [Online]. Available: https://cloud.google.com/artifact-registry/docs/docker/pushing-and-pulling. [Accessed: 29 June 2023].

[9] "Create a new react app," *Reactjs.org*. [Online]. Available: https://legacy.reactjs.org/docs/create-a-new-react-app.html. [Accessed: 29 June 2023].

[10] S.Gandotra, "ReactJS router," *GeeksforGeeks*, 13-Dec-2019. [Online]. Available: https://www.geeksforgeeks.org/reactjs-router/. [Accessed: 29 June 2023].

[11] "LaTeX Listings package JSON formatting," *TeX Stack Exchange*, Available: https://tex.stackexchange.com/questions/560830/latex-listings-package-json-formatting. [Accessed 29 June 2023].

[12] V. Venkatapathi, "B00936916_VikramVenkatapathi_A1_Report.pdf," *Dalhousie University*, Jan. 2023. [Online]. Available: https://git.cs.dal.ca/vikramv/csci5408_w23_b00936916_vikram_-venkatapathi/-/blob/main/Assignment_1/B00936916_VikramVenkatapathi_A1_Report.pdf. [Accessed 29 June 2023].

[13] Shanmuganathan, Vishakan. "OOAD-Project." *GitHub*, 2021, https://github.com/svishakan/OOAD-Project. [Accessed 29 June 2023].

[14] AWS. "Amazon Lex V1",*Amazon.com*[Online]. Available: https://docs.aws.amazon.com/lex/latest/dg/gs-bp-create-bot.html. [Accessed: 29 June 2023].

[15] "Swarm mode overview," *Docker Documentation*, 30-Jun-2023. [Online]. Available: https://docs.docker.com/engine/swarm/. [Accessed: 03 July 2023].

[16] R. Powell, "Docker Swarm vs Kubernetes: how to choose a container orchestration tool," *CircleCI*, 12-Oct-2021. [Online]. Available: https://circleci.com/blog/docker-swarm-vs-kubernetes/. [Accessed: 03 July 2023].

[17] "Production-Grade Container Orchestration," *Kubernetes*. [Online]. Available: https://kubernetes.io/. [Accessed: 03 July 2023].

[18] "Lightweight kubernetes," *K3s.io*. [Online]. Available: https://k3s.io/. [Accessed: 03 July 2023].