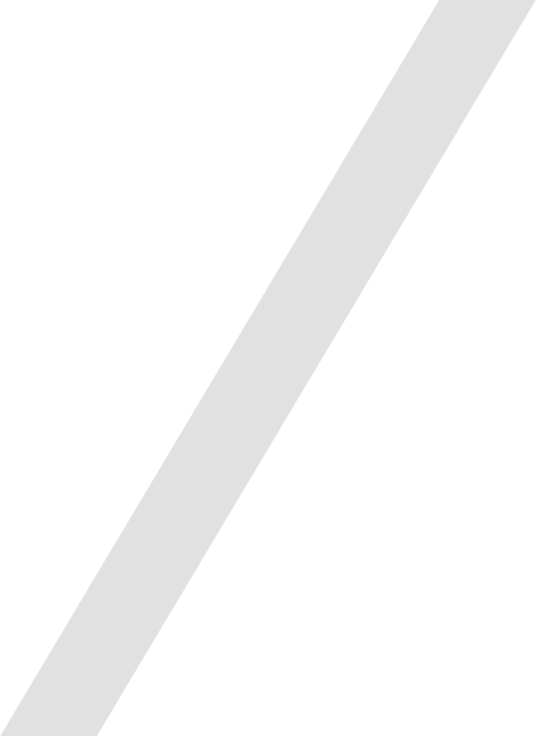
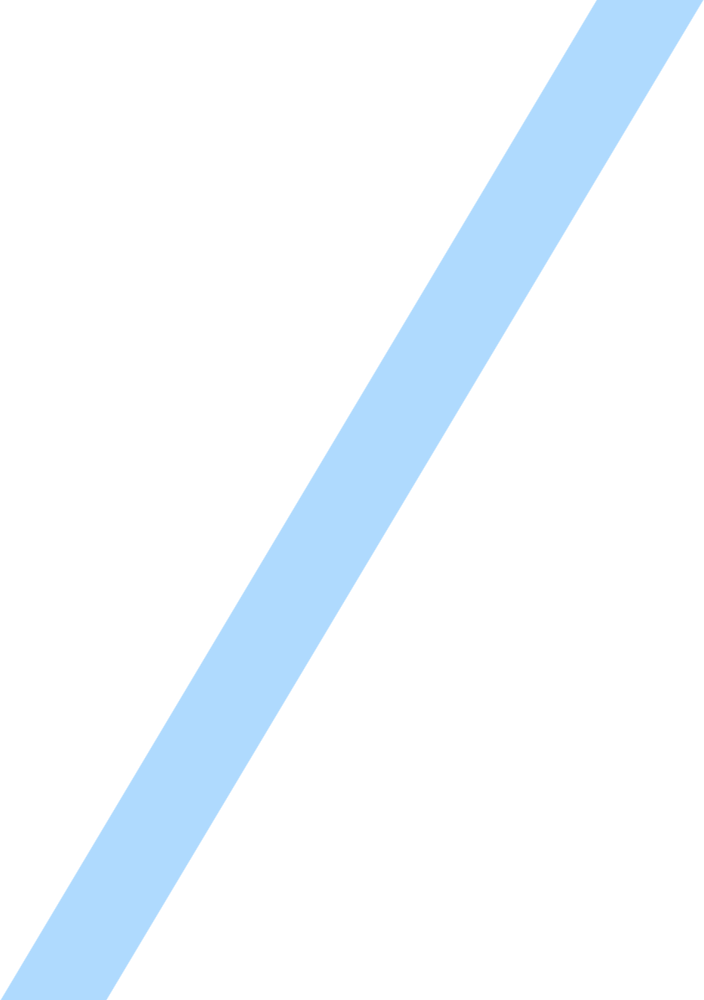
|  |
| --- |
| Real-Time Website Traffic Analytics Using AWS Cloud Services |

|  |
| --- |
| DSCI- 6007-01  Distributed & Scalable Data Engineering  Instructor: Dr. Ardiana Sula  2025-04-28 |
| Team Members:  Hrishabh Mahaju - hmaha4@unh.newhaven.edu  Yaminipoojitha Rayadurgam - yraya1@unh.newhaven.edu  Sahil Ghelani - sghel1@unh.newhaven.edu  *GitHub Repository:*  *https://github.com/hrishabh360/NasaWebsiteLogsDataEngineering/tree/master* |
|  |



Contents

[Team Member Role Assignment 2](#_Toc196853416)

[Teamwork Reflection 3](#_Toc196853417)

[Abstract 4](#_Toc196853418)

[Introduction 5](#_Toc196853419)

[Literature Review 6](#_Toc196853420)

[Methodology 7](#_Toc196853421)

[Project Workflow Diagram 8](#_Toc196853422)

[Findings 9](#_Toc196853423)

[🔹 1. Top 10 Most Visited URLs 9](#_Toc196853424)

[🔹 2. Top Active IP Addresses 10](#_Toc196853425)

[🔹 3. Hourly Traffic Pattern 10](#_Toc196853426)

[🔹4. HTTP Status Code Overview 11](#_Toc196853427)

[Discussion 12](#_Toc196853428)

[Conclusion 13](#_Toc196853429)

[Contributions/References 14](#_Toc196853430)

[Appendices: 15](#_Toc196853431)

## Team Member Role Assignment

| Team Member | Role | Responsibilities |
| --- | --- | --- |
| Hrishabh Mahaju | Project Manager & Data Engineer | - Oversee the project - Lead architecture design - Set up AWS services (Kinesis, S3, Lambda) - Design and implement the data pipeline |
| Yaminipoojitha  Rayadurgam | Data Ingestion and Processing Engineer | - Develop Python script for log parsing - Set up data streaming to Kinesis - Assist in Lambda function development - Handle schema design for Glue/Athena |
| Sahil Ghelani | Data Analyst and Visualization Specialist | - Query data using Athena - Design visualizations in Power BI - Analyze traffic patterns, status codes, and user behaviors - Prepare data insights for the report and presentation |

## Teamwork Reflection

Throughout the project, effective teamwork and collaboration were key to the team's success. Each member brought their strengths to specific stages of the project, simulating a real-world distributed data engineering environment.

As the Team Leader and Project Manager, Hrishabh was responsible for overseeing the entire project lifecycle, coordinating tasks, and ensuring timely execution. Hrishabh also led the AWS architecture setup, configuring services such as Kinesis, Lambda, and S3.

Yaminipoojitha, acting as the Data Ingestion and Processing Engineer, developed the Python scripts to parse and stream NASA HTTP logs into Kinesis and contributed to the Lambda function that processed real-time records.

Sahil, in the role of Data Analyst and Visualization Specialist, focused on querying the processed data using Athena, building insightful visualizations in Power BI, and extracting meaningful traffic patterns to support business understanding.

The team maintained consistent communication through scheduled meetings, shared GitHub repositories, and collaborative planning sessions. Tasks were divided according to each member’s expertise, promoting efficiency and continuous learning. Challenges such as limited AWS account permissions were collectively identified and resolved through brainstorming and research.

The distributed teamwork approach not only ensured the technical success of the project but also simulated professional data engineering collaboration, fostering accountability, technical skill development, and shared ownership of the project's outcomes.

.

## Abstract

In the fast-paced realm of system operations and software development, reading website logs is an essential task for monitoring system health, enhancing user experience, and decision-making. Drawing upon past industry experience as a Software Engineer, Hrishabh saw the potential for applying real-time website traffic logs to develop useful insights that influence the Software Development Life Cycle (SDLC). Having seen this, the team conceptualized and deployed a real-time website traffic analytics pipeline using scalable cloud technologies.

The solution is to implement a full end-to-end data engineering pipeline to simulate the ingestion, processing, storage, and analysis of real real-time web server logs. NASA's HTTP Web Server Logs were used as the source dataset to simulate high-speed traffic information. A Python script loads the logs into Amazon Kinesis where AWS Lambda functions to process the records and store them as structured JSON objects in Amazon S3. It is then cataloged by AWS Glue and queried by Amazon Athena, and finally, analysis is visualized by Power BI.

This project demonstrates a distributed, scalable system that can inspect real-time web site traffic, identify the most popular pages, monitor status code distributions, and identify traffic patterns over time. The team, with this pipeline, demonstrates the value of log analysis in contemporary system administration and demonstrates the way real-time observations can influence improved performance monitoring, system debugging, and user experience work. The solution emphasizes cloud-native, serverless architecture aligned with today's data engineering practices.

## Introduction

In the digital era, website traffic logs represent a valuable source of information for understanding user behavior, system performance, and potential operational bottlenecks. Organizations rely heavily on real-time insights from server logs to enhance decision-making, optimize user experiences, and proactively address technical issues. Inspired by practical industry experience in software engineering, this project was designed to explore how real-time log analysis can support the Software Development Life Cycle (SDLC) and operational excellence.

The primary objective of this project is to build a scalable, distributed data engineering pipeline capable of ingesting, processing, storing, and analyzing real-time website traffic logs. Using NASA’s HTTP Web Server Logs as the foundational dataset, the project simulates high-velocity traffic and demonstrates how modern cloud-native tools can be leveraged to handle real-time data streams efficiently.

Amazon Kinesis serves as the ingestion layer for streaming data, while AWS Lambda functions process and structure the logs. Amazon S3 provides a reliable and scalable storage solution, and AWS Glue facilitates schema inference and cataloging. The processed data is queried using Amazon Athena and visualized through Power BI, enabling interactive analysis of traffic patterns, status codes, and user interactions.

This project reflects the core principles of distributed and scalable data engineering, integrating cloud-based services to create a seamless end-to-end pipeline. It showcases the importance of real-time data analytics and demonstrates how actionable insights can be derived from seemingly raw and unstructured web server logs.

## Literature Review

A critical step in understanding the relevance of website traffic log analysis is to review prior research and existing industry practices in the domain of real-time data ingestion, processing, and analytics. The growing demand for real-time system monitoring, fault detection, and user behavior analysis has driven significant advancements in scalable log analytics pipelines.

Previous studies have explored the importance of server log analysis in system operations and cybersecurity. Early frameworks, as described by Lee and Stolfo (2000), utilized batch processing models to retrospectively extract insights from large web server logs. However, the evolution of cloud computing and stream processing technologies has enabled a shift toward real-time analytics, significantly reducing detection and response times (Marz & Warren, 2015).

Amazon Web Services (AWS) technical whitepapers emphasize the use of event-driven, serverless architectures — particularly services like Amazon Kinesis and AWS Lambda — for building highly scalable, fault-tolerant real-time data pipelines (AWS Architecture Center, 2020). These services allow near-instantaneous ingestion and transformation of log records, enabling system administrators to monitor performance, detect anomalies, and influence operational decisions rapidly.

Historical datasets, such as NASA’s HTTP Web Server Logs, have been widely used in academic settings to simulate high-traffic scenarios and benchmark real-time data systems. Studies such as Zaharia et al. (2012) demonstrated the effectiveness of distributed systems like Spark Streaming for managing fast data streams, influencing modern cloud-native designs that favor managed services over self-hosted clusters.

Furthermore, the integration of tools such as AWS Glue and Amazon Athena facilitates seamless cataloging and querying of structured data derived from unstructured logs. Visualization platforms like Power BI have also been recognized for translating complex analytics outputs into actionable insights for both technical and non-technical stakeholders (Chaudhuri, Dayal, & Narasayya, 2011).

Overall, the literature highlights the growing importance of real-time web log analysis in the Software Development Life Cycle (SDLC), particularly for proactive debugging, performance optimization, and enhancing user experiences. Building upon these foundations, the current project implements a distributed, serverless pipeline aligned with contemporary best practices to demonstrate the critical role of cloud-native log analytics in modern system administration and development workflows.

## Methodology

This project employs the CRISP-DM (Cross Industry Standard Process for Data Mining) methodology to systematically organize the data engineering, processing, and analysis tasks.

The project, titled "Real-Time Website Traffic Analytics Using Distributed and Scalable AWS Cloud Services," focuses on demonstrating how real-time website traffic logs can be leveraged to generate meaningful insights that aid operational decision-making and enhance system performance.

Business Understanding:

Recognizing the importance of log analysis in the Software Development Life Cycle (SDLC), the project aims to showcase how real-time analytics on website traffic can improve system monitoring, detect anomalies, and support infrastructure optimization.

Data Understanding:

The NASA HTTP Web Server Logs from July 1995 were chosen due to their real-world structure and high-frequency request patterns. These logs contain crucial attributes such as IP addresses, timestamps, HTTP methods, URLs, status codes, and response sizes.

Data Preparation:

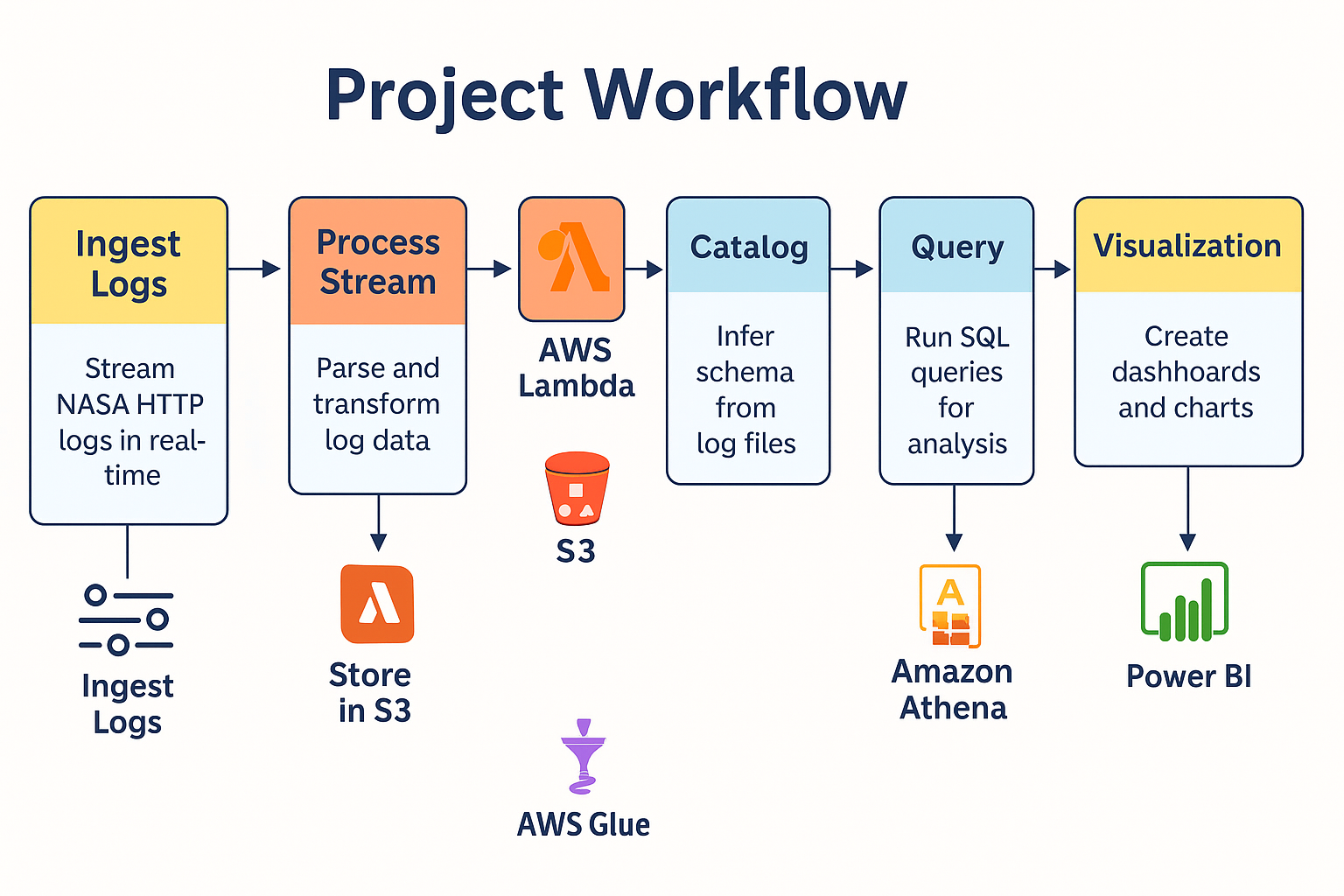
A Python-based ingestion script was developed to parse and convert the semi-structured logs into structured JSON format. Data was streamed into Amazon Kinesis, processed by AWS Lambda functions, and stored in Amazon S3. AWS Glue was used to catalog the data, preparing it for query and analysis.

Modeling:  
The pipeline architecture utilized Amazon Kinesis for real-time ingestion, AWS Lambda for on-the-fly processing, Amazon S3 for storage, AWS Glue for schema detection, Athena for querying, and Power BI for visualization. This serverless, scalable model closely mirrors modern industry best practices for real-time data engineering.

Evaluation:  
The system was evaluated based on the successful ingestion, processing, and storage of log data, and the ability to derive insights through Athena queries and Power BI dashboards. Visualizations such as top URLs, traffic timelines, and HTTP status code distributions validated the effectiveness of the pipeline in extracting operational intelligence from raw log data.

This methodology ensures a comprehensive, scalable, and practical approach to real-time website traffic analytics using cloud-native technologies.

## Project Workflow Diagram



*Fig. Project workflow diagram for Real-Time Website Traffic Analytics Using AWS Cloud Services*

The project workflow begins with the ingestion of NASA HTTP server logs, which are streamed in real-time using a Python script into Amazon Kinesis. From there, AWS Lambda functions process and transform the incoming data into a structured JSON format. The processed data is then stored in Amazon S3, providing scalable and durable storage. AWS Glue is used to catalog the stored data and infer the schema, making it query able. Using Amazon Athena, SQL queries are executed on the structured data to extract meaningful insights. Finally, the results are visualized through Power BI dashboards and charts, enabling interactive analysis of traffic patterns, popular URLs, and system health metrics.

## Findings

The Power BI dashboard developed for this project provides a comprehensive overview of real-time website traffic behavior derived from the NASA HTTP Web Server Logs. Key insights from the visualizations are summarized below:

A close-up of a graph

AI-generated content may be incorrect.

*Fig. NASA HTTP Web Server Logs Power-BI Dashboard.*

🔹 1. Top 10 Most Visited URLs

*SQL Query:*

*-- Most Requested URLs*

*SELECT url*

*,COUNT(\*) AS hits*

*FROM nasa\_logs\_nasa\_kinesis\_output\_logs*

*GROUP BY url*

*ORDER BY hits DESC LIMIT 10;*

The most frequently accessed resources on the server were primarily image files, such as /images/NASA-logosmall.gif and /images/KSC-logosmall.gif, each with hundreds of requests. This suggests that users heavily accessed graphical content, possibly for educational or media usage. Pages related to the /shuttle/countdown/ and mission logs also ranked high, indicating significant user interest in specific NASA missions and launch events.

🔹 2. Top Active IP Addresses

*SQL Query:*

*-- Top IP Addresses (most active users)*

*SELECT ip*

*,COUNT(\*) AS request\_count*

*FROM nasa\_logs\_nasa\_kinesis\_output\_logs*

*GROUP BY ip*

*ORDER BY request\_count DESC LIMIT 10;*

Analysis of IP activity revealed a handful of highly active IP addresses, with the top contributor (burger.letters.com) generating nearly 200 requests. This pattern may reflect institutional scraping, automated data retrieval, or high user engagement from specific networks or locations.

🔹 3. Hourly Traffic Pattern

*SQL Query:*

*-- Traffic Over Time (Hourly Breakdown)*

*SELECT substr(TIMESTAMP, 1, 14) AS hour\_block*

*,COUNT(\*) AS requests*

*FROM nasa\_logs\_nasa\_kinesis\_output\_logs*

*GROUP BY substr(TIMESTAMP, 1, 14)*

*ORDER BY hour\_block;*

The line chart of hourly traffic reveals peak activity between midnight and early morning hours (specifically between 00:00 and 02:00 UTC), with request volume gradually tapering off throughout the day. This trend may be attributed to global interest across different time zones or automated traffic concentrated in early hours.

🔹4. HTTP Status Code Overview

*SQL Query:*

-- HTTP Status Code Distribution

SELECT status\_code

    ,COUNT(\*) AS count

FROM nasa\_logs\_nasa\_kinesis\_output\_logs

GROUP BY status\_code

ORDER BY count DESC;

Most requests (~90%) returned a 200 OK status, confirming that most resources were successfully served. A small percentage of 302 and 304 responses represent redirections and cache validations, respectively. The presence of 404 Not Found errors (~1.5%) also highlights some broken or outdated links within the logs.

Interpretation

These findings demonstrate the practical value of log data analysis in understanding user access patterns, identifying popular content, monitoring system health, and detecting anomalies. By analyzing traffic trends, frequent request sources, and response codes, organizations can make data-driven improvements to performance, usability, and content delivery.

## Discussion

The findings from this project affirm the practical relevance of real-time web log analysis in system monitoring, traffic optimization, and user behavior tracking. From the outset, our research question centered around understanding whether serverless, distributed architecture could be used to transform unstructured log data into actionable insights for system-level decision-making. The results show that this approach is both feasible and highly effective.

The visualizations generated through Power BI clearly demonstrate patterns in user access behavior. For instance, the overwhelming interest in graphical resources like NASA mission logos points to a content-driven traffic structure, potentially from media sites or academic usage. This validates the assumption that logs are not merely technical artifacts but reflections of user intent.

Equally important are the patterns observed in IP activity and hourly usage. Identifying high-volume IPs or unusual surges in traffic equips system administrators with the tools to flag scraping behavior, detect bot traffic, or plan for server scaling. The hourly traffic decline, for example, reveals time-zone-driven engagement, further showcasing how operational strategies can be shaped through log-derived evidence.

However, as with all analytical processes, the findings offer a partial view. This dataset reflects a single month from 1995, meaning broader trends across months or modern systems are not represented. Additionally, since the NASA logs were anonymized and cleaned, the project assumes generalizability without incorporating elements like user sessions, geolocation, or authentication logs.

Still, this project provides a strong proof-of-concept. It bridges a clear knowledge gap between real-time data engineering and business analytics. By successfully deploying and evaluating a fully cloud-native log analysis pipeline, the project reinforces the value of integrating scalable tools like AWS and Power BI into modern SDLC practices — not only for developers but for analysts, system architects, and decision-makers alike.

## Conclusion

This project successfully demonstrated the design and implementation of a real-time, serverless data pipeline to analyze website traffic logs using modern cloud technologies. By simulating real-world traffic through NASA's HTTP Web Server Logs and applying a distributed architecture involving Amazon Kinesis, Lambda, S3, Glue, and Athena, the project showcased how scalable systems can transform raw log data into valuable operational insights.

Through structured data processing and cloud-native orchestration, the pipeline enabled near real-time analysis of user access patterns, top URLs, HTTP status distributions, and traffic trends. The final visualizations created in Power BI illustrated how technical data can be transformed into intuitive dashboards that inform system monitoring, debugging, and content strategy.

The project also highlights the value of adopting cloud services in data engineering education and practice. It emphasized not only technical implementation but also teamwork, problem-solving, and architectural thinking in a simulated industry environment. The use of serverless components ensured low overhead and high scalability — aligning with best practices in modern data infrastructure.

While the dataset and scope were constrained for academic purposes, the methodology is extensible and adaptable to live systems. Future enhancements could include incorporating more recent log sources, applying user segmentation, performing geolocation analysis, or integrating machine learning models for anomaly detection.

Overall, this project reinforces the critical role of real-time log analytics in supporting system resilience, user-centric development, and operational intelligence — key pillars of today’s data-driven organizations.

## Contributions/References

1. [Lee, W., & Stolfo, S. J. (2000). A framework for constructing features and models for intrusion detection systems.](https://dl.acm.org/doi/10.1145/382912.382914)
2. [Marz, N., & Warren, J. (2015). Big Data: Principles and best practices of scalable real-time data systems.](https://www.manning.com/books/big-data)
3. [AWS Architecture Center (2020). Serverless Architectures with AWS Lambda: Overview and Best Practices.](https://docs.aws.amazon.com/lambda/latest/dg/serverless-architecture.html)
4. [Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., & Stoica, I. (2012). Discretized Streams: Fault-tolerant streaming computation at scale.](https://dl.acm.org/doi/10.1145/2387880.2387892)
5. [Chaudhuri, S., Dayal, U., & Narasayya, V. (2011). An overview of business intelligence technology.](https://dl.acm.org/doi/10.1145/1978542.1978562)

## Appendices:

A computer screen shot of colorful text

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.