Journal of Computer Science
https://dergipark.org.tr/en/pub/bbd
Anatolian Science

ISSN,e-ISSN: 2548-1304
Volume: IDAP-2023, pp: 165-171, 2023
https://doi.org/10.53070/bbd.1344271
Research Paper

Designing a platform for Tweet Collection, Analytics and Storage (TweetCASP)

Tuğba Beril DOĞUÇ[1], Ahmet Arif AYDIN*[1]
[1]Bilgisayar Mühendisliği Bölümü, İnönü Üniversitesi, Malatya, Türkiye
(berildoguc@hotmail.com, arif.aydin@inonu.edu.tr)

Received:Aug.16,2023 Accepted:Aug.27,2023 Published:Oct.18,2023

Abstract—In this era of big data, the streaming, storage, and analysis of large amounts of data present a variety of challenges. Several challenges must be addressed by designers of data-intensive systems in order to retrieve useful information. Collecting, storing, and analyzing data requires a collection and analytics platform comprised of an appropriate choice of data processing and analytics technologies in order to acquire meaningful insight. In this paper, we report on TweetCASP (Tweet Collection, Analytics and Storage Platfrom), which gathers tweets based on user-entered keywords using Twitter's Streaming API, providing an environment for real-time analytics on streaming data and permanently storing data in an Apache Cassandra NoSQL datastore to fulfill future batch-oriented data processing requirements. Moreover, The TweetCASP presents an example of a data-intensive system used by software developers, designers, and researchers for data collecting and analytics.

Keywords: *Big data, data-intensive systems, real-time analytics, streaming analytics, NoSQL.*

1.Introduction

In the era of big data, we have been bombarded with data in a variety of formats (video, image, audio, text) and speeds, generated by sources such as web pages, servers (logs), microblogging sites (Twitter, Facebook, LinkedIn, etc.), Internet of Things (IoT) devices, sensors (position, temperature, speed, humidity, pressure, etc.), health records, and transactional records (Yaqoob et al., 2016)(Lv et al., 2017)(Domo Company, 2023). In Gartner's report (Gartner Inc, 2022), big data was defined as "*big data is a high-volume, high-volume form of computing that requires low-cost, innovative forms of computing that provide advanced information, decision-making, and process automation that are high-speed and/or diverse information assets*."

Each stage of collecting, storing, and analyzing large amounts of data in different formats creates various challenges for developers of data-intensive systems, such as handling the speed of incoming data to capture it without losing it *(velocity),* storing large amounts of data under a feasible data model of a suitable data storage technology *(volume),* and utilizing the right set of tools for analyzing data in order to gain insight from it *(value)* (Anderson et al., 2015)(Aydin, 2016)(Aydin & Anderson, 2020).

Gleaning useful information out of data is generally performed via batch or streaming data processing (Doguc & Aydin, 2019)(Syed et al., 2021). Using data mining, machine learning, and statistical techniques, batch-style data processing performs operations on completed datasets with the intent of discovering hidden patterns, making predictions, or performing exploratory analytics such as customer behavior forecasting, product recommendation, trend determination, fraud detection, or disease detection. Real-time or streaming analytics, on the other hand, are performed on data in motion (fast data). Real-time analytics is crucial since it seeks to deliver rapid responses in a short period of time (KEKEVİ & AYDIN, 2022). Real-time analytics is in great demand in the following application domains: banking applications for fraud detection; network applications for attack detection; and crisis informatics apps for fast emergency response (Han et al., 2014).

Due to the demand for real-time data processing and analytics, TweetCASP, a platform for the collection, storage, and analysis of tweets in real time, has been developed for this research. It is essential to have a data collection, storage, and analytics system in order to meet a wide range of analytics and domain needs. TweetCASP is an example of a data-intensive system that intends to offer a set of appropriate technologies for constructing a collecting, storage, and analysis environment for Twitter data. In addition, TweetCASP enables tweet collection based on user-entered keywords via the Twitter Streaming API (Twitter, 2022) and Python Tweepy module (Roesslein, 2022), creates a data processing and analysis environment on captured tweets in real-time via RabbitMQ, and stores data in Apache Cassandra (ApacheCassandra, 2022), for future exploratory analytics. These technologies are suitable for data collection and analysis tasks(Anderson & Schram, 2011; Aydin & Anderson, 2017).

This paper is organized as follows: Section 2 outlines related work. In section 3, materials and techniques are provided by detailing the TweetCASP's design, architecture, and utilized technology. In Section 4, an example of usage of TweetCASP is presented. In Section 5, a discussion and future work is presented. In Section 6, the undertaken research and contributions are summarized.

2. Literature Review

In this big data era, data-intensive systems are a popular topic, and numerous data-intensive systems have been developed for a variety of domains to address a variety of user requirements. However, due to space constraints, we only provide related works on data-intensive systems, specifically designed for Twitter data collection, storage, and analytics, which is one of our study's limitations. Table 1 also includes a list of the investigated related works, purposes, and tools used in their applications.

Table 1. Summary of selected related works

Reference	Utilized Tools	Purpose
(Aydin & Anderson, 2017)	Apache Spark, Redis, RabbitMQ, Cassandra	Real-time tweet collection and analytics for Crisis Informatics research
(Jambi & Anderson, 2017)	Apache Kafka, Spark, Cassandra	Deal with large twitter data
(Gehring et al., 2019)	Apache Storm, Trident, Spark, Flink, Samza	Real-time analysis with stock market data
(Amghar et al., 2020)	Twitter API, Programming Language	Finding the most appropriate NoSQL technology for managing tweets
(Aswathy et al., 2022)	Tweepy, Twython, Python twitter tools	Real-time tweet collection and analytics for Crisis Informatics research
(Vanam & R, 2023)	Tweepy, Apache Spark, TwitterAPI, HDFS	Performing sentiment analysis with using Apache Spark and Machine Learning on Tweet data
(Satauri et al., 2023)	Python, Sklearn Twitterscrapper, TextBlob	Collecting Tweets to perform sentiment analysis for online consumers

In (Aydin & Anderson, 2017), IDCAP (Incremental Data Collection and Analytics Platform) was developed. The IDCAP utilizes Apache Spark, Redis, RabbitMQ, and Apache Cassandra, and it utilizes Twitter's Streaming API to get tweets related to current topics based on user-entered keywords. The IDCAP is a real-time data processing platform designed to execute operations and analyses such as sorting, and it enables the users to query, filter, and search the streaming twitter data.

In (Jambi & Anderson, 2017), the demand for tools to perform streaming data analysis was mentioned for various reasons. The authors focused on facilitating queries on Twitter data and providing flexibility to these queries, and they described a prototype application called the "EPIC Real-Time" platform. The performance of EPIC Real-Time was tested based on performance, usability, scalability, and reliability.

In (Gehring et al., 2019), an increase in the number of data processing systems has been mentioned, and the following Apache technologies: Storm, Trident, Spark, Flink, and Samza are examined in detail, and their similarities and differences are presented. Unlike our study, the analysis was performed with stock market data instead of Twitter data.

In (Amghar et al., 2020), a comparison analysis is presented to determine the best suitable NoSQL data storage solution for handling Twitter data. In addition, the authors compare the read, write, and analytical performance of Redis, Cassandra, MongoDB, Couchbase, and Neo4j, and they recommend Couchbase. However, they don't give enough information about their tweet collection mechanism and the interface they use.

In (Aswathy et al., 2022), the importance of accurate and fast data analytics is stressed based on the disasters that occur in the world and the losses caused by these disasters. The focus is on collecting tweets about natural disasters such as floods, landslides, and rain. The authors stated that social media privacy policies such as media and location information are not violated while analyzing the collected data. In addition, unlike our study, this work also collects and translates tweets in Indian languages.

In (Vanam & R, 2023), tweet collection is performed using the Twitter API, Tweepy, and Spark Streaming. The main purpose of this work is to show the advantages of using Apache Spark technology for collecting and analyzing tweet data. Unlike our study, the authors used both CNN and logistic Regression when performing sentiment analysis.

In (Satauri et al., 2023), the importance of the collection, processing, and analysis of big social media data for sales companies in making marketing decisions and managing their strategies is mentioned. The authors presented a framework to provide sentiment analysis on reviews posted by online customers using brands and products on Twitter. To accomplish their goals, Python's Twitterscrapper module is used for collecting tweets. Sklearn and TextBlob libraries are utilized to perform analysis on tweet datasets.

3. Materials and Methods

This section describes the layered system architecture of TweetCASP and the capabilities of each layer's technology. The application, service, and storage layers of the TweetCASP are shown at a high level in Figure 1. The following subsections provide the description of each layer and the utilized technologies.

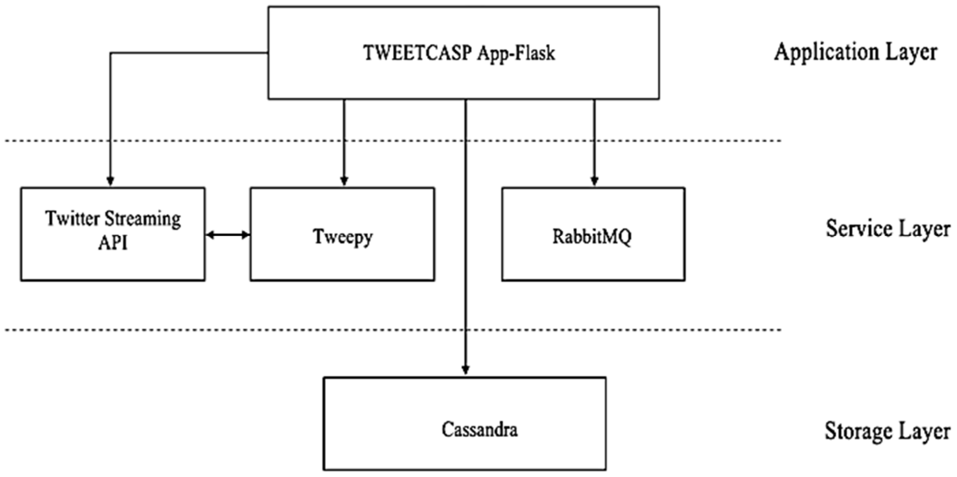
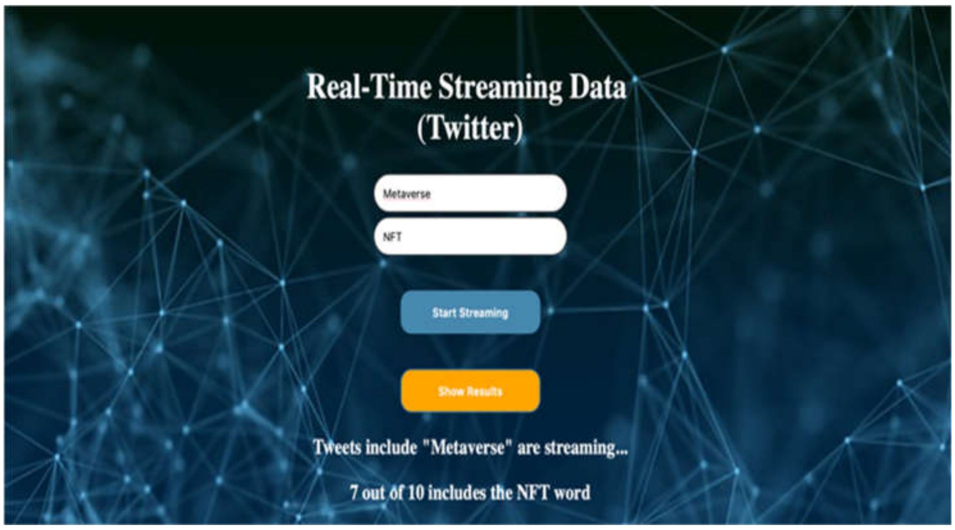


Figure 1. TweetCASP System Architecture

3.1. Application Layer

In the application layer, a web application has been developed by using the Flask web framework (*Flask Documentation (2.3.x)*, 2023). Flask is a popular Python web development framework, and it has a small and easy to expand core with features such as URL forwarding, a template engine, and portable structure. Also, Flask is utilized by well-known companies such as Netflix, Reddit, and Lyft.



Tweets include "Metaverse" are streaming...

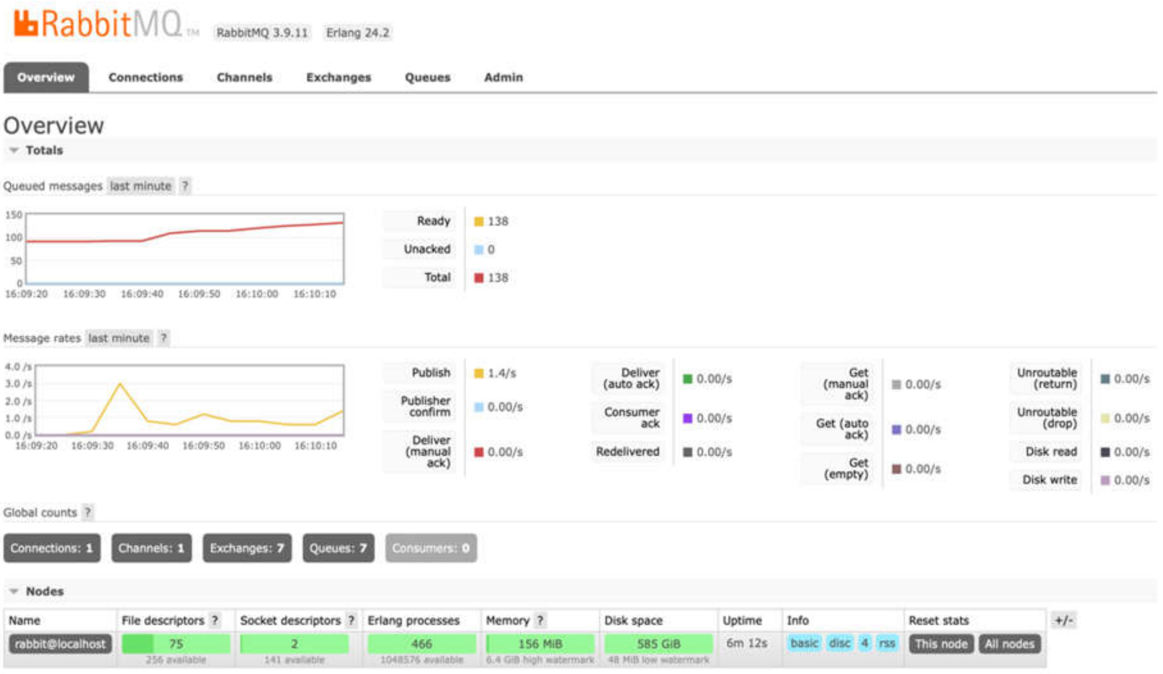
7 out of 10 includes the NFT word

Figure 2. Flask Application Interface

Flask is employed in our study because to its convenience and usability while constructing a web interface and offering a user-friendly user interface to hide the complexity of TweetCASP from its users. Figure 2 depicts the main user interface of our flask application. It enables its users to specify keywords related to the tweets they intend to collect from Twitter, initiate the data extraction process, and then at any time obtain a percentage analysis with the other keywords they specify.

3.2. Service Layer

RabbitMQ (*RabbitMQ*, 2023) and Tweepy python module (Roesslein, 2022) are employed at the service layer. RabbitMQ is an open-source message broker and queue server that enables applications to communicate data over a standard protocol or simply queue jobs for processing by distributed workers. It supports several communications protocols, STOMP and AMQP in particular (Rostanski et al., 2014). The primary objective of deploying RabbitMQ in our work is to manage the frequency of tweets and establish an environment for streaming analytics without losing any collected tweets. In addition, RabbitMQ is a resilient and persistent queuing platform that can be easily integrated with Apache Cassandra to store tweets in a scalable manner permanently. Figure 3 illustrates the user interface for monitoring an overview of the services offered by RabbitMQ, including queues, queued messages, rate limit, memory, and disk space.



Ready 138
Unacked 0
Total 138

Publish 1.4/s
Publisher confirm 0.00/s
Deliver (manual ack) 0.00/s
Deliver (auto ack) 0.00/s
Consumer ack 0.00/s
Redelivered 0.00/s
Get (manual ack) 0.00/s
Get (auto ack) 0.00/s
Get (empty) 0.00/s
Unroutable (return) 0.00/s
Unroutable (drop) 0.00/s
Disk read 0.00/s
Disk write 0.00/s

Connections: 1 Channels: 1 Exchanges: 7 Queues: 7 Consumers: 0

Name	File descriptors ?	Socket descriptors ?	Erlang processes	Memory ?	Disk space	Uptime	Info	Reset stats
rabbit@localhost	75 256 available	2 141 available	466 1048576 available	156 MiB 6.4 GiB high watermark	585 GiB 48 MiB low watermark	6m 12s	basic disc 4 rss	This node All nodes

Figure 3. RabbitMQ Interface

Tweepy is an easy-to-use and intuitive Python library for gaining access to Twitter's Streaming APIs. Additionally, it provides very helpful and comprehensible documentation for its potential users. Tweepy is employed in our work because to its excellent support for connecting to the Twitter Streaming API, its straightforward documentation, and its compatibility with Apache Cassandra and RabbitMQ.

In the TweetCASP, the Tweepy module is utilized to connect to the Twitter Streaming API using Twitter-supplied credentials: Consumer Key (API Key), Consumer Secret (API Secret), and Access Token. After connecting, TweetCASP can stream tweets depending on user-specified keywords supplied via the user interface seen in Figure 2. In addition, TweetCASP's user interface design hides the complexity of connecting to the Twitter API, entering multiple keywords (up to 400 words), and storing collected tweets.

3.3. Storage Layer

Apache Cassandra is one of the popular (DB-ENGINES, 2022) open source, NoSQL wide-column (columnar) data storage platform that provides distributed data storage at the storage layer. Apache Cassandra is utilized by globally well-known companies such as Twitter, CERN, Comcast, eBay, GitHub, GoDaddy, Hulu, Apple, Instagram, Intuit, Netflix, Reddit, and The Weather Channel.

Apache Cassandra was developed by Facebook in 2008 (Laksham Avinash & Prashant Malik, 2010) to address various storage-related difficulties associated with unstructured data. Regarding the CAP Theorem (Brewer, 2012), Apache Cassandra meets the partition tolerance and availability requirement (AP) (Marungo, 2018). The eventual consistency provided by Apache Cassandra guarantees the creation of replications after a certain amount of time (*eventual consistency).* Additionally, Apache Cassandra has similarities with Amazon's Dynamo and Google's BigTable databases and is developed in Java (Laksham Avinash & Prashant Malik, 2010). Cassandra Query Language (CQL) is a specialized, user-friendly query language for processing stored data.

Moreover, Apache Cassandra offers replication across many data centers, since data is replicated automatically to several nodes for fault tolerance and horizontal scalability. It enables the recovery of failing nodes without downtime, provides fault tolerance, eliminates single points of failure, stores massive volumes of data without data loss, offers high-speed read/write and high-rate data compression (Doguc & Aydin, 2019). Accordingly, TweetCASP employs Apache Cassandra because of the aforementioned advantages to support scalable data collection and permanent distributed storage for heavy read and write requests.

4. Use Case

It is crucial to conceal the complexity of tweet collection. Accordingly, a web UI is created utilizing the Flask Framework in order to provide a user-friendly interface. As seen in Figure 2, the TweetCASP working principle contains two input fields. The first allows users to input keywords into the Twitter Streaming API in order to stream instant tweets, while the second allows users to define a term on demand for filtering analysis on gathered tweets.

TweetCASP is performed as shown in Figure 2 by collecting tweets containing the term "Metaverse" and then providing filtering analysis on collected tweets to show only user-requested tweets containing NFT phrases and the proportion, such as 7 out of 10. Moreover, the “Start Streaming” button enables users to start collecting tweets from Twitter depending on keywords specified by the user. Also, when TweetCASP started collecting tweets, we made sure that the tweets from the Twitter Streaming API are written directly into RabbitMQ (see Figure 3) without any changes or delays. This is done to avoid time loss and interruptions while the tweets are being collected.

The “Show Results” button in Figure 2 enables users to perform real-time analytics on tweets stored in RabbitMQ to display instantaneous results regarding the number of tweets containing a certain user-entered term (NFT). For this case, we are able to provide a percentage of tweet text containing the user-specified expression. In addition, the TweetCASP only persists tweet id, tweet date, account information, tweet content, and geolocation information during the writing of tweets into Apache Cassandra for the current version. The primary objective of saving the just mentioned portions of tweets is to decrease the data size for the purpose of conducting rapid and interactive data analysis. On the other hand, if a user asks for it, we can easily allow the storage of all fields of the collected tweets. After doing data analysis on the tweets stored in RabbitMQ, the tweets that were temporarily queued are removed from RabbitMQ and moved to Apache Cassandra for permanent storage.

5. Discussion and Future Work

The concept of developing a platform for real-time data collection and analytics is both crucial and difficult. Developers of data collection and analytics platforms must overcome challenges such as velocity, volume, and diversity in order to meet analytics requirements and generate value. In addition, a data collection platform is the most important prerequisite for performing real-time analytics. Consequently, in this study, TweetCASP was designed to perform real-time Twitter data collection and real-time analytics on streaming tweets while addressing the aforementioned issues through the use of the appropriate technologies, tools, and frameworks.

In this initial version of TweetCASP, we give an example of real-time analytics to filter the complete collection of tweets in order to discover the specified subset of tweets based on user-entered keywords and report the percentage of tweets that were filtered. This version of TweetCASP is not the final version of our work, nor is it capable of handling all forms of data analytics requests for now. Nevertheless, the TweetCASP platform can be utilized as a foundation for building additional real-time data processing activities based on users' requirements. On Apache Cassandra-stored Twitter data, we can also support batch-style processing for exploratory analysis.

In future development, we also aim to include Apache Spark into the TweetCASP platform to facilitate exploratory batch and real-time data analytics. In addition, TweetCASP will get a more interactive UI to provide its users with more customized choices for filtering, searching, visualizing, and exporting analytics information on tweets queued in RabbitMQ with "id, date, account, tweet, and location" tags. Furthermore, graphical representations are crucial for presenting analytics data. Thus, we shall focus on including graphical representations into reports of analytical outcomes. Last, the presented version of TweetCASP has been using standard Twitter Streaming API. In our future development, we will be migrating to the new supported Twitter API v2 endpoint.

6. Conclusion

In this work, we report on development of the TweetCASP platform, which was designed to collect tweets from Twitter based on keywords specified by users and provide real-time analytics. The following are the study's contributions: presenting a data-intensive system design example to collect tweets based on user-requested keywords, designing a platform to perform real-time analytics on fast data, and storing data permanently in a popular NoSQL data store (Apache Cassandra) to meet users' future detailed data analytics requirements in a batch-oriented fashion. In addition, TweetCASP provides a prototype system that effectively incorporates technologies for gathering Twitter data, conducting real-time analytics, and scalable and distributed storage without data loss for data-intensive system designers and researchers in data processing and analytics.

References

Amghar, S., Cherdal, S., & Mouline, S. (2020). *Storing , preprocessing and analyzing tweets : finding the suitable noSQL system*. https://doi.org/10.1080/1206212X.2020.1846946

Anderson, K. M., Aydin, A. A., Barrenechea, M., Cardenas, A., Hakeem, M., & Jambi, S. (2015). Design Challenges/Solutions for Environments Supporting the Analysis of Social Media Data in Crisis Informatics Research. *2015 48th Hawaii International Conference on System Sciences*, *2015-March*, 163–172. https://doi.org/10.1109/HICSS.2015.29

Anderson, K. M., & Schram, A. (2011). Design and implementation of a data analytics infrastructure in support of crisis informatics research: NIER track. *2011 33rd International Conference on Software Engineering (ICSE)*, 844–847. https://doi.org/10.1145/1985793.1985920

ApacheCassandra. (2022). *ApacheCassandra.pdf*. https://cassandra.apache.org/_/index.html

Aswathy, A., Prabha, R., Gopal, L. S., Pullarkatt, D., & Ramesh, M. V. (2022). An efficient twitter data collection and analytics framework for effective disaster management. *2022 IEEE Delhi Section Conference, DELCON 2022*. https://doi.org/10.1109/DELCON54057.2022.9753627

Aydin, A. A. (2016). *INCREMENTAL DATA COLLECTION & ANALYTICS THE DESIGN OF NEXT-GENERATION CRISIS INFORMATICS SOFTWARE* [Ph.D., University of Colorado Boulder]. https://www.proquest.com/pagepdf/1834583278/Record/9F7C2D640FDE4BCCPQ/3?accountid=16268

Aydin, A. A., & Anderson, K. M. (2017). Batch to Real-Time : Incremental Data Collection & Analytics Platform. *Proceedings of the 50th Hawaii International Conference on System Sciences*, 5911–5920. http://hdl.handle.net/10125/41876

Aydin, A. A., & Anderson, K. M. (2020). Data modelling for large-scale social media analytics: design challenges and lessons learned. *International Journal of Data Mining, Modelling and Management*, *12*(4), 386. https://doi.org/10.1504/IJDMMM.2020.111409

Brewer, E. (2012). CAP Twelve Years Later: How the “Rules” Have Changed. *Computer*, *February*. https://doi.org/10.1109/MC.2012.37

DB-ENGINES. (2022). *DB-Engines Ranking*. https://db-engines.com/en/ranking

Doguc, T. B., & Aydin, A. A. (2019). CAP-based Examination of Popular NoSQL Database Technologies in Streaming Data Processing. *2019 International Artificial Intelligence and Data Processing Symposium (IDAP)*, 1–6. https://doi.org/10.1109/IDAP.2019.8875874

Domo Company. (2023). *Data Never Sleeps 9.0*. https://www.domo.com/learn/infographic/data-never-sleeps-9

Flask Documentation (2.3.x). (2023). https://flask.palletsprojects.com/en/2.3.x/tutorial/database/

Gartner Inc. (2022). *Gartner*. https://www.gartner.com/en/glossary/all-terms

Gehring, M., Charfuelan, M., & Markl, V. (2019). A comparison of distributed stream processing systems for time series analysis. *Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft Fur Informatik (GI)*, *P-290*, 205–214. https://doi.org/10.18420/btw2019-ws-21

Han, H., Yonggang, W., Tat-Seng, C., & Xuelong, L. (2014). Toward Scalable Systems for Big Data Analytics: A Technology Tutorial. *Access, IEEE*, *2*, 652–687. https://doi.org/0.11 09/ACCESS.2014.2332453

Jambi, S., & Anderson, K. M. (2017). Engineering scalable distributed services for real-time big data analytics. *Proceedings - 3rd IEEE International Conference on Big Data Computing Service and Applications, BigDataService 2017*, 131–140. https://doi.org/10.1109/BigDataService.2017.22

KEKEVİ, U., & AYDIN, A. A. (2022). Real-Time Big Data Processing and Analytics: Concepts, Technologies, and Domains. *Computer Science*, *55*(35), 1–100. https://doi.org/10.53070/bbd.1204112

Laksham Avinash, & Prashant Malik. (2010). Cassandra: a decentralized structured storage system. *ACM SIGOPS Operating Systems Review*, 1–6. https://doi.org/10.1145/1773912.1773922

Lv, Z., Song, H., Basanta-Val, P., Steed, A., & Jo, M. (2017). Next - Generation Big Data Analytics : State of the. *IEEE Transactions on Industrial Informatics*, *3203*(4), 1891–1899.

Marungo, F. (2018). A primer on NoSQL databases for enterprise architects: The cap theorem and transparent data access with MongoDB and Cassandra. *Proceedings of the Annual Hawaii International Conference on System Sciences*, *2018-Janua*, 4621–4630. https://doi.org/10.24251/hicss.2018.583

RabbitMQ. (2023). https://www.rabbitmq.com/

Roesslein, J. (2022). *Tweepy*. https://www.tweepy.org/

Rostanski, M., Grochla, K., & Seman, A. (2014). Evaluation of highly available and fault-tolerant middleware clustered architectures using RabbitMQ. *2014 Federated Conference on Computer Science and Information Systems, FedCSIS 2014*, 879–884. https://doi.org/10.15439/2014F48

Satauri, I., Satouri, B., & El Beqqali, O. (2023). Big Data Analysis in Commercial Social Networks: Analysis of Twitter Reviews for Marketing Decision Making. *European Journal of Information Technologies and Computer Science*, *3*(2), 1–6. https://doi.org/10.24018/compute.2023.3.2.94

Syed, D., Zainab, A., Ghrayeb, A., Refaat, S. S., Abu-Rub, H., & Bouhali, O. (2021). Smart Grid Big Data Analytics: Survey of Technologies, Techniques, and Applications. *IEEE Access*, *9*, 59564–59585. https://doi.org/10.1109/ACCESS.2020.3041178

Twitter. (2022). *Twitter Streaming API*. https://developer.twitter.com/en/docs/twitter-api/v1/tweets/filter-realtime/overview

Vanam, H., & R, J. R. R. (2023). Sentiment Analysis of Twitter Data Using Big Data Analytics and Deep Learning Model. *2023 International Conference on Artificial Intelligence and Knowledge Discovery in Concurrent Engineering (ICECONF)*, 1–6. https://doi.org/10.1109/ICECONF57129.2023.10084281

Yaqoob, I., Hashem, I. A. T., Gani, A., Mokhtar, S., Ahmed, E., Anuar, N. B., & Vasilakos, A. v. (2016). Big data: From beginning to future. *International Journal of Information Management*, *36*(6), 1231–1247. https://doi.org/10.1016/j.ijinfomgt.2016.07.009