HUUT Case Study

<Data and Analytics>

Revision History

Version	Date of Revision	Description	Author	Reviewed By	Approved By
1.0	14-11-2022	First Draft	Mohmmad Shahnawaz Ahangar/ Sanket Gaikwad	Anusha D Shanbhag	Arihant Bengani
2.0	17-02-2023	Customer Challenges, Outcome updated	Mohmmad Shahnawaz Ahangar/ Sanket Gaikwad	Anusha D Shanbhag	Arihant Bengani
3.0	20.02.2023	AWS Services titles are updated appropriately	Mohmmad Shahnawaz Ahangar	Anusha D Shanbhag	Arihant Bengani
4.0	10.03.2023	Additional Arhictecture Diagramd added	Mohmmad Shahnawaz Ahangar	Anusha D Shanbhag	Arihant Bengani

Table of Contents

Name of the Customer:
HUUT

Customer Background

Huut is a social media platform based out of India. It has a large audience from urban to rural areas where peoples from different languages come to share their thoughts using their native language and voice their opinions. Here, the primary focus is on freedom of speech, allowing individuals to express themselves in authentic & unforced conversations.

Customer Challenges

As a social media platform, the client generates a considerable amount of data regularly, and there is a lack of expert data engineering resources. They want to leverage the modern data-driven solution and improve their customer experience/acquisition to derive better insights, build complex business strategies and improve decision-making capabilities. The customer reached out to CloudThat with the requirement to make the end-to-end data engineering pipeline.

Here are a few technical challenges the client experienced in achieving their goals:

- The client had no existing solution to gain an in-depth understanding of their social media data, its transformation, and how it could be leveraged to drive insights and improve their platform.
- The client required a highly available and scalable solution to facilitate real-time data processing and transformation.
- The application captures the data in a semi-structured format, inefficient for real-time processing. They wanted to create a separate data lake solution catering to all their data analytics and AI/ML requirements.
- The platform allows anyone to express their thoughts in their native language and any language they prefer. Still, the challenge lies in ensuring that a language does not become exclusive to individuals from different linguistic backgrounds.

- The client also wanted to improve the user interaction on the platform and wanted a data-driven approach to provide better content for new and existing users.
- The client also wanted to improve their search application using AI-generated content.
- The client wanted a fully automated process catering to all the data-driven solutions.
- The client also wanted to set up a monitoring and alerting workflow for their data analytics system.

Assessment

The client lacks a Data Science team and operates a social media platform. They have expressed a need to enhance customer interaction, increase engagement, and prepare the platform for expansion in the future. A separate Data engineering account was to be created to cater to all the data-driven activities such as creating a data lake solution, scalable data streaming and data pipelines, event-driven analytics, and AI/ML solutions like speech-to-text, multi-language translation and many more for their future growth.

Business Objectives

- To build a solid DataLake solution that will be highly Scalable, Available, Durable, and Secure
- To build a future-ready Data engineering pipeline that can harness the power of data
- To build AI solutions to improve the overall platform and make customer engagement more personalized
- To build a data-driven solution that helps businesses make better decisions than before

Proposed Solution and Architecture diagram

Considering the customer's business needs and technical specifications, CloudThat proposed an end-to-end solution that can implement features and service with reduced downtime, optimizing costs and providing a highly secure and scalable solution.

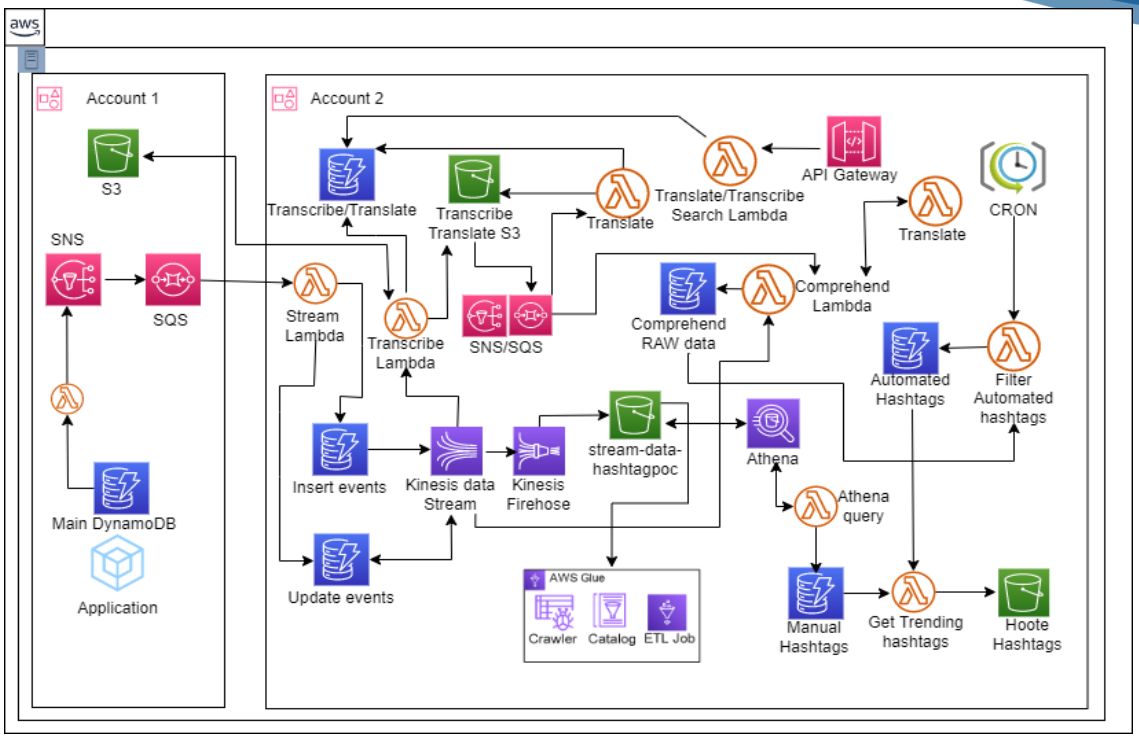


Fig 1: Architecture flow diagram

- A separate AWS account for building data-driven solutions since the customer wanted to distinguish the data engineering from the primary applications workload
- A cross-account to access the primary AWS account where application data is replicated to the data engineering account.
- In data engineering account:
  - Creating a centralized data lake solution using Amazon S3 to store all the source data (which is in raw JSON format) that originates from the primary AWS account. Storing the data in Amazon S3 using proper prefixes and appropriate data formatting structures to help to query and access the data efficiently.
  - Data is transferred between two AWS accounts using Amazon SQS and AWS Lambda. The data is then pushed to Amazon DynamoDB tables for two separate types of events/payloads. These are then streamed to the Amazon Kinesis stream and Amazon Kinesis Data Firehose, which dumps the data into Amazon S3.
  - The Amazon Transcribe lambda pulls the data from the Amazon Kinesis Data Stream, which transcribes the audio to text for various languages and then stores the data in the Amazon DynamoDB table and the Amazon S3 bucket.
  - The transcribed text is translated into SRT format for various languages and stored in the same Amazon DynamoDB table and the Amazon S3 bucket.

- Request from Amazon API Gateway will be transcribed/translated and stored in Amazon DynamoDB
- The Amazon SQS sends the raw transcribed text to the comprehend and translate AWS lambda functions and will store it in an Amazon DynamoDB table as raw data. The raw data is then filtered for hashtags which are then saved into an Amazon DynamoDB table, and a cron job will make sure this lambda is at a particular point in time to fetch more data from the Amazon DynamoDB
- Data from Amazon Kinesis Data Firehose is stored in an Amazon S3 bucket. Using the AWS Glue Crawler, data from the Amazon S3 bucket will be fetched, and on the fetched data ETL job will be triggered to transform the data.
- Using Amazon Athena, data can be queried from the Amazon S3 bucket, and AWS Lambda will fetch the required feature from Amazon Athena and store it in Amazon DynamoDB.
- Using AWS Lambda data from the automated hashtag and manual hashtag Amazon DynamoDB can be stored in an Amazon S3 bucket.

Third-Party Applications or Solutions used

- **GIT:** It is used as a code repository for creating various versions of code that are used for transforming the data
- **Slack**: To integrate and expose multiple APIs for regular updates on alarms, notifications, errors, and others.

AWS Services Used

- **Amazon Kinesis Data Firehose:** is an extract, transform, and load (ETL) service that reliably captures, transforms and delivers streaming data to data lakes. It stores the raw data in an Amazon S3 bucket.
- **Amazon Kinesis Data Stream:** is a serverless data streaming solution that offers real-time ingestion and storage of streaming data. It is used to send data to multiple sources.

- **Amazon Simple Storage Service (Amazon S3):** Amazon S3 is used as Data Lake to store data from multiple sources. The buckets are configured with fine-grained policies which allow access to only specific users and applications.
- **AWS Glue Service:** is an ETL service that validates, logs, and transforms the raw data. The source data is transformed by performing data cleaning using ETL jobs to extract valuable data and store the transformed data back into the Amazon S3 bucket.
- **Amazon Athena:** a serverless querying service, used to fetch the transformed data in the Amazon S3 bucket for data information gathering and analysis.
- **Amazon DynamoDB:** is a fully managed, serverless, key-value NoSQL database designed to run high-performance applications at any scale. **It** has been used as a database as it is a fully managed, serverless, key-value NoSQL database designed to run high-performance applications at any scale. Amazon DynamoDB offers built-in security, continuous backups, automated multi-Region replications, in-memory caching, and data export tools.
- **AWS Lambda:** runs the code in a serverless environment. It is used to write and deploy code for streaming, transcribing, and translating data.
- **Amazon SQS:** is a distributed message queuing service. Queues the data from the primary AWS account and Amazon S3 bucket. The Amazon SQS sends the raw transcribed text to the AWS Lambda functions.
- **AWS Identity and Access Management (IAM):** Access is controlled using AWS IAM Roles for resources or services with the least privileged access policies
- **Amazon Translate:** is a translation service that is used to translate from English to different languages, or vice-versa.
- **Amazon Transcribe:** is used to generate subtitles for audio and video data files
- **Amazon SNS:** a simple notification service that is used to deliver/push data from cross-account.
- **Amazon API Gateway:** is used for API management and giving access to the external client or trusted party to the data.
- **Amazon Comprehend:** an NLP service that is used extract text with insights and valuable information. It is used for sentiment analysis.
- **Amazon CloudFront:** is used, with OAI enabled, to securely access Amazon S3 data

by acting as a mediator and preventing direct public access to the Amazon S3 data.

- **Amazon CloudFormation:** is an IaaS service used for easy provisioning and modelling of cloud services/resources. Automation was achieved using Amazon CloudFormation for AWS Glue service for fast and continuous deployment and will be used for RTO/RPO
- **Amazon CloudTrail:** is used for auditing and compliance. It is used to audit, understand, and quickly resolve issues or actions a user, role, or service takes quickly without downtime.
- **Amazon CloudWatch:** is used for collecting and visualizing real-time logs and metrics. It is used for monitoring and logging errors and specific events by various AWS services.

How AWS was used as part of the solution

Serverless architecture has been created using Amazon API Gateway, AWS Lambda, Amazon Kinesis Data Streams and Amazon Kinesis Data Firehose, Amazon S3, and AWS Glue for setting up data-driven solutions

- Amazon SQS and AWS Lambda are used for accessing the source data in the Data Engineering account. The source data (in raw JSON format) contains information like audio file, image, created and updated timestamp, number of likes, number of times it was heard, text, and tags in it. This data is then dumped into Amazon DynamoDB.
- We have used Amazon Kinesis Data Streams for batch streaming of source data, and Amazon Kinesis Data Firehose is used to store in a centralized data lake Amazon S3 solution. The source data is in JSON format, and the information comes continuously. We have used Change Data Capture for Amazon DynamoDB events and stored the data in Amazon S3 using Dynamic partitioning of Amazon Kinesis Data Firehose. We must segregate the data based on the time and date the source data is created.
- Dynamic partitioning is used for generating custom prefixes from the records coming from Amazon Kinesis data streams. It has helped us differentiate the data from the

records eliminating the need to perform manual prefixes.

- Amazon Kinesis Data Streams will get data from Amazon DynamoDB. This data is then fed to Amazon transcribe, which is used to transcribe audio/video, which will be translated into different languages and stored into text for various languages stored in Amazon DynamoDB and Amazon S3 bucket. From the Amazon S3 bucket, the text will be queued to Amazon Translate, which will be translated into different languages and stored in the same Amazon DynamoDB.
- Cross Account Amazon S3 access is also set up to fetch the Audio File which is stored in the primary AWS account. Fine-grained AWS IAM policies provide limited access to particular services so that the data cannot be misused. We also have set up the cross-region Amazon S3 objects replication for Disaster Recovery. This has also helped us achieve High Performance, Availability, and Data Integrity.
- From Amazon Kinesis Data Firehose, the source data is stored in Amazon S3. The AWS Glue crawler service will run to identify if any new partitions or folders are generated in the Amazon S3 data lake and store all the partition information in the AWS Glue Data Catalog.
- By using AWS Glue Data Catalog and partitions, we are running AWS Glue ETL Job to transform the data. Mapping will be done on a variable, and their type will also be changed. Tags will be converted into a proper format. After ETL transformation, data will be in CSV format and stored the structured data back into the Amazon S3 data lake.
- In AWS Glue ETL Job, we enable job bookmarks to track previously processed Amazon S3 data by the ETL job. Eliminating data duplication will ensure that Job runs the ETL job only on the new data.
- We have used AWS Lambda to run Amazon Athena queries on structured data in Amazon S3 to derive the required insights the customer wants. We have stored the insights in the Amazon DynamoDB table using the Date and Time keys to present the data meaningfully.
- We have set up another AWS Lambda function to translate/ transcribe the data and Amazon API Gateway to fetch the insights based on customer needs by providing query parameters to it and storing them in an Amazon DynamoDB table.

Additionally, the data received from the API is integrated with the application platform as new features

Start Date of the Project

25th November 2021

End Date of the Project

16th February 2022

Date the project entered Production

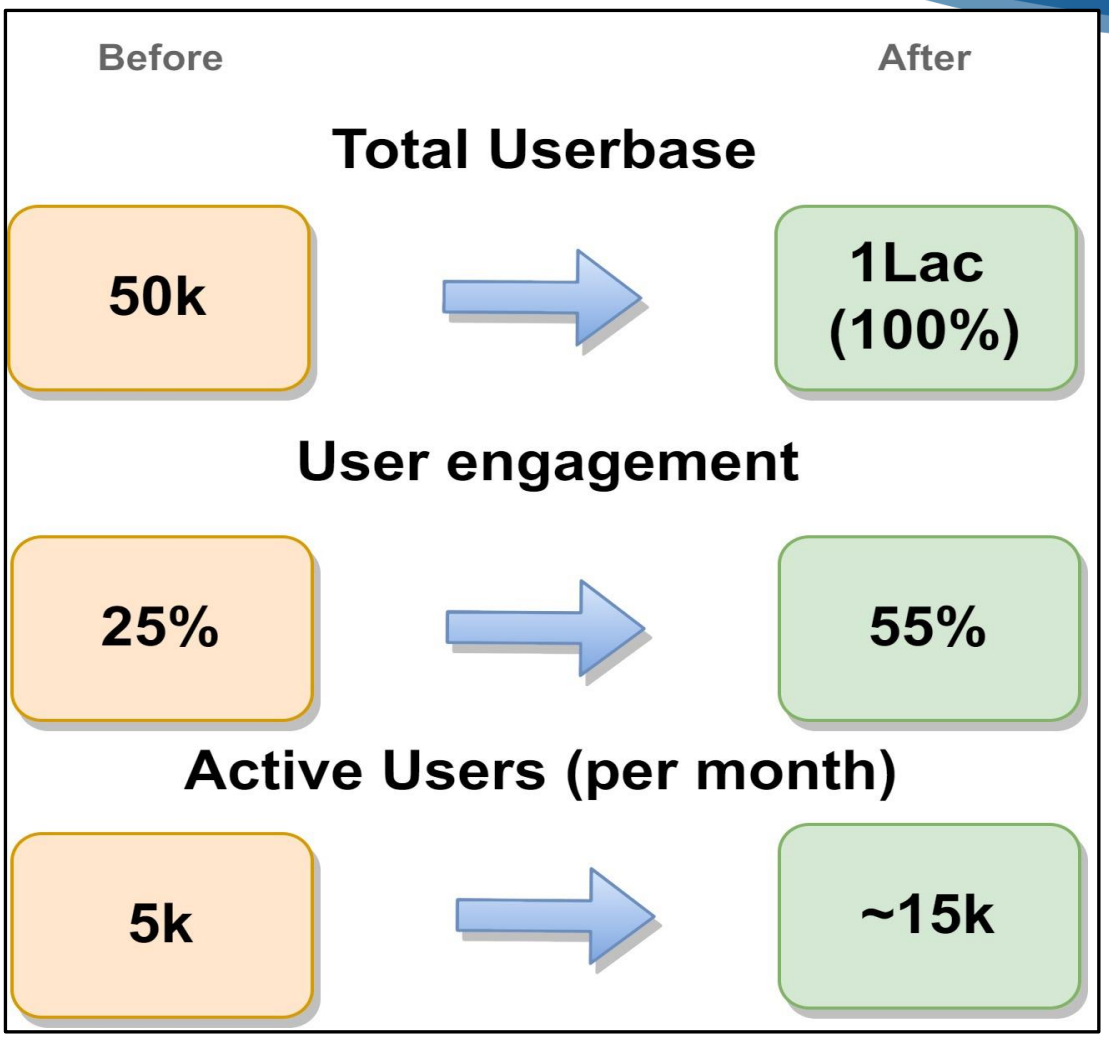
28th December 2021

Outcome

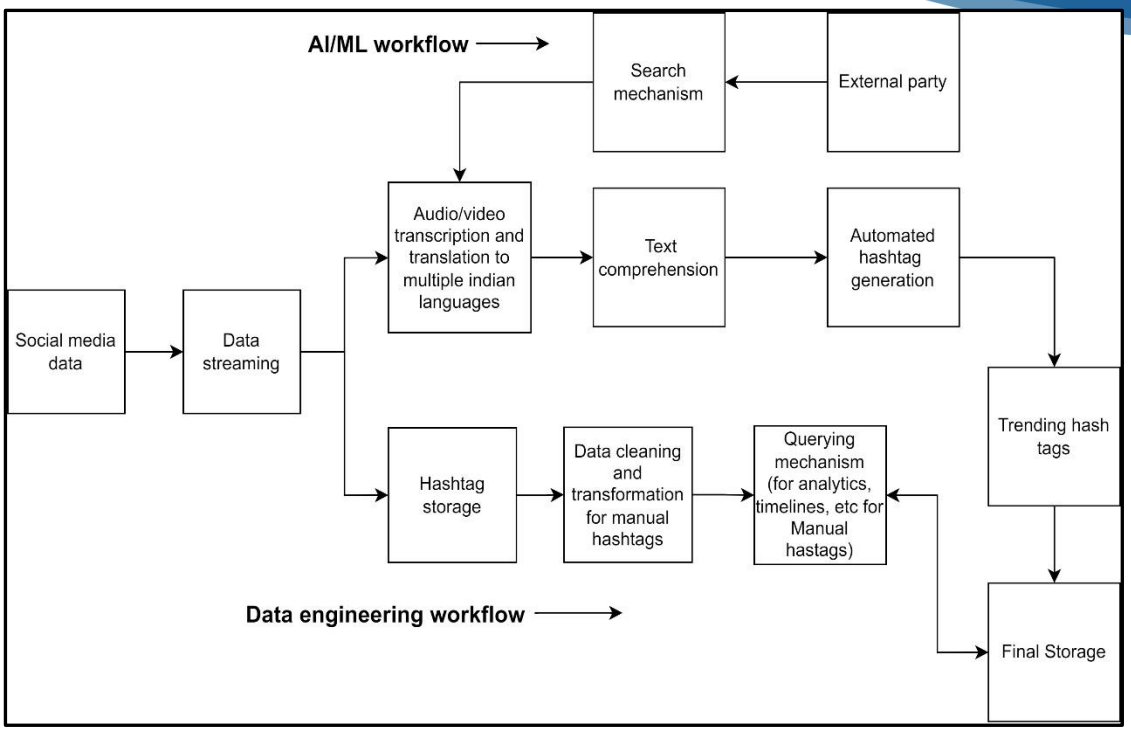
- Customer now has an end-to-end system that covers all aspect for their data engineering operations and analytics tasks:
    - A fully managed data pipeline is in place to streamline the streaming and cleaning of raw data. The ETL jobs were set up for the collection of the data and transformation at specific schedules.
    - Necessary steps were taken to ensure the same pipeline scales effortlessly as their user base grows.
    - Queries can be run on hashtag data to scan and retrieve information according to specified timelines. This enabled the customer to view and understand user trends, user behaviors and providing much more valuable insights for business decisions.
    - A data lake solution was provided for better data governance and more efficient storage of hashtag data. The data lake solution is also designed to cater to future requirements for AI/ML applications. As AI/ML algorithms require large volumes of data to train and learn from, the data lake solution can provide a central location for storing and accessing the data required for

such applications

- Customer now has the ability to see the global trending hashtags on their home pages, improving user experience and delievering better content for the user.
- The users of the customer application can now see closed-captions generated for their audio/video posts on their feed.
    - The customer userbase is made up of people who speak multiple languages. To better serve these users and ensure a wider audience reach, an AI/ML feature was introduced and implemented to extract and translate captions for their audio and video data into multiple languauges. This allowed customer to provide inclusive user experiences for different linguistic backgrounds.
    - The extracted caption data mechanism was pipelined with translation services that accurately translates to multiple indian language.
- By implementing sentiment analysis in their cloud infrastructure, they can now accurately and dyamically detect the emotions conveyed in their social media posts and generate appropriate hashtags that resonate with their audience.
    - Text extraction mechanism was introducted to convert audio/video files into captions data for sentiment analysis and the most probable hastags were generated for the user posts.
    - This allowed users to receive suggested hastags along side with the manual hashtags that they provided.
- By integrating with Slack channels, customers can receive updates regarding monitoring alerts and issues. Any necessary process changes will also be handled appropriately.
- The other key performance indicators include total size of the userbase, User engagement metrics with posts and hastags, and active users per month which were observed by the customer after implementing the features and solutions:



Additional Architectural Designs and Diagrams



Lessons Learned

- **Separation and targeting of data:** At the initial stage, the data is directly coming to the Amazon DynamoDB, an AWS lambda was introduced between cross-account, which was used to separate events on type events such as insert or update event data. This allowed for more efficient and targeted processing of data, improving system performance and reducing the risk of errors. It also allowed for a more flexible and scalable approach to data processing.

- **Understanding data control and flow:** To control the flow the data, a control mechanism was introduced using Amazon SNS and Amazon SQS with buffering and sent to appropriate Amazon S3 buckets. The buffering mechanism of Amazon SQS allowed for the control of the rate at which data was sent and received, preventing system overloads and ensuring proper processing and storage of data. This hugely allowed for creating bigger bandwidth in the data processing. With Amazon SNS, it was possible to separate concerns and allow different downstream services to subscribe to specific topics and receive only relevant data.

- **Data tranformation and cleaning:** There was a single data stream because of which unnecessary data was also getting processed. To fix this, we made an ETL process in AWS Glue for each new event data coming from the stream and stripped off the unnecessary information from the same. Now it was possible to ensure that only the required data was processed, leading to more efficient use of system resources and improved system performance.