

# **Bans and Bubbles: Analysis of Moderation in VTuber Streams**

A Project Report

Presented to

DATA-228-12

Fall, 2023

By

Abdul Sohail Ahmed (016769610)

Poojan Gagrani (016795285)

Kashish Thakur (016919383)

Somna Sattoor (014640769)

Swetha Neha Kutty Sivakumar (016780712)

December 10, 2023

Copyright © 2023

[Abdul Sohail Ahmed, Poojan Gagrani, Somna Sattoor, Swetha Neha Kutty Sivakumar, Kashish Thakur]

ALL RIGHTS RESERVED

## ABSTRACT

### **Bans and Bubbles: Analysis of Moderation in VTuber Streams**

Abdul Sohail Ahmed, Poojan Gagrani, Somna Sattoor, Swetha Neha Kutty Sivakumar, &

Kashish Thakur

The rise of Virtual YouTubers (VTubers) has become a prominent trend in the expanding digital and virtual entertainment industry. This research investigates the complex dynamics of VTuber events, with a specific focus on measures related to audience involvement, including live chat conversations, Super Chat donations, and moderating activities. Using a comprehensive dataset obtained from Kaggle, the study thoroughly examines these variables to identify the characteristics that contribute to successful VTuber engagements. By utilizing Azure's powerful array of data tools, the project extensively investigates the data using Azure Databricks and Azure Synapse, resulting in a comprehensive study. The acquired insights seek to aid VTubers in enhancing their content development and engagement strategies, while also enabling viewers to navigate the VTuber ecosystem more efficiently. Expected results encompass the cultivation of a sophisticated comprehension of audience preferences and behaviors, resulting in more tailored and influential viewer experiences. The project aims to improve the digital interface between VTubers and their audiences, thereby promoting the long-term viability and expansion of this pioneering type of entertainment.

### **Acknowledgments**

The creation of this study was greatly aided by Professor Andrew H. Bond and TA Venkatasai Rao Dheekonda, for whom the authors are extremely grateful.

## **Table of Contents**

### **Chapter 1 Introduction**

- 1.1 Project goals and objectives
- 1.2 Problem and Motivation
- 1.3 Project Results and Expected Deliverables

### **Chapter 2 Background and Related Work**

- 2.1 Background and Used Technologies
- 2.2 Literature Survey

### **Chapter 3 Data Overview**

- 3.1 Data Description
- 3.2 Dataset Samples

### **Chapter 4 Architecture and Project Flow**

- 4.1 Architecture
- 4.2 Data Modeling and ER Diagram
- 4.3 Project Flow

### **Chapter 5 Data Engineering**

- 5.1 Importing data
- 5.2 Data Cleaning
- 5.3 ETL process
  - 1. Extract Data
  - 2. Transform Data
  - 3. Load Data

### **Chapter 6 Data Analytics and Visualization**

- 6.1 Data Analytics Using SQL Script

## **Chapter 7 Visualizations**

7.1 Data Visualization

7.2 Web Application to Host Dashboard

## **Chapter 8 Collaboration**

8.1 GitHub

8.2 Team Members and IDs

## **Chapter 9 Conclusion and Future Work**

9.1 Conclusion

9.2 Future Work

## **References**

## **List of Tables**

Table 1. Channel table.....	Page No. 17
Table 2. Chat Data table.....	Page No. 17
Table 3. Superchat Data table.....	Page No. 18
Table 4. Deletion Events table.....	Page No. 18
Table 5. Ban Events table.....	Page No. 19
Table 6. Chat Statistics table.....	Page No. 19
Table 7. Super Chat Statistics table.....	Page No. 20
Table 8. Tools used along with the purpose.....	Page No. 27

## List of Figures

Figure 1. Sample from the unprocessed Channel dataset .....	Page No. 21
Figure 2. Sample from the unprocessed Chat dataset .....	Page No. 22
Figure 3. Sample from the unprocessed Chat Statistics dataset .....	Page No. 22
Figure 4. Sample from the unprocessed SuperChat dataset .....	Page No. 23
Figure 5. Sample from the unprocessed SuperChat Statistics dataset .....	Page No. 23
Figure 6. Sample from the unprocessed Deletion Events dataset .....	Page No. 24
Figure 7. Sample from the unprocessed Ban Events dataset .....	Page No. 24
Figure 8. Diagram of the Project Architecture .....	Page No. 25
Figure 9. ER Diagram.....	Page No. 26
Figure 10. Raw data files loaded into Azure data lake container .....	Page No. 28
Figure 11. Setting up Azure Databricks spark notebook and reading files .....	Page No. 29
Figure 12. Filtering and reducing ban data from Jan to Feb 2022 .....	Page No. 29
Figure 13. Filtering and reducing chat stats from Jan to Feb 2022 .....	Page No. 30
Figure 14. Filtering and reducing deletion from Jan to Feb 2022 .....	Page No. 30
Figure 15. Filtering and reducing superchat stats from Jan to Feb 2022 .....	Page No. 31
Figure 16. Filtering and reducing superchat stats from Jan to Feb 2022 .....	Page No. 31
Figure 17. Dropping Name, Group, and Photo columns from channel .....	Page No. 32
Figure 18. Mapping the color for the significance field in superchat .....	Page No. 32
Figure 19. Generating chatId for chat .....	Page No. 33

Figure 20. Generating superChatId for chat .....	Page No. 33
Figure 21. Extracting cleaned files from the pipeline .....	Page No. 34
Figure 22. Merging Chat data .....	Page No. 35
Figure 23. Merging Chat and Channel data.....	Page No. 35
Figure 24. Merging Superchat data .....	Page No. 36
Figure 25. Merging Superchat and Channel data .....	Page No. 36
Figure 26. Merging Chat Stats and Channel data .....	Page No. 37
Figure 27. Merging Superchat Stats and Channel data .....	Page No. 37
Figure 28. Pipeline for moving final data from source to destination.....	Page No. 38
Figure 29. SQL query with output to show the total number of Super Chats and Unique Super Chatters in each Period .....	Page No. 39
Figure 30. Horizontal Bar plot of the total number of Super Chats and Unique Super Chatters over the Period .....	Page No. 40
Figure 31. SQL query with output to show the total number of Banned Chatters by Period .....	Page No. 40
Figure 32. Line plot to show the trend of total Banned Chatters over the Period .....	Page No. 41
Figure 33. SQL query with output to show top 10 Currencies by total Super Chat Amount .....	Page No. 42
Figure 34. Pie chart to show total Super Chat Amount from top 10 Currency .....	Page No. 42
Figure 35. SQL query and output to show the top 10 Channels with the highest Banned Chatters percentage by Period .....	Page No. 43

Figure 36. SQL query and output containing Affiliation details about top 10 Affiliations by Average Subscribers .....	Page No. 44
Figure 37. Top 10 Affiliations by Channels .....	Page No. 46
Figure 38. Top 10 Affiliations by count of Video uploads .....	Page No. 47
Figure 39. Top 15 Channels by Member Chats .....	Page No. 48
Figure 40. Member's trend for top 5 Affiliations .....	Page No. 49
Figure 41. Count of Super Chats in top 10 Channels .....	Page No. 50
Figure 42. Total Amount gained by the top 10 Channels .....	Page No. 51
Figure 43. Count of Banned Chats for top 10 Affiliations .....	Page No. 52
Figure 44. Count of Deleted Chats for top 10 Channels .....	Page No. 53
Figure 45. Website Landing page .....	Page No. 54
Figure 46. Login with Google OAuth (SSO).....	Page No. 55
Figure 47. VTuber Dashboard hosted on the website .....	Page No. 55
Figure 48. Github repository of the project .....	Page No. 56

# **Chapter 1**

## **1.1 Project Goals and Objectives**

The primary aims and objectives of this study are to explore the dynamic realm of VTubers, virtual performers who have captivated a worldwide audience through their online personalities. The project will utilize sophisticated data analysis methods to analyze viewer involvement on several platforms. This analysis will include studying chat interactions, donation patterns, and moderator actions to assess the effectiveness of the material and understand audience attitude. The objective is to create a comprehensive depiction of the factors that motivate viewer engagement and loyalty within the VTuber community, highlighting material that strongly resonates with the audience.

Moreover, the initiative is carefully intended to function as a navigational tool for affiliation agencies, identifying VTubers who continuously create material that resonates positively with the public, thereby revealing opportunities for brand collaborations. The project offers full insights by utilizing Azure's extensive data capabilities, including data input to Azure Data Lake, processing with Azure Databricks, and rich exploratory data analysis in Synapse. The outcome of this analysis will be a website that is easy for users to navigate, displaying condensed insights using PowerBi visuals. This platform aims to not only direct viewers to top-notch VTuber material but also offer rising VTubers valuable data to improve their skills and increase their presence in the competitive realm of digital entertainment.

## **1.2 Problem and Motivation**

The emergence of Virtual YouTubers (VTubers) has introduced a novel era of digital entertainment, captivating global audiences with their distinctive fusion of animation and interactivity. Nevertheless, as this virtual phenomenon expands, the intricacy of comprehending

audience involvement and content impact also increases. This project aims to analyze large volumes of live stream data to reveal trends in user engagement and the effectiveness of content. This project employs Azure's data orchestration technologies to systematically monitor chat participation, Super Chat donations, and moderation events. The goal is to support VTubers in providing impactful content and assist viewers in making informed choices. The objective of this study is to offer practical insights for VTubers and affiliation agencies, to promote constructive content production and establish long-lasting brand collaborations.

The endeavor is motivated by the potential to greatly improve the VTuber ecosystem, making it more captivating for viewers and also more profitable and manageable for content creators. The project aims to provide a strong analytical framework that utilizes cloud-based services to handle and visualize data. This will enable a thorough understanding of the VTuber landscape. This involves creating a flexible and protected framework for handling and displaying data, resulting in a user-friendly website that highlights important discoveries. The motive goes beyond simple analysis; it involves fostering a dynamic online community that flourishes through well-informed decision-making and strategic development, benefiting both viewers and VTubers.

### **1.3 Project Results and Expected Deliverables**

The primary result of this project is the creation of a customized interactive analytics dashboard for VTuber event data, offering comprehensive information on VTuber engagement. The dashboard is an important output that provides a detailed perspective on the dynamics of VTuber streams, encompassing chat interactions and moderating occurrences. Another notable result is the acquired knowledge of content moderation, which unveils the real-time management of VTuber streams to promote constructive interactions.

Expected deliverables from this project are manifold:

1. A Power BI dashboard that visually represents VTuber event data dynamically and interactively.
2. Comprehensive analytical reports providing detailed insights about VTuber engagement and trends of content moderation.
3. An efficient and reliable implementation of Google OAuth Single Sign-On for secure and uncomplicated access to the dashboard.
4. A website designed using the .NET Framework that is easy to use, allowing users to easily explore and understand the presented information.

This extensive collection of deliverables will function as a significant asset for comprehending and promoting constructive content generation and audience involvement within the VTuber community.

## **Chapter 2 Background and Related Work**

### **2.1 Background and Used Technologies**

#### **Background**

The project is focused on the concept of Vtubers, which is a virtual YouTube community, where online entertainers use a virtual avatar in place of their actual identity to interact with the users. During online streaming, Vtubers connect with their fans, using Chats and Superchats. Chats are the general unpaid medium of interaction used by the fans, whereas Super chats are the medium of interaction used by fans with paid memberships. The dataset used in the project consists of Vtuber content consisting of a large quantity of data about Chats, Super chats, and moderation events like Deletion and Ban events.

## Technologies Used

All the project tasks are accomplished using the data orchestration tools of the Azure ecosystem, which are explained below:

1. **Azure Data Lake:** The data is ingested into the Data Lake, which enables it to perform Big Data analytics. It allows the unlimited storage of structured, semi-structured, and unstructured data of any size. This property of Azure Data Lake makes it suitable for big data projects.
2. **Azure Databricks:** It is an Apache Spark-based analytics platform, which has a lake house architecture that along with Databricks SQL helps in performing data warehousing tasks. It is used to perform Data Cleaning and Preprocessing, with the help of Python data frames. All the null values are removed and missing values are either imputed or removed based on the context of the information.
3. **Azure Synapse Analytics:** It is an integrated analytics service, which is used for performing exploratory data analysis. This analysis is used for initial data understanding and for identifying the outliers in the data. The insights obtained can be used to perform further data analytics while creating visualization dashboards.
4. **Power BI:** It is an interactive visualization dashboard service offered by Microsoft which is integrated with other Azure services. It enables interactive visualizations and business intelligence capabilities with an interface simple enough for end users to create their reports and dashboards.
5. **DotNet Framework:** It is a software development framework that is used to build the web application.

6. **GoogleOAuth:** It is a security feature that allows a user to successfully sign in to several linked but independent software systems with a single ID. It is used to sign in to the website via Gmail ID.

## 2.2 Literature Survey

Kim and Yoo (2021) have done a comparative study between the user experience of existing general and virtual YouTubers. The data is collected using a survey, which is prepared by the AttrakDiff evaluation method. This questionnaire is used for quantitative analysis of four dimensions of usability which are pragmatic quality, hedonic quality-identity, hedonic quality-simulation, and attractiveness with 28 items (pairs of opposite adjectives positioned at both ends) rated on a 7-point scale (-3: negative adjective ~ 3: positive adjective). The survey was collected from a total of 50 people (25 men and 25 women) in their 20s to 30s. The study aims to further identify the preferences and features of the user experience using AttrakDiff evaluation techniques from the simple rating comparison for existing virtual YouTubers (Vtubers).

The research done by Advantia and NoHikari (2023) aims to explore the factors that influence the growth of subscribers to Virtual YouTubers (VTubers) in Vietnam. The dataset includes an intake of 97 VTubers chosen at random, and the study uses regression analysis to examine the link between the number of videos, views, and subscribers. According to the study's findings, increasing the number of video uploads does improve the number of subscribers, however the number of views has a direct impact on the number of subscribers. Hence the view count is identified to be a high impact factor in increasing the subscriber count. The main goal of the study is to understand the different growth strategies for Vtubers in the context of Digital media, majorly focusing on the Vietnamese market.

Mamat et al. (2022) performed a study to identify Malaysian youth's interest in anime culture and its subcultures such as VTube. Since the 1990s, anime, manga, and drama series have been among the most popular and well-received forms of Japanese popular culture, and their translation into multiple languages has made them globally recognized. An online survey utilizing Google Forms was completed by 104 people. Respondents were Japanese language students from two Malaysian public universities. Some of the responders are members of a Japanese Cultural Club at one of the universities. The survey was divided into three sections: respondents' background, respondents' interest in anime culture, and their interest in VTubers. The results showed that the most popular culture is anime, followed by manga, songs, and seiyuu (voice actor/actress). The study identifies fantasy as the most popular anime genre, followed by humor, love stories (romance), and mystery. The majority of respondents had heard the term "VTuber" from anime, friends, or internet outlets such as YouTube. The statistics show that VTubers are an important and well-known product of Japanese culture among Japanese culture lovers and that they will certainly gain widespread momentum in the future.

## **Chapter 3 Data Overview**

### **3.1 Data Description**

The VTubers dataset is obtained from Kaggle, where the data is uploaded by a reliable source. The data is in the form of Parquet files, which is an efficient column storage format. The size of the data is about 13GB consisting of information related to Vtuber Channel, Vtuber Chat data, Vtuber Superchat Data, Vtuber Chat Statistics, Vtuber Super Chat Statistics, and information about Ban and Deletion Events. The dataset collected covers three months in the year 2020. The data is ingested into the Azure Data Lake, where all the collected Parquet files

are stored and fetched for further Data Processing. As described earlier the complete dataset consists of seven files. The following tables describe the columns in each file.

The Table below shows the Column names, data type, and Description of the corresponding columns of the Channel table.

**Table 1: Channel table**

Column Name	Type	Description
channelId	string	A unique identifier for each channel
name	string	Name of the Channel
englishName	nullable string	Name of the Channel in English
affiliation	string	Channel Affiliation
group	nullable string	Group
subscriptionCount	number	The count of the subscriptions
videoCount	number	Total number of videos uploaded by the channel
photo	string	Channel Icon

The Table below shows the Column names, data type, and Description of the corresponding columns of the Chat Data table.

**Table 2: Chat Data table**

Column Name	Type	Description
timestamp	string	ISO 8601 UTC timestamp
author channelId	string	author channel id
bodyLength	number	Chat message length

isMember	nullable boolean	is a member
videoId	string	source video id
channelId	string	source channel id

The Table below shows the Column names, data type, and Description of the corresponding columns of the Superchat Data table.

**Table 3: Superchat Data table**

Column Name	Type	Description
timestamp	string	ISO 8601 UTC timestamp
authorChannelId	string	author channel id
amount	number	purchased amount
currency	string	three-letter currency symbol
significance	number	significance
channelId	string	source channel id

The Table below shows the Column names, data type, and Description of the corresponding columns of the Deletion Events table.

**Table 4: Deletion Events table**

Column Name	Type	Description
timestamp	string	UTC timestamp
id	string	chat id

retracted	boolean	is deleted by author oneself
videoId	string	source video id
channelId	string	source channel id

The Table below shows the Column names, data type, and Description of the corresponding columns of the Bans Events table.

**Table 5: Ban Events table**

Column Name	Type	Description
timestamp	string	UTC timestamp
authorChannelId	string	channel id
videoId	string	source video id
channelId	string	source channel id

The Table below shows the Column names, data type, and Description of the corresponding columns of the Chat Statistics table.

**Table 6: Chat Statistics table**

Column Name	Type	Description
channelId	string	A unique identifier for each channel
period	string	Interested period (%Y-%M)
chats	number	Number of chats
memberChats	number	Number of chats with membership status attached

uniqueChatters	number	Number of unique chatters
uniqueMembers	number	Number of unique members appeared on the live chat
bannedChatters	number	Number of unique chatters marked as banned by mods
deletedChats	number	Number of chats deleted by mods

The Table below shows the Column names, data type, and Description of the corresponding columns of the Super Chat Statistics table.

**Table 7: Super Chat Statistics table**

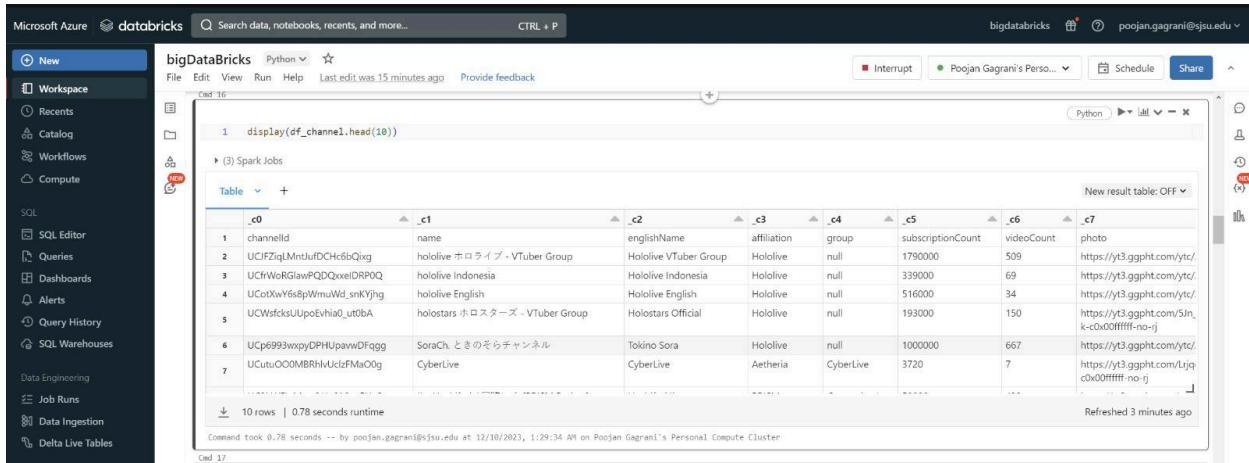
Column Name	Type	Description
channelId	string	A unique identifier for each channel
period	string	Interested period (%Y-%M)
superChats	number	Number of super chats
uniqueSuperChatters	number	Number of unique super chatters
totalSC	number	Total amount of super chats
averageSC	number	The average amount of super chat
totalMessageLength	number	Total message length
averageMessageLength	number	Average message length
mostFrequentCurrency	string	Most frequent currency
mostFrequentColor	string	Most frequent color

The Collected files are further merged into 4 separate files based on the context and relevance of the information available in them:

- Merged Data Files for Channel, Chats, Ban, and Deletion files
- Merged Data File for Channel and Chat Statistics File
- Merged Data File for Channel and Super Chat Statistics File
- Merged Data File for Channel and Super Chat File

## 3.2 Dataset Samples

The following figures show the sample VTubers raw data collected from Kaggle.



The screenshot shows a Databricks workspace interface. On the left, there's a sidebar with various options like New, Workspace, Recents, Catalog, Workflows, Compute, SQL Editor, Queries, Dashboards, Alerts, Query History, SQL Warehouses, Data Engineering, Job Runs, Data Ingestion, and Delta Live Tables. The main area has a search bar at the top. Below it, there's a notebook titled "bigDataBrics" with a Python tab selected. The code cell contains the command `display(df_channel.head(10))`. The resulting DataFrame, titled "Table", displays 10 rows of data with columns labeled c0 through c7. The data includes information such as channel ID, name, English name, affiliation, group, subscription count, video count, and photo URL. The photo URLs are all identical to <https://yt3.ggpht.com/yc/>. The last row shows a timestamp indicating the command took 0.78 seconds to run.

c0	c1	c2	c3	c4	c5	c6	c7
1	channelId	name	englishName	affiliation	group	subscriptionCount	photo
2	UCFZiqLMnJufDCHc6bQixg	hololive ホロライブ - VTuber Group	Hololive VTuber Group	Hololive	null	1790000	<a href="https://yt3.ggpht.com/yc/">https://yt3.ggpht.com/yc/</a>
3	UCfrWoRGlavPQDQxelDRP0Q	hololive Indonesia	Hololive Indonesia	Hololive	null	339000	69
4	UCotXWY6s8jWmuWd_snkYfhg	hololive English	Hololive English	Hololive	null	516000	34
5	UCWsfcksUUpoEvhqA_ut0BA	holostars ホロスターズ - VTuber Group	Holostars Official	Hololive	null	193000	150
6	UCp6993wpxyDPHUpawwDfqqg	SoraCh. ときのそらチャンネル	Tokino Sora	Hololive	null	1000000	667
7	UCutuO0M0BRHmUclzfMa0g	CyberLive	CyberLive	Aetheria	CyberLive	3720	7
							<a href="https://yt3.ggpht.com/yc/">https://yt3.ggpht.com/yc/</a>
							<a href="https://yt3.ggpht.com/yc/">https://yt3.ggpht.com/yc/</a>
							<a href="https://yt3.ggpht.com/yc/">https://yt3.ggpht.com/yc/</a>

**Fig. 1. Sample from the unprocessed Channel dataset**

**bigDataBrics** Python

File Edit View Run Help Last edit was 16 minutes ago Provide feedback

Interrupt Poojan Gagrani's Personal Compute Cluster Schedule Share

Cmd 18

```
1 display(df_chat1.head(10))
```

(5) Spark Jobs

Table + New result table: OFF

	timestamp	authorChannelId	videoId	channelId	isMember	bodyLength
1	2022-01-01T00:00:00.004000+00:00	bb3a93db098b21e406b996db3fb214dc0e2da1b2	d1C-at0rh-I	UCO_aKKYn4tvrqPjctZ6EQ	false	25
2	2022-01-01T00:00:00.008000+00:00	1ba78928726f633b6a6146e1859631388fc72	VQN6fyIm1pQ	UCU-J8uXuLZh16G-cT1naw	false	30
3	2022-01-01T00:00:00.029000+00:00	a72ae292ab609b19d582abde70a0b3aaad7509	d1C-at0rh-I	UCO_aKKYn4tvrqPjctZ6EQ	false	1
4	2022-01-01T00:00:00.032000+00:00	986dc5f63387a35dc2820affd45764ae01e	MlxU28lmal	UCIM92Ok_spNKLVB5TgwseQ	false	1
5	2022-01-01T00:00:00.044000+00:00	8cc4f6c1dc1ce5b8327464b7428964a08d0376	VQN6fyIm1pQ	UCU-J8uXuLZh16G-cT1naw	true	17
6	2022-01-01T00:00:00.045000+00:00	48ca5116ce7d7409babaa38bb5207d20ceb561	d1C-at0rh-I	UCO_aKKYn4tvrqPjctZ6EQ	false	15
7	2022-01-01T00:00:00.060000+00:00	14011-f7545710a1hv7n027h4frh7r7794R099	MlvIIJ8lmal	UICIM92Ok_spNKLVB5TgwseQ	false	1

↓ 10 rows | 42.25 seconds runtime Refreshed 3 minutes ago

Cmd 19

```
1 display(df_chat2)
```

(3) Spark Jobs

Table + New result table: OFF

	timestamp	authorChannelId	videoId	channelId	isMember	bodyLength
1	2022-01-01T00:00:00.094000+00:00	728975dd0e90dd277dde6f2b58eb399ed683cd	Ef31tOuCwAY	UCU-J8uXuLZh16G-cT1naw	false	1
2	2022-02-01T00:00:00.163000+00:00	6aaada079cd1c730802d8d33928c08a4bea027	Ef31tOuCwAY	UCU-J8uXuLZh16G-cT1naw	false	20
3	2022-02-01T00:00:00.529000+00:00	3519b50b7d7e7b78f7699dfdf4391d1edcaad16	ipV4jSMhY	UCW1KvNjqrflsfdfdzS2a	false	59
4	2022-02-01T00:00:00.534000+00:00	004cd1f4955b4a89e79fbcae33a05d2d4ca16b	CWuNAV9L5gA	UCjmCrq4TP14kguOtJ-31w	true	19
5	2022-02-01T00:00:00.590000+00:00	9d8cfdf445e264e0b8055978df0bfffde6277e4	CWuNAV9L5gA	UCjmCrq4TP14kguOtJ-31w	false	4
6	2022-02-01T00:00:00.604000+00:00	2a21be0d25642c105607bf0101565a15b2cf541	JlU0-lq5nE	UCR6qlspn62WVxvCBk1dLow	false	3
7	2022-02-01T00:00:00.647000+00:00	Na927-n090ufef163913eadaafea699rf541r803	127nNfIxFVF	UICFKCYvNvNvGvNvGvNvF5H3iw	true	3

Fig. 2. Sample from the unprocessed Chat dataset

**bigDataBrics** Python

File Edit View Run Help Last edit was 15 minutes ago Provide feedback

Interrupt Poojan Gagrani's Personal Compute Cluster Schedule Share

Cmd 17

```
1 display(df_chat_stats.head(10))
```

(3) Spark Jobs

Table + New result table: OFF

	channelId	period	chats	memberChats	uniqueChatters	uniqueMembers	bannedChatters	deletedChats
1	UC-A24dwZW7-M2klD0N6_jfA	2021-03-01	32116	7874	1112	141	1	6
2	UC-hM6YjuNYVAmUWxel9FeA	2021-03-01	1569241	378133	60399	4959	650	2243
3	UC-o-E631C2q8sAoAuM6Umg	2021-03-01	220362	25920	13101	307	79	23
4	UC01gb866djh23Ngk3A1OLQ	2021-03-01	30686	12753	1082	130	15	30
5	UC00wc36lJ90cyG9c9c-4cg	2021-03-01	446411	178018	7413	710	8	29
6	UC0QzQ6nK1EOZv512769Zw	2021-03-01	1443	0	58	0	0	2
7	Iw07TYw-IV724rauN2XMu4c_kw	2021-04-01	42976	64576	22375	1048	27	267

↓ 10 rows | 0.80 seconds runtime Refreshed 4 minutes ago

Cmd 18

```
1 display(df_chat1.head(10))
```

(5) Spark Jobs

Table + New result table: OFF

	timestamp	authorChannelId	videoId	channelId	isMember	bodyLength
1	2022-01-01T00:00:00.04000+00:00	bb3a93db098b21e406b996db3fb214dc0e2da1b2	d1C-at0rh-I	UCO_aKKYn4tvrqPjctZ6EQ	false	25
2	2022-01-01T00:00:00.08000+00:00	1ba78928726f633b6a6146e1859631388fc72	VQN6fyIm1pQ	UCU-J8uXuLZh16G-cT1naw	false	30
3	2022-01-01T00:00:00.29000+00:00	a72ae292ab609b19d582abde70a0b3aaad7509	d1C-at0rh-I	UCO_aKKYn4tvrqPjctZ6EQ	false	1
4	2022-01-01T00:00:00.32000+00:00	986dc5f63387a35dc2820affd45764ae01e	MlxU28lmal	UCIM92Ok_spNKLVB5TgwseQ	false	1
5	2022-01-01T00:00:00.044000+00:00	8cc4f6c1dc1ce5b8327464b7428964a08d0376	VQN6fyIm1pQ	UCU-J8uXuLZh16G-cT1naw	true	17
6	2022-01-01T00:00:00.045000+00:00	48ca5116ce7d7409babaa38bb5207d20ceb561	d1C-at0rh-I	UCO_aKKYn4tvrqPjctZ6EQ	false	15
7	2022-01-01T00:00:00.060000+00:00	14011-f7545710a1hv7n027h4frh7r7794R099	MlvIIJ8lmal	UICIM92Ok_spNKLVB5TgwseQ	false	1

Fig. 3. Sample from the unprocessed Chat Statistics dataset

**bigDataBrics** Python

File Edit View Run Help Last edit was 18 minutes ago Provide feedback

COMMAND TOOK 0.79 seconds -- by pojan.gagrani@jsu.edu at 12/18/2023, 1:29:43 AM on Poojan Gagrani's Personal Compute Cluster

Run all | Poojan Gagrani's Perso... | Schedule | Share |

Cmd 24

```
1 display(df_superchat1.head(10))

▶ (3) Spark Jobs
```

Table New result table: OFF

	timestamp	amount	currency	significance	authorChannelId	videoId	channelId	box
1	2022-01-01T00:00:01.329000+00:00	2	USD	2	499e0ea29912c7161c8d55016ceae1c6cbf8378c	Mix0U28lmal	UCIM92Ok_spNLV5tgwseQ	30
2	2022-01-01T00:00:01.605000+00:00	10	GBP	4	f3c2e99024eb6bc8a87a9ff2f8bc4184a396a9	B80L8ShGg0	UCaum0e1MvZf69bolulgA0bg	22
3	2022-01-01T00:00:03.979000+00:00	5	USD	3	1e195b6e99e853378d0457d14bf7fd7a4571e7	MuFMD403gbg	UCqm3BQLJfvkTsX_hmv0UmA	51
4	2022-01-01T00:00:05.695000+00:00	75	TWD	3	a9fa16a3886b001082460f9ced2e536644a40	8xFvwcfj9Og	UCsUj0dszADCGf3gNrQEu5Q	23
5	2022-01-01T00:00:06.820000+00:00	10	GBP	4	24b6ae1d29d1505b72af1daa32767028cd562d06	d1C_atlh-h	UCO_aKKYxNtvrgPjIz26EQ	33
6	2022-01-01T00:00:09.172000+00:00	400	HUF	2	81a47b531f9ad97b659615bc4a3a0ebc92092	Mix0U28lmal	UCIM92Ok_spNLV5tgwseQ	20
7	2022-01-01T00:00:09.490000+00:00	50	ARS	3	95a6d17nf5a92014df7a4f46511h1ea6f65817y9	RnRfFa4H1	IICRIIRMR54H7-R7NCle4R7NA	na
								Refreshed 1 minute ago

Command took 0.89 seconds -- by pojan.gagrani@jsu.edu at 12/18/2023, 1:29:43 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 25

```
1 display(df_superchat2)

▶ (1) Spark Jobs
```

Table New result table: OFF

	timestamp	amount	currency	significance	authorChannelId	videoId	channelId	box
1	2022-02-01T00:00:04.662000+00:00	10	USD	4	07aa633be0ac204a6dd614bd953503c5d890f15	VojnkQ3B4zU	UCy1z3o3XR1r1lFKGSUAg	
2	2022-02-01T00:00:05.012000+00:00	50	ARS	3	1e7b29d69f0645df6ba296a03f1aaacf87d96de	VojnkQ3B4zU	UCy1z3o3XR1r1lFKGSUAg	
3	2022-02-01T00:00:06.095000+00:00	250	JPY	2	0984ded7985303b3t51659ffbe1d1cf57ed3b	1i2qNfRfVE	UCFKOVgVbGmX65Rx03EtH3iw	
4	2022-02-01T00:00:22.883000+00:00	200	JPY	2	cb6705909306c4b5a41f59978ebc216625f	erR-Wc5aiKY	UCWNUxGdnN085Qf2lH4o2g	
5	2022-02-01T00:00:34.323000+00:00	610	JPY	3	5ba3f82c19c848777be205a121d523e2b622c	1i2qNfRfVE	UCFKOVgVbGmX65Rx03EtH3iw	
6	2022-02-01T00:00:42.132000+00:00	5	USD	3	8f4324b56af53b401bf862eadb899d6e35653c49	jJU0-fqSnE	UCR6qhtLpn62WVxCBK1dkLow	
7	2022-02-01T00:00:49.340000+00:00	300	IPV	2	u4d47f652773aef4x3ad504f4f0a117f-l5v0sw3	1i2qNfRfVE	IICRKNvAnVvNmX55vN3FH43iw	
								Refreshed 1 minute ago

Command took 0.80 seconds -- by pojan.gagrani@jsu.edu at 12/18/2023, 1:29:43 AM on Poojan Gagrani's Personal Compute Cluster

**Fig. 4. Sample from the unprocessed SuperChat dataset**

**bigDataBrics** Python

File Edit View Run Help Last edit was 18 minutes ago Provide feedback

COMMAND TOOK 0.59 seconds -- by pojan.gagrani@jsu.edu at 12/18/2023, 1:29:43 AM on Poojan Gagrani's Personal Compute Cluster

Run all | Poojan Gagrani's Perso... | Schedule | Share |

Cmd 24

```
1 display(df_superchat_stats.head(10))

▶ (3) Spark Jobs
```

Table New result table: OFF

	channelId	period	superChats	uniqueSuperChatters	totalSC	averageSC	totalMessageLength	averageMessageLength	mostFrequentCur
1	UCFKOVgVbGmX65Rx03EtH3iw	2021-03-01	5552	1043	3482098	627	182687	35	JPY
2	UCV5ZLJkMKMgJ3Un0bzv	2021-03-01	220	121	199004	904	5747	29	JPY
3	UcvxTidFTWBGv3MKjXK83qvJvCw	2021-03-01	2653	796	1576726	594	64393	26	JPY
4	UCNVEyYb5jH5QLmGe5gTs2g	2021-03-01	166	124	205611	1238	4853	35	USD
5	UCLgCy0RlgOW6Qb4jzQ	2021-03-01	3468	835	3773715	1088	110789	34	JPY
6	UcbcBfwndlUNqj-1995Su4A	2021-03-01	25	22	41713	1668	742	33	JPY
7	UUD94H7yDxDxem84H7vYGiRw	2021-03-01	112	55	80600	737	2238	21	IPV
									Refreshed 1 minute ago

Command took 0.59 seconds -- by pojan.gagrani@jsu.edu at 12/18/2023, 1:29:43 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 25

```
1 display(df_superchat1.head(10))

▶ (3) Spark Jobs
```

Table New result table: OFF

	timestamp	amount	currency	significance	authorChannelId	videoId	channelId	box
1	2022-01-01T00:00:01.329000+00:00	2	USD	2	499e0ea29912c7161c8d55016ceae1c6cbf8378c	Mix0U28lmal	UCIM92Ok_spNLV5tgwseQ	30
2	2022-01-01T00:00:01.605000+00:00	10	GBP	4	f3c2e99024eb6bc8a87a9ff2f8bc4184a396a9	B80L8ShGg0	UCaum0e1MvZf69bolulgA0bg	22
3	2022-01-01T00:00:03.979000+00:00	5	USD	3	1e195b6e99e853378d0457d14bf7fd7a4571e7	MuFMD403gbg	UCqm3BQLJfvkTsX_hmv0UmA	51
4	2022-01-01T00:00:05.695000+00:00	75	TWD	3	a9fa16a3886b001082460f9ced2e536644a40	8xFvwcfj9Og	UCsUj0dszADCGf3gNrQEu5Q	23
5	2022-01-01T00:00:06.820000+00:00	10	GBP	4	24b6ae1d29d1505b72af1daa32767028cd562d06	d1C_atlh-h	UCO_aKKYxNtvrgPjIz26EQ	33
6	2022-01-01T00:00:09.172000+00:00	400	HUF	2	81a47b531f9ad97b659615bc4a3a0ebc92092	Mix0U28lmal	UCIM92Ok_spNLV5tgwseQ	20
7	2022-01-01T00:00:09.490000+00:00	50	ARS	3	95a6d17nf5a92014df7a4f46511h1ea6f65817y9	RnRfFa4H1	IICRIIRMR54H7-R7NCle4R7NA	na
								Refreshed 1 minute ago

Command took 0.59 seconds -- by pojan.gagrani@jsu.edu at 12/18/2023, 1:29:43 AM on Poojan Gagrani's Personal Compute Cluster

**Fig. 5. Sample from the unprocessed SuperChat Statistics dataset**

```

1 display(df_deletion.head(10))

(3) Spark Jobs

Table ▾ + New result table: OFF ▾
timestamp id retracted videoId channelId
1 2021-02-11T09:03:53.73Z CKUGKgNQM21yOTY4NGU0Q0ZUEbZ0fKm05nRHBRIdDUGfob3R1ODRINENGY2pBRVFnZGFBb05FUTE2MTMwMzQyMjQ3MT true b208RqGxwM UCC2Uf0BKV0kV
2 2021-02-11T09:04:09.45Z CjsKGhNLvF0T1c4NGU0Q0ZUOHnyUVLk13TePReh1DSm0bk5hINTRINENGWtbnWUfVZEINTUYQS05Mw%3D%3D true bISX7M_n9RA UCI_gCybO/RlgO
3 2021-02-11T09:04:40.21Z CxUKGhNLeI4dcC04NGU0Q0Zld3.yUvkUVB:RGIREidD7mF5LUlhODRINENGUUIKs2dvZEpVNEx5QTE2MTMwMzQyMjM0NDYh3D true bISX7M_n9RA UCI_gCybO/RlgO
4 2021-02-11T09:06:26.54Z CjsKGhNMVdNmkhNGU0Q0ZUV9UyUvkUVy:FRZBh1DsplLXFMTRINENGWVhWUfZF4hNEiZyQmQ9%3D%3D true FBHae81fR3U UCNVEsYbzjH5C
5 2021-02-11T09:06:36.51Z CKUGKgNNejAO53E5NGU0Q0ZmR0f3UW9hNS04UNREidGS9TMfbNzRINENGZnvOfzFizGppvUFVZzE2MTMwMzQd44N true 2RLmw5Wb5k UCHsx4Hqa-10Rj
6 2021-02-11T09:07:07.19Z CKUGKgNMZUN4cmk5NGU0Q0ZmOHfyUVldmJBUFRREidDTWlOUXTORINENGWsh5SandvZEhTdN6QTE2MTMwMzQ0MTAyNT true FBHae81fR3U UCNVEsYbzjH5C
A%3D
7 2021-02-11T09:07:47.56Z CKUGKgNSehNy3E5NGU0Q0ZOffyUVkUVy:Sk93EdDUGj6aHNXTRINENGVB0VfZfpYSUPZzE2MTMwMzQ0NDUyNg%3D true 2RLmw5Wb5k UCHsx4Hqa-10Rj
8 2021-02-11T09:07:49.36Z CKUGKgNLvNtxh5NGU0Q0ZUOHnyUvkUVb:RIdTm1NEtpNzRINENGWhMy3dFZddLVzLdzE2MTMwMzQ0NTExNDI true 0vDlyNgVyzI UCYz_5n-uDuCh
3D

↓ 10 rows | 3.45 seconds runtime

```

```

Command took 3.45 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:29:43 AM on Poojan Gagrani's Personal Compute Cluster

```

```

1 display(df_superchat_stats.head(10))

(3) Spark Jobs

Table ▾ + New result table: OFF ▾
channelId period superChats uniqueSuperChatters totalSC averageSC totalMessageLength averageMessageLength mostFrequentCurre
1 UCFKOVgVbGmX65Rx0EtH3iw 2021-03-01 5552 1043 3482098 627 182687 35 JPY

```

```

Command took 3.45 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:29:43 AM on Poojan Gagrani's Personal Compute Cluster

```

**Fig. 6. Sample from the unprocessed Deletion Events dataset**

```

1 display(df_ban.head(10))

(3) Spark Jobs

Table ▾ + New result table: OFF ▾
timestamp authorChannelId videoId channelId
1 2021-02-11T09:52:28.631Z 28b911656881fefa650d89961c8614bf62f80f 0vDlyNgVyzI UCYz_5n-uDuChHtLo7My1HnQ
2 2021-02-11T09:54:47.501Z 20035eb4b063278be862274f66618a2cd474ee 0vDlyNgVyzI UCYz_5n-uDuChHtLo7My1HnQ
3 2021-02-11T09:58:28.22Z 524e0b589091a1cdccac7365b1ea8aacd5c657b bISX7M_n9RA UCL_gCybO/RlgOxw6Qb4qjzQ
4 2021-02-11T09:58:33.21Z 524e0b589091a1cdccac7365b1ea8aacd5c657b bISX7M_n9RA UCL_gCybO/RlgOxw6Qb4qjzQ
5 2021-02-11T10:08:31.674Z 8a6ff699826307fc3311aa3a20b91c7024fce86 -UbSxTkVjU UC-hM6YjuNYVAmUWxelr9FeA
6 2021-02-11T10:16:17.987Z 3786132c735586b0f8120bb62a59b9442b96a9b -UbSxTkVjU UC-hM6YjuNYVAmUWxelr9FeA
7 2021-02-11T10:22:01.197Z a3a9f1eRhnH51ASdnf5anfa40a9a2c5-7977RA .1JhsVtVaiII 111-hM6YjuNYVAmUWxelr9FeA

↓ 10 rows | 4.40 seconds runtime

```

```

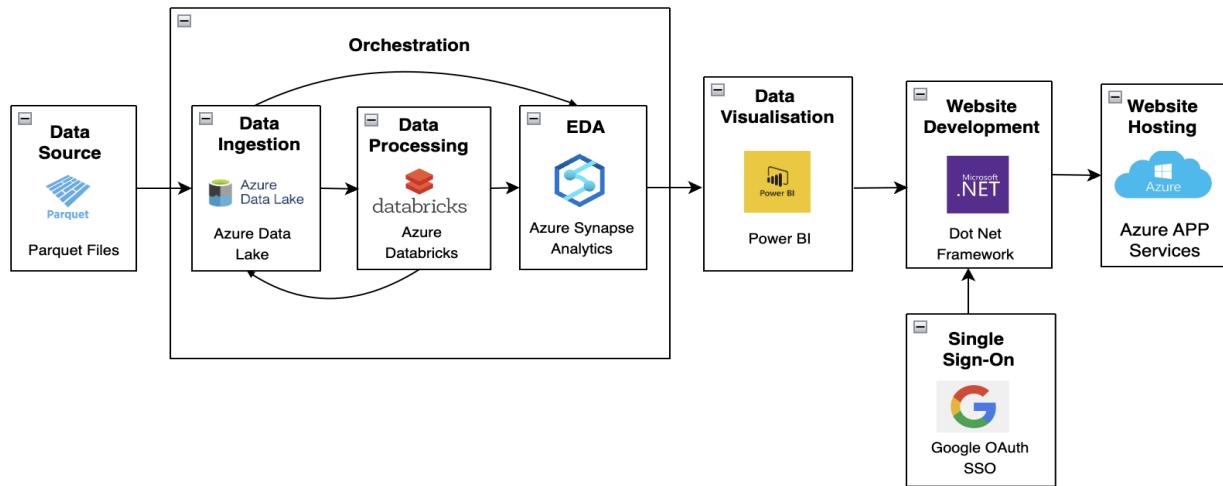
Command took 4.40 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:29:33 AM on Poojan Gagrani's Personal Compute Cluster

```

**Fig. 7. Sample from the unprocessed Ban Events dataset**

# Chapter 4 Architecture and Project Flow

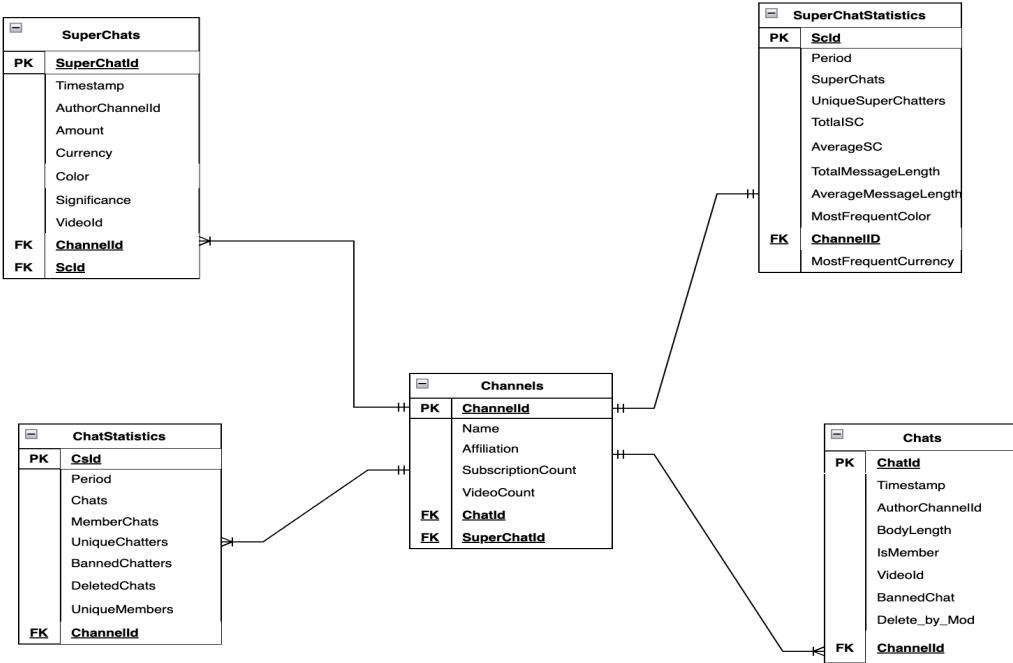
## 4.1 Architecture



**Fig. 8. Diagram of the Project Architecture**

## 4.2 Data Modeling and ER Diagram

The merged files were created in the Azure Data Lake following which a series of data cleaning operations were performed in Azure Databricks. The data model designed for Vtubers data consists of four-dimensional tables and one fact table. The dimensional table includes the SuperChats, Chats, SuperChat statistics, and Chat Statistics data. Apart from the Super Chat Statistics table, all three tables have a mandatory one-to-many relationship with the Channel fact table on the one side and the dimensional table on the many sides. The channel fact table consists of quantitative data such as the total number of videos and subscription counts. Each dimensional table provides context to the channel as it contains descriptive attributes. The dimensional tables maintain a unique identifier column to determine every instance of dimension and are connected to the channel fact table by the ChannelID column. The below figure represents the data modeling designs of this study.



**Fig. 9. ER Diagram**

### 4.3 Project Flow

The project is implemented by utilizing a comprehensive set of tools within the Azure environment for effective data flow. The Azure Data Lake serves as the starting point for ingesting a humongous volume of data extracted from Kaggle. After importing the data to Data Lake, rigorous steps of pre-processing the data are performed using Azure Databricks. The preprocessed data is then used to understand the underlying pattern, find outliers, and examine the characteristics of the data. The analytical results are then visualized using PowerBI dashboards. The dashboard shows the in-depth analysis and is interactive to further filter the visualization as required. The visualizations are then published to the website using the Dot Net framework. The project incorporates Google OAuth Single Sign-On (SSO) to ensure safe and efficient user authentication. This way users can seamlessly access the website. Finally, the built

web applications are deployed in a dependable and scalable manner through the use of Azure Web Services for website hosting. The overall data flow of the project reflects an integrated approach for data management, analysis, visualization, and website development in the Azure ecosystem. The table below shows the list of tools used in this project along with the corresponding purpose for each tool.

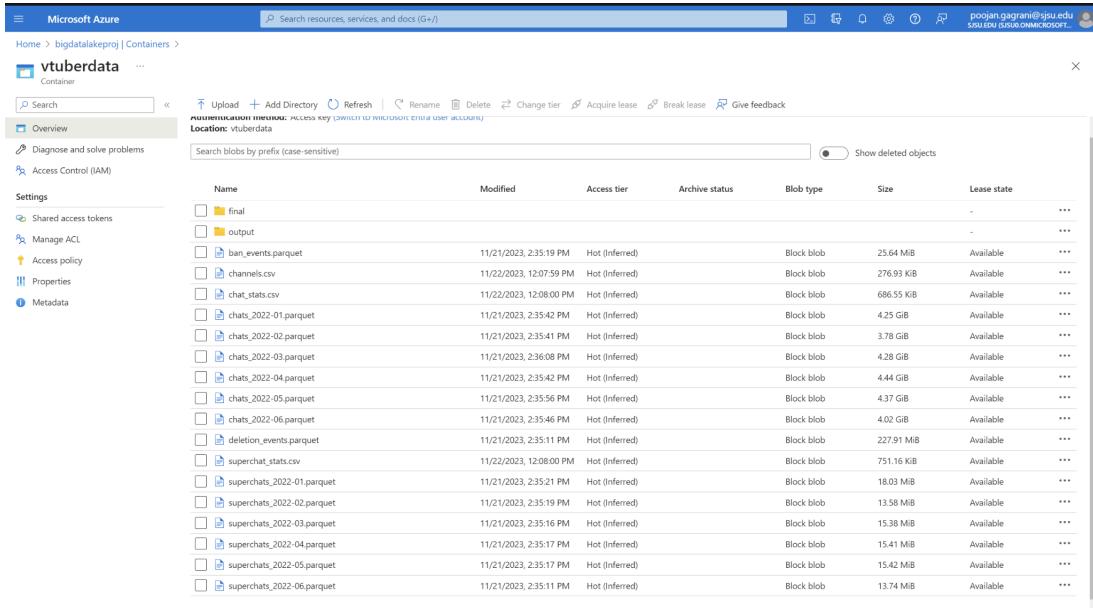
**Table 8: Tools used along with the purpose**

Tool	Purpose
Azure Data Lake	For Scalable Data Ingestion and allows storage of vast amounts of data
Azure Databricks	Used for large-scale data processing
Azure Synapse Analytics	Exploratory Data Analysis
Power BI	Analysis and Visualizations
Dot Net Framework	Website Development
Google OAuth SSO	Secure and Streamlined User Authentication.
Azure Web Services	Website Hosting

## **Chapter 5 Data Engineering**

### **5.1 Importing Data**

The raw parquet files are first loaded into the Azure data lake. The data files loaded contained Vtubers ban events, channel details, chat statistics details, deletion events, super chat statistics, and super chat details.



**Fig. 10. Raw data files loaded into Azure data lake container**

## 5.2 Data Cleaning

It is a crucial phase in any data analytics project and involves various methods that help ensure data consistency and integrity. All the raw Vtuber data was loaded into Azure Databricks and then the data was read using spark data frames and following data cleaning operations were performed. The data cleaning process involves the following steps.

1. Setting up Azure Databricks spark notebook and reading files
2. Filtering and reducing ban data from Jan to Feb 2022
3. Filtering and reducing chat stats from Jan to Feb 2022
4. Filtering and reducing deletion from Jan to Feb 2022
5. Filtering and reducing superchat stats from Jan to Feb 2022
6. Filtering and reducing superchat stats from Jan to Feb 2022
7. Dropping Name, Group, and Photo columns from channel
8. Mapping the color for the significance field in Superchat

## 9. Generating chatId for chat

## 10. Generating superChatId for a chat

The data cleaning code snippets are attached below.

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P bigdataanalysis Python File Edit View Run Help Last edit was 9 minutes ago Provide feedback Run all Schedule Share

Code 1:

```
1 from pyspark.sql.functions import col, to_timestamp
2 from datetime import datetime
3 from pyspark.sql.functions import col, lit, count, when, to_timestamp, monotonically_increasing_id, sum, to_date, row_number
4 from pyspark.sql.types import StructType, StructField, StringType, IntegerType
5 from pyspark.sql.window import Window
```

Code 2:

```
1 storage_account_name = "bigdatalakeproj"
2 filesystem_name = "vtuberdata"
3 access_key = [REDACTED]
4
5 spark.conf.set(f"fs.azure.account.key.{storage_account_name}.dfs.core.windows.net", access_key)
```

Code 3:

```
1 file_format_parquet = "parquet" # e.g., "csv", "parquet", "json", etc.
2 file_format_csv = "csv"
```

Code 4:

```
1 file_path_ban = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/ban_events/ban_events.csv"
2 file_path_channel = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/channel/channels.csv"
3 file_path_chat_stats = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/chat/chat_stats.csv"
4 file_path_chat1 = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/chat1/chat1.csv"
5 file_path_chat2 = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/chat2/chat2.csv"
6 file_path_chat3 = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/chat3/chat3.csv"
7 file_path_deletion = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/deletion/deletion.csv"
8 file_path_superchat_stats = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/superchat_stats/superchat_stats.csv"
9 file_path_superchat1 = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/superchat1/superchat1.csv"
10 file_path_superchat2 = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/superchat2/superchat2.csv"
11 file_path_superchat3 = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/output/superchat3/superchat3.csv"
```

Fig. 11. Setting up Azure Databricks spark notebook and reading files

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P bigdataanalysis Python File Edit View Run Help Last edit was now Provide feedback Interrupt Schedule Share

Call 27:

Data Cleaning

```
1 start_date = datetime(2022, 1, 1)
2 end_date = datetime(2022, 3, 31, 23, 59, 59)
3
4 df_ban = df_ban.withColumn("timestamp", to_timestamp("timestamp"))
5 df_ban_filtered = df_ban.filter((col("timestamp") > start_date) & (col("timestamp") < end_date))

# df_ban: pyspark.sql.DataFrame = [timestamp: timestamp, authorChannelId: string ... 2 more fields]
# df_ban_filtered: pyspark.sql.DataFrame = [timestamp: timestamp, authorChannelId: string ... 2 more fields]
```

Command took 0.14 seconds -- by poojan.gagrani@jsu.edu at 12/18/2023, 1:36:45 AM on Poojan Gagrani's Personal Compute Cluster

Call 28:

```
1 display(df_ban_filtered.tail(10))
```

(3) Spark Jobs

Table + New result table: OFF

timestamp	authorChannelId	videoId	channelId
2022-03-31T23:40:19.976Z	a43961849e0de99cb5f3600a2b68660f42f25	q9yjd0ThsA	UCm34Ig0Zk2TinWVYXpA
2022-03-31T23:42:05.262Z	a44cd9d57ed539772574dacf8f300b14d88e	d5rre3bpEbw	UCLAU_k_A_uFje7H1TgJnWr
2022-03-31T23:42:05.262Z	a44cd9d57ed539772574dacf8f300b14d88e	d5rre3bpEbw	UCLAU_k_A_uFje7H1TgJnWr
2022-03-31T23:45:03.386Z	5a396c8088e1c629f56de9ab90669ef813a	d5rre3bpEbw	UCLAU_k_A_uFje7H1TgJnWr
2022-03-31T23:45:28.176Z	5d457870fb309ba9004ff0fded5367799586b74	Wg7VshyOE	UClyNcoz4pWzXflub0DvULQ
2022-03-31T23:46:06.632Z	1b4dad1d510158814037d63d384046950fa6	aMSPS-hklJg	UCYfTzj03XRt1lFKGSUAg
2022-03-31T23:47:11.457Z	c779e99e2e1shv1aa3a7a16496664cfh-1-42%7d	HNvGivYndII	UCf6cR0TnCvH11Qew9lFf7A
10 rows   2.09 seconds runtime			

Command took 2.09 seconds -- by poojan.gagrani@jsu.edu at 12/18/2023, 1:36:46 AM on Poojan Gagrani's Personal Compute Cluster

Call 29:

```
1 start_peroid = "2022-01"
2 end_peroid = "2022-02"
```

Fig. 12. Filtering and reducing ban data from Jan to Feb 2022

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P bigDataBrics Data Engineering Data Ingestion Machine Learning Features Serving Collapse menu

**bigDataBrics Python**

File Edit View Run Help Last edit was 2 minutes ago Provide feedback

Command took 2.09 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:36:46 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 29

```

1 start_period = "2022-01"
2 end_period = "2022-03"
3
4 # Filter the Dataframe
5 df_chat_stats_filtered = df_chat_stats.filter((col("period") >= start_period) & (col("period") <= end_period))
6 df_chat_stats_filtered = df_chat_stats_filtered.withColumn("period", col("period").substr(1, 7))

# df_chat_stats_filtered: pyspark.sql.DataFrame = [channelid: string, period: string ... 6 more fields]

```

Command took 0.18 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:36:47 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 30

```

1 display(df_chat_stats_filtered)

```

(1) Spark Jobs

Table New result table: OFF

channelid	period	chats	memberChats	uniqueChatters	uniqueMembers	bannedChatters	deletedChatters
1 UC-A2dnZW7-M2kID0N6_ifA	2022-01	68741	27091	2203	208	0	0
2 UC-zuE0NEFxPvLq0Kwbuw	2022-01	61448	0	927	0	2	0
3 UC-6rZgmxZSlbq786j3RD5ow	2022-01	687872	373457	13830	2515	6	15
4 UC-DzOMtruxzMD0HtG9gHQ	2022-01	1840	1219	89	31	1	3
5 UC-ENagPkl0LoJu2lnlTZVZDHQ	2022-01	13	1	12	1	0	0
6 UC-LsfNFBby1AwPFxbqPUQTEQ	2022-01	9068	2418	437	33	0	0
7 UC-CCeuvnnnRhf7wVnu	2022-01	57873	36210	1170	151	6	0

2,538 rows | 0.58 seconds runtime

Command took 0.58 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:36:48 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 31

```

1 start_date = datetime(2022, 1, 1)
2 end_date = datetime(2022, 3, 31, 23, 59, 59)
3

```

10 rows | 0.01 seconds runtime

Fig 13. Filtering and reducing chat stats from Jan to Feb 2022

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P bigDataBrics Data Engineering Data Ingestion Machine Learning Features Serving Collapse menu

**bigDataBrics Python**

File Edit View Run Help Last edit was 3 minutes ago Provide feedback

Command took 0.15 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:36:49 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 32

```

1 start_date = datetime(2022, 1, 1)
2 end_date = datetime(2022, 3, 31, 23, 59, 59)
3
4 df_deletion = df_deletion.withColumn("timestamp", to_timestamp("timestamp"))
5 df_deletion_filtered = df_deletion.filter((col("timestamp") >= start_date) & (col("timestamp") <= end_date))

# df_deletion: pyspark.sql.DataFrame = [timestamp: timestamp, id: string ... 3 more fields]
# df_deletion_filtered: pyspark.sql.DataFrame = [timestamp: timestamp, id: string ... 3 more fields]

```

Command took 0.15 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:36:49 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 33

```

1 display(df_deletion_filtered)

```

(1) Spark Jobs

Table New result table: OFF

timestamp	id	retracted	videoid	channelid
1 2022-01-01T00:00:24.216Z	CKUKGkNMJ3JuOe9mal9VQ0ZTM0R3Z1FkbmDRTGIREidDUG5jOE9xZWpFVUNGZFZNHdJZgWY0dSUTE2NDASOTUyMDcwMjA	true	d1C-at0rh-l	UCO_aKKYxn4b
2 2022-01-01T00:00:29.856Z	CjoGkGNMN28tUltma9VQ0Z2WEr3Z1fkcdZjQXBREhxDStdelay1PZWpfVUNGUUTzVmfdZFdFaQUfQ50x	true	Mix0U28imal	UCIM920K_spn
3 2022-01-01T00:00:35.066Z	CKUKGkNNNrnTxVma9VQ0Z2KNHfyUVikSG9R0p3EidDTENObWJlZmpfVUNGWFdrd2dvZG9abONldzE2NDASOTUyMjM1ODQ	true	Mix0U28imal	UCIM920K_spn
4 2022-01-01T00:01:00.137Z	CKUKGkNkZlpnZEmal9VQ0Z1VXRlyUlkVwpZTndRIdDSxpQMGf5ZmpfVUNGvZkOVFvZE4d0hYzzE2NDASOTUyMtzbODc%3D	true	d1C-at0rh-l	UCO_aKKYxn4b
5 2022-01-01T00:01:37.813Z	CKUKGkNQLvd3dWma9VQ0ZSSU1yUVikelpjQn3EidDSUNONHVxZmpfVUNGWpvn10FZZH3d01ndzE2NDASOTUyODMzdNk	true	40f5povAZM	UCBifsaCvgBa1
6 2022-01-01T00:02:27.846Z	CKUKGkNJS2xnXy1mal9VQ0ZXVE3Z1FkdnJwTxNREidDUG5jOE9xZWpFVUNGZFZNHdJZgWY0dSUTE2NDASOTUzMzE1NTQ%	true	d1C-at0rh-l	UCO_aKKYxn4b
7 2022-01-01T00:02:29.71Z	CKUKGkNlcVNzNeInal9VQ0Zt0cxZ0Fkc2NjTD3EidDUHY4Z3VpZmpfVUNGWUqlqfVZzINE5RUte2NDASOTUzMTMyMz%3D	false	BBDL8EshGg0	UCAmu0e1Mv2

10,000 rows | Truncated data | 3.55 seconds runtime

Command took 3.55 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:37:54 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 33

Fig. 14. Filtering and reducing deletion from Jan to Feb 2022

```

File Edit View Run Help Last edit was 3 minutes ago Provide feedback
bigDataBricks Interrupt Poojan Gagrani's Perso... Share
New Workspace Recents Catalog Workflows Compute SQL SQL Editor Queries Dashboards Alerts Query History SQL Warehouses Data Engineering Job Runs Data Ingestion Delta Live Tables Machine Learning Experiments Features Models Serving Marketplace Partner Connect Collapse menu
Search data, notebooks, recents, and more...
CTRL + P
bigDataBricks Python
4 # Filter the DataFrame
5 df_superchat_stats_filtered = df_superchat_stats.filter((col("period") >= start_period) & (col("period") <= end_period))
6 df_superchat_stats_filtered = df_superchat_stats_filtered.withColumn("period", col("period").substr(1, 7))
7
8 df_superchat_stats_filtered: pyspark.sql.dataframe.DataFrame = [channelId: string, period: string, ... 8 more fields]
Command took 0.17 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:37:55 AM on Poojan Gagrani's Personal Compute Cluster
Ced 34
1 display(df_superchat_stats_filtered.tail(10))
2
3 (3) Spark Jobs
Table + New result table: OFF
channelId period superChats uniqueSuperChatters totalISC averageSC totalMessageLength averageMessageLength mostFrequentCur
1 UC4zWdCrInXmpVkjOgAHsDw 2022-03 2 2 1000 500 103 51 JPY
2 UC0yRTjbPuS1ZA1Hxdsg4A 2022-03 77 43 73913 959 1248 16 JPY
3 UCAhL_Wjym8XA75algwV-Q 2022-03 59 44 88059 1492 1256 21 JPY
4 UCDtKU6WRdcKAh-6o_zmA 2022-03 18 15 20031 1112 193 10 JPY
5 UCKDhdDpoE9lurDuyAGauNnw 2022-03 1 1 500 500 13 13 JPY
6 UCEV4UTLqkWq0lkkgCf0WKQ 2022-03 2 2 323 161 18 9 PEN
7 11rvcvfnRwA+nVM1? VMRn9aw 2022-03 3 3 3536 1178 29 9 TWD
↓ 10 rows | 0.90 seconds runtime
Command took 0.90 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:37:55 AM on Poojan Gagrani's Personal Compute Cluster
Ced 35
1 print("Chat 1", df_chat1.count())
2 print("Chat 2", df_chat2.count())
3 print("Chat 3", df_chat3.count())
4
5 (6) Spark Jobs
Chat 1 100378944
Chat 2 89228784
Chat 3 100558615
Command took 2.60 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:37:56 AM on Poojan Gagrani's Personal Compute Cluster

```

**Fig. 15. Filtering and reducing superchat stats from Jan to Feb 2022**

```

File Edit View Run Help Last edit was 5 minutes ago Provide feedback
bigDataBricks Interrupt Poojan Gagrani's Perso... Share
New Workspace Recents Catalog Workflows Compute SQL SQL Editor Queries Dashboards Alerts Query History SQL Warehouses Data Engineering Job Runs Data Ingestion Delta Live Tables Machine Learning Experiments Features Models Serving Marketplace Partner Connect Collapse menu
Search data, notebooks, recents, and more...
CTRL + P
bigDataBricks Python
1 null_counts_ban = df_ban_filtered.select([count(when(col(c).isNull(), c)).alias(c) for c in df_ban_filtered.columns])
2 null_counts_ban.show()
3
4 (2) Spark Jobs
+-----+-----+-----+
|timestamp|author channelId|videoId|channelId|
+-----+-----+-----+
| 0| 0| 0| 0|
+-----+-----+-----+
Command took 1.16 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:37:56 AM on Poojan Gagrani's Personal Compute Cluster
Ced 37
1 null_counts_channel = df_channel.select([count(when(col(c).isNull(), c)).alias(c) for c in df_channel.columns])
2 null_counts_channel.show()
3
4 (2) Spark Jobs
+-----+-----+-----+
|_c0|_c1|_c2|_c3|_c4|_c5|_c6|_c7|
+-----+-----+-----+
| 0| 0| 0| 0| 929| 0| 0| 0|
+-----+-----+-----+
Command took 0.79 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:37:56 AM on Poojan Gagrani's Personal Compute Cluster
Ced 38
1 display(df_channel.head(1))
2
3 (3) Spark Jobs
Table + New result table: OFF
_c0 _c1 _c2 _c3 _c4 _c5 _c6 _c7
1 channelId name englishName affiliation group subscriptionCount videoCount photo

```

**Fig. 16. Filtering and reducing channel stats from Jan to Feb 2022**

```

1 df_channel = df_channel.drop("name", "group", "photo")
2
3 display(df_channel)

```

Command took 0.09 seconds -- by pojan.gagrani@jsu.edu at 12/10/2023, 1:37:59 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 43

channelId	englishName	affiliation	subscriptionCount	videoCount
1 UCXW4MqCQn-jCaxIX-nn-BYg	Nagao Kei	Nijisanji	324000	884
2 UCG0zBZV_QMP4MtWg6ljhEA	Shu Yamine	Nijisanji	681000	173
3 UCG6DR5VzrPAp0ZV7-HB0w	Maizuru Yokato	Independents	26900	599
4 UCCGnJy6m4RQHQH_2mkfjCQ	Jaret Fajrianto	Independents	85000	567
5 UCZSDNzqjBzbIO_ldnXg	Taka Radjiman	Nijisanji	115000	764
6 UCNUS0XhN1Tf7ldd0MpA9	Tsurugi Nen	Tsunderia	6110	102
7 i1rFQFwvRvnl4d_ArIHeR-Nuv	Shinni Aila	WACTRNR	121000	157

1,358 rows | 2.20 seconds runtime

Command took 2.20 seconds -- by pojan.gagrani@jsu.edu at 12/10/2023, 1:38:00 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 44

```

1 null_counts_chat_stats = df_chat_stats_filtered.select([count(when(col(c).isNull(), c)).alias(c) for c in df_chat_stats_filtered.columns])
2 null_counts_chat_stats.show()

```

Command took 0.09 seconds -- by pojan.gagrani@jsu.edu at 12/10/2023, 1:38:00 AM on Poojan Gagrani's Personal Compute Cluster

Fig. 17. Dropping Name, Group, and Photo columns from channel

Cmd 56

```

1 df_superchat1 = df_superchat1.withColumn(
2   "color",
3   when(col("significance") == 1, "blue")
4   .when(col("significance") == 2, "lightblue")
5   .when(col("significance") == 3, "green")
6   .when(col("significance") == 4, "yellow")
7   .when(col("significance") == 5, "orange")
8   .when(col("significance") == 6, "magenta")
9   .when(col("significance") == 7, "red")
10  .otherwise("unknown") # Handling cases where significance is outside 1-7
11 )

```

df\_superchat1: pyspark.sql.dataframe.DataFrame = [timestamp: string, amount: double ... 7 more fields]

Command took 0.19 seconds -- by pojan.gagrani@jsu.edu at 12/10/2023, 1:38:08 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 57

```

1 df_superchat2 = df_superchat2.withColumn(
2   "color",
3   when(col("significance") == 1, "blue")
4   .when(col("significance") == 2, "lightblue")
5   .when(col("significance") == 3, "green")
6   .when(col("significance") == 4, "yellow")
7   .when(col("significance") == 5, "orange")
8   .when(col("significance") == 6, "magenta")
9   .when(col("significance") == 7, "red")
10  .otherwise("unknown") # Handling cases where significance is outside 1-7
11 )

```

df\_superchat2: pyspark.sql.dataframe.DataFrame = [timestamp: string, amount: double ... 7 more fields]

Command took 0.19 seconds -- by pojan.gagrani@jsu.edu at 12/10/2023, 1:38:08 AM on Poojan Gagrani's Personal Compute Cluster

Cmd 58

```

1 df_superchat3 = df_superchat3.withColumn(
2   "color",
3   when(col("significance") == 1, "blue")
4   .when(col("significance") == 2, "lightblue")

```

Fig. 18. Mapping the color for the significance field in superchat

```

1 df_chat1_filtered = df_chat1_filtered.withColumn("chatId", monotonically_increasing_id())
2
3 df_chat1_filtered: pyspark.sql.dataframe.DataFrame = [timestamp: string, authorChannelId: string ... 5 more fields]
Command took 0.09 seconds -- by poojan.gagrani@jsu.edu at 12/10/2023, 1:38:15 AM on Poojan Gagrani's Personal Compute Cluster

```

```

1 display(df_chat1_filtered)
Cancel Running command...
(1) Spark Jobs

```

```

1 # Assuming the row counts are given as follows
2 count_df_chat_filtered1 = 3180000
3 count_df_chat_filtered2 = 2800000
4
5 # Calculate the offset for df_chat2 and df_chat3
6 offset_df_chat_filtered2 = count_df_chat_filtered1
7 offset_df_chat_filtered3 = count_df_chat_filtered1 + count_df_chat_filtered2

```

```

1 df_chat2_filtered = df_chat2_filtered.withColumn("chatId", monotonically_increasing_id() + offset_df_chat_filtered2)
2 df_chat3_filtered = df_chat3_filtered.withColumn("chatId", monotonically_increasing_id() + offset_df_chat_filtered3)

```

```

1 display(df_chat2_filtered)

```

**Fig. 19. Generating chatId for chat**

```

1 df_superchat1 = df_superchat1.withColumn("superChatId", monotonically_increasing_id() + offset_df_superchat1)
2 df_superchat2 = df_superchat2.withColumn("superChatId", monotonically_increasing_id() + offset_df_superchat2)
3 df_superchat3 = df_superchat3.withColumn("superChatId", monotonically_increasing_id() + offset_df_superchat3)

```

```

1 df_superchat1: pyspark.sql.dataframe.DataFrame = [timestamp: string, amount: double ... 8 more fields]
2 df_superchat2: pyspark.sql.dataframe.DataFrame = [timestamp: string, amount: double ... 8 more fields]
3 df_superchat3: pyspark.sql.dataframe.DataFrame = [timestamp: string, amount: double ... 8 more fields]

```

```

1 display(df_superchat1)
Cancel Waiting to run...
(1) Spark Jobs

```

timestamp	amount	currency	significance	authorChannelId	videoId	channelId
2022-01-01T00:00:01.329000+00:00	2	USD	2	499e0ea29912c7161cd55016ceae1c6cbf8378c	Mix0U28lmal	UCIM92Ok_spNKLVB5TgwseQ
2022-01-01T00:00:01.605000+00:00	10	GBP	4	f3c2e990284eb6bcab87af91f28bc4194a396g9	BBL08EshGg0	UCAume01MVZf69b0luxA0Bg
2022-01-01T00:00:03.979000+00:00	5	USD	3	1e195b6e99fe053357d0457d14b7fd7a4571e7	MuFMD403gbg	UCqm3BQLUfvTsX_hvm0UmA
2022-01-01T00:00:05.695000+00:00	75	TWD	3	a9fa16a3886b0001082b460fa9ced2e53664a40	8Fwvcl9jOg	UCJuj0dszADCGrb3NgfEuSgQ
2022-01-01T00:00:06.820000+00:00	10	GBP	4	24b6aa7d2d915b027a2f1da32f6702cd562d06	d1C-at0h-l	UCO_akKXvn4vrqfjCzZ6EQ
2022-01-01T00:00:09.172000+00:00	400	HUF	2	81a47b931fd9ada97b659615bca46a30ebc92092	Mix0U28lmal	UCIM92Ok_spNKLVB5TgwseQ
2022-01-01T00:00:09.172000+00:00	50	ARS	3	94ef41rnfk=2037n4f7a+d4f1h1+auhV65R17+rQ	RnRxFhAfHrI	11^RI IRMRQd1 H7rP7YvtaoR7na

```

1 df_chat_stats_filtered = df_chat_stats_filtered.withColumn("chatStatsId", monotonically_increasing_id())
2
3 df_chat_stats_filtered: pyspark.sql.dataframe.DataFrame = [channelId: string, period: string ... 7 more fields]

```

**Fig. 20. Generating superChatId for chat**

## 5.3 ETL process

The ETL pipeline was built in with the source being Datalake the data was loaded into Databricks where transform operations were performed, and a schema was achieved having four tables Channels, Chats, Chat Statistics, SuperChat, Superchat Statistics, and the destination was Azure Synapse Analytics where the Load operation was done.

On successful completion of the initial data cleaning the data was then extracted from the data lake using spark dataframes on Azure Databricks and the data was merged hence the transformations were applied where the final output tables were Channels, Chats, Chat Statistics, SuperChat, Superchat Statistics.

### a. Extracting Data

The data is extracted from the data lake using the Azure data bricks. The data is read into spark data frames and further transformations are applied.

```
bigDataAnalysis Python ▾ ★
File Edit View Run Help Last edit was.now Provide feedback
Run all Poojan Gagrani's Perso... Schedule Share
bigDataAnalysis
1 from pyspark.sql.functions import col, to_timestamp
2 from datetime import datetime
3 from pyspark.sql.functions import col, lit, count, when, to_timestamp, monotonically_increasing_id, sum, to_date, row_number
4 from pyspark.sql.types import StructType, StructField, StringType, IntegerType
5 from pyspark.sql.window import Window
Command took 0.30 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:59:42 AM on Poojan Gagrani's Personal Compute Cluster
Cud 2
1 storage_account_name = "bigdatalakeproj"
2 file_system_name = "vtuberdata"
3 access_key = "015rP4tM0Th/Tq9Bslp9KxxXashfCKSY03bs9II64gk8xEoy4+POv@asASbrSzSBCCuXBK9kg+AStB0GyKg="
4
5 spark.conf.set("fs.azure.account.key.{storage_account_name}.dfs.core.windows.net", access_key)
Command took 0.10 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:59:45 AM on Poojan Gagrani's Personal Compute Cluster
Cud 3
1 file_format_parquet = "parquet" # e.g., "csv", "parquet", "json", etc.
2 file_format_csv = "csv"
Command took 0.07 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:59:46 AM on Poojan Gagrani's Personal Compute Cluster
Cud 4
1 file_path_ban = f"abfss://{{file_system_name}}@{{storage_account_name}}.dfs.core.windows.net/output/ban_events/ban_events.csv"
2 file_path_channel = f"abfss://{{file_system_name}}@{{storage_account_name}}.dfs.core.windows.net/output/channel/channels.csv"
3 file_path_chat_stats = f"abfss://{{file_system_name}}@{{storage_account_name}}.dfs.core.windows.net/output/chat_stats/chat_stats.csv"
4 file_path_chat1 = f"abfss://{{file_system_name}}@{{storage_account_name}}.dfs.core.windows.net/output/chat1/chat1.csv"
5 file_path_chat2 = f"abfss://{{file_system_name}}@{{storage_account_name}}.dfs.core.windows.net/output/chat2/chat2.csv"
6 file_path_chat3 = f"abfss://{{file_system_name}}@{{storage_account_name}}.dfs.core.windows.net/output/chat3/chat3.csv"
7 file_path_deletion = f"abfss://{{file_system_name}}@{{storage_account_name}}.dfs.core.windows.net/output/deletion/deletion.csv"
8 file_path_superchat1 = f"abfss://{{file_system_name}}@{{storage_account_name}}.dfs.core.windows.net/output/superchat1/superchat1.csv"
9 file_path_superchat2 = f"abfss://{{file_system_name}}@{{storage_account_name}}.dfs.core.windows.net/output/superchat2/superchat2.csv"
10 file_path_superchat3 = f"abfss://{{file_system_name}}@{{storage_account_name}}.dfs.core.windows.net/output/superchat3/superchat3.csv"
11
Command took 0.06 seconds -- by poojan.gagrani@sjtu.edu at 12/10/2023, 1:28:26 AM on Poojan Gagrani's Personal Compute Cluster
Cud 5
```

Fig. 21. Extracting cleaned files from the pipeline

### b. Transforming Data

The data transformations are applied to where the Chat and Superchat data are merged

first, then after merging the chat and deletion events data frames are combined, and later chat and ban events are merged. Finally, the Chat and Superchats are merged with Channel, and the final four resultant tables are created

The screenshot shows a Microsoft Azure DataBricks workspace. The left sidebar includes sections for Microsoft Azure, DataBricks, Workspace, Catalog, Workflows, Compute, SQL, SQL Editor, Queries, Dashboards, Alerts, Query History, SQL Warehouses, Data Engineering, Job Runs, Data Ingestion, Delta Live Tables, Machine Learning, Experiments, Features, Models, Serving, Marketplace, Partner Connect, and a Collapse menu. The main area displays a notebook titled "bigDataAnalysis" in Python. The notebook contains several code cells and their outputs. One cell shows the creation of a DataFrame "df\_chat" by unioning three datasets. Another cell displays the DataFrame. A third cell shows a table of data with columns: timestamp, authorChannelId, videoid, channelId, isMember, bodyLength, and chatid. The table has 10 rows of sample data. The bottom of the screen shows the DataBricks navigation bar with options like Run all, Schedule, Share, and a user profile.

```
1 df_chat = df_chat1.union(df_chat2).union(df_chat3)
2
3 # df_chat: pyspark.sql.DataFrameType = [timestamp: timestamp, authorChannelId: string ... 5 more fields]
4
5 Command took 0.12 seconds -- by pojan.gagrani@sjsu.edu at 12/10/2023, 2:08:46 AM on Poojan Gagrani's Personal Compute Cluster
6 Cud 20
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
```

	timestamp	authorChannelId	videoid	channelId	isMember	bodyLength	chatid
1	2022-01-31T00:00:00.016Z	e06b70be1759107a92c1aa23e66968477f540	SjPE092z5w	UC6tSB9Tn0Of010Beo9UEJZA	false	8	0
2	2022-01-31T00:00:00.035Z	319c30b506c44dcf39ef65e2f679a27cd99d6	SjPE092z5w	UC6tSB9Tn0Of010Beo9UEJZA	false	1	1
3	2022-01-31T00:00:00.112Z	4f9387e97dc1a92c26fb246fd624936ee78228	SjPE092z5w	UC6tSB9Tn0Of010Beo9UEJZA	true	4	2
4	2022-01-31T00:00:00.133Z	ce161eff5470204686df69d9455e9432d950e	SjPE092z5w	UC6tSB9Tn0Of010Beo9UEJZA	false	13	3
5	2022-01-31T00:00:00.146Z	8ca95be338f06fbcb394fe62088484585b1	SjPE092z5w	UC6tSB9Tn0Of010Beo9UEJZA	false	13	4
6	2022-01-31T00:00:00.209Z	e011f54d30ae1e642a093d559f5968	SjPE092z5w	UC6tSB9Tn0Of010Beo9UEJZA	true	11	5
7	2022-01-31T00:00:00.216Z	07774ad57a71ff74767297c4b4667rrrrd600	SjPE092z5w	UC6tSB9Tn0Of010Beo9UEJZA	false	11	6

10,000 rows | Truncated data | 0.58 seconds runtime  
Refreshed 6 minutes ago

Command took 0.58 seconds -- by pojan.gagrani@sjsu.edu at 12/10/2023, 2:08:52 AM on Poojan Gagrani's Personal Compute Cluster

Cud 21

```
1 df_deletion_filtered_test = df_deletion.dropDuplicates(["videoid", "channelId"])
2
3 # df_deletion_filtered_test: pyspark.sql.DataFrameType = [timestamp: timestamp, id: string ... 4 more fields]
4
5 Command took 0.12 seconds -- by pojan.gagrani@sjsu.edu at 12/10/2023, 2:08:55 AM on Poojan Gagrani's Personal Compute Cluster
6 Cud 22
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
```

1 deletion\_cols = df\_deletion\_filtered\_test.select("videoid", "channelId", "deleted\_by\_mod")
2
3 df\_chat\_merged = df\_chat.join(deletion\_cols, on=["videoid", "channelId"], how="left")

**Fig. 22. Merging Chat data**

Microsoft Azure Search data, notebooks, recents, and more... CTRL + P bigdatabricks pojan.gagrani@sjjsu.edu

New Workspace Catalog Compute SQL Queries Alerts Data Engineering Machine Learning Features Serving Partner Connect

bigDataAnalysis Python File Edit View Help Last edit was 6 minutes ago Provide feedback Run all Pojan Gagrani's Personal Compute Cluster Share

File Edit View Help Last edit was 6 minutes ago Provide feedback Run all Pojan Gagrani's Personal Compute Cluster

Command Took 0.08 seconds -- by pojan.gagrani@sjjsu.edu at 12/10/2023, 2:01:05 AM on Pojan Gagrani's Personal Compute Cluster

Cmd 36

1 display(df\_chat\_channel\_merged)

(?) Spark Jobs

Table + New result table: OFF

channelId	authorChannelId	videoId	timestamp	isMember	bodyLength	chatId	deletedBy	
1	UCge897n0010Be9j9UEZA	e06b70be1759107a98321aa32ce6698477540	SjPENO92x5w	2022-01-31T00:00:00.016Z	false	8	0	true
2	UCge897n0010Be9j9UEZA	31930b506c192d39efc652fe79a7cd95de	SjPENO92x5w	2022-01-31T00:00:00.035Z	false	1	1	true
3	UCMvGHROBfZLunyTFSNvg	49387e07dc1a02c260246af0d24936ee78282	GRzBALD_E	2022-01-31T00:00:13Z	true	4	2	true
4	UCQBiqXXWVGeJn9bdw	ce816ef54702046bed09d945e9a32ed950	O2du12fH-ow	2022-01-31T00:00:13Z	false	13	3	true
5	UCge897n0010Be9j9UEZA	8cc9595e33fb0f69d794fed20e88485d081	SjPENO92x5w	2022-01-31T00:00:14Z	true	13	4	true
6	UCU-Bu8dUL1t6g-Ct1naw	e011f5430cae11e42a21172462b03d599f5968	C_FbKCC012g	2022-01-31T00:00:20Z	true	11	5	true
7	19c5vn-44c0crnnwh1717rnf	(777+ad30c0+11f7477K+2934hebab69+rcwad9h	iUT3Fevi6uf	2022-01-31T00:00:21Z	false	13	6	true

10,000 rows | Truncated data: 4.15 seconds runtime Refreshed 8 minutes ago

Command Took 4.15 seconds -- by pojan.gagrani@sjjsu.edu at 12/10/2023, 2:01:05 AM on Pojan Gagrani's Personal Compute Cluster

Cmd 35

1 display(df\_chat\_channel\_merged.count())

(?) Spark Jobs

8999581

Command Took 9.68 seconds -- by pojan.gagrani@sjjsu.edu at 12/10/2023, 2:01:05 AM on Pojan Gagrani's Personal Compute Cluster

Cmd 36

1 df\_superchat\_merged = df\_superchat.union(df\_superchat2).union(df\_superchat3)

2 df\_superchat\_merged. pyspark.sql.DataFrame. Dataframe = [timestamp: timestamp, amount: double ... 8 more fields]

Command Took 0.08 seconds -- by pojan.gagrani@sjjsu.edu at 12/10/2023, 2:01:08 AM on Pojan Gagrani's Personal Compute Cluster

**Fig. 23. Merging Chat and Channel data**

The screenshot shows the Microsoft Azure Databricks workspace interface. On the left, the sidebar includes options like New, Workspace, Recents, Catalog, Workflows, Compute, SQL, SQL Editor, Queries, Dashboards, Alerts, Query History, SQL Warehouses, Data Engineering, Job Runs, Data Ingestion, Delta Live Tables, Machine Learning, Experiments, Features, Models, Serving, Marketplace, Partner Connect, and a Collapse menu. The main area is titled 'bigDataAnalysis' and contains a Python notebook. The notebook has three cells:

- Cell 37:** Displays the merged DataFrame. The output table has columns: timestamp, amount, currency, significance, authorChannelId, videoid, channelid, and bodylen. It lists 10 rows of data.
- Cell 38:** Prints the total count of Superchat data. The output shows 'Superchat len: 1033436'.
- Cell 39:** Prints the total count of the merged DataFrame. The output shows 'Superchat merged len: 1033436'.

At the bottom of the notebook, it says 'Refreshed 8 minutes ago'.

Fig. 24. Merging Superchat data

This screenshot is similar to Fig. 24, showing the Databricks workspace. The sidebar and notebook structure are identical. The notebook contains three cells:

- Cell A1:** Displays the merged DataFrame. The output table has columns: channelid, timestamp, amount, currency, significance, authorChannelId, videoid, and bodylen. It lists 10 rows of data.
- Cell A2:** Prints the total count of the merged DataFrame. The output shows '1032402'.
- Cell A3:** Displays the merged DataFrame. The output table has columns: channelid, period, chats, memberChats, uniqueChatters, uniqueMembers, bannedChatters, deletedChats, and chatStatsId. It lists 10 rows of data.

At the bottom of the notebook, it says 'Refreshed 10 minutes ago'.

Fig. 25. Merging Superchat and Channel data

The screenshot shows the Databricks workspace interface. On the left is the sidebar with various notebooks, queries, and data engineering options. The main area is titled "bigDataAnalysis" and contains two code cells.

**Code Cell 45:**

```
1 display(df_chat_stats_channel_merged)
```

This cell displays a merged DataFrame with the following schema and data:

channelId	period	chats	memberChats	uniqueChatters	uniqueMembers	bannedChatters	deletedChats	chatStatsId	englishName
1	UC--A2dwZw7-M2klD0N6_ifA	2022-01-01	68741	27091	2203	208	0	0	Shishio Chi
2	UC-zUEf0NeFPxLqxDKvbwu	2022-01-01	61448	0	927	0	2	0	Iori Matsuz
3	UC-62gmz2Slbq78j3RD9ow	2022-01-01	68782	373457	13830	2515	6	15	Leos Vincen
4	UC-DzOMtuxuzMD0HtTg9fQ	2022-01-01	1840	1219	89	31	1	3	Aozora Kuru
5	UC-EN4gkllO3u2nh7ZVZDHO	2022-01-01	13	1	12	1	0	0	Ayame Hyo
6	UC-LsNFByJawPfbpQUOTFQ	2022-01-01	9068	2418	437	33	0	0	Alta Il
7	UC-0CecovinmhnRHf77wAkw	2022-01-01	57874	36210	1170	151	6	6	Akima Saki

**Code Cell 46:**

```
1 display(df_superchat_stats)
```

This cell displays a merged DataFrame with the following schema and data:

channelId	period	superChats	uniqueSuperChatters	totalSC	averageSC	totalMessageLength	averageMessageLength	mostFrequentCu	
1	UCIM920K_spNKLVB5tgswsQ	2022-01-01	3581	2085	5328995	1488	220186	63	TWD
2	UCAum0e1MVZF69boluxgAOBg	2022-01-01	261	73	196033	751	15966	61	USD
3	UCqm3BQLUfVtX_hrm0UmA	2022-01-01	5692	1497	5512879	968	208507	42	JPY
4	UCsUj0dszADCGBf3gNQEusQ	2022-01-01	916	460	1252511	1367	51686	64	USD
5	UCO_aKKYxn4rvqPjTzZ6EQ	2022-01-01	1153	544	1862014	1614	75978	66	USD
6	UCBRM854LH7cRZO2Cle9RDA	2022-01-01	1426	412	1116987	783	72917	52	USD

Fig. 26. Merging Chat Stats and Channel data

The screenshot shows the Databricks workspace interface. On the left is the sidebar with various notebooks, queries, and data engineering options. The main area is titled "bigDataAnalysis" and contains three code cells.

**Code Cell 48:**

```
1 display(df_superchat_stats_channel_merged)
```

This cell displays a merged DataFrame with the following schema and data:

channelId	period	superChats	uniqueSuperChatters	totalSC	averageSC	totalMessageLength	averageMessageLength	mostFrequentCu	
1	UCIM920K_spNKLVB5tgswsQ	2022-01-01	3581	2085	5328995	1488	220186	63	TWD
2	UCAum0e1MVZF69boluxgAOBg	2022-01-01	261	73	196033	751	15966	61	USD
3	UCqm3BQLUfVtX_hrm0UmA	2022-01-01	5692	1497	5512879	968	208507	42	JPY
4	UCsUj0dszADCGBf3gNQEusQ	2022-01-01	916	460	1252511	1367	51686	64	USD
5	UCO_aKKYxn4rvqPjTzZ6EQ	2022-01-01	1153	544	1862014	1614	75978	66	USD
6	UCBRM854LH7cRZO2Cle9RDA	2022-01-01	1426	412	1116987	783	72917	52	USD

**Code Cell 49:**

Saving final dataframe

```
1 storage_account_name = "bigdatalakeproj"
2 filesystem_name = "vtuberdata"
```

**Code Cell 50:**

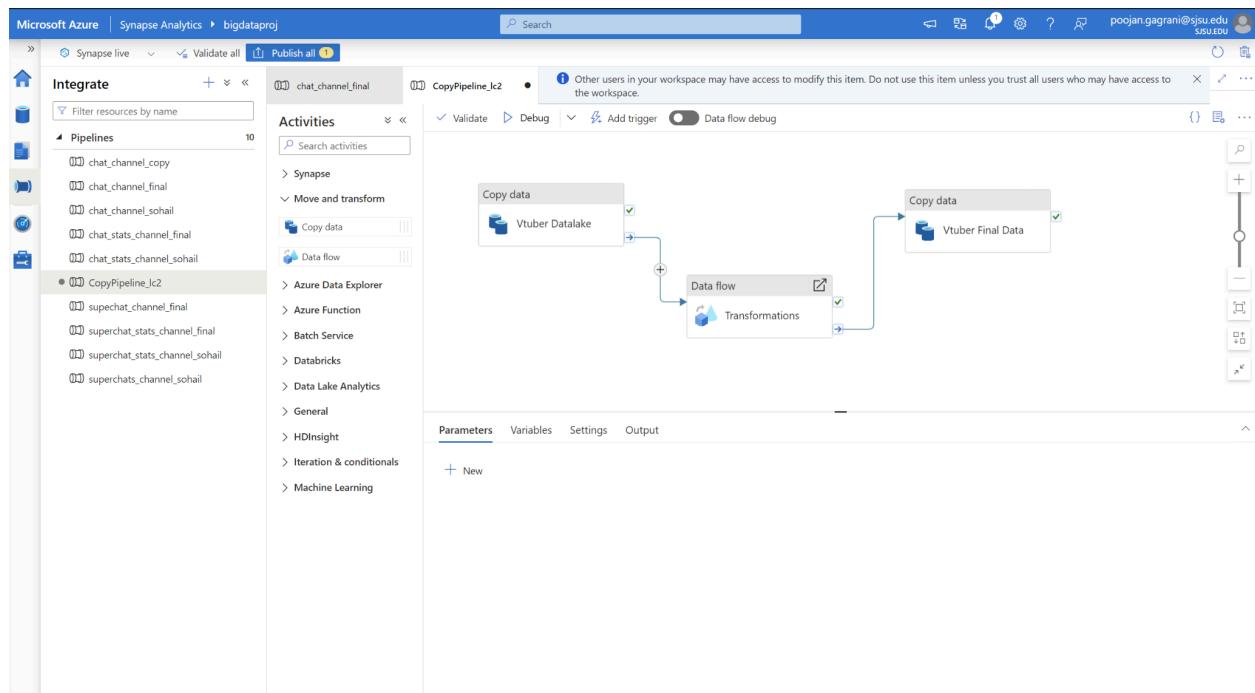
```
1 output_path_chat_channel = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/final/chat_channel_merged"
2 df_chat_channel_merged.coalesce(1).write.option("header", "true").csv(output_path_chat_channel)

1 output_path_superchat_channel = f"abfss://{{filesystem_name}}@{{storage_account_name}}.dfs.core.windows.net/final/superchat_channel_merged"
2 df_superchat_channel_merged.coalesce(1).write.option("header", "true").csv(output_path_superchat_channel)
```

Fig. 27. Merging Superchat Stats and Channel data

### c. Loading data

The merged data is moved from the source Azure data lake to the destination Azure Synapse Analytics where the EDA is performed using built-in SQL serverless to understand better the data and Visualization is done using the built-in Power-BI to extract data in the BI tool where the dashboarding is done.



**Fig. 28. Pipeline for moving final data from source to destination**

## Chapter 6 Data Analytics

### 6.1 Data Analytics using SQL Script

Once the data was cleansed, transformed, and merged in Azure Databricks it was loaded back into the Azure Synapse Analytics to perform analysis using SQL script capability. The analysis is performed on these datasets to get meaningful insights about the affiliations and channels using chat and super chat statistics as well as performing trend analysis like observing the number of banned chatters over the period to understand the behavior of the audience. Below

are some of the queries that were used to perform analysis along with the insights gained from it.

### Query 1: Total number of Super Chats and Unique Super Chatters in each Period

```

1 -- EDA: Total Super Chats and Total Unique Super Chatters by Period
2
3     SELECT
4         period,
5             SUM(superChats) AS totalSuperChats,
6             SUM (uniqueSuperChatters) AS totaluniqueSuperChatters
7
8     FROM
9         OPENROWSET(
10             BULK 'https://bigdatalakeproj.dfs.core.windows.net/vtuberdata/output/superchat_stats/superchat_stats.csv',
11             FORMAT = 'CSV',
12             PARSER_VERSION = '2.0',
13             FIELDTERMINATOR = ',',
14             FIRSTROW = 2,
15             CODEPAGE = '65001' -- UTF-8 code page
16         ) WITH (
17             channelId VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS,
18             period VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS,
19             superChats INT,
20             uniqueSuperChatters INT,
21             totalSC INT,
22             uniqueSC INT
23         )

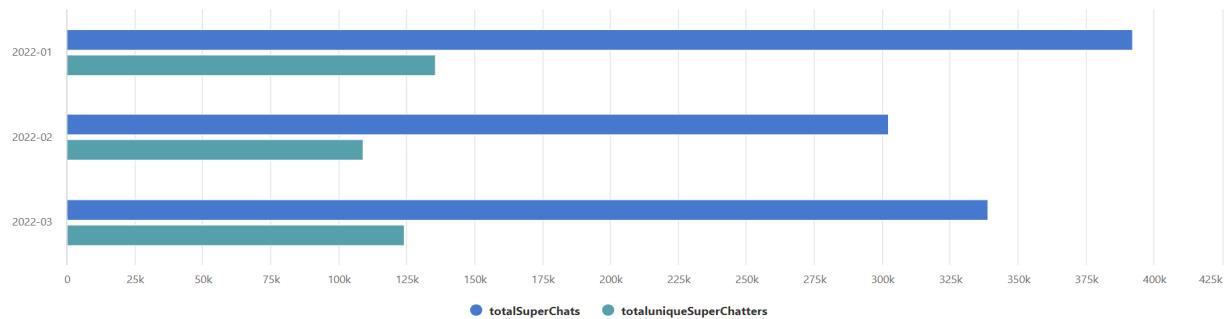
```

The screenshot shows the SQL Server Management Studio interface. The query window contains the provided T-SQL code. Below the code, the results pane is visible, showing a table with three columns: period, totalSuperChats, and totaluniqueSuperChatters. The data is as follows:

period	totalSuperChats	totaluniqueSuperChatters
2022-01	392073	135813
2022-02	302486	108958
2022-03	338877	124309

A message bar at the bottom indicates "00:00:02 Query executed successfully."

**Fig. 29. SQL query with output to show the total number of Super Chats and Unique Super Chatters in each Period**



**Fig. 30. Horizontal Bar plot of the total number of Super Chats and Unique Super Chatters over the Period**

This query result and horizontal bar plot shows the total number of unique super chatters that are involved and the total number of super chats that are done in all the channels over 3 months. It is quite visible that the number of super chats and the number of unique super chats were highest in January then it decreased in February and it increased again in March. A probable reason to see this trend could be that the engagement between the streamer and the viewer got reduced during this time or the content delivered by most of the channels was not satisfactory for the viewers. This is important for streamers to determine the strategy that could be used to optimize their approach towards viewers, it could also help to provide information about the periods where maximum chances of super chats are there maybe due to some holidays or festivals this could be helpful for streamers to create more attractive content and gain more viewers.

### Query 2: Total number of Banned Chatters by Period

```

1  -- Exploratory Data Analysis (EDA) for Total Banned Chatters by Period
2
3  SELECT
4      period AS [Period],
5      SUM(bannedChatters) AS [Total Banned Chatters]
6  FROM
7      OPENROWSET(
8          BULK 'https://bigdatalakeproj.dfs.core.windows.net/vtuberdata/output/chat_stats/chat_stats.csv',
9          FORMAT = 'CSV',
10         PARSE_VERSION = '2.0',
11         FIELDTERMINATOR = ',',
12         FIRSTROW = 2,
13         CODEPAGE = '65001' -- UTF-8 code page
14     ) WITH (
15         channelId VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS,
16         period VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS,
17         chats INT,
18         memberChats INT,

```

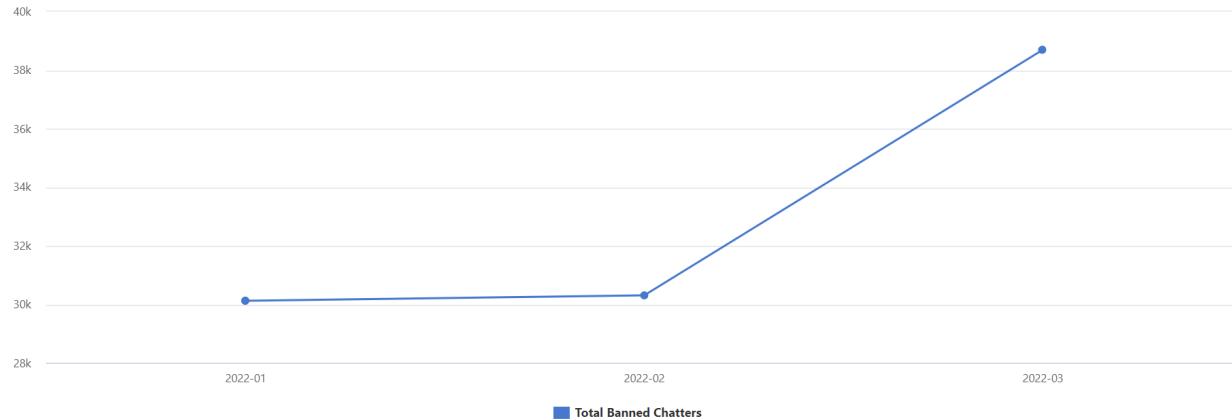
Results Messages

View Table Chart Export results

Search

Period	Total Banned Chatters
2022-01	30124
2022-02	30305
2022-03	38694

**Fig. 31. SQL query with output to show the total number of Banned Chatters by Period**



**Fig. 32. Line plot to show the trend of total Banned Chatters over the Period**

This query result and line plot shows the total number of chatters that have been banned over 3 months. It is easy to see that over months the number of chatters that are getting banned is increasing. A steep increase can be seen in March. A probable reason could be that there were a large number of chatters who were violating the channel guidelines or were using abusive language in the charts resulting in an instant ban of the id. It could be a useful indicator for the streamers as well as the streaming platform to incorporate stricter guidelines so that a safe and peaceful environment is maintained. It could also be helpful for new streamers to determine which kind of content they should be away from so that it would result in less number of guideline violations by the chatters.

#### **Query 3:** Top 10 Currencies by total Super Chat Amount

```

1  -- EDA: Total Super Chat Amount by Currency
2  SELECT TOP 10
3      mostFrequentCurrency AS currency,
4      SUM(totalsC) AS totalSuperChatAmount
5  FROM
6      OPENROWSET(
7          BULK 'https://bigdatalakeproj.dfs.core.windows.net/vtuberdata/output/superchat_stats/superchat_stats.csv',
8          FORMAT = 'CSV',
9          PARSER_VERSION = '2.0',
10         FIELDTERMINATOR = ',',
11         FIRSTROW = 2,
12         CODEPAGE = '65001' -- UTF-8 code page
13     ) WITH (
14         channelId VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS,
15         period VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS,
16         superChats INT,
17         uniqueSuperChatters INT,
18         totalsC INT,
19         averageS INT,
20         totalMessageLength INT,
21         averageMessageLength INT,
22         mostFrequentCurrency VARCHAR(10) COLLATE SQL_Latin1_General_CI_AS,
23         mostFrequentColor VARCHAR(10) COLLATE SQL_Latin1_General_CI_AS,
24         superChatStatsId INT
25     ) AS [result]
26 GROUP BY
27     mostFrequentCurrency
28 ORDER BY
29     totalSuperChatAmount DESC;
30

```

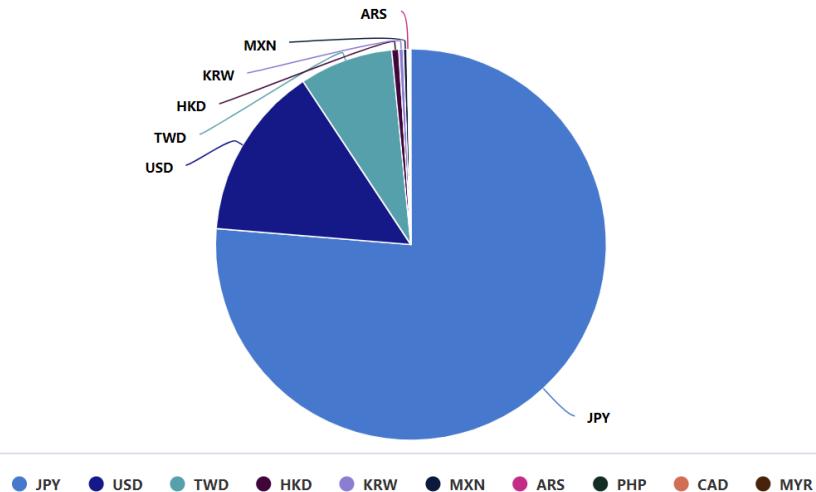
Results Messages

View Table Chart Export results ▾

Search

currency	totalSuperChatAmount
JPY	954849472
USD	180244882
TWD	96875715
HKD	7288098
KRW	47080020
MXN	3669933
ARS	1535878
PHP	1034491
CAD	948038
MYR	387312

**Fig. 33. SQL query with output to show top 10 Currencies by total Super Chat Amount**



**Fig. 34. Pie chart to show total Super Chat Amount from top 10 Currency**

The above query result and the pie chart shows the top 10 currencies by total super chat amount. It is quite evident that the maximum amount of super chats was in Japanese Yen followed by USD. One probable reason could be that the maximum number of viewers and the streamers are from Japan and America which would inevitably lead to a large amount of donation from the currency local to that country. Another reason could be that some of the top VTubers are from Japan and America and they accept super chats only in specific currency which could also lead to a contribution towards the total amount donated through that currency. This information could be really helpful for the new viewers as it can help them understand the payment preferences of the community. It could also be helpful for new streamers to create content related to that country so that more viewers from that country can be attracted which could lead to the growth of that streamer.

#### Query 4: Top 10 Channels with the highest Banned Chatters percentage by Period

channelId	englishName	affiliation	Period	bannedChattersPercentage
UCdCzCzCvPoqSHfbx0tz_A	Hoshino Char	Independents	2022-03	6.25
UCu5nZavAhsqskh-rhvoRA	Minerva Rosaline	Independents	2022-01	2.92533098844482
UCF2jQlMmtuIDCh6bQig	Hololive VTuber Group	Hololive	2022-02	2.98268401766822
UCu5nZavAhsqskh-rhvoRA	Minerva Rosaline	Independents	2022-02	2.05879631057778
UCCk072ns5Bt1nqX1CoaZ-g	Mythia Battford	Independents	2022-03	1.93266381541179
UCu5nZavAhsqskh-rhvoRA	Minerva Rosaline	Independents	2022-03	1.91258318538769
UCCk072ns5Bt1nqX1CoaZ-g	Mythia Battford	Independents	2022-01	1.90815727105151
UCCk072ns5Bt1nqX1CoaZ-g	Mythia Battford	Independents	2022-02	1.66282664776196
UCge_6fjHyOCiRtWCmaVTAQ	Lumi Celestia	MAHAS	2022-03	1.44587991164067
UCTRoCB07jyboSldaqKqfQg	Nakiri Ayame	Hololive	2022-03	1.36549840691853

**Fig. 35. SQL query and output to show the top 10 Channels with the highest Banned**

**Chatters percentage by Period**

Figure 35 shows the query result of the top 10 channels having the highest percentage of banned chatters grouped by period. It is quite evident that the streamer named “Hoshino Char” without any affiliation has the highest percentage of banned chatters in March. The probable reason could be the shift in the content of the stream which some percentage of viewers didn’t like or might have found offensive which could have resulted in the chats by the viewers that needed to be moderated. This information could be really helpful for the streamer to determine the factors leading to the banned chats and modify the content of the stream accordingly so that it could result in healthy engagement by the community and a positive environment maintained on the streaming platform. It could be helpful for new viewers to understand which channels have strict moderation policies and can determine based on that whether they want to watch that particular streamer or not.

#### Query 5: Affiliation details of top 10 Channels by Average Subscribers

```

1  -- Query to get details for the top 10 affiliations
2  WITH TopAffiliations AS (
3      SELECT TOP 10
4          affiliation,
5          COUNT(*) AS total_channels,
6          AVG(CONVERT(INT, subscriptionCount)) AS avg_subscribers,
7          MAX(CONVERT(INT, subscriptionCount)) AS max_subscribers,
8          MIN(CONVERT(INT, subscriptionCount)) AS min_subscribers,
9          SUM(CONVERT(INT, subscriptionCount)) AS total_subscribers
10     FROM
11        OPENROWSET(
12            BULK 'https://bigdatalakeproj.dfs.core.windows.net/vtuberdata/output/channel/channels.csv',
13            FORMAT = 'CSV',
14            PARSER_VERSION = '2.0',
15            FIELDTERMINATOR = ',',
16            FIRSTROW = 2,
17            CODEPAGE = '65001' -- UTF-8 code page
18        ) WITH (
19            channelId VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS,
20            englishName VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS,
21            affiliation VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS,
22            subscriptionCount VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS,
23            videoCount VARCHAR(255) COLLATE SQL_Latin1_General_CI_AS
24        ) AS [result]
25    GROUP BY
26        affiliation
27    ORDER BY
28        total_subscribers DESC
29 )
30 CTE FCT

```

**Results**

affiliation	total_channels	avg_subscribers	max_subscribers	min_subscribers	total_subscribers
Kizuna AI Inc.	3	1551633	3080000	84900	4654900
Hololive	92	785527	4100000	9320	72268520
VShojo	15	402273	1430000	52600	6034100
Nijisanji	207	293853	1540000	383	60827723
VSpo	16	256093	453000	84500	4097500
Twitch Independents	60	207092	2060000	7950	12425550
Nori Pro	13	147830	883000	38200	1921800
KAMITSUBAKI	18	116690	706000	5850	2100420
Independents	470	105500	2510000	0	49585391
774inc	31	94726	602000	9920	2936520

**Fig. 36. SQL query and output containing Affiliation details about top 10 Affiliations by Average Subscribers**

The above query result shows the details of the top 10 affiliations ordered based on average subscribers which include details like the total number of channels that are part of the affiliation, average number of subscribers, maximum number of subscribers that is equal to the subscriber count of a particular channel of that affiliation, minimum number of subscribers that is equal to the subscriber count of a particular channel of that affiliation, and total number of subscribers. It can be easily seen from the output that the affiliation named “Kizuna Ai inc.” has the highest number of average subscribers but only three channels are part of it which shows that each channel is performing well and have decent viewership and followers. Even though the number of channels involved in Hololive and Nijisanji is high the average subscriber count is still good which shows that affiliation is famous among the audience and people prefer to watch channels related to them. This is useful for the new incoming streamer to determine which affiliation to associate with to reach a larger audience and also it could be helpful for the people who are new to this streaming platform to determine whose content to watch.

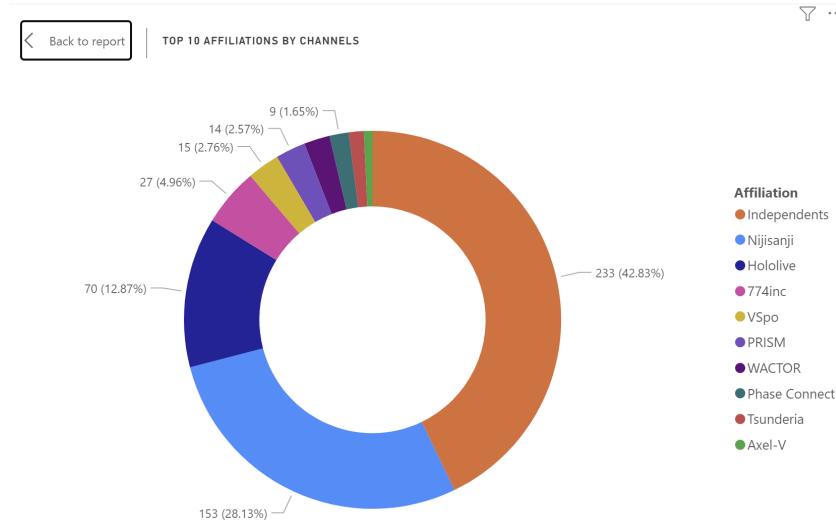
## **Chapter 7 Data Analytics**

### **7.1 Data Visualization**

The data visualization process was performed by creating visualizations using merged datasets to determine the underlying pattern and gain valuable insights about affiliations and channels using channel statistics, super chat statistics, and chat statistics. Power BI software was used to create various visualizations and they were incorporated into three dashboards named chat stats, super chat stats, and banned and deleted.

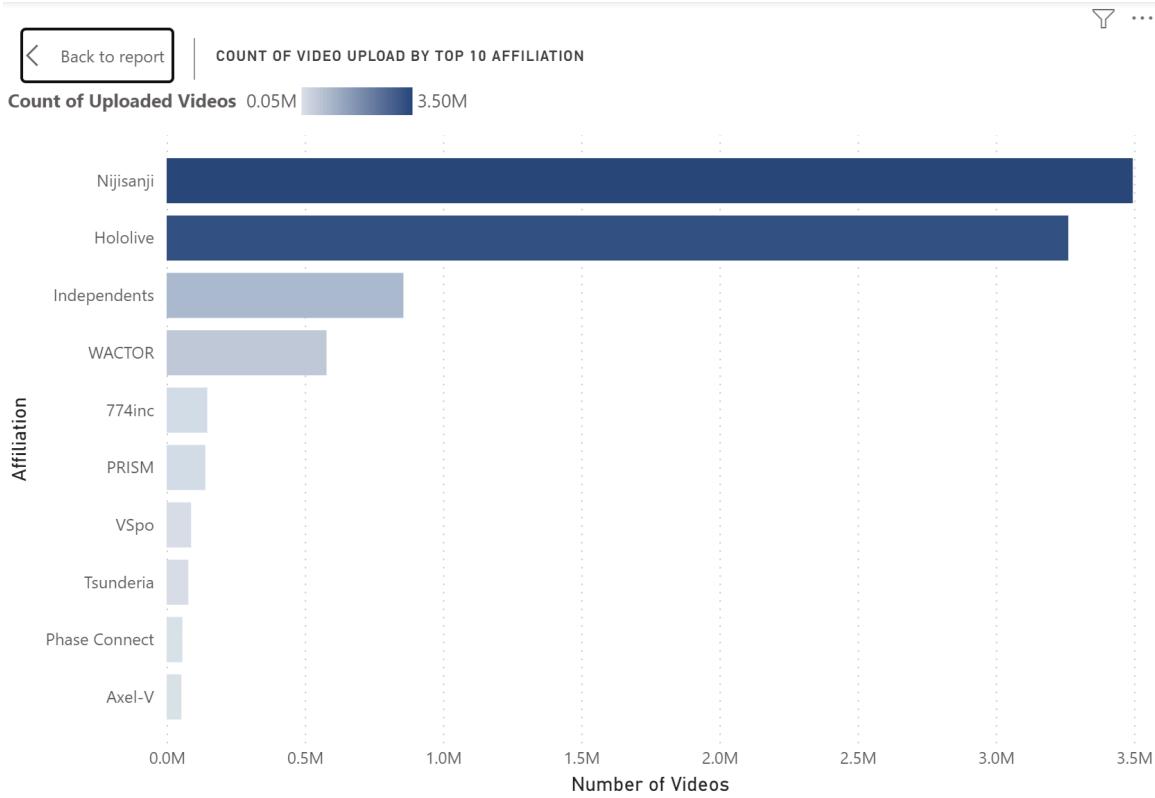
Figure 37 illustrates the top 10 affiliations based on the total number of channels associated with them. It is quite evident from the pie chart that Nijisanji has the highest number

of channels. Independents refer to the channels that are not associated with any affiliations. Hololive follows the Nijisanji. This information is useful for new streamers to determine the affiliation to choose from based on their popularity among streamers. It could also be helpful for a new viewer to understand which channel to explore based on the popularity of the affiliations.



**Fig. 37. Top 10 Affiliations by Channels**

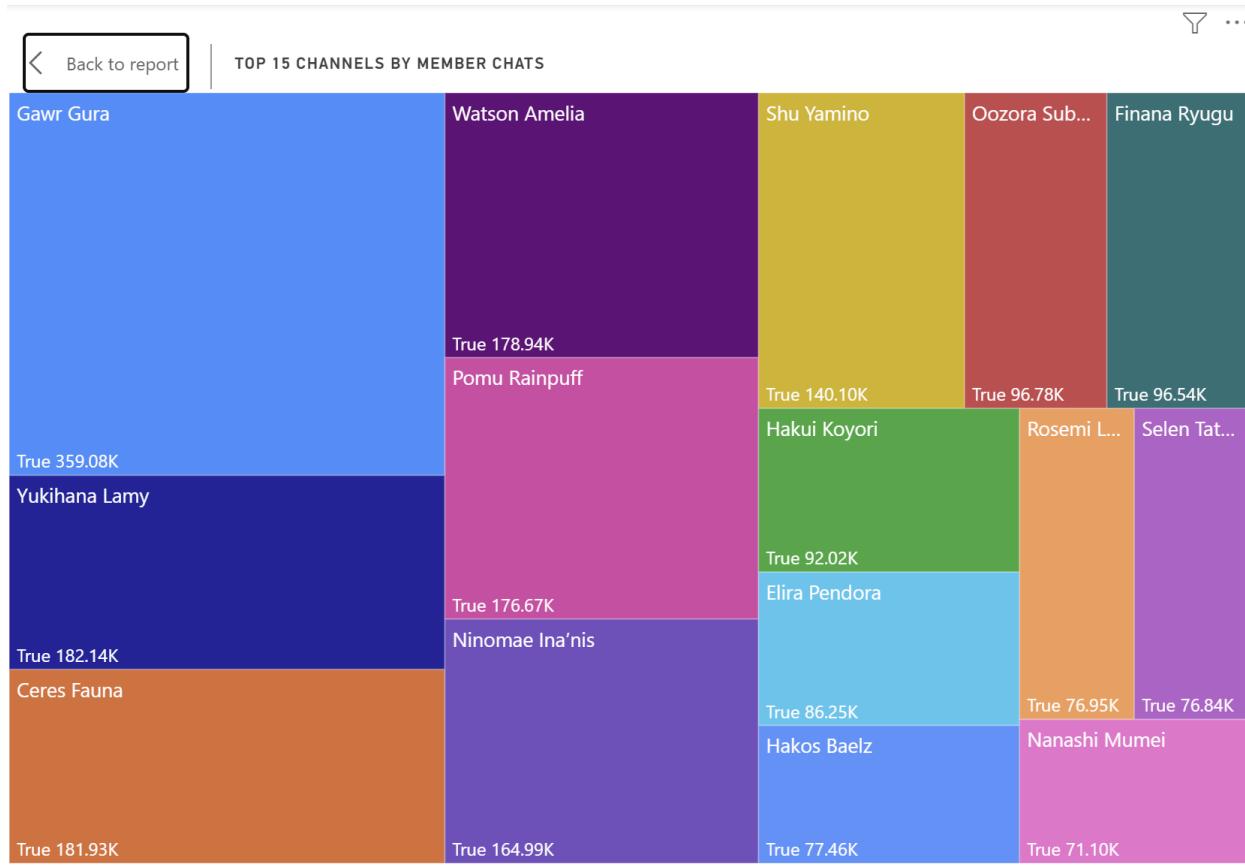
Figure 38 shows the total number of videos uploaded by the top 10 affiliations. It can be seen that Nijisanji and Hololive have a very large number of video uploads. The probable reason could be due to a large number of channels present in them which on a cumulative level results in a large number of uploads. These many videos could attract new viewers as they allow the ground to explore varied content uploaded by the channels associated with these affiliations which could ultimately lead to an increase in popularity and the subscriber count. It could be useful for existing streamers to determine whether to stay with their current affiliation or move on to the popular ones where content is uploaded more frequently and has a larger viewer base.



**Fig. 38. Top 10 Affiliations by count of Video uploads**

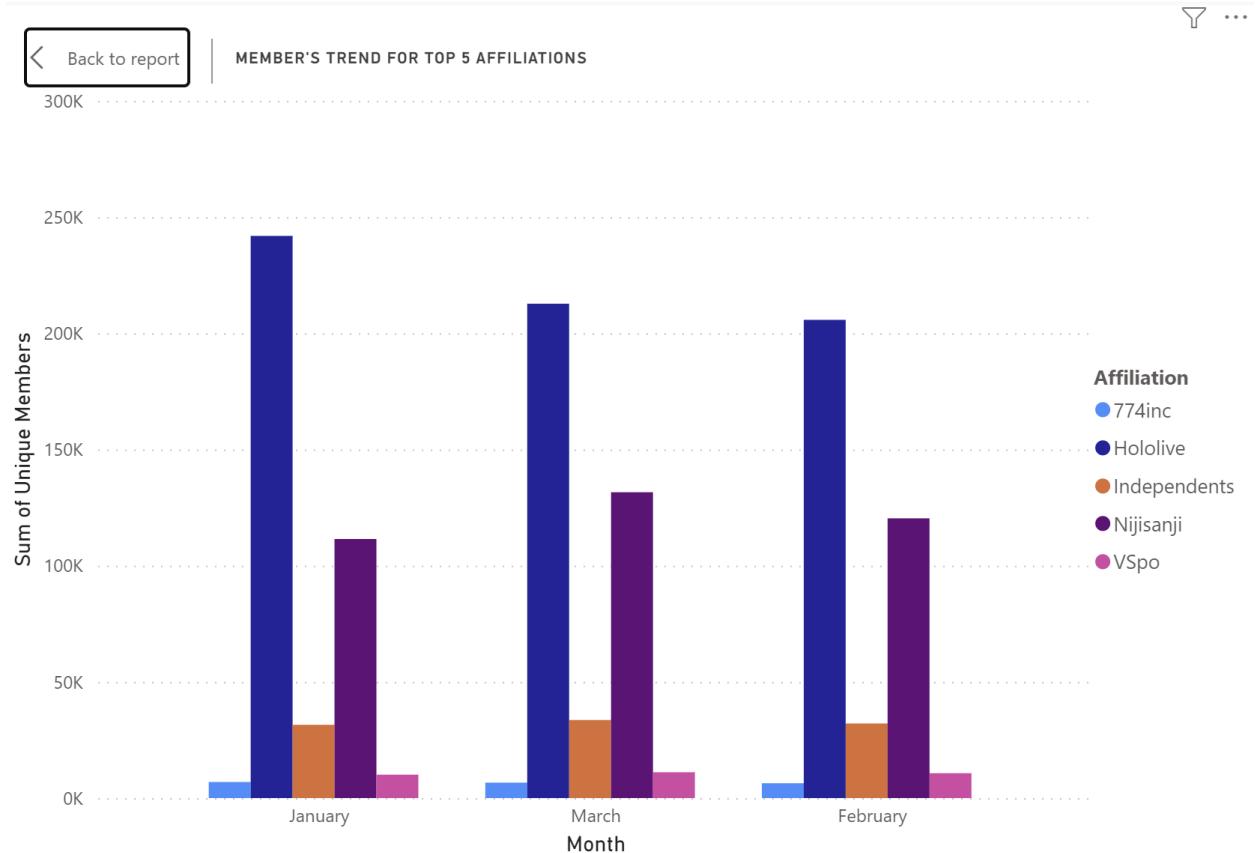
Figure 39 shows the top 15 channels based on the total number of member chats. The channel named Gaur Gura has the highest number of member chats followed by Yukihana Lamy and Ceres Fauna. This is useful for viewers to determine which channel has a more engaged community associated with it. This could allow the viewer to get a better understanding of the content by interacting with the members. It could also be a nice starting point for new streamers where they can understand channels like Gaur Gura by performing in-depth research on their content, strategies followed for streaming, and how the streamer keeps the community together.

to get a brief idea about how to gain more popularity and get higher viewership.



**Fig. 39. Top 15 Channels by Member Chats**

Figure 40 depicts the trend of the number of members in the top 5 affiliations over 3 months. It is quite evident from the plot that Hololive has the highest number of members in all three periods followed by Nijisanji and it is quite understandable because the number of channels involved with it is also very large. In Hololive the number of new members that are getting added is decreasing every month and in the case of Nijisanji it increased for February but then it decreased again in March. It could help new viewers gauge the popularity of the affiliations. Hololive is consistently able to maintain a strong membership and fanbase that shows the trust of the audience. This information is useful for new streamers to determine the affiliation they could join based on community support.

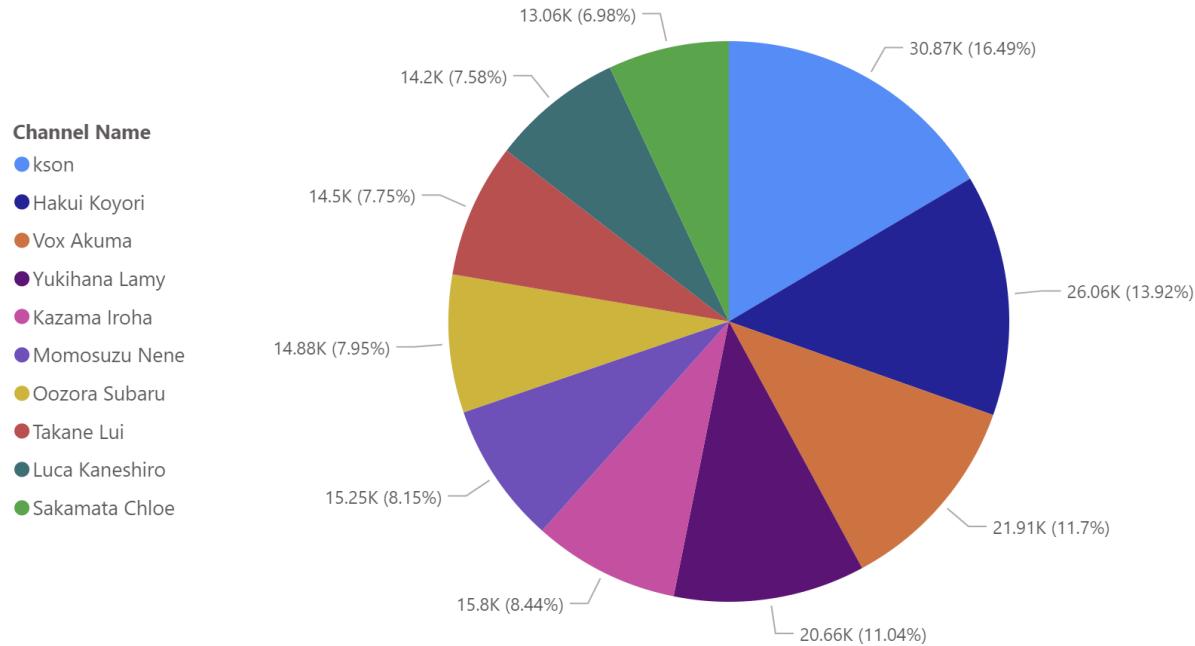


**Fig. 40. Member's trend for top 5 Affiliations**

Figure 41 shows the top 10 channels based on the total number of super chats. In the plot, the channel named Kson has the highest number of super chats followed by the channel named Hakui Koyori. The probable reasons could be the interesting content created by these streamers that attract more viewers to donate or the streamer maintains a healthy engagement with the viewers. This information could be helpful for new incoming streamers to determine the type of content or the strategies of community engagement that attract more audience and inspire people to contribute financially to the channel and can incorporate those insights to improve their streams. For new viewers, it provides information about the popularity of the channel and also the creation of a supportive and interactive environment where viewers contribute willingly.

[Back to report](#)

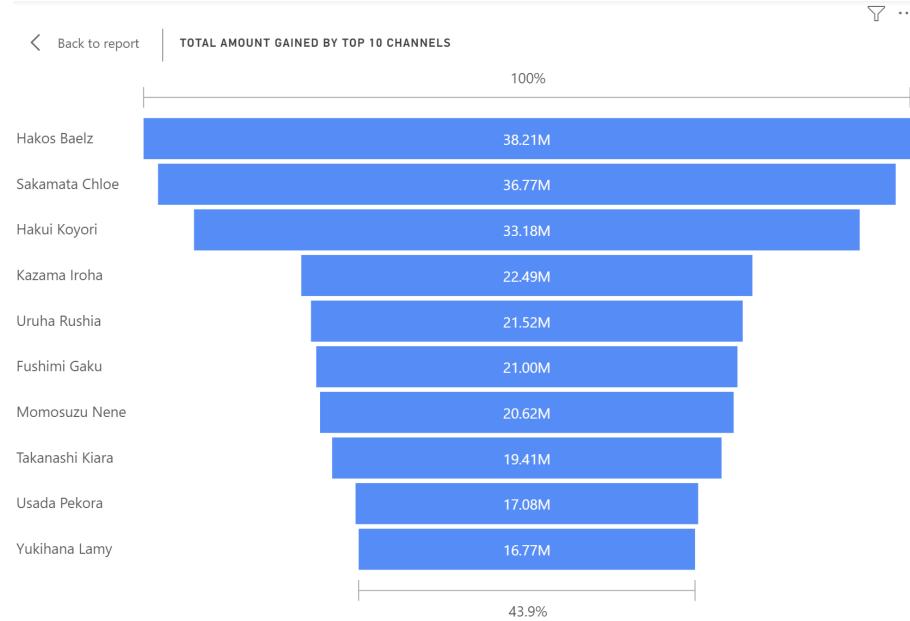
## COUNT OF SUPER CHATS IN TOP 10 CHANNELS



**Fig. 41. Count of Super Chats in top 10 Channels**

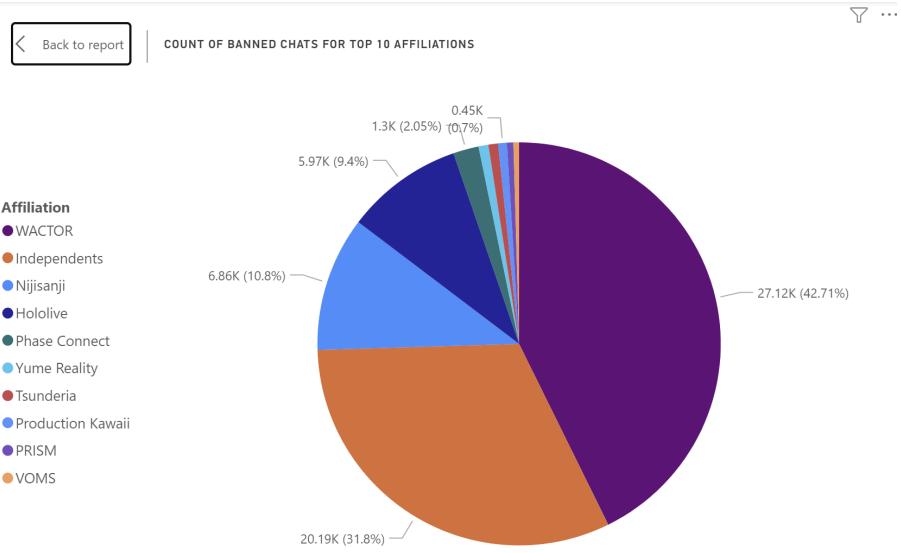
Figure 42 shows the top 10 channels that gained the maximum amount. The channel named Hakos Balez gained the highest amount through the streams followed by the channel named Sakamata Chloe. A probable reason could be the content created by these streamers was good and engaging which resulted in attracting a large amount of viewers and encouraged them to contribute to the streams financially. Another reason could be that these channels might have a decent fanbase but they are constantly supporting the channel financially which results in a large amount of money received by the streamer as a whole. It could be helpful for streamers to create more engaging content and follow the monetization techniques of these channels to get a large amount of money and become more popular. For viewers, it could portray the ability of the

streamer to maintain a loyal fanbase by highly appreciating the viewer who contributes financially and also maintaining a healthy relationship with the viewer community.



**Fig. 42. Total Amount gained by the top 10 Channels**

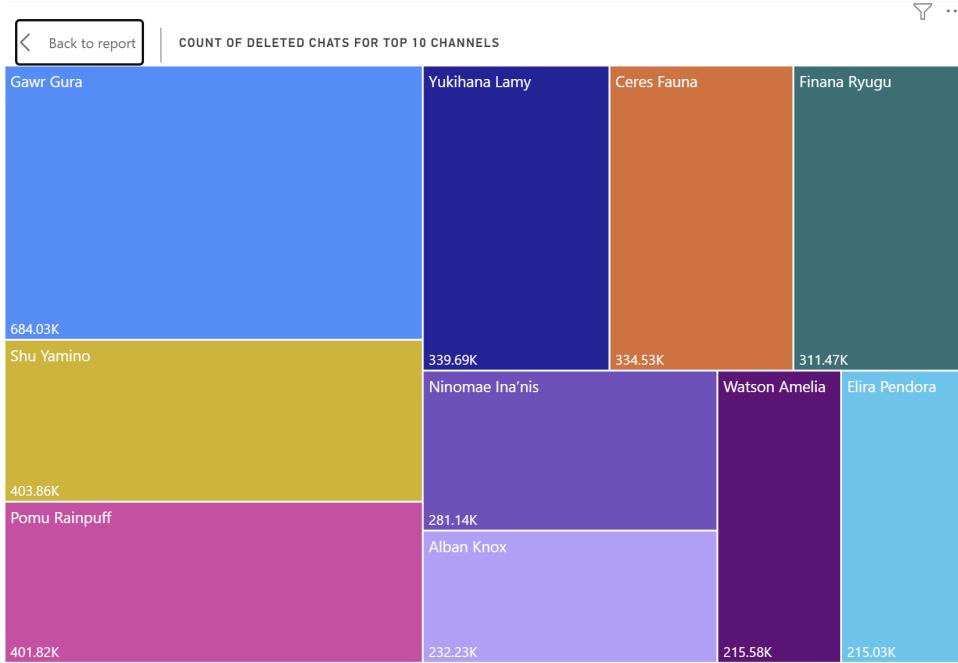
Figure 43 depicts the top 10 affiliations by the count of banned chats. It is quite visible that the affiliation named WACTOR has the highest number of banned chats followed by Nijisanji. Here, independence is not an affiliation rather it is a collection of streamers with no affiliation associated with them. This indicates that an affiliation named WACTOR might have stricter community guidelines in place which could have resulted in frequent bans of the chats and might show the enforcement of a healthy environment. Another reason could be the content created by the streamer might have been offensive or revolting to some viewers which resulted in the chats that were not according to the guidelines. This shows the information about the community guideline policy of the affiliation and it also portrays the information that may be the streamer is showing negative behavior towards the audience. This could be helpful for the streaming platform and also the affiliation to take strict action against that channel.



**Fig. 43. Count of Banned Chats for top 10 Affiliations**

Figure 44 depicts the top 10 channels by total number of deleted chats. It is quite evident that Gawr Gura has the highest number of deleted chats followed by Shu Yamino. A probable reason could be the moderators are trying to maintain a healthy environment by deleting chats that are offensive or contain abusive language. It could also be possible that the streamer is having a fight over the stream or might be saying something offensive that would have resulted in chats that were automatically deleted by the moderation policy. There is a possibility of both positive and negative take on this. If it is positive then it means that the streamer is trying to maintain a healthy environment and trying to ensure that new viewers will feel welcome and will be able to enjoy the streams without any issues. It would also look positive in the eyes of the viewer as it portrays that the streamer is trying to maintain proper community guidelines. This could prove useful for new streamers to determine the strategies needed to be followed for moderation such that there is proper communication between streamer and audience but at the same also ensured that no viewer will feel disheartened or lose interest due to negative

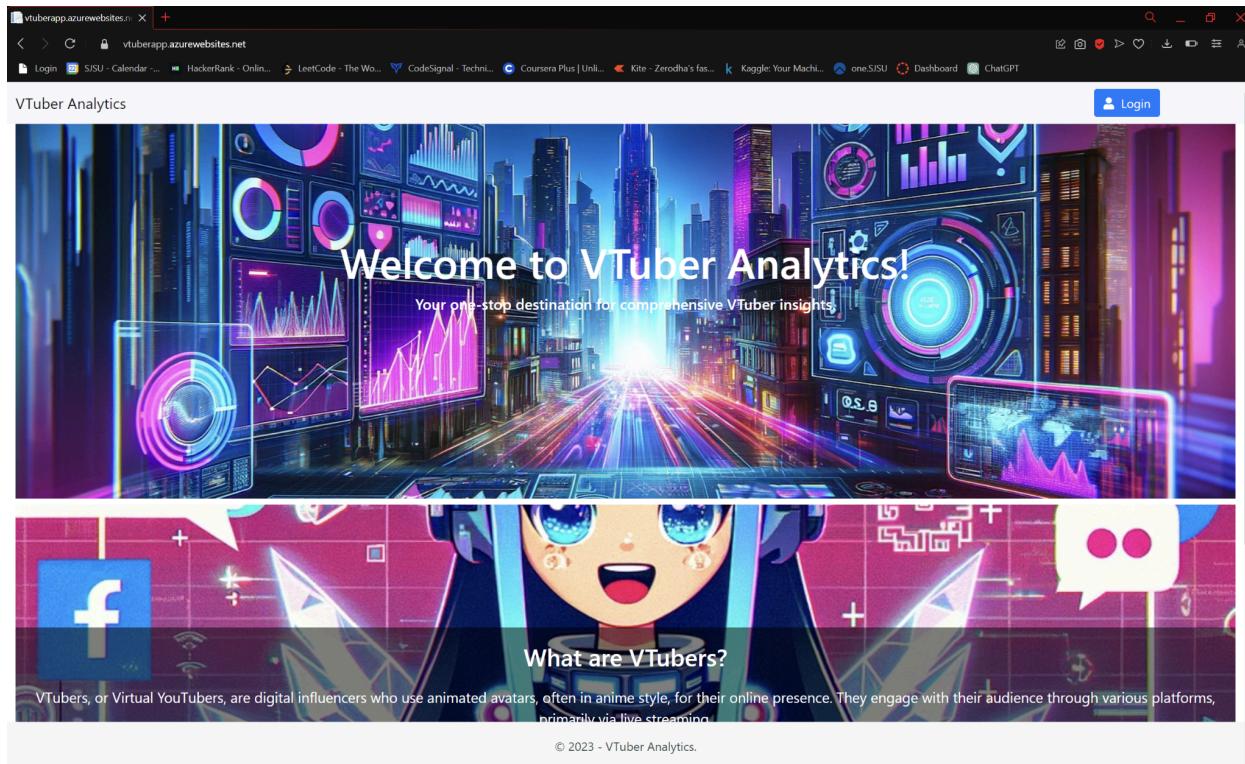
comments. It could also help new streamers understand what kind of things to look out for so that it does not result in any offensive content or arguments with some viewers.



**Fig. 44. Count of Deleted Chats for top 10 Channels**

## 7.2 Web Application to Host Dashboard

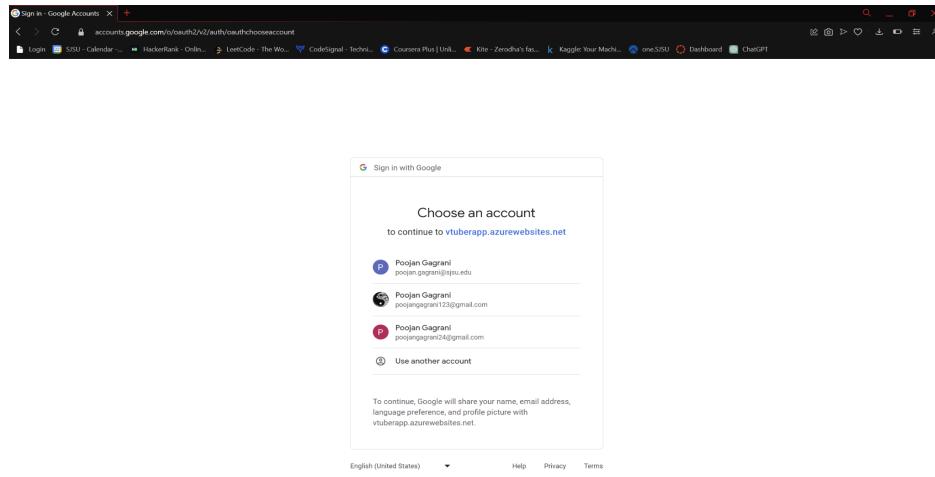
The web application was created using the ASP.NET core framework to host the Power BI dashboard that contains various plots that provide useful insights about the VTuber's data. ASP.NET is the most commonly used web development framework that is used to create more dynamic websites. In this study, ASP.NET is used to create a landing page that contains some brief details about the objective of the study and the benefits that this study will have for new viewers and aspiring VTubers. Figure 45 shows the landing page of the website.



**Fig. 45. Website Landing page**

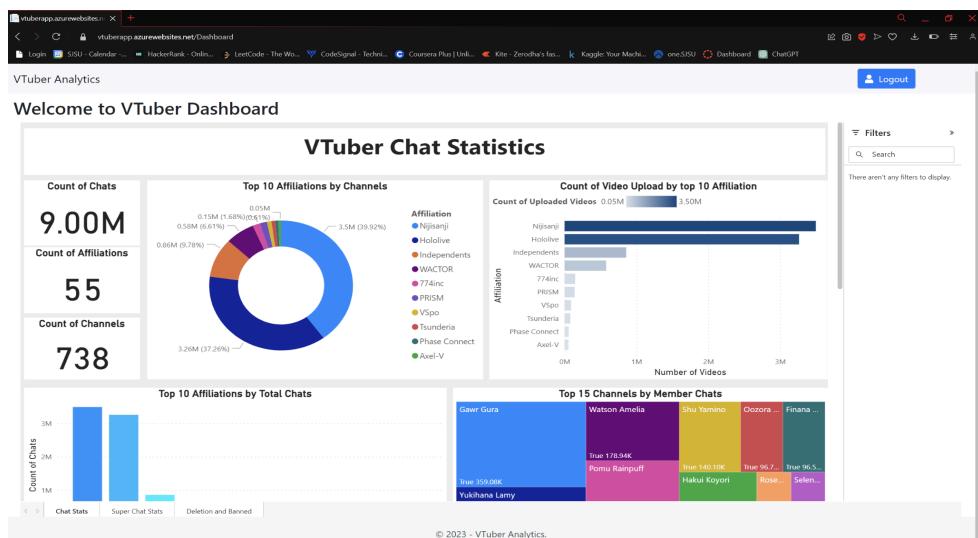
The Single Sign On is also implemented on this landing page using Google OAuth SSO.

In this using the single login credentials of Google the user can access multiple software systems and websites. Users can log into the website by clicking on the login button at the top right-hand side of the homepage. When it is pressed, the user will be directed to the sign-in with Google option where the user will be asked to use either the existing Gmail ID that was already used to log in or sign in using a new ID. Figure 46 shows the sign-in with the Google page.



**Fig. 46. Login with Google OAuth (SSO)**

After successful login, the user can access multiple tabs or sheets of the Power BI dashboard that is hosted on this website. Since the dashboard is interactive users that are either streamers or viewers can play around with the plots to get useful insights. After using the website the user can log out by clicking on the logout button at the top right-hand side of the website and when it is pressed the user will be redirected to the home page or landing page. Figure 47 shows the website with the hosted dashboard.



**Fig.47. VTuber Dashboard hosted on the website**

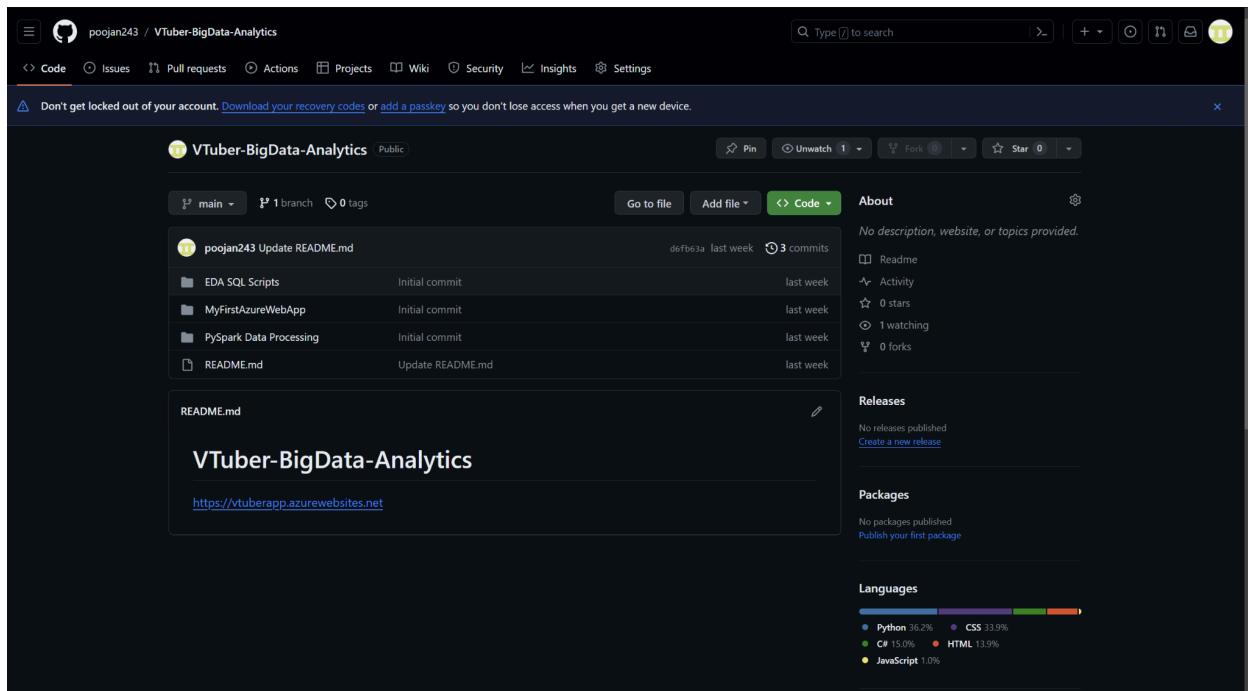
# Chapter 8

## 8.1 Github

It is essential to store the final code used for the project as part of the disaster management and recovery plan. For the part of saving data the data is stored in the online repository platform GitHub where the up-to-date code's backup is stored at all times, this tool not only facilitates code backup but also facilitates a collaborative environment where multiple contributors can work together ensuring that they have up-to-date code available and can also build on top of it and can save their changes and update the code base at all times.

For this project, a GitHub repository was created and the code files were committed. It comprises data cleaning, transformation, and the web application code which was implemented for the project.

**GitHub repository URL:** <https://github.com/poojan243/VTuber-BigData-Analytics>



**Fig. 48. Github repository of the project**

## 8.2 Team Members and IDs

Name	Student ID
Abdul Sohail Ahmed	016769610
Poojan Gagrani	016795285
Kashish Thakur	016919383
Swetha Neha Kutty Sivakumar	016780712
Somna Sattoor	014640769

## Chapter 9

### 9.1 Conclusion

#### 9.2 Future Work

Creating an improved pipeline capable of handling even bigger amounts of data (TBs) at scale. Better data preparation can be done by using more capable spark clusters that can manage and preprocess larger datasets. When a greater set of data is available, including additional statistics. Enhancing the website by incorporating numerous login choices such as AzureAD, Facebook, and Twitter logins. Enabling user-specific dashboards on a website, for example, a chat metrics dashboard for a streamer that focuses on stats related to that broadcaster and allows them to easily access insights.

## **References**

- Ahmad, N., & Mamat, R. (2022). The Consumption of Popular Culture Products among Japanese Language Learners at Universiti Teknologi Mara. *International Journal of Academic Research in Business & Social Sciences*, 12(1).
- <https://doi.org/10.6007/ijarbss/v12-i1/12183>
- Amalia, S. I., & Mohammad, W. (2023). Analysis of the Effect of the Number of Views and Number of Videos on the Number of Virtual YouTuber Subscribers in Vietnam. *Himeka: Journal of Interdisciplinary Social Sciences*, 1(1), 108–114. Retrieved from <https://journal.chishikinh.my.id/index.php/himeka/article/view/23>

Kim, D., & Yoo, H. (2021). A comparative study of user experience according to one-person media Virtual YouTuber (VTuber) and general YouTuber. *Asia-Pacific Journal of Convergent Research Interchange*, 7(5), 1–10. <https://doi.org/10.47116/apjcri.2021.05.01>