Project Report

Team Insight Hub: - Social Media Analysis Project

❖ Introduction:

In an era dominated by digital presence, social media has become an indispensable tool for businesses to connect with their audience. The sheer volume of users and data on platforms like Twitter, Instagram, and Facebook makes it challenging for organizations to navigate and optimize their social media strategies. This project aims to provide comprehensive social media analysis, specifically focusing on identifying inactive accounts, detecting bots, tracking trending hashtags, offering geographical insights, and assessing user engagement. We aim to empower our clients with actionable insights to enhance their social media presence and maximize engagement.

❖ Project Objectives:

- **Identification of Inactive Accounts:**

1. Utilize advanced SQL Queries to analyze user activity patterns.
2. Identify accounts with prolonged periods of inactivity.
3. Categorize and prioritize inactive accounts based on factors such as follower count, comments, number of posts and historical engagement.

- **Trending Hashtags Analysis:**

1. Develop an interactive dashboard using PowerBI for analysing trending hashtags across multiple social media platforms.
2. Provide insights into the velocity and longevity of trending hashtags.
3. Recommend relevant hashtags based on the client's industry and target audience.

- **Geographical Insights:**

1. Implement geospatial analysis to understand the regional distribution of followers and engagement.
2. Identify key geographical areas of influence and potential areas for expansion.
3. Tailor content strategies based on regional preferences and trends.

- **User Engagement Assessment:**

1. Analyze user engagement metrics, including likes, comments, and posts.
2. Identify the hashtags used by users for engagement trends.
3. Provide personalized recommendations for increasing user engagement.

❖ **Why We Selected This Project:**

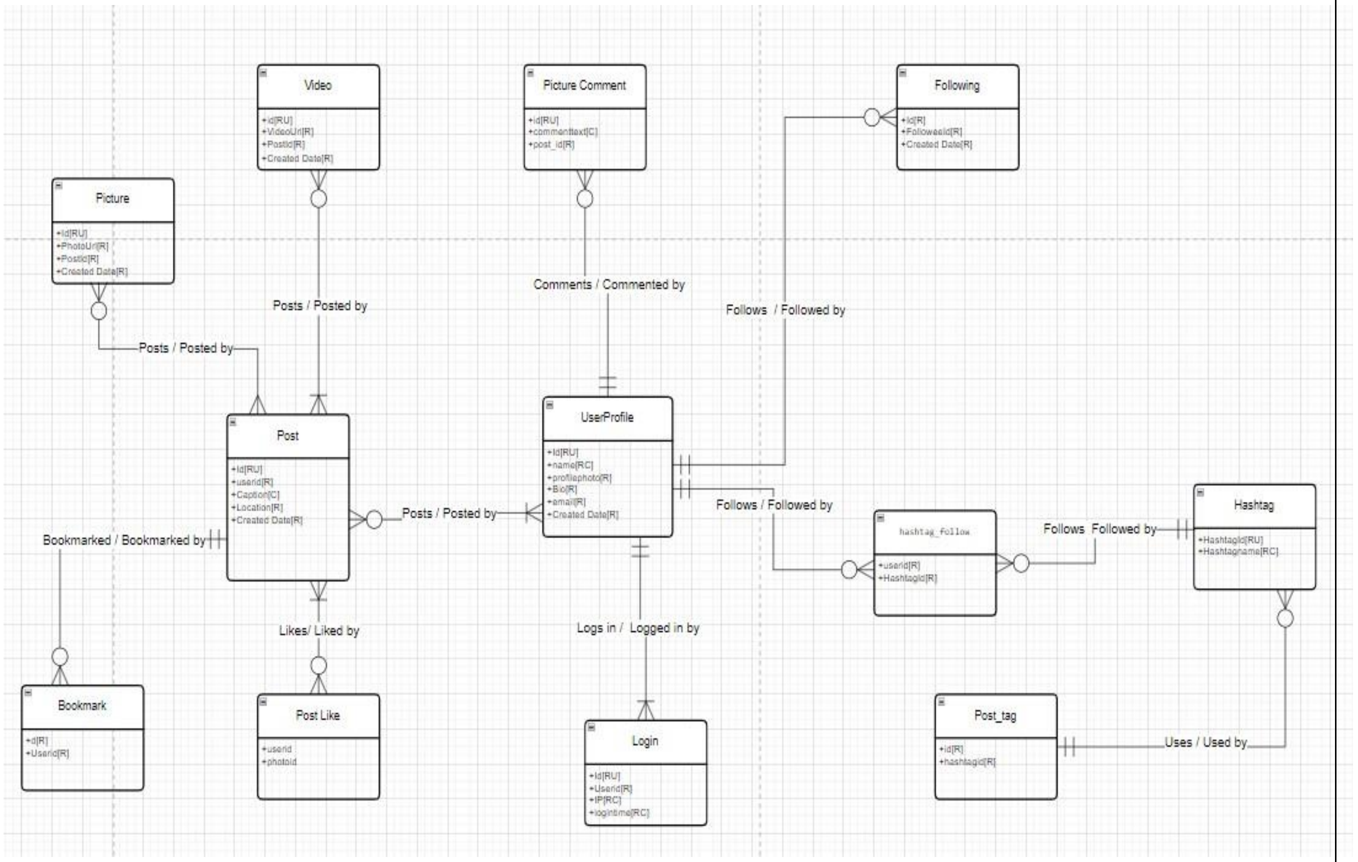
The landscape of social media is dynamic and complex, requiring sophisticated tools to derive meaningful insights. We selected this project because it addresses a critical need for businesses striving to optimize their social media presence. By leveraging advanced analytics and data visualization tools, we aim to provide our clients with a holistic understanding of their social media ecosystem, enabling them to make informed decisions and enhance their online visibility.

❖ **Business Problem Addressed:**

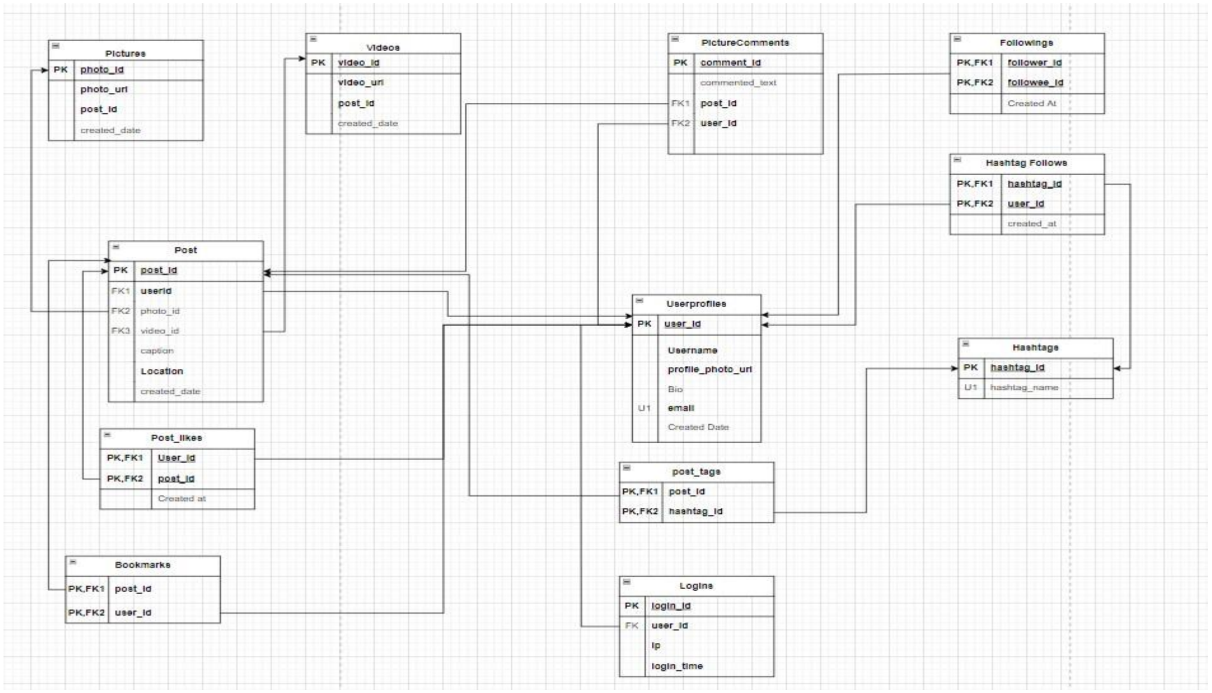
The primary challenge this project addresses is the difficulty businesses face in navigating the vast and intricate social media landscape. Many organizations struggle to identify inactive accounts, bot interference, and stay ahead of trending topics. This lack of clarity hampers their ability to tailor content strategies for maximum impact. Our solution directly tackles these issues, offering actionable insights that empower businesses to refine their social media approach, increase engagement, and ultimately expand their reach.

In conclusion, the Social Media Analysis project focuses on delivering a robust and data-driven solution to the multifaceted challenges of social media management. By identifying inactive accounts, detecting bots, analyzing trending hashtags, providing geographical insights, and assessing user engagement, our goal is to equip clients with the tools they need to thrive in the ever-evolving digital landscape.

❖ Conceptual & Logical Diagram



Relationship	Entity 1	Entity 2	Rule	Min	Max
Userprofiles - post	Userprofile	Post	one-to-many	0	many
Userprofiles - picturecomments	Userprofile	picturecomments	One-to-many	0	many
Userprofiles-following	Userprofile	following	One-to-many	0	many
Userprofile-hashtag_follow	Userprofile	hashtag_follow	One-to-Many	0	Many
Userprofile-Login	Userprofile	Login	one-to-many	1	many
post-userprofiles	post	userprofiles	one-to-many	1	many
Post-video	post	video	one-to-many	0	many
post-picture	post	picture	one-to-many	0	many
post-bookmark	post	bookmark	one-to-many	0	many
post-postlike	post	post_like	one-to-many	0	many
hashtag_follow-hashtag	hashtag_follow	hashtag	one-to-one	1	1
hashtag-post_tag	hashtag	post_tag	one-to-one	1	1



Post_Likes	user_id	FK, RU	References from table userprofiles
	post_id	FK	References from table post
	created_at		Default current timestamp
Followings	follower_id	PK, RU	Unique identifier for a follower
	followee_id	PK,RU	Unique identifier for a followee
	created_at		Default current timestamp
Hashtags	hashtag_id	PK,RU	Unique identifier for a hashtag
	hashtag_name	RU	Name of the hashtag used
	created_at		Default current timestamp
Hashtag_follow	user_id	PK, RU	References from table userprofiles
	hashtag_id	FK	References from table hashtags
	created_at		Default current timestamp
Post_tags	post_id	FK	References from table post
	hashtag_id	FK	References from table hashtags
Bookmarks	post_id	FK	References from table post
	user_id	FK	References from table userprofiles
	created_at		Default current timestamp
Login	login_id	PK, RU	Unique identifier for every login by user
	user_id	FK	References from table userprofiles
	ip		ip address
	login_time		Total time logged in

	Entities and Attributes		
Entity	Attribute	Props	Description
User_profiles	User_id	PK, RU	Unique identifier for a user.
	User_name	RU	Unique username for login purposes.
	Profile_photo_url	RU	Profile photo of the user
	Bio		Denotes the bio of user
	Email	RU	email id of the user
	Created_at		Default current timestamp
Pictures	photo_id	PK, RU	Unique identifier for a picture
	photo_url	FK	Foreign key linked to User Authentication, identifying the user.
	post_id	FK,RU	Unique identifier for a picture
	created_at		Default current timestamp
Videos	video_id	PK, RU	Unique identifier for a video.
	video_url		Video URL posted by the user
	post_id	FK	References from table pictures
	created_at		Default current timestamp
Post	post_id	PK, RU	Unique Identifier for a post
	photo_id	FK	References from table pictures
	video_id	FK	References from table videos
	user_id	FK	References from table userprofiles
	caption		caption under every post
	location		location from where the user has posted
	created_at		Default current timestamp
Picture_Comments	comment_id	PK, RU	Unique identifier for an comment.
	comment_text		Text from the comment
	post_id	FK	References from table post
	user_id	FK	References from table userprofiles
	created_at		Default current timestamp
Post_Likes	user_id	FK, RU	References from table userprofiles
	post_id	FK	References from table post
	created_at		Default current timestamp

- **Project Log:**

Project Name: Team Insight Hub: - Social Media Analysis

Project Start Date: 12th November 2023

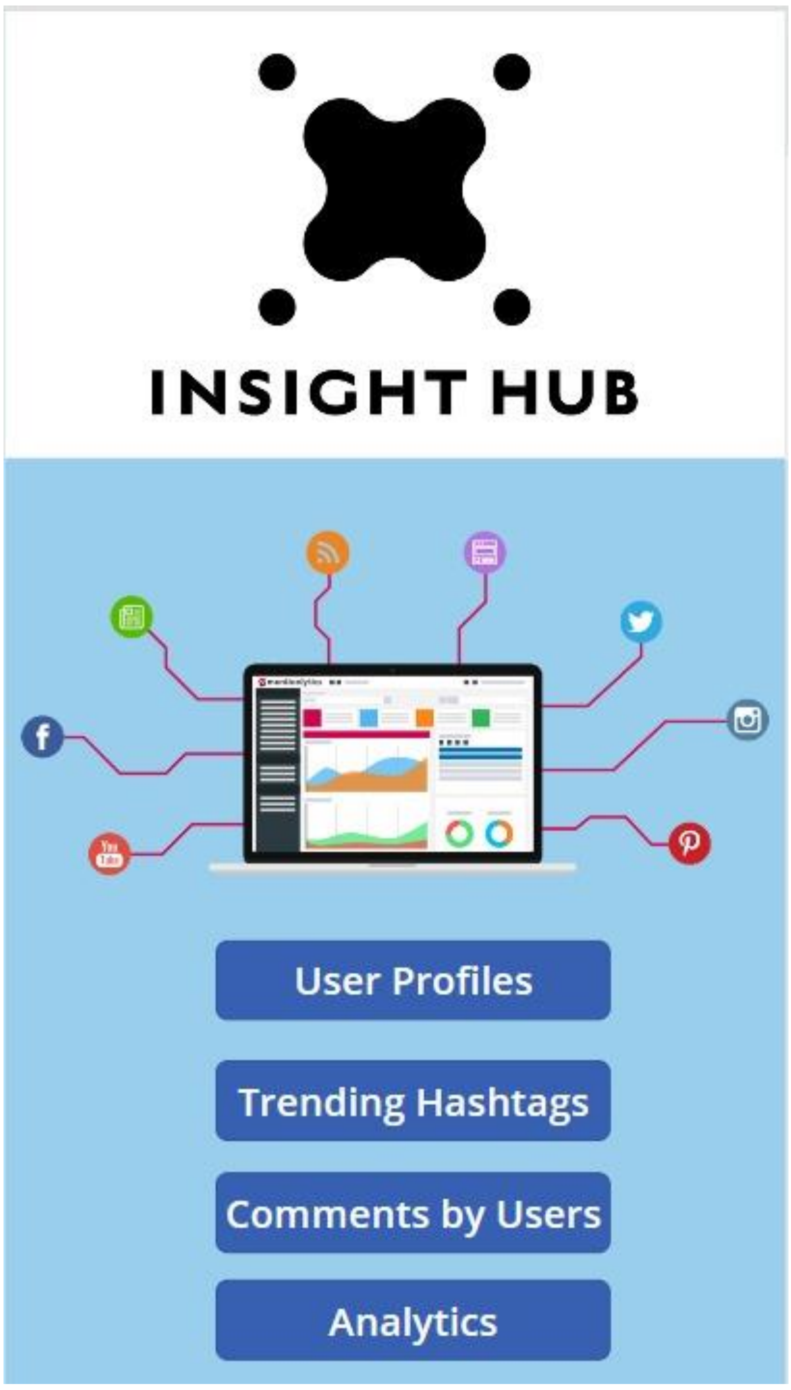
Project End Date: 5th December 2023

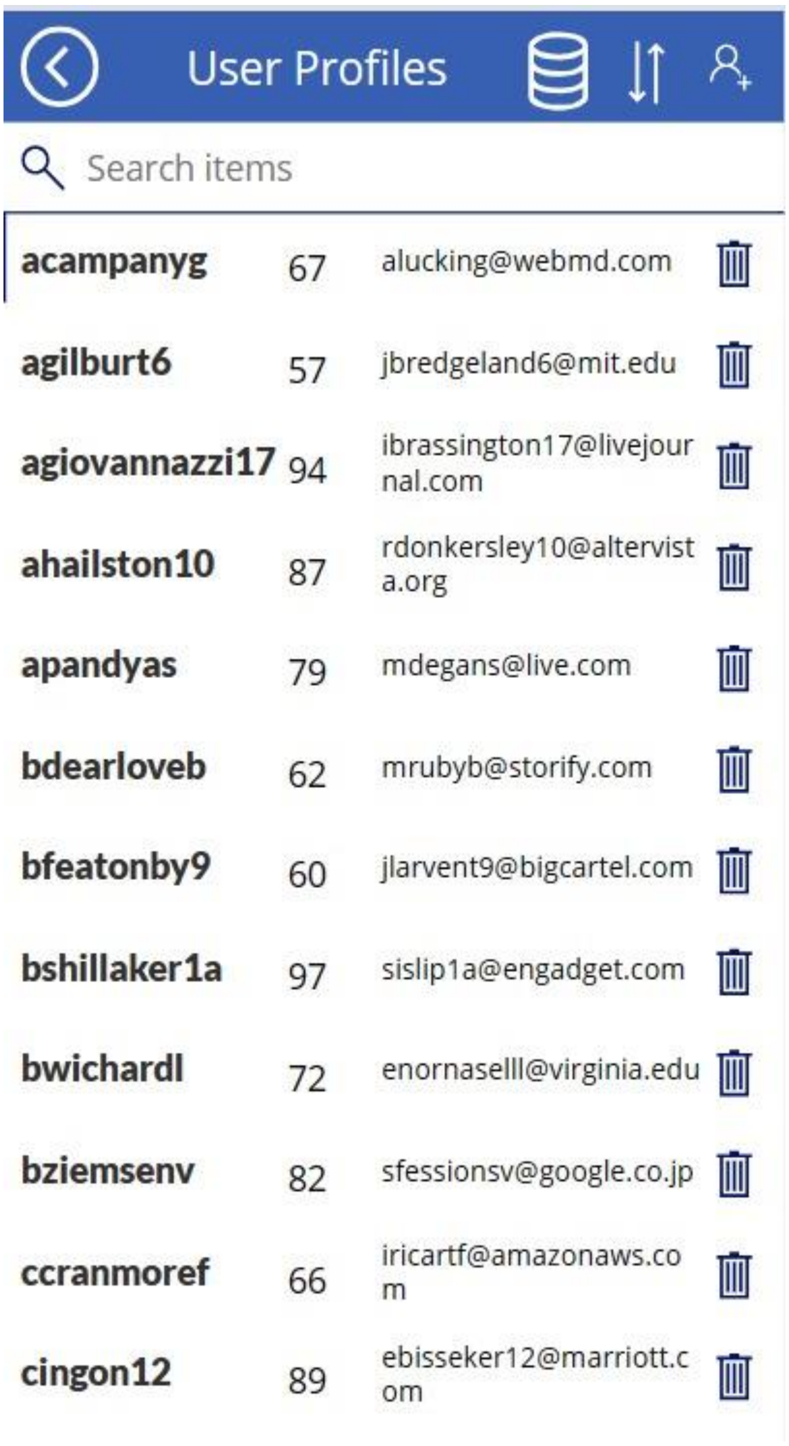
Project Team: Saarthak Joshi & Divyanshu Srivastava

Log Entries:

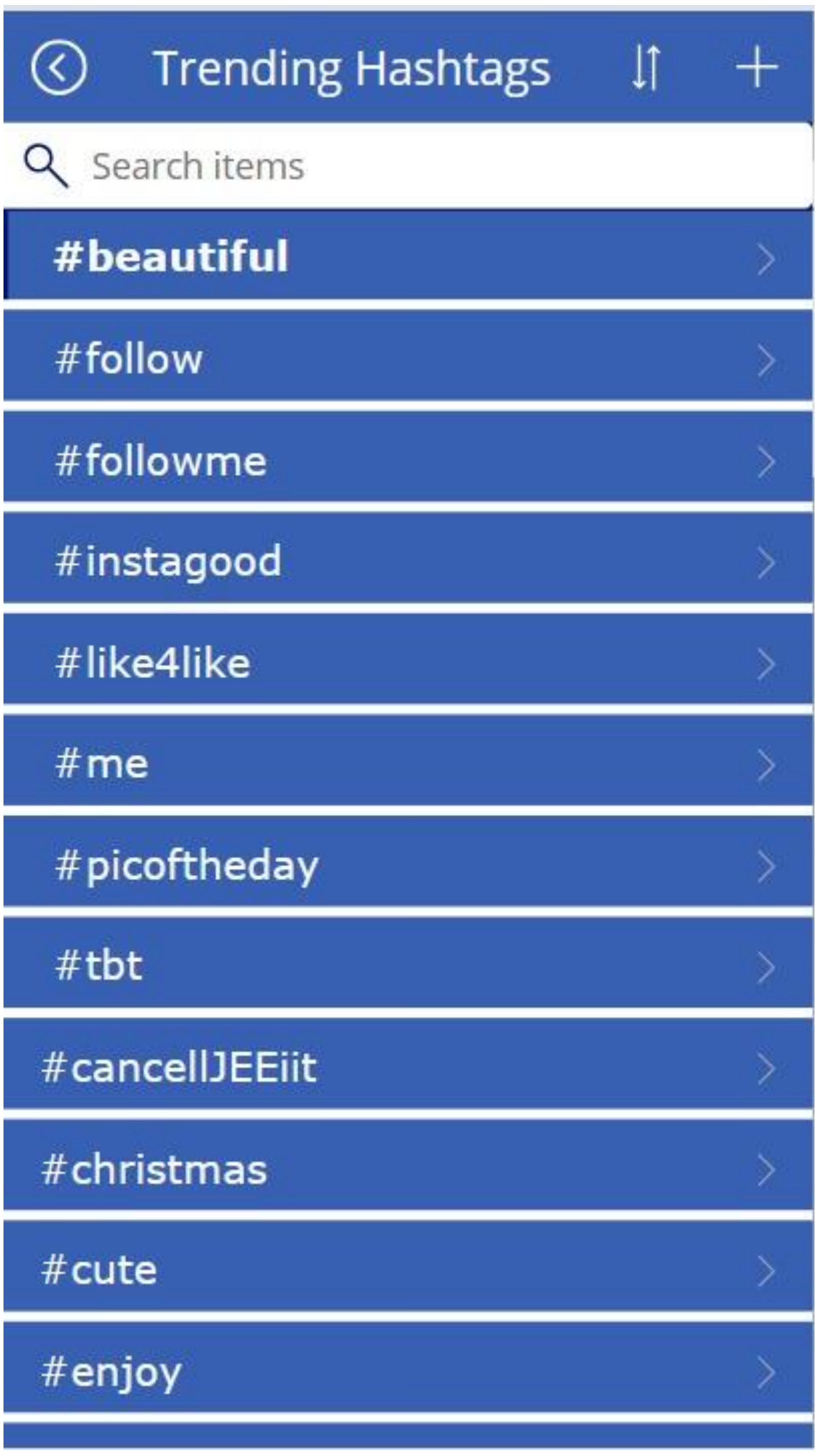
Date	Contributor Name	Contribution
12th Nov 2023	Saarthak Joshi	Data Collection
12th Nov 2023	Divyanshu Srivastava	Data Cleaning
15th Nov 2023	Divyanshu + Saarthak	Creating a rough DB Schema
15th Nov 2023	Divyanshu Srivastava	Creating Tables in Azure
15th Nov 2023	Saarthak Joshi	Adding constraints
18th Nov 2023	Divyanshu + Saarthak	Data Manipulation
23rd Nov 2023	Divyanshu + Saarthak	Writing SQL Queries
23rd Nov 2023	Divyanshu + Saarthak	MS Powerapps
28th Nov 2023	Saarthak Joshi	Data Visualization
28th Nov 2023	Divyanshu Srivastava	Embedding PowerBi Dashboard to Powerapps
2nd Dec 2023	Divyanshu Srivastava	PPT and App Demo
8th Dec 2023	Saarthak Joshi	Creating Project Report

❖ **APPLICATION SCREENS:**

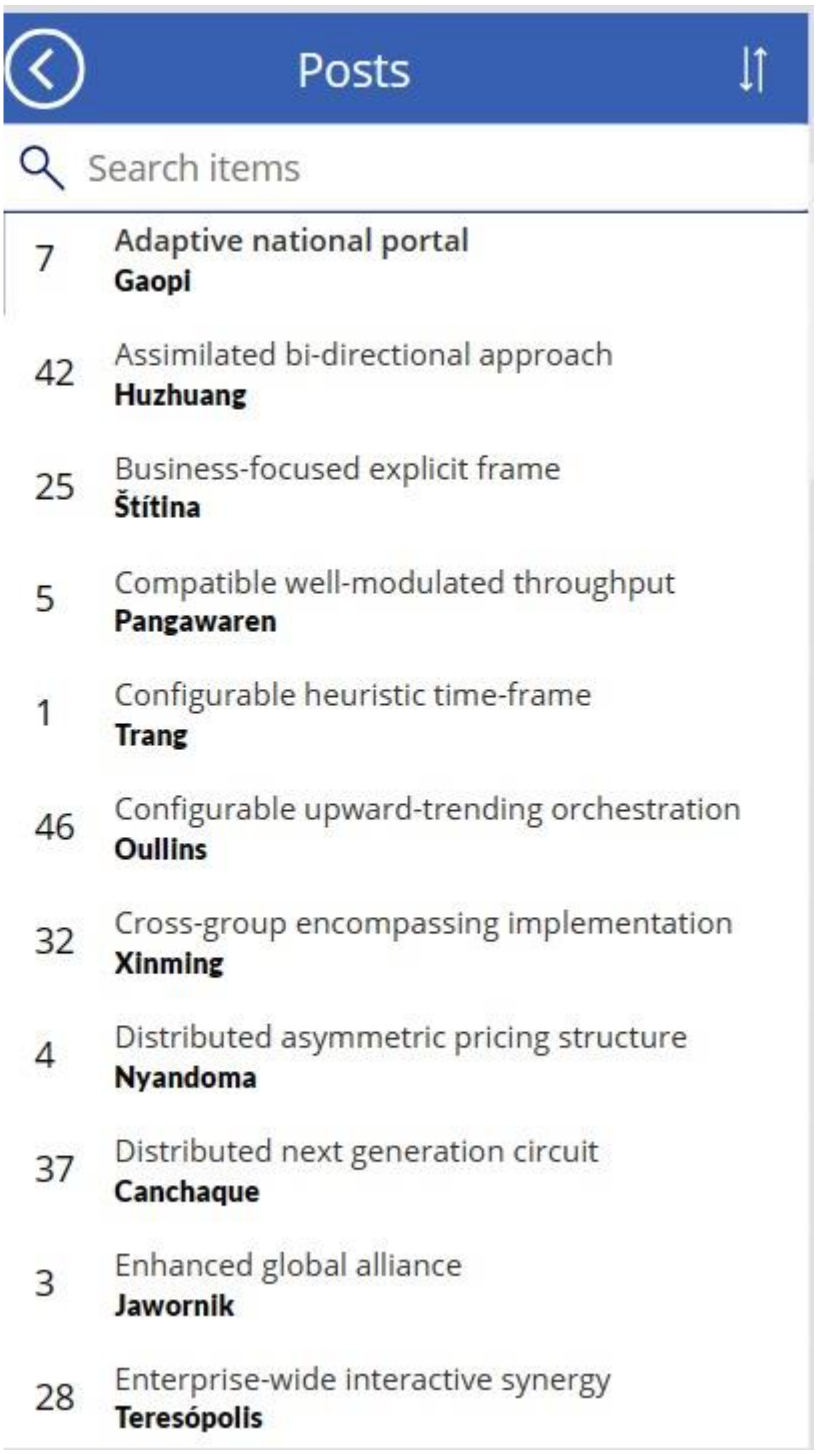




acampanyg	67	alucking@webmd.com
agilburt6	57	jbredgeland6@mit.edu
agiovannazzi17	94	ibrassington17@livejournal.com
ahailston10	87	rdonkersley10@altervista.org
apandyas	79	mdegans@live.com
bdearloveb	62	mrubyb@storify.com
bfeatonby9	60	jlarvent9@bigcartel.com
bshillaker1a	97	sislip1a@engadget.com
bwichardl	72	enornaselll@virginia.edu
bziemsenv	82	sfessionsv@google.co.jp
ccranmoref	66	iricartf@amazonaws.com
cingon12	89	ebisseker12@marriott.com

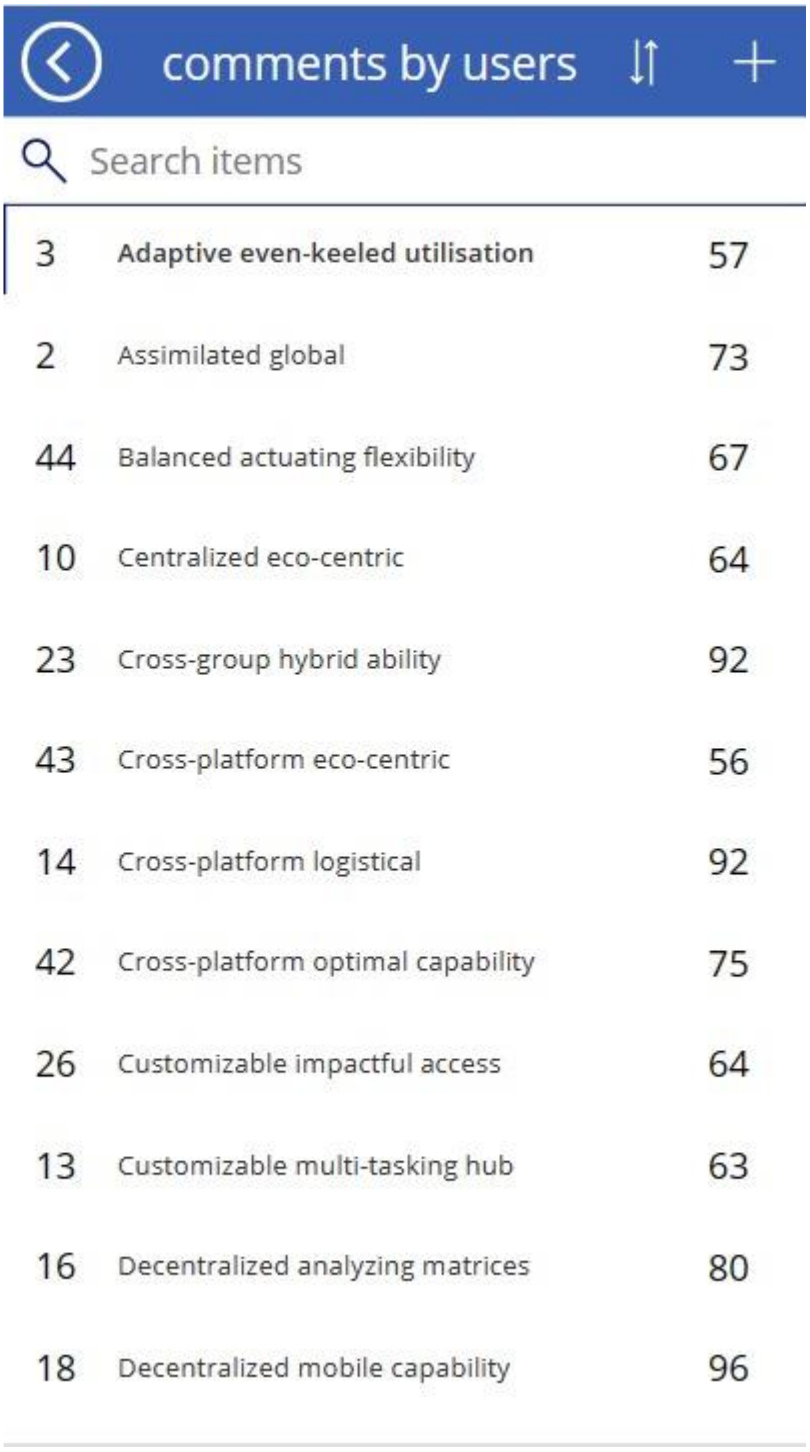


#beautiful
#follow
#followme
#instagood
#like4like
#me
#picoftheday
#tbt
#cancellJEEiit
#christmas
#cute
#enjoy

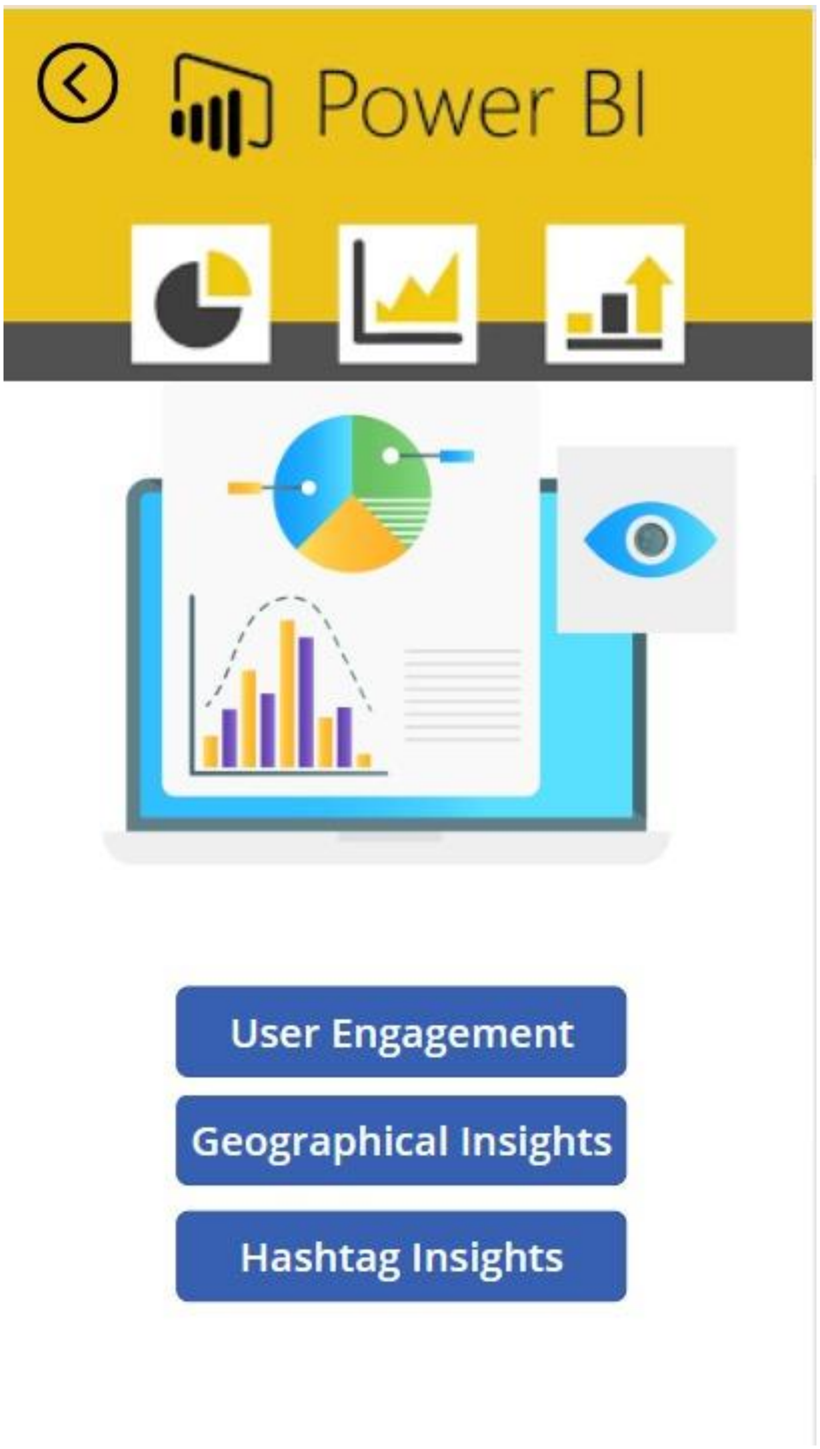

Posts

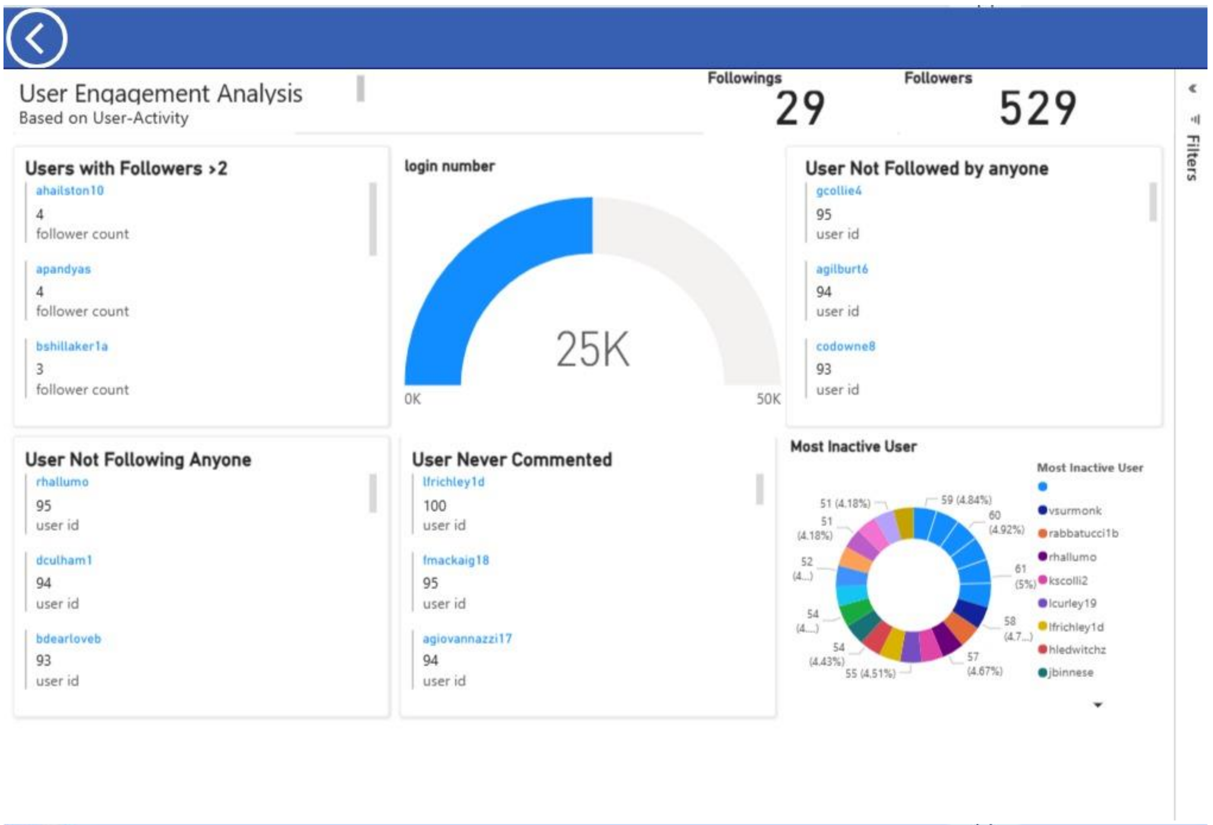
Search items

7	**Adaptive national portal** **Gaopi**
42	Assimilated bi-directional approach **Huzhuang**
25	Business-focused explicit frame **Štítina**
5	Compatible well-modulated throughput **Pangawaren**
1	Configurable heuristic time-frame **Trang**
46	Configurable upward-trending orchestration **Oullins**
32	Cross-group encompassing implementation **Xinming**
4	Distributed asymmetric pricing structure **Nyandoma**
37	Distributed next generation circuit **Canchaque**
3	Enhanced global alliance **Jawornik**
28	Enterprise-wide interactive synergy **Teresópolis**



3	**Adaptive even-keeled utilisation**	57
2	Assimilated global	73
44	Balanced actuating flexibility	67
10	Centralized eco-centric	64
23	Cross-group hybrid ability	92
43	Cross-platform eco-centric	56
14	Cross-platform logistical	92
42	Cross-platform optimal capability	75
26	Customizable impactful access	64
13	Customizable multi-tasking hub	63
16	Decentralized analyzing matrices	80
18	Decentralized mobile capability	96


Power BI
User Engagement
Geographical Insights
Hashtag Insights



User Engagement Analysis
Based on User-Activity

Followings 29
Followers 529

Users with Followers >2

ahailston10
4
follower count

apandyas
4
follower count

bshillaker1a
3
follower count

login number

User Not Followed by anyone

gcollie4
95
user id

agilburt6
94
user id

codowne8
93
user id

User Not Following Anyone

rhallumo
95
user id

dculham1
94
user id

bdearloveb
93
user id

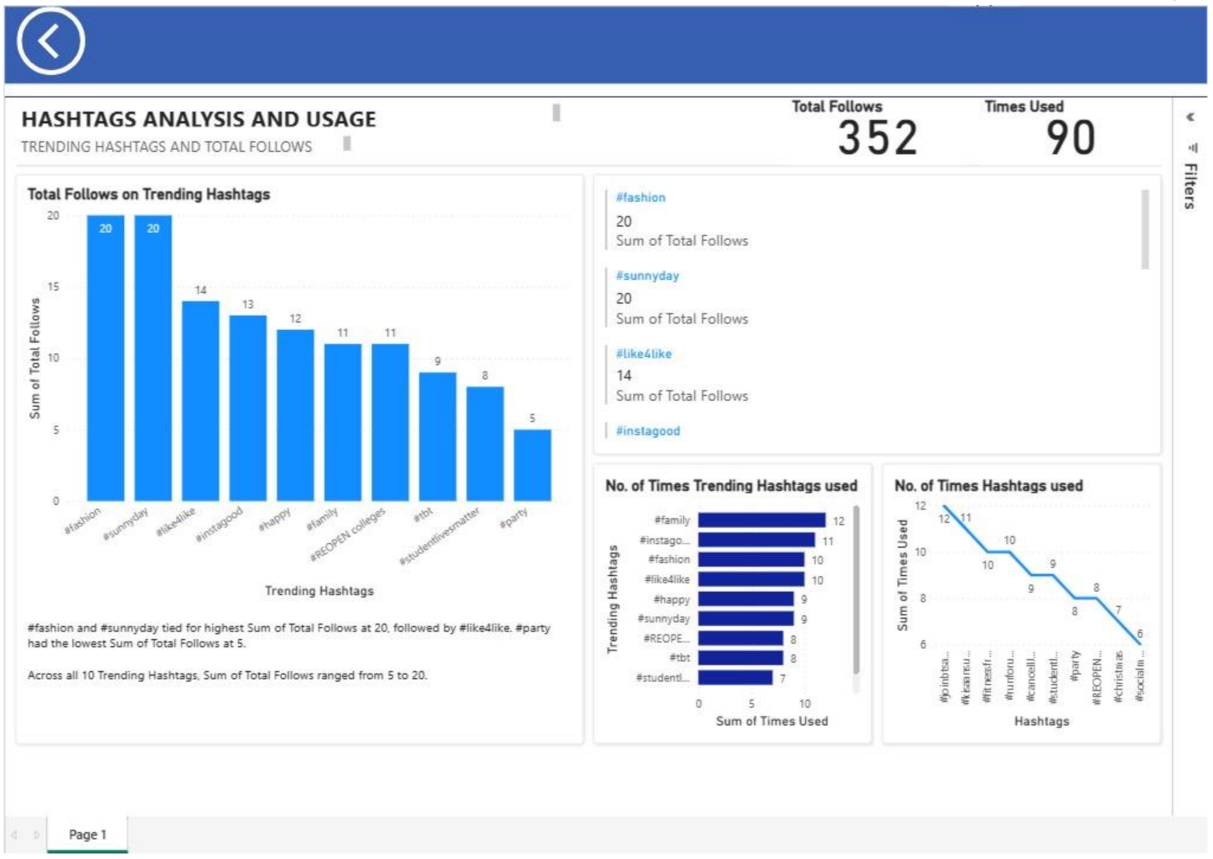
User Never Commented

lfrichley1d
100
user id

fmackaig18
95
user id

agiovannazzi17
94
user id

Most Inactive User



HASHTAGS ANALYSIS AND USAGE
TRENDING HASHTAGS AND TOTAL FOLLOWS

Total Follows 352
Times Used 90

Total Follows on Trending Hashtags

#fashion and #sunnyday tied for highest Sum of Total Follows at 20, followed by #like4like. #party had the lowest Sum of Total Follows at 5.

Across all 10 Trending Hashtags, Sum of Total Follows ranged from 5 to 20.

#fashion
20
Sum of Total Follows

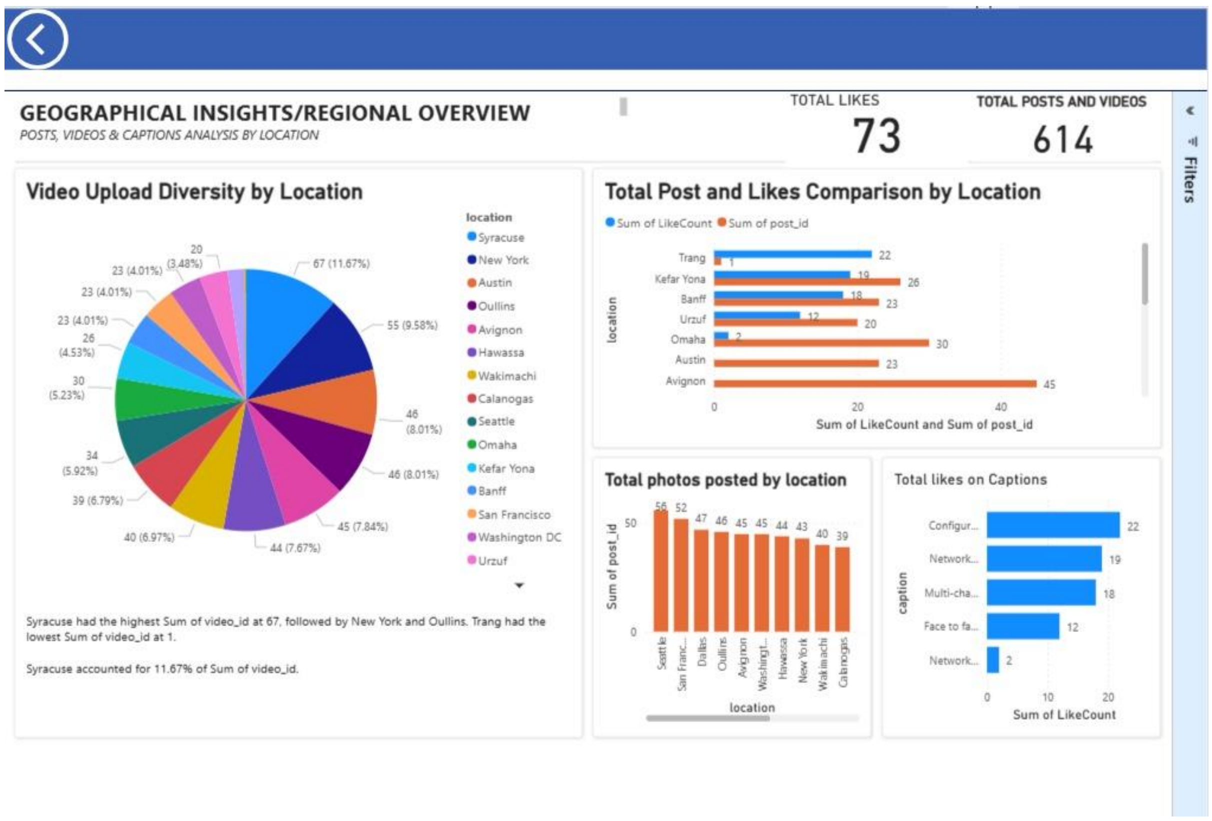
#sunnyday
20
Sum of Total Follows

#like4like
14
Sum of Total Follows

#instagood

No. of Times Trending Hashtags used

No. of Times Hashtags used



Syracuse had the highest Sum of video_id at 67, followed by New York and Oullins. Trang had the lowest Sum of video_id at 1.

Syracuse accounted for 11.67% of Sum of video_id.

<u>APPLICATION DEMO LINK</u>: **https://video.syr.edu/media/t/1_ae5q7h4z**

❖ SQL Up/Down script to implement the internal model with initial data.

```
-- Create userprofiles table
CREATE TABLE userprofiles (
    user_id INT PRIMARY KEY IDENTITY(1,1),
    username NVARCHAR(255) UNIQUE NOT NULL,
    profile_photo_url NVARCHAR(255) DEFAULT 'https://picsum.photos/100',
    bio NVARCHAR(255),
    email NVARCHAR(30) NOT NULL,
    created_at DATETIME2 DEFAULT CURRENT_TIMESTAMP
);

-- Create pictures table
CREATE TABLE pictures (
    photo_id INT PRIMARY KEY IDENTITY(1,1),
    photo_url NVARCHAR(255) NOT NULL UNIQUE,
    post_id INT NOT NULL,
    created_at DATETIME2 DEFAULT CURRENT_TIMESTAMP,
    size FLOAT CHECK (size < 5)
);

-- Create videos table
CREATE TABLE videos (
    video_id INT PRIMARY KEY IDENTITY(1,1),
    video_url NVARCHAR(255) NOT NULL UNIQUE,
    post_id INT NOT NULL,
    created_at DATETIME2 DEFAULT CURRENT_TIMESTAMP,
    size FLOAT CHECK (size < 10)
);

-- Create post table
CREATE TABLE post (
    post_id INT PRIMARY KEY IDENTITY(1,1),
    photo_id INT,
    video_id INT,
    user_id INT NOT NULL,
    caption NVARCHAR(200),
    location NVARCHAR(50),
    created_at DATETIME2 DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY(user_id) REFERENCES userprofiles(user_id),
    FOREIGN KEY(photo_id) REFERENCES pictures(photo_id),
    FOREIGN KEY(video_id) REFERENCES videos(video_id)
);

-- Create picturecomments table
CREATE TABLE picturecomments (
    comment_id INT PRIMARY KEY IDENTITY(1,1),
    comment_text NVARCHAR(255) NOT NULL,
    post_id INT NOT NULL,
    user_id INT NOT NULL,
    created_at DATETIME2 DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY(post_id) REFERENCES post(post_id),
    FOREIGN KEY(user_id) REFERENCES userprofiles(user_id)
);

-- Create post_likes table
CREATE TABLE post_likes (
    user_id INT NOT NULL,
    post_id INT NOT NULL,
    created_at DATETIME2 DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY(user_id) REFERENCES userprofiles(user_id),
    FOREIGN KEY(post_id) REFERENCES post(post_id),
    PRIMARY KEY(user_id, post_id)
);

-- Create comment_likes table
CREATE TABLE comment_likes (
    user_id INT NOT NULL,
    comment_id INT NOT NULL,
    created_at DATETIME2 DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY(user_id) REFERENCES userprofiles(user_id),
    FOREIGN KEY(comment_id) REFERENCES picturecomments(comment_id),
    PRIMARY KEY(user_id, comment_id)
);

-- Create followings table
CREATE TABLE followings (
    follower_id INT NOT NULL,
    followee_id INT NOT NULL,
    created_at DATETIME2 DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY(follower_id) REFERENCES userprofiles(user_id),
    FOREIGN KEY(followee_id) REFERENCES userprofiles(user_id),
    PRIMARY KEY(follower_id, followee_id)
);

-- Create hashtags table
CREATE TABLE hashtags (
    hashtag_id INT PRIMARY KEY IDENTITY(1,1),
    hashtag_name NVARCHAR(255) UNIQUE,
    created_at DATETIME2 DEFAULT CURRENT_TIMESTAMP
);

-- Create hashtag_follow table
CREATE TABLE hashtag_follow (
    user_id INT NOT NULL,
    hashtag_id INT NOT NULL,
    created_at DATETIME2 DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY(user_id) REFERENCES userprofiles(user_id),
    FOREIGN KEY(hashtag_id) REFERENCES hashtags(hashtag_id),
    PRIMARY KEY(user_id, hashtag_id)
);

-- Create post_tags table
```

```
CREATE TABLE post_tags (
    post_id INT NOT NULL,
    hashtag_id INT NOT NULL,
    FOREIGN KEY(post_id) REFERENCES post(post_id),
    FOREIGN KEY(hashtag_id) REFERENCES hashtags(hashtag_id),
    PRIMARY KEY(post_id, hashtag_id)
);
```

-- Drop tables in reverse order to maintain referential integrity

```
-- Drop post_tags table
DROP TABLE IF EXISTS post_tags;

-- Drop hashtag_follow table
DROP TABLE IF EXISTS hashtag_follow;

-- Drop hashtags table
DROP TABLE IF EXISTS hashtags;

-- Drop followings table
DROP TABLE IF EXISTS followings;

-- Drop comment_likes table
DROP TABLE IF EXISTS comment_likes;

-- Drop post_likes table
DROP TABLE IF EXISTS post_likes;

-- Drop picturecomments table
DROP TABLE IF EXISTS picturecomments;

-- Drop post table
DROP TABLE IF EXISTS post;
```

```
-- Drop videos table

DROP TABLE IF EXISTS videos;


-- Drop pictures table

DROP TABLE IF EXISTS pictures;


-- Drop userprofiles table

DROP TABLE IF EXISTS userprofiles;
```

All the insert commands for the below tables will come under their respective statements.

(Showcasing single insert commands for reference)

```
insert into userprofiles (username, profile_photo_url, bio, email)

insert into followings (follower_id, followee_id)
INSERT INTO HASHTAGS(hashtag_name)
INSERT INTO pictures(photo_url, post_id, size)
INSERT INTO videos(video_url, post_id, size)
insert into post (post_id, photo_id, video_id, user_id, caption, location)
INSERT INTO post_tags(post_id, hashtag_id)
insert into post_likes (user_id, post_id)
insert into bookmarks (post_id, user_id)
insert into picturecomments (comment_text, post_id, user_id)
insert into hashtag_follow (user_id, hashtag_id)
insert into login (user_id, IP)
```

- ## <u>SQL up script for data load/migration</u>

```
-- Insert data into userprofiles table

INSERT INTO userprofiles (username, profile_photo_url, bio, email, created_at)

VALUES

  ('john_doe', 'https://example.com/john_photo', 'Passionate about technology', 'john@example.com',
'2023-01-01 00:00:00'),
```

  -- We can add more insert statements as per our convenience;

```
-- Insert data into followings table

INSERT INTO followings (follower_id, followee_id, created_at)

VALUES

  (1, 2, '2023-01-01 00:00:00'),
```

  -- We can add more insert statements as per our convenience;

```
-- Insert data into hashtags table
INSERT INTO hashtags (hashtag_name, created_at)
VALUES
  ('technology', '2023-01-01 00:00:00'),
  -- We can add more insert statements as per our convenience;

-- Insert data into pictures table
INSERT INTO pictures (photo_url, post_id, size, created_at)
VALUES
  ('https://example.com/photo1', 1, 3.5, '2023-01-01 00:00:00'),
  -- We can add more insert statements as per our convenience;

-- Insert data into videos table
INSERT INTO videos (video_url, post_id, size, created_at)
VALUES
  ('https://example.com/video1', 2, 8.8, '2023-01-01 00:00:00'),
  -- We can add more insert statements as per our convenience;

-- Insert data into post table
INSERT INTO post (photo_id, video_id, user_id, caption, location, created_at)
VALUES
  (1, 2, 1, 'Exciting moments', 'City Park', '2023-01-01 00:00:00'),
  -- We can add more insert statements as per our convenience;

-- Insert data into post_tags table
INSERT INTO post_tags (post_id, hashtag_id)
VALUES
  (1, 1),
  -- We can add more insert statements as per our convenience;

-- Insert data into post_likes table
INSERT INTO post_likes (user_id, post_id, created_at)
VALUES
```

```
  (1, 1, '2023-01-01 00:00:00'),
  -- We can add more insert statements as per our convenience;


-- Insert data into bookmarks table
INSERT INTO bookmarks (post_id, user_id, created_at)
VALUES
  (1, 2, '2023-01-01 00:00:00'),
  -- We can add more insert statements as per our convenience;


-- Insert data into picturecomments table
INSERT INTO picturecomments (comment_text, post_id, user_id, created_at)
VALUES
  ('Great shot!', 1, 2, '2023-01-01 00:00:00'),
  -- We can add more insert statements as per our convenience;


-- Insert data into hashtag_follow table
INSERT INTO hashtag_follow (user_id, hashtag_id, created_at)
VALUES
  (1, 1, '2023-01-01 00:00:00'),
  -- We can add more insert statements as per our convenience;


-- Insert data into login table
INSERT INTO login (user_id, IP, created_at)
VALUES
  (1, '192.168.0.1', '2023-01-01 00:00:00');
  -- We can add more insert statements as per our convenience;
```

SQL DOWN script

```
-- Remove data from login table
DELETE FROM login WHERE user_id = 1 AND IP = '192.168.0.1';


-- Remove data from hashtag_follow table
DELETE FROM hashtag_follow WHERE user_id = 1 AND hashtag_id = 1;


-- Remove data from picturecomments table
```

```
DELETE FROM picturecomments WHERE comment_text = 'Great shot!' AND post_id = 1 AND
user_id = 2;


-- Remove data from bookmarks table

DELETE FROM bookmarks WHERE post_id = 1 AND user_id = 2;


-- Remove data from post_likes table

DELETE FROM post_likes WHERE user_id = 1 AND post_id = 1;


-- Remove data from post_tags table

DELETE FROM post_tags WHERE post_id = 1 AND hashtag_id = 1;


-- Remove data from post table

DELETE FROM post WHERE photo_id = 1 AND video_id = 2 AND user_id = 1 AND caption =
'Exciting moments' AND location = 'City Park';


-- Remove data from videos table

DELETE FROM videos WHERE video_url = 'https://example.com/video1' AND post_id = 2 AND size =
8.8;


-- Remove data from pictures table

DELETE FROM pictures WHERE photo_url = 'https://example.com/photo1' AND post_id = 1 AND size
= 3.5;
```

- **<u>Queries</u>**

```
    --Location of Users

SELECT * FROM post

WHERE location IN ('Trang ' ,'Omaha','Banff','Urzuf','Kefar
Yona','Hawassa','Oullins','Calanogas','Avignon','Wakimachi');


--Most Used Hashtags

SELECT hashtag_name AS 'Trending Hashtags',

    COUNT(post_tags.hashtag_id) AS 'Times Used'

FROM hashtags
```

```
Inner JOIN post_tags ON hashtags.hashtag_id = post_tags.hashtag_id
GROUP BY hashtags.hashtag_name
ORDER BY COUNT(post_tags.hashtag_id) DESC



-- Users who have not posted checked in post table to find out inactive  users
SELECT user_id, username AS 'Most Inactive User'
FROM userprofiles
WHERE user_id NOT IN
(SELECT user_id FROM post);




-- Count of Logins by user
SELECT count( distinct login.login_id) as login_counts,userprofiles.user_id, userprofiles.username
FROM userprofiles
LEFT JOIN login ON userprofiles.user_id = login.user_id
group by userprofiles.user_id, userprofiles.username

-- Most Liked Posts
SELECT TOP 5 post_likes.user_id, post_likes.post_id, COUNT(post_likes.post_id) AS 'LikeCount'
FROM post_likes
JOIN post ON post.post_id = post_likes.post_id
GROUP BY post_likes.user_id, post_likes.post_id
ORDER BY COUNT(post_likes.post_id) DESC;



select TOP 5  post_likes.user_id,userprofiles.username, COUNT(post_likes.post_id) AS 'LikeCount'
FROM post_likes
inner join userprofiles on post_likes.user_id = userprofiles.user_id
 group by post_likes.user_id , userprofiles.username
```

```
-- Average post per user
SELECT ROUND((COUNT(post_id) / COUNT(DISTINCT user_id) ),2) AS 'Average Post per User'
FROM post;

-- no. of login by per user
SELECT userprofiles.user_id, userprofiles.username, login.login_id AS login_number
FROM userprofiles
RIGHT JOIN login ON userprofiles.user_id = login.user_id;



-- User who liked every single post (CHECK FOR BOT)
SELECT userprofiles.username, COUNT(post_likes.post_id) AS num_likes
FROM userprofiles
INNER JOIN post_likes ON userprofiles.user_id = post_likes.user_id
GROUP BY userprofiles.user_id, userprofiles.username
HAVING COUNT() = (SELECT COUNT() FROM post);

--  Users who Never Comment
SELECT user_id, username AS 'Users who Never Comment'
FROM userprofiles
WHERE user_id NOT IN (SELECT user_id FROM comments);

-- User who commented on every post (CHECK FOR BOT)
SELECT userprofiles.username, Count(*) AS num_comment
FROM userprofiles
INNER JOIN picturecomments ON picturecomments.user_id = userprofiles.user_id
GROUP  BY picturecomments.user_id
HAVING COUNT() = (SELECT COUNT() FROM picturecomments );



-- User Not Followed by anyone
SELECT user_id, username AS 'User Not Followed by anyone'
FROM userprofiles
WHERE user_id NOT IN (SELECT followee_id FROM follows);
```

```sql
-- User Not Following Anyone
SELECT user_id, username AS 'User Not Following Anyone'
FROM userprofiles
WHERE user_id NOT IN (SELECT follower_id FROM follows);

-- 13. Posted more than 2 times
SELECT user_id, COUNT(*) AS post_count
FROM post
GROUP BY user_id
HAVING COUNT(*) > 2
ORDER BY post_count DESC;


-- 14. Followers > 2
SELECT followee_id, COUNT(follower_id) AS follower_count
FROM follows
GROUP BY followee_id
HAVING COUNT(follower_id) > 2
ORDER BY follower_count DESC;


-- 15. Any specific word in comment
SELECT *
FROM picturecomments
WHERE comment_text LIKE '%eco%';

-- 16. Longest captions in post
SELECT TOP 5 user_id, caption, LEN(post.caption) AS caption_length
FROM post
ORDER BY caption_length DESC
```