Problem Statment :

In the competitive restaurant industry, stakeholders often lack clear insights into which user engagement factors contribute most to business success. This project seeks to address the question: How do user engagement metrics (e.g., reviews, tips, and check-ins) correlate with key business performance indicators (such as review count and ratings) for How do user engagement metrics, such as reviews, tips, and check-ins, influence key business performance indicators like review counts and ratings for restaurants? In the highly competitive restaurant industry, understanding these relationships is crucial for stakeholders aiming to enhance business success. providing data-driven insights that can guide strategic decision-making and promote sustainable growth .? By analyzing the Yelp dataset, the goal is to identify actionable patterns that can guide strategic decisions and enhance restaurant performance.

General Research Objectives :

Quantify the Correlation Between User Engagement and Business Metrics: Assess the relationship between user engagement factors—such as reviews, tips, and check-ins—and business metrics, including review count and average star rating. This will reveal whether restaurants with higher user engagement tend to have higher ratings and more reviews.

Analyze the Impact of Sentiment on Business Performance: Investigate whether positive sentiment in reviews and tips is associated with

higher star ratings and an increase in the total number of reviews. This analysis will help determine if sentiment influences business success.

Identify Time Trends in User Engagement: Explore whether consistent user engagement over time serves as a stronger indicator of long-term success compared to sporadic bursts of activity. This will provide insights into engagement patterns that contribute to sustained business growth.

Adapt Research Based on Emerging Insights: Modify the research focus as necessary, based on findings and insights gained during analysis, to capture additional factors relevant to business success.

Hypothesis Testing :

User Engagement and Business Performance: Higher levels of user engagement, such as increased reviews, tips, and check-ins, are positively correlated with higher review counts and ratings for restaurants.

Sentiment Impact on Ratings and Review Counts: Positive sentiments expressed in reviews and tips contribute to higher overall ratings and increased review counts for restaurants.

Consistent Engagement and Long-term Success: Consistent user engagement over time is positively associated with sustained business success for restaurants.

About Dataset :

This Yelp dataset contains information across eight metropolitan areas in the USA and Canada, organized into five primary tables for analysis:

`Business`: Contains details on 131,930 businesses, including over 1.2 million attributes like hours, parking, availability, and ambiance.

`Review`: Includes user-generated reviews with star ratings, text, and timestamps.

`User`: Information on 1,987,897 users, including activity and interaction data.

`Tip`: Contains 908,915 tips from users, providing short advice about businesses.

`Check`-in: Aggregates check-in data over time for each business, enabling time-based analysis of user visits.

Importing Necessary Libraries Below :

In [79]:
```
from inspect import stack

import pandas as pd
import numpy as np
from matplotlib.pyplot import subplot
from rich.jupyter import display
from sqlalchemy import create_engine, text
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.colors import LinearSegmentedColormap
from datetime import datetime
import folium
from geopy.geocoders import Nominatim
import warnings
warnings.filterwarnings('ignore')
```

Creating MySQL connection Using SQLAlchemy Engine :

In [2]:
```
from sqlalchemy import create_engine

# Create an engine that connects to your MySQL database
engine = create_engine('mysql+mysqlconnector://root:Vermasuryanshu%40110906@localho

# Verify connection
try:

    # Here, you can use pandas to directly interact with the database
    print("Successfully connected to the database using SQLAlchemy!")
except Exception as e:
    print(f"Error: {e}")
```

```
Successfully connected to the database using SQLAlchemy!
```

Reading the tables using sqlAlchemy :

In [3]:
```
try:
    # Open the connection using the engine and use it as a context manager
    with engine.connect() as sql:
        print("Successfully connected to the database using SQLAlchemy!")
        # Execute SHOW TABLES query
        result = sql.execute(text("SHOW TABLES;"))

        # Fetch and print the tables
        print("Tables in the database:")
        for row in result:
            print(row[0])  # Print the name of each table

except Exception as e:
    print(f"Error: {e}")
```

```
Successfully connected to the database using SQLAlchemy!
Tables in the database:
business
checkin
checkin_df_csv
review
tip
user
```

In [4]:
```
# Reading data from the 'business' table using pandas
business = pd.read_sql("SELECT * FROM business;", engine)
business # Display the first few rows of the DataFrame
```

Out[4]:

	business_id	name	address	city	state	postal_code
0	Pns2l4eNsfO8kk83dixA6A	Abby Rappoport, LAC, CMQ	1616 Chapala St, Ste 2	Santa Barbara	CA	93101
1	mpf3x-BjTdTEA3yCZrAYPw	The UPS Store	87 Grasso Plaza Shopping Center	Affton	MO	63123
2	tUFrWirKiKi_TAnsVWINQQ	Target	5255 E Broadway Blvd	Tucson	AZ	85711
3	MTSW4McQd7CbVtyjqoe9mw	St Honore Pastries	935 Race St	Philadelphia	PA	19107
4	mWMc6_wTdE0EUBKIGXDVfA	Perkiomen Valley Brewery	101 Walnut St	Green Lane	PA	18054
...	...	...	...	...	...	...
150341	IUQopTMmYQG-qRtBk-8QnA	Binh's Nails	3388 Gateway Blvd	Edmonton	AB	T6J 5H2
150342	c8GjPIOTGVmIemT7j5_SyQ	Wild Birds Unlimited	2813 Bransford Ave	Nashville	TN	37204
150343	_QAMST-NrQobXduilWEqSw	Claire's Boutique	6020 E 82nd St, Ste 46	Indianapolis	IN	46250
150344	mtGm22y5c2UHNXDFAjaPNw	Cyclery & Fitness Center	2472 Troy Rd	Edwardsville	IL	62025
150345	jV_XOycEzSlTx-65W906pg	Sic Ink	238 Apollo Beach Blvd	Apollo beach	FL	33572

150346 rows × 12 columns

In [5]:
```
# Reading data from the 'checkin' table using pandas
checkin = pd.read_sql("SELECT * FROM checkin;", engine)
checkin # Display the first few rows of the DataFrame
```

Out[5]:

	business_id	date
0	---kPU91CF4Lq2-WlRu9Lw	2020-03-13 21:10:56, 2020-06-02 22:18:06, 2020...
1	--0iUa4sNDFiZFrAdIWhZQ	2010-09-13 21:43:09, 2011-05-04 23:08:15, 2011...
2	--30_8IhuyMHbSOcNWd6DQ	2013-06-14 23:29:17, 2014-08-13 23:20:22
3	--7PUidqRWpRSpXebiyxTg	2011-02-15 17:12:00, 2011-07-28 02:46:10, 2012...
4	--7jw19RH9JKXgFohspgQw	2014-04-21 20:42:11, 2014-04-28 21:04:46, 2014...
...	...	...
131718	zznJox6-nmXlGYNWgTDwQQ	2013-03-23 16:22:47, 2013-04-07 02:03:12, 2013...
131719	zznZqH9CiAznbkV6fXyHWA	2021-06-12 01:16:12
131720	zzu6_r3DxBJuXcjnOYVdTw	2011-05-24 01:35:13, 2012-01-01 23:44:33, 2012...
131721	zzw66H6hVjXQEt0Js3Mo4A	2016-12-03 23:33:26, 2018-12-02 19:08:45
131722	zzyx5x0Z7xXWWvWnZFuxlQ	2015-01-06 17:51:53

131723 rows × 2 columns

In [6]:
```
# Reading data from the 'review' table using pandas
review = pd.read_sql("SELECT * FROM review;", engine) # Display the first few rows
review
```

Out[6]:

	review_id	user_id	business_
0	KU_O5udG6zpxOg-VcAEodg	mh_-eMZ6K5RLWhZyISBhwA	XQfwVwDr-v0ZS3_CbbE5X
1	BiTunyQ73aT9WBnpR9DZGw	OyoGAe7OKpv6SyGZT5g77Q	7ATYjTIgM3jUlt4UM3Iyp
2	saUsX_uimxRlCVr67Z4Jig	8g_iMtfSiwikVnbP2etR0A	YjUWPpI6HXG530lwP-fb
3	AqPFMleE6RsU23_auESxiA	_7bHUi9Uuf5__HHc_Q8guQ	kxX2SOes4o-D3ZQBkiMR
4	Sx8TMOWLNuJBWer-0pcmoA	bcjbaE6dDog4jkNY91ncLQ	e4Vwtrqf-wpJfwesgvdg
...	...	...	
6990275	H0RIamZu0B0Ei0P4aeh3sQ	qskILQ3k0I_qcCMI-k6_QQ	jals67o91gcrD4DC81Vk
6990276	shTPgbgdwTHSuU67mGCmZQ	Zo0th2m8Ez4gLSbHftiQvg	2vLksaMmSEcGbjI5gywp

	review_id	user_id	business_
6990277	YNfNhgZlaaCO5Q_YJR4rEw	mm6E4FbCMwJmb7kPDZ5v2Q	R1khUUxidqfaJmcpmGd4a
6990278	i-I4ZOhoX70Nw5H0FwrQUA	YwAMC-jvZ1fvEUum6QkEkw	Rr9kKArrMhSLVE9a53q-
6990279	RwcKOdEuLRHNJe4M9-qpqg	6JehEvdoCvZPJ_XlxnzIlw	VAeEXLbEcI9Emt9KGYq9

6990280 rows × 9 columns

In [7]:
```
# Reading data from the 'tip' table using pandas
tip = pd.read_sql("SELECT * FROM tip;", engine) # Display the first few rows of the
tip
```

Out[7]:

	user_id	business_id	text	date	com
0	AGNUgVwnZUey3gcPCJ76iw	3uLgwr0qeCNMjKenHJwPGQ	Avengers time with the ladies.	2012-05-18 02:17:21	
1	NBN4MgHP9D3cw--SnauTkA	QoezRbYQncpRqyrLH6Iqjg	They have lots of good deserts and tasty cuban...	2013-02-05 18:35:10	
2	-copOvldyKh1qr-vzkDEvw	MYoRNLb5chwjQe3c_k37Gg	It's open even when you think it isn't	2013-08-18 00:56:08	
3	FjMQVZjSqY8syIO-53KFKw	hV-bABTK-glh5wj31ps_Jw	Very decent fried chicken	2017-06-27 23:05:38	
4	ld0AperBXk1h6UbqmM80zw	_uN0OudeJ3Zl_tf6nxg5ww	Appetizers.. platter special for lunch	2012-10-06 19:43:09	
...	...	...	...	...	
908910	eYodOTF8pkqKPzHkcxZs-Q	3lHTewuKFt5IlmbXJoFeDQ	Disappointed in one of your managers.	2021-09-11 19:18:57	
908911	1uxtQAuJ2T5Xwa_wp7kUnA	OaGf0Dp56ARhQwIDT90w_g	Great food and service.	2021-10-30 11:54:36	
908912	v48Spe6WEpqehsF2xQADpg	hYnMeAO77RGyTtIzUSKYzQ	Love their Cubans!!	2021-11-05 13:18:56	
908913	ckqKGM2hl7I9Chp5IpAhkw	s2eyoTuJrcP7I_XyjdhUHQ	Great pizza great price	2021-11-20 16:11:44	
908914	4tF1CWdMxvvwpUIgGsDygA	_cb1Vg1NIWry8UA0jyuXnQ	Food is good value but a bit hot!	2021-12-07 22:30:00	

908915 rows × 5 columns

In [8]:

```
# Reading data from the 'user' table using pandas
user = pd.read_sql("SELECT * FROM user;", engine) # Display the first few rows of t
user
```

Out[8]:

	user_id	name	review_count	yelping_since	C5	funny
0	qVc8ODYU5SZjKXVBgXdI7w	Walker	585	2007-01-25 16:47:26	7217	1259
1	j14WgRoU_-2ZE1aw1dXrJg	Daniel	4333	2009-01-25 04:35:42	43091	13066
2	2WnXYQFK0hXEoTxPtV2zvg	Steph	665	2008-07-25 10:41:00	2086	1010
3	SZDeASXq7o05mMNLshsdIA	Gwen	224	2005-11-29 04:38:33	512	330
4	hA5lMy-EnncsH4JoR-hFGQ	Karen	79	2007-01-05 19:40:59	29	15
...	...	...	...	...	...	...
1983818	fB3jbHi3m0L2KgGOxBv6uw	Jerrold	23	2015-01-06 00:31:31	7	0
1983819	68czcr4BxJyMQ9cJBm6C7Q	Jane	1	2016-06-14 07:20:52	0	0
1983820	1x3KMskYxOuJCjRz70xOqQ	Shomari	4	2017-02-04 15:31:58	1	1
1983821	ulfGl4tdbrH05xKzh5lnog	Susanne	2	2011-01-14 00:29:08	0	0
1983822	wL5jPrLRVCK_Pmo4lM1zpA	Isa	2	2020-12-19 02:32:39	0	0

1983823 rows × 22 columns

Data Analysis :

In [9]:
```
business_Open_Restaurants=pd.read_sql("select * from business where is_open = 1 AND
```

In [10]:
```
business_Open_Restaurants # Open Restaurants
```

Out[10]:

	business_id	name	address	city	state	postal_code
0	MTSW4McQd7CbVtyjqoe9mw	St Honore Pastries	935 Race St	Philadelphia	PA	19107
1	CF33F8-E6oudUQ46HnavjQ	Sonic Drive-In	615 S Main St	Ashland City	TN	37015
2	bBDDEgkFA1Otx9Lfe7BZUQ	Sonic Drive-In	2312 Dickerson Pike	Nashville	TN	37207
3	eEOYSgkmpB90uNA7lDOMRA	Vietnamese Food Truck	None	Tampa Bay	FL	33602
4	il_Ro8jwPlHresjw9EGmBg	Denny's	8901 US 31 S	Indianapolis	IN	46227
...	...	...	...	...	...	...
34999	w_4xUt-1AyY2ZwKtnjW0Xg	Bittercreek Alehouse	246 N 8th St	Boise	ID	83702
35000	l9eLGG9ZKpLJzboZq-9LRQ	Wawa	19 N Bishop Ave	Clifton Heights	PA	19018
35001	cM6V90ExQD6KMSU3rRB5ZA	Dutch Bros Coffee	1181 N Milwaukee St	Boise	ID	83704
35002	WnT9NIzQgLlILjPT0kEcsQ	Adelita Taqueria & Restaurant	1108 S 9th St	Philadelphia	PA	19147
35003	2O2K6SXPWv56amqxCECd4w	The Plum Pit	4405 Pennell Rd	Aston	DE	19014

35004 rows × 12 columns

Out of 150K Businesses, 35K are Restaurant Business and are Open.

Q. What is the descriptive stats for review count and star rating for businesses ?

In [11]:
```
query = """
SELECT
    AVG(review_count) AS average_review,
    MIN(review_count) AS min_review,
    MAX(review_count) AS max_review,
    AVG(stars) AS average_star_rating,
    MIN(stars) AS min_star_rating,
    MAX(stars) AS max_star_rating
FROM (
    SELECT *
    FROM business
    WHERE is_open = 1
    AND LOWER(categories) LIKE '%restaurant%'
) AS business_Open_Restaurants;
"""



# Execute the query and display the result
review_business = pd.read_sql(query, engine)
```

In [12]:
```
# Calculating the median for the both Stars & Review_count
review_business.insert(3,'median_review',[business['review_count'].median()])
review_business.insert(7,'median_stars',[business['stars'].median()])
```

In [13]:
```
review_business # As per the stats, We can conclude that the review columns include
```

Out[13]:

	average_review	min_review	max_review	median_review	average_star_rating	min_star_r
0	104.0978	5	7568	15.0	3.523969	

In [14]:
```
review_business=review_business.T
```

In [15]:
```
review_business
```

Out[15]:

	0
average_review	104.097800
min_review	5.000000
max_review	7568.000000
median_review	15.000000
average_star_rating	3.523969
min_star_rating	1.000000
max_star_rating	5.000000
median_stars	3.500000

The Max_review is outlier and data is skewed, To address this I decided to remove restaurants with outlier review counts, For this I created a Function to identify & remove outliers using IQR method.

In [16]:

```
# Convert column to integer, if it contains boolean values by mistake
business_Open_Restaurants['review_count'] = business_Open_Restaurants['review_count
```

In [17]:

```
#  Removing Outliers using IQR
# Defining a function called --> remove_outlier
def remove_outlier(df,col):
    q1 = df[col].quantile(0.25)
    q3 = df[col].quantile(0.75)
    iqr = q3-q1
    lower_bound = q1 - 1.5 * iqr
    upper_bound = q3 + 1.5 * iqr
    df = df[(df[col] >= lower_bound) & (df[col] <= upper_bound)]
    return df


# calling the function
business_Open_Restaurants = remove_outlier(business_Open_Restaurants,'review_count'
```

In [18]:

```
business_Open_Restaurants
```

Out[18]:

	business_id	name	address	city	state	postal_code
0	MTSW4McQd7CbVtyjqoe9mw	St Honore Pastries	935 Race St	Philadelphia	PA	19107
1	CF33F8-E6oudUQ46HnavjQ	Sonic Drive-In	615 S Main St	Ashland City	TN	37015
2	bBDDEgkFA1Otx9Lfe7BZUQ	Sonic Drive-In	2312 Dickerson Pike	Nashville	TN	37207
3	eEOYSgkmpB90uNA7lDOMRA	Vietnamese Food Truck	None	Tampa Bay	FL	33602
4	il_Ro8jwPlHresjw9EGmBg	Denny's	8901 US 31 S	Indianapolis	IN	46227
...	...	...	...	...	...	...
34998	sf_oQ62L8UEnOOLf00nNGA	Pizza Hut	5028 Old Hickory	Hermitage	TN	37076
35000	l9eLGG9ZKpLJzboZq-9LRQ	Wawa	19 N Bishop Ave	Clifton Heights	PA	19018
35001	cM6V90ExQD6KMSU3rRB5ZA	Dutch Bros Coffee	1181 N Milwaukee St	Boise	ID	83704
35002	WnT9NIzQgLlILjPT0kEcsQ	Adelita Taqueria & Restaurant	1108 S 9th St	Philadelphia	PA	19147
35003	2O2K6SXPWv56amqxCECd4w	The Plum Pit	4405 Pennell Rd	Aston	DE	19014

31537 rows × 12 columns

In [19]:
```
business_Open_Restaurants['review_count'].describe() # its reflected the change and
```

Out[19]:
```
count    31537.000000
mean        55.975426
std         56.559679
min          5.000000
25%         14.000000
50%         33.000000
75%         79.000000
max        248.000000
Name: review_count, dtype: float64
```

After Removing outliers, Now I get average review count as 55.975 for the restaurant business.

Open Businesses Which Are Restaurant :

In [20]:
```
business_Open_Restaurants
```

Out[20]:

	business_id	name	address	city	state	postal_code
0	MTSW4McQd7CbVtyjqoe9mw	St Honore Pastries	935 Race St	Philadelphia	PA	19107
1	CF33F8-E6oudUQ46HnavjQ	Sonic Drive-In	615 S Main St	Ashland City	TN	37015
2	bBDDEgkFA1Otx9Lfe7BZUQ	Sonic Drive-In	2312 Dickerson Pike	Nashville	TN	37207
3	eEOYSgkmpB90uNA7lDOMRA	Vietnamese Food Truck	None	Tampa Bay	FL	33602
4	il_Ro8jwPlHresjw9EGmBg	Denny's	8901 US 31 S	Indianapolis	IN	46227
...	...	...	...	...	...	...
34998	sf_oQ62L8UEnOOLf00nNGA	Pizza Hut	5028 Old Hickory	Hermitage	TN	37076
35000	l9eLGG9ZKpLJzboZq-9LRQ	Wawa	19 N Bishop Ave	Clifton Heights	PA	19018
35001	cM6V90ExQD6KMSU3rRB5ZA	Dutch Bros Coffee	1181 N Milwaukee St	Boise	ID	83704
35002	WnT9NIzQgLlILjPT0kEcsQ	Adelita Taqueria & Restaurant	1108 S 9th St	Philadelphia	PA	19147
35003	2O2K6SXPWv56amqxCECd4w	The Plum Pit	4405 Pennell Rd	Aston	DE	19014

31537 rows × 12 columns

Q. Which restaurant have the higest number of reviews ?

In [21]:
```
pd.read_sql(""" select name, SUM(review_count) AS Higest_Review_Counts, AVG(stars)
```

Out[21]:

	name	Higest_Review_Counts	Average_Rating
0	McDonald's	16490.0	1.868702
1	Chipotle Mexican Grill	9071.0	2.381757
2	First Watch	8688.0	3.896552
3	Acme Oyster House	8343.0	4.000000
4	Taco Bell	8017.0	2.141813
5	Chick-fil-A	7967.0	3.373418
6	Oceana Grill	7400.0	4.000000
7	Buffalo Wild Wings	6810.0	2.347458
8	Panera Bread	6613.0	2.661905
9	Hattie B's Hot Chicken - Nashville	6093.0	4.500000

Q. Which restaurant have the highest number of highest rating ?

In [22]:
```
pd.read_sql(""" select name, SUM(review_count) AS Higest_Review_Counts, AVG(stars)
```

Out[22]:

	name	Higest_Review_Counts	Average_Rating
7	Vegan International Co. Kitchen & Market	269.0	5.0
6	The Foundry Bakery	185.0	5.0
9	Jet City Espresso Hyde Park	152.0	5.0
1	bb.q Chicken - O'Fallon	42.0	5.0
3	Asia Mix Restaurant	10.0	5.0
5	Healthy Soul Indy	9.0	5.0
8	Antojitos Carmen Restaurante Y Taqueria	9.0	5.0
2	Tacos Don Vicente	8.0	5.0
0	YWCA Corazon Cafe & Catering	5.0	5.0
4	In and Out Express Food Market	5.0	5.0

NOTE :

No direct correlation: Higher rating do not guarantee a higher review count and vice versa, The review cannot reflect user engagement, but do not necessarily States the overall customer satisfaction or business performance, Successes in the restaurant business is not solely determined by rating or review counts.

Q. Do restaurants with higer engagement tends to have higher rating ?

In [23]:
```
pd.read_sql(""" select business_id, sum(length(date) - length(replace(date, ',', ''
```

Out[23]:

	business_id	checkin_count
0	3wo9jODQnuvBm8Gkem6qXg	3110.0
1	MkF4gosEaJqJ3tNk1BZiwg	3106.0
2	ctHjyadbDQAtUFfkcAFEHw	3104.0
3	h6IzeUVASeDtvKhd2PEsKA	3101.0
4	6dDC5PSmPEoJYuM8r8dN_A	3096.0
...	...	...
131718	i5tex_2_UNEsPUh6oaVREA	1.0
131719	i6-xjNGY_8Co3DwHDrBZ4w	1.0
131720	i69x-7o4wuLbWC3sf4ek7Q	1.0
131721	i6n5Cv7C2OOfrgGAj5weiw	1.0
131722	i6nsOTszsTWJw_vRrBkAJg	1.0

131723 rows × 2 columns

In [24]:
```
# tip_counts per business_id
pd.read_sql(""" select business_id, count(*) as tip_counts from  tip group by busin
```

Out[24]:

	business_id	tip_counts
0	FEXhWNCMkv22qG04E83Qjg	2571
1	-QI8Qi8XWH3D8y8ethnajA	1011
2	_ab50qdWOk0DdB6XOrBitw	932
3	ytynqOUb3hjKeJfRj5Tshw	827
4	Eb1XmmLWyt_way5NNZ7-Pw	826
...	...	...
106188	54hp8YAMnI0baeSwbnBxDA	1
106189	GP9X_N5vMHYTuwcEiz-Xnw	1
106190	qgZtdbGuASSA-Av7_-rgCw	1
106191	H5TSeRUNkoCRtL2RcB9WKQ	1
106192	uE_4H__4kj4Xjb115_LBvA	1

106193 rows × 2 columns

In [25]:
```
review_counts = pd.read_sql(F""" select total.avg_rating as rating,
AVG(total.review_count) as avg_review_count,
AVG(total.checkin_count) as avg_checkin_count,
```

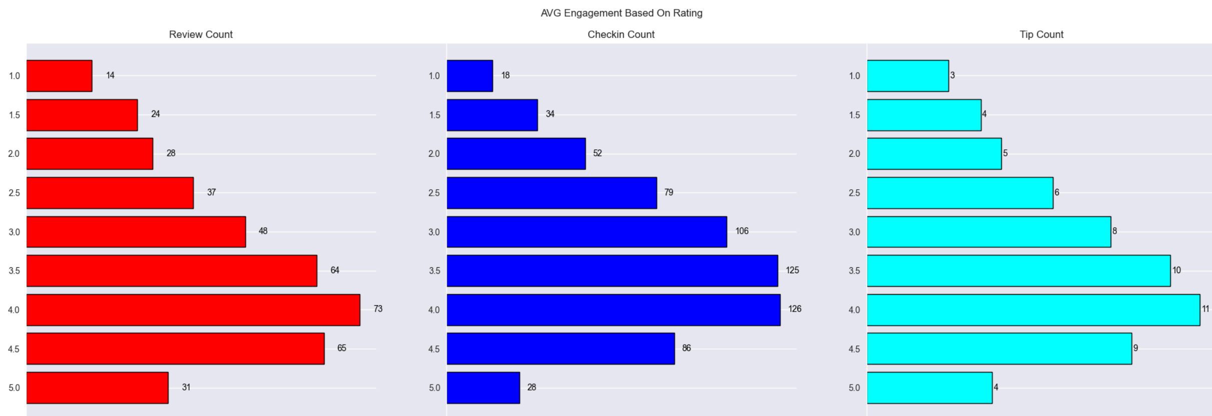
```
AVG(total.tip_count) as avg_tip_count
from
(select
    business.business_id,
    SUM(business.review_count) AS review_count,
    AVG(business.stars) AS avg_rating,
    SUM(LENGTH(checkin.date) - LENGTH(replace(checkin.date, ',', '')) +1) AS checki
    SUM(tip.tip_count) as tip_count
from
    business
left join
    checkin ON business.business_id = checkin.business_id
left join
    (select business_id, count(business_id) as tip_count from tip GROUP BY business
where business.business_id IN {tuple(business_Open_Restaurants['business_id'])}
GROUP BY
    business.business_id) as total
GROUP BY total.avg_rating
 """,engine)
```

In [26]:
```
review_counts=review_counts.sort_values('rating', ascending=False)
```

Ploting `Bar` Graphs :

In [27]:
```
# Creating Void and Axis for ploting the graphs
plt.figure(figsize=(25,8))
plt.title('AVG Engagement Based On Rating\n\n')
plt.xticks([])
plt.yticks([])

# Review Count Plot
plt.subplot(1,3,1)
plt.title('Review Count')
plt.barh(review_counts['rating'].astype('str'), review_counts['avg_review_count'],
plt.gca().spines['right'].set_visible(False)
for i, value in enumerate(review_counts['avg_review_count']):
    plt.text(value+3,i,str(round(value)), color='Black', va = 'center')
plt.xticks([])

# Checkin Count Plot
plt.subplot(1,3,2)
plt.title('Checkin Count')
plt.barh(review_counts['rating'].astype('str'), review_counts['avg_checkin_count'],
plt.gca().spines['right'].set_visible(False)
for i, value in enumerate(review_counts['avg_checkin_count']):
    plt.text(value+3,i,str(round(value)), color='Black', va = 'center')
plt.xticks([])

# Tip Count Plot
plt.subplot(1,3,3)
plt.title('Tip Count')
plt.barh(review_counts['rating'].astype('str'), review_counts['avg_tip_count'], edg
plt.gca().spines['right'].set_visible(False)
```

```
for i, value in enumerate(review_counts['avg_tip_count']):
    plt.text(value+0.05,i,str(round(value)), color='Black', va = 'center')
plt.xticks([])
plt.show()
```



NOTE :

Data show a general increase in average review check in and tip counts as rating improves from one to four stars, restaurants created four stars. Exhibit the highest engagement across reviews, check insurance and tips, suggesting a peak in user interaction, interestingly, engagement matrix. (Review, check in). Dip for restaurant rated 4.5 and significantly more at five stars, They dropped an engagement at 5 stars might suggest either a situation point where fewer customer feel compended or to add their reviews for a selective believer only a small satisfied audience. Frequents these establishment.

Q. Is there a correclation between the number of reviews, tip, and check - ins for a business ?

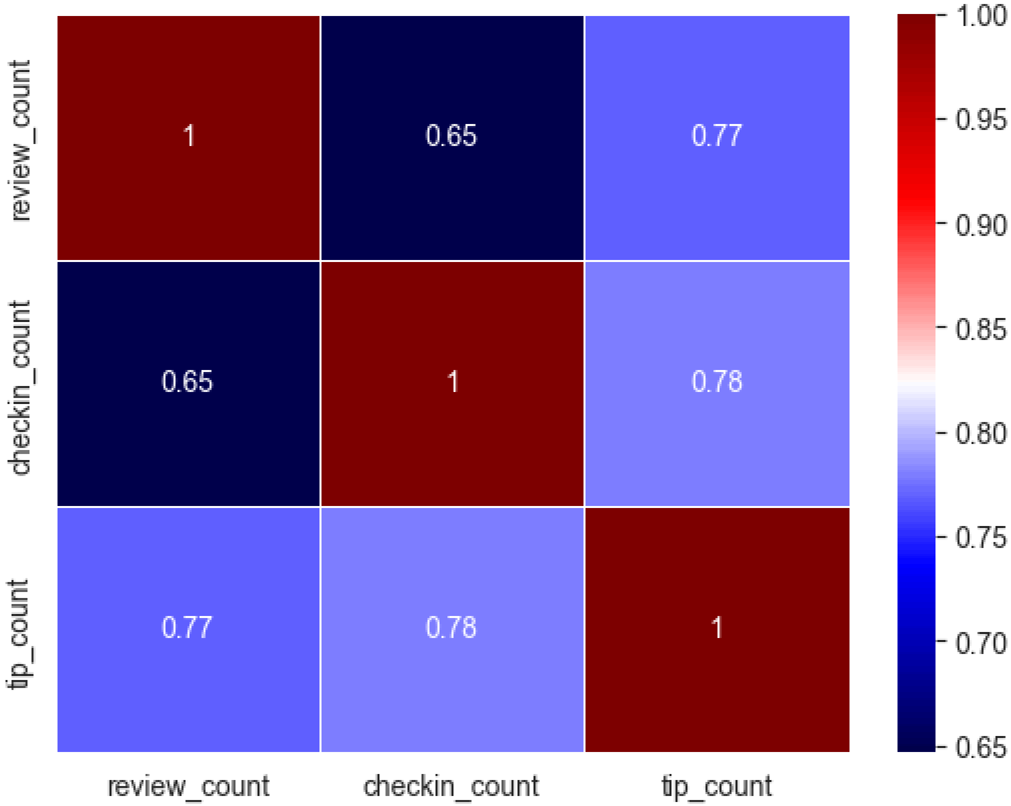
In [28]:
```
engage_df = pd.read_sql(F""" select business.business_id,
 sum(business.review_count) as review_count,
 AVG(business.stars) as avg_rating,
 SUM(LENGTH(checkin.date) - LENGTH(replace(checkin.date, ',', '')) +1) as checkin_c
 SUM(tip.tip_count) as tip_count
 from
 business
 left join
    checkin ON business.business_id = checkin.business_id
left join
    (select business_id, count(business_id) as tip_count from tip GROUP BY business
where business.business_id IN {tuple(business_Open_Restaurants['business_id'])}
GROUP BY
    business.business_id
```

```
""",engine).dropna()
```

In [29]:
```
engage_df_corr = engage_df[['review_count','checkin_count','tip_count']].corr()
```

In [30]:
```
# Ploting the HeatMap
sns.heatmap(engage_df_corr,cmap='seismic', annot=True, linewidths=0.5,linecolor='wh
```

Out[30]: <Axes: >



In [31]:
```
engage_dff = pd.read_sql(F"""
    SELECT
        business.business_id,
        SUM(business.review_count) AS review_count,
        AVG(business.stars) AS avg_rating,
        SUM(LENGTH(checkin.date) - LENGTH(REPLACE(checkin.date, ',', '')) + 1) AS c
        SUM(tip.tip_count) AS tip_count,
        CASE
            WHEN business.stars >= 3.5 THEN 'High-Rated'
            ELSE 'Low-Rated'
        END AS category
    FROM
        business
    LEFT JOIN
        checkin ON business.business_id = checkin.business_id
    LEFT JOIN
        (SELECT business_id, COUNT(business_id) AS tip_count
         FROM tip
         GROUP BY business_id) AS tip ON business.business_id = tip.business_id
```

```
    WHERE
        business.business_id IN {tuple(business_Open_Restaurants['business_id'])}
    GROUP BY
        business.business_id, business.stars
""", engine).dropna()
```

In [32]:
```
engage_dff.groupby('category')[['review_count','tip_count','checkin_count']].mean()
```

Out[32]:

category	review_count	tip_count	checkin_count
High-Rated	72.274446	10.148378	121.375576
Low-Rated	42.123420	6.541689	88.880828

NOTE :

The data set shows a strong positive correlation among review counts, checking counts and tip counts, These correlations suggest that user engagement across different platforms, such as reviews, tips and check insurance is interlinked, Higher activity in one area tends to be associated with higher activity others. Businesses should focus on strategies that boost all types of user investment as increases in one type of engagement are likely to drive increase in others. And hence, in overall visibility and interaction with customers.

In [33]:
```
# Function to calculate the sucess score based on the avg rating and total review c
def calculate_sucess_metric(df):
    sucess_score=[]
    for idx, row in df.iterrows():
        score = row['avg_rating'] * np.log(row['review_count'] + 1 )
        sucess_score.append(score)
    return sucess_score
```

MAP Plot :

Q. How do the sucess metrics (review_count or avg_rating) of restaurant vary across different states and cities?

In [34]:
```
city_df = pd.read_sql(F"""
SELECT
    city,
    state,
```

```
    latitude,
    longitude,
    AVG(stars) AS avg_rating,
    SUM(review_count) AS review_count,
    COUNT(*) AS restaurant_count
FROM business
WHERE business_id IN {tuple(business_Open_Restaurants['business_id'])}
GROUP BY state, city, latitude, longitude
ORDER BY review_count DESC
LIMIT 10
""", engine)
```

In [35]:
```
city_df = city_df.reset_index()
```

In [36]:
```
city_df = city_df.drop('index',axis = 1)
```

In [37]:
```
city_df['success_score'] = calculate_sucess_metric(city_df)
```

In [38]:
```
city_df
```

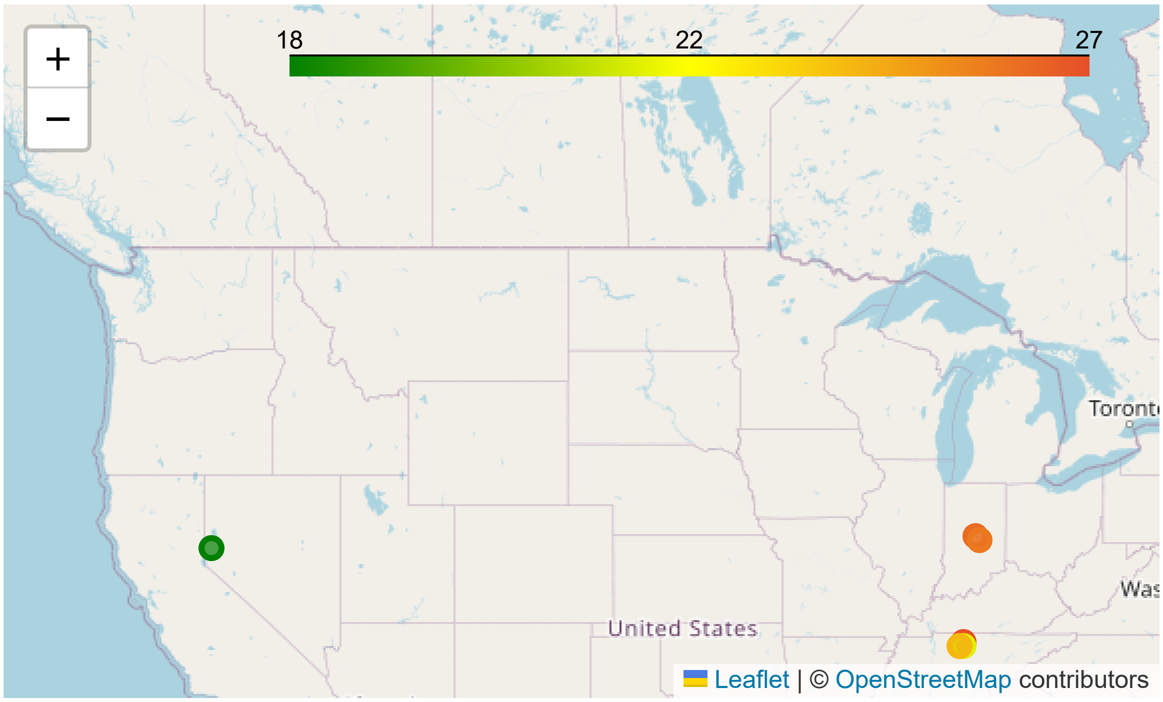
Out[38]:

	city	state	latitude	longitude	avg_rating	review_count	restaurant_count
0	Philadelphia	PA	39.953159	-75.159098	4.000000	541.0	6
1	Nashville	TN	36.163685	-86.782598	4.000000	517.0	3
2	Carmel	IN	39.978599	-86.128981	4.166667	503.0	3
3	Sparks	NV	39.541452	-119.716242	3.000000	452.0	4
4	Hendersonville	TN	36.302820	-86.619056	4.375000	431.0	4
5	Nashville	TN	36.170064	-86.665561	3.625000	424.0	4
6	Nashville	TN	36.138603	-86.800358	4.000000	417.0	2
7	Philadelphia	PA	39.958359	-75.195393	4.250000	416.0	6
8	Indianapolis	IN	39.858230	-85.978565	4.250000	411.0	2
9	Philadelphia	PA	39.949756	-75.148062	3.583333	379.0	12

In [39]:
```
# Create a base Map
m = folium.Map(location = [city_df['latitude'].mean(), city_df['longitude'].mean()]

# Define a color scale
color_scale = folium.LinearColormap(colors=['green','yellow','#E54F29'], vmin = cit

# Add markers to the map

for idx, row in city_df.iterrows():
    folium.CircleMarker(
        location = [row['latitude'], row['longitude']],
        radius = 5,
        color = color_scale(row['success_score']),
```

```
        fill = True,
        fill_color = color_scale(row['success_score']),
        fill_opacity = 0.7,
        popup=F"Success Score: {row ['success_score']}"
    ).add_to(m)

# Add color scale to the map

m.add_child(color_scale)
```

Out[39]:



NOTE :

Philadelphia emerges as the top city with the highest success code, indicating a combination of high ratings and active user engagements. following Philadelphia, Tampa, Indianapolis and tucson rank among the top cities with significant success score, suggesting thriving restaurant success in these areas. The success matrix vary significantly across different state and cities, highlighting regional differences in dining Preferences, culinary sceneries and customer engagement levels, identifying city with the highest success score present opportunities for Restaurant chains to Expend or invest further while areas with low score may require targeted efforts to improve rating and increase user engagement.

Time :

Q. Are there any patterns in the user engagement over time for sucessfull business compared to less sucessfull ones, Are there any seasonal trends in the user engagement for restaurant ?

In [40]:
```
high_rated_engagement = pd.read_sql_query(f"""
SELECT review.month_year, review.review_count, tip.tip_count FROM
(SELECT DATE_FORMAT(date, '%m-%Y') AS month_year, COUNT(*) AS review_count
FROM review
WHERE business_id IN {tuple(business_Open_Restaurants['business_id'])} AND stars >=
GROUP BY month_year
ORDER BY month_year) AS review
JOIN
(SELECT AVG(b.stars) AS avg_stars, DATE_FORMAT(tip.date, '%m-%Y') AS month_year, CO
FROM tip
JOIN business AS b
ON tip.business_id = b.business_id
WHERE tip.business_id IN {tuple(business_Open_Restaurants['business_id'])} AND b.st
GROUP BY month_year
ORDER BY month_year) AS tip
ON review.month_year = tip.month_year
;""", engine)

low_rated_engagement = pd.read_sql_query(f"""
SELECT review.month_year, review.review_count, tip.tip_count FROM
(SELECT DATE_FORMAT(date, '%m-%Y') AS month_year, COUNT(*) AS review_count
FROM review
WHERE business_id IN {tuple(business_Open_Restaurants['business_id'])} AND stars <
GROUP BY month_year
ORDER BY month_year) AS review
JOIN
(SELECT AVG(b.stars) AS avg_stars, DATE_FORMAT(tip.date, '%m-%Y') AS month_year, CO
FROM tip
JOIN business AS b
ON tip.business_id = b.business_id
WHERE tip.business_id IN {tuple(business_Open_Restaurants['business_id'])} AND b.st
GROUP BY month_year
ORDER BY month_year) AS tip
ON review.month_year = tip.month_year
;""", engine)
```

In [41]:
```
high_rated_engagement
```

Out[41]:

	month_year	review_count	tip_count
0	01-2010	1218	79
1	01-2011	2171	621
2	01-2012	3086	1321
3	01-2013	3801	1230
4	01-2014	4973	1357
...	...	...	...
149	12-2017	10161	1477
150	12-2018	12870	1163
151	12-2019	13756	1161
152	12-2020	11294	937
153	12-2021	12652	652

154 rows × 3 columns

In [42]:
```
low_rated_engagement
```

Out[42]:

	month_year	review_count	tip_count
0	01-2010	613	25
1	01-2011	1103	297
2	01-2012	1748	538
3	01-2013	2196	548
4	01-2014	2769	607
...	...	...	...
149	12-2017	5970	441
150	12-2018	7574	338
151	12-2019	7591	275
152	12-2020	5014	148
153	12-2021	6937	122

154 rows × 3 columns

In [43]:
```
time_rating = pd.read_sql(f"""
SELECT DATE_FORMAT(date, '%m-%Y') AS month_year, AVG(stars) AS avg_rating
FROM review
WHERE business_id IN {tuple(business_Open_Restaurants['business_id'])}
```

```
GROUP BY month_year
ORDER BY month_year;
""", engine)
```

In [44]:
```
time_rating
```

Out[44]:

	month_year	avg_rating
0	01-2006	4.000000
1	01-2007	3.897436
2	01-2008	3.603960
3	01-2009	3.690661
4	01-2010	3.724194
...	...	...
198	12-2017	3.613415
199	12-2018	3.608687
200	12-2019	3.665246
201	12-2020	3.833701
202	12-2021	3.672673

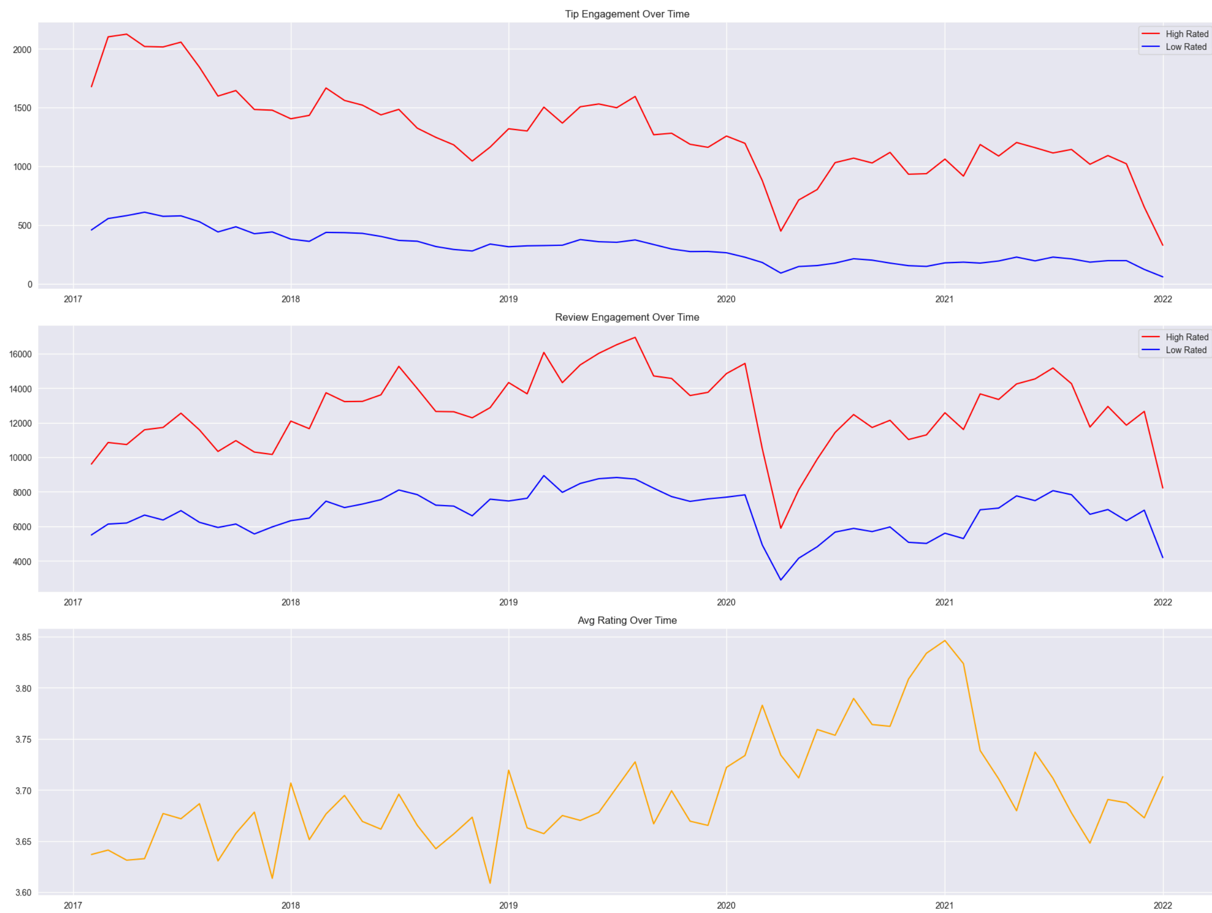
203 rows × 2 columns

In [45]:
```
# time_rating
time_rating['month_year'] = pd.to_datetime(time_rating['month_year'])
time_rating.sort_values('month_year',inplace= True)
time_rating = time_rating[time_rating['month_year']>'2017']

# high_rated_engagement
high_rated_engagement['month_year'] = pd.to_datetime(high_rated_engagement['month_y
high_rated_engagement.sort_values('month_year',inplace=True)
high_rated_engagement = high_rated_engagement[high_rated_engagement['month_year']>'

# low_rated_engagement
low_rated_engagement['month_year'] =  pd.to_datetime(low_rated_engagement['month_ye
low_rated_engagement.sort_values('month_year',inplace=True)
low_rated_engagement = low_rated_engagement[low_rated_engagement['month_year']>'201
```

In [46]:
```
# Creating a col called avg_rating in high_rated_engagement
high_rated_engagement['avg_rating'] = time_rating['avg_rating'].values
```

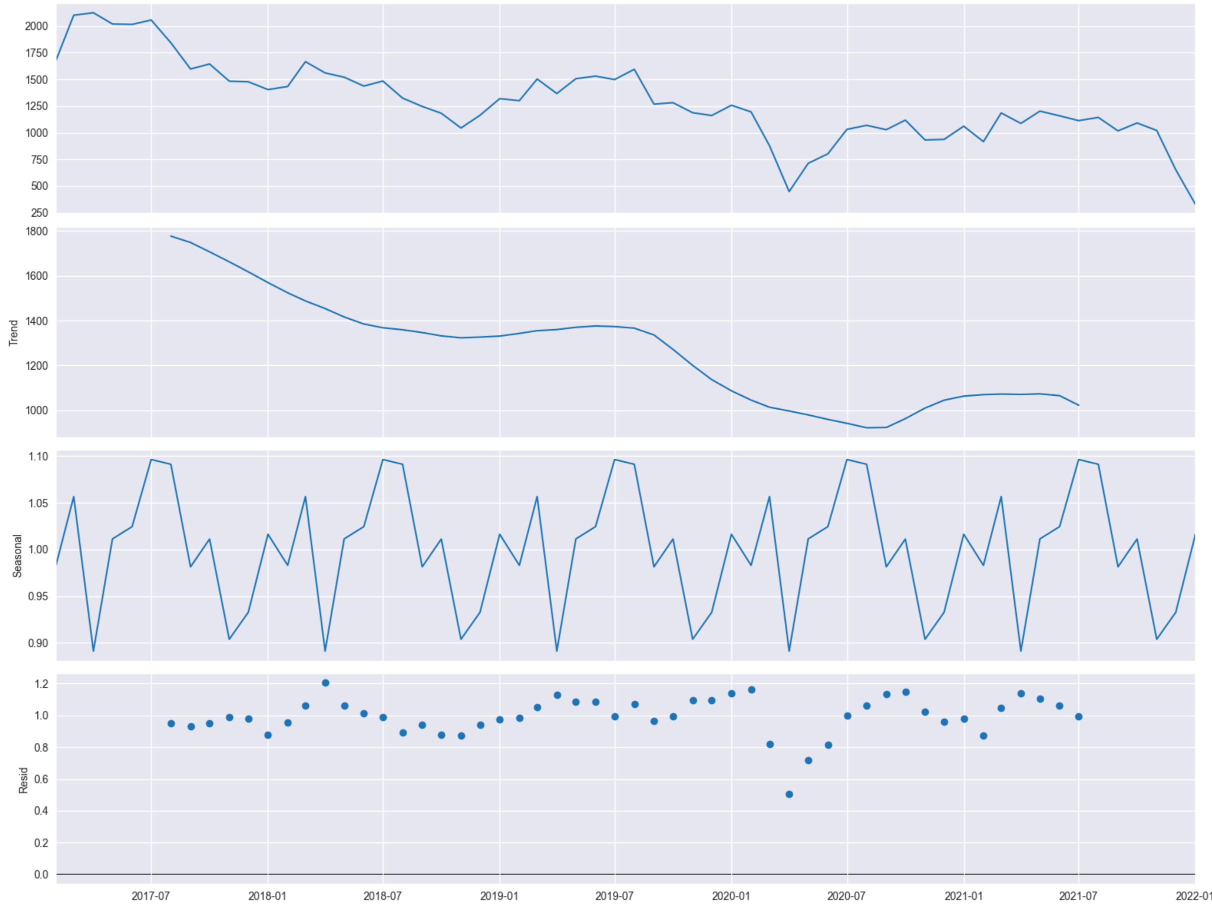
In [47]:
```
# ploting the time trends
plt.figure(figsize=(20, 15))
plt.subplot(3,1,1)
plt.title('Tip Engagement Over Time')
plt.plot(high_rated_engagement['month_year'], high_rated_engagement['tip_count'], l
plt.plot(low_rated_engagement['month_year'], low_rated_engagement['tip_count'], lab
```

```
plt.legend()

plt.subplot(3,1,2)
plt.title('Review Engagement Over Time')
plt.plot(high_rated_engagement['month_year'], high_rated_engagement['review_count']
plt.plot(low_rated_engagement['month_year'], low_rated_engagement['review_count'],
plt.legend()

plt.subplot(3,1,3)
plt.title('Avg Rating Over Time')
plt.plot(time_rating['month_year'], time_rating['avg_rating'], color = 'ORANGE')
plt.tight_layout()
plt.show()
```

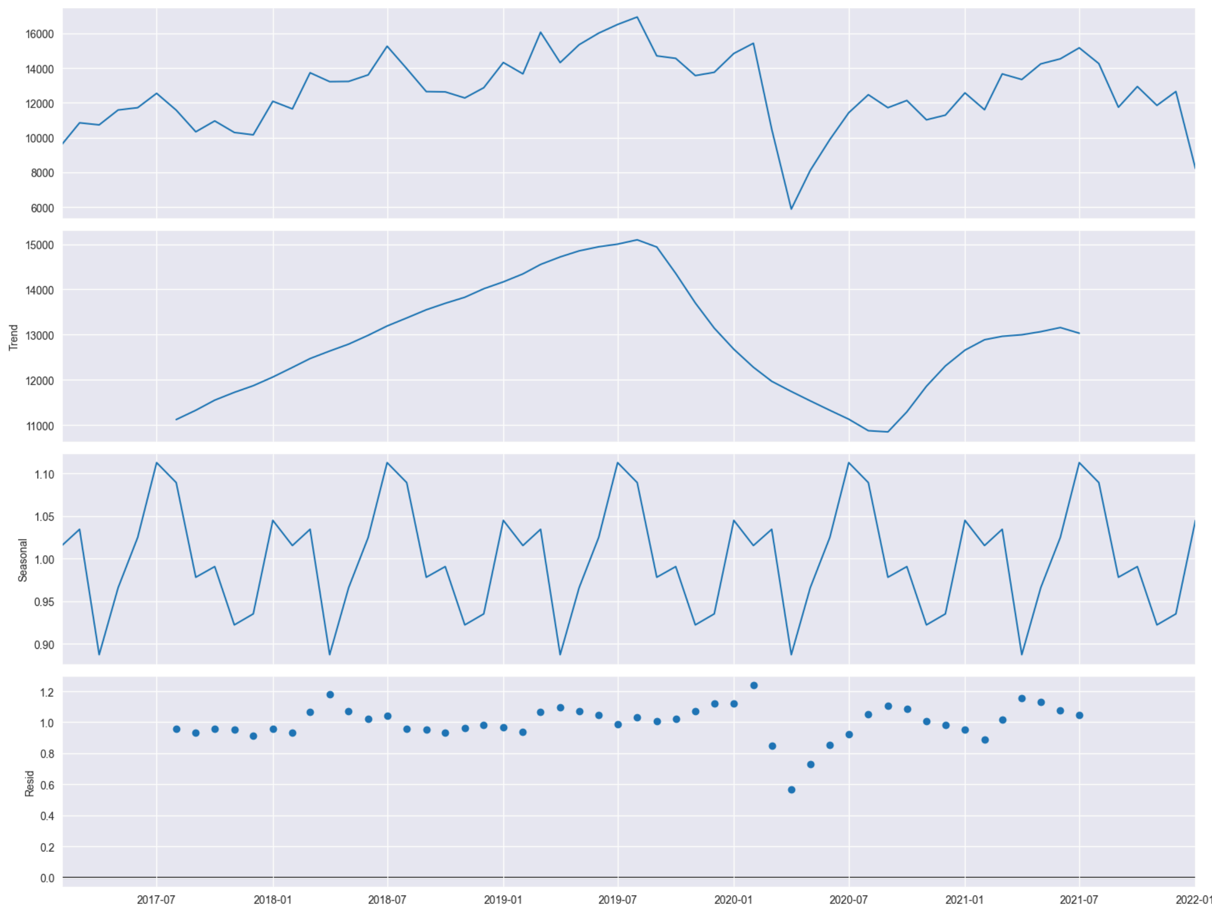


In [48]:
```
tip_high_rated = high_rated_engagement[['month_year','tip_count']].set_index('month
review_high_rated = high_rated_engagement[['month_year', 'review_count']].set_index
rating_df = time_rating[['month_year', 'avg_rating']].set_index('month_year')
```

In [49]:
```
# seasonal_decompose of tip_high_rated
from statsmodels.tsa.seasonal import seasonal_decompose
multiplication_decompose = seasonal_decompose(tip_high_rated, model = 'multiplicati
plt.rcParams.update({'figure.figsize':(16,12)})
multiplication_decompose.plot()
plt.show()
```



In [50]:
```
# seasonal_decompose of  review_high_rated
from statsmodels.tsa.seasonal import seasonal_decompose
multiplication_decompose = seasonal_decompose(review_high_rated, model = 'multiplic
plt.rcParams.update({'figure.figsize':(16,12)})
multiplication_decompose.plot()
plt.show()
```



NOTE :

Successful businesses, particularly those with higher ratings above 3.5 exhibit consistent and possibly increase user engagement overtime, High rated restaurants maintain a steady or blowing level of user management over time, reflecting ongoing customer interest and satisfaction, Keep count is showing a downward trend, whereas review count is showing an up on trend with time, Years starting at year ending from around November and March is highly engaging in seasonal.

Sentiment Analysis Using NLTK :

Q. Retrive Top Five Restaurant with High Success Score And Sentiments Included (Positive, Negetive & Neutral)

In [51]:
```
# import nltk
# from nltk.sentiment import SentimentIntensityAnalyzer
#
# # Create a sentiment col
# sentiment = review['text']
#
# # Perform sentiment analysis using SentimentIntensityAnalyzer:
#
# sia = SentimentIntensityAnalyzer()
#
# # Define a function to get sentiment score
# def get_sentiment(comment):
#     score = sia.polarity_scores(comment)
#     return score['compound']  # compound score for overall sentiment
#
# # Apply sentiment analysis to the 'comments' column
# sentiment_score = sentiment.apply(get_sentiment)
# sentiments = sentiment_score.apply(lambda x: 'Positive' if x > 0 else ('Negative'
#
# sentiments
```

In [52]:
```
sentim = pd.read_csv('D:\\VSCODE\\SQL_PROJECT\\Senti.csv')
```

In [53]:
```
senti = pd.concat([review[['business_id','text']], sentim], axis = 1)
```

In [54]:
```
# Sentiments
senti
```

Out[54]:

	business_id	text	Unnamed: 0.1	Unnamed: 0	text
0	XQfwVwDr-v0ZS3_CbbE5Xw	If you decide to eat here, just be aware it is...	0	0	Positive
1	7ATYjTIgM3jUlt4UM3IypQ	I've taken a lot of spin classes over the year...	1	1	Positive
2	YjUWPpI6HXG530lwP-fb2A	Family diner. Had the buffet. Eclectic assortm...	2	2	Positive
3	kxX2SOes4o-D3ZQBkiMRfA	Wow! Yummy, different, delicious. Our favo...	3	3	Positive
4	e4Vwtrqf-wpJfwesgvdgxQ	Cute interior and owner (?) gave us tour of up...	4	4	Positive
...	...	...	...	...	...
6990275	jals67o91gcrD4DC81Vk6w	Latest addition to services from ICCU is Apple...	6990275	6990275	Negative
6990276	2vLksaMmSEcGbjI5gywpZA	This spot offers a great, affordable east week...	6990276	6990276	Positive
6990277	R1khUUxidqfaJmcpmGd4aw	This Home Depot won me over when I needed to g...	6990277	6990277	Positive
6990278	Rr9kKArrMhSLVE9a53q-aA	For when I'm feeling like ignoring my calorie-...	6990278	6990278	Positive
6990279	VAeEXLbEcI9Emt9KGYq9aA	Located in the 'Walking District' in Nashville...	6990279	6990279	Positive

6990280 rows × 5 columns

In [55]:

```
business
```

Out[55]:

	business_id	name	address	city	state	postal_code
0	Pns2l4eNsfO8kk83dixA6A	Abby Rappoport, LAC, CMQ	1616 Chapala St, Ste 2	Santa Barbara	CA	93101
1	mpf3x-BjTdTEA3yCZrAYPw	The UPS Store	87 Grasso Plaza Shopping Center	Affton	MO	63123
2	tUFrWirKiKi_TAnsVWINQQ	Target	5255 E Broadway Blvd	Tucson	AZ	85711
3	MTSW4McQd7CbVtyjqoe9mw	St Honore Pastries	935 Race St	Philadelphia	PA	19107
4	mWMc6_wTdE0EUBKIGXDVfA	Perkiomen Valley Brewery	101 Walnut St	Green Lane	PA	18054
...	...	...	...	...	...	...
150341	IUQopTMmYQG-qRtBk-8QnA	Binh's Nails	3388 Gateway Blvd	Edmonton	AB	T6J 5H2
150342	c8GjPIOTGVmIemT7j5_SyQ	Wild Birds Unlimited	2813 Bransford Ave	Nashville	TN	37204
150343	_QAMST-NrQobXduilWEqSw	Claire's Boutique	6020 E 82nd St, Ste 46	Indianapolis	IN	46250
150344	mtGm22y5c2UHNXDFAjaPNw	Cyclery & Fitness Center	2472 Troy Rd	Edwardsville	IL	62025
150345	jV_XOycEzSlTx-65W906pg	Sic Ink	238 Apollo Beach Blvd	Apollo beach	FL	33572

150346 rows × 12 columns

In [56]:
```
business_data = business[['business_id','name']]
senti = senti.iloc[:,[0,4]]
```

In [57]:
```
senti
```

Out[57]:

	business_id	text
0	XQfwVwDr-v0ZS3_CbbE5Xw	Positive
1	7ATYjTIgM3jUlt4UM3IypQ	Positive
2	YjUWPpI6HXG530lwP-fb2A	Positive
3	kxX2SOes4o-D3ZQBkiMRfA	Positive
4	e4Vwtrqf-wpJfwesgvdgxQ	Positive
...	...	...
6990275	jals67o91gcrD4DC81Vk6w	Negative
6990276	2vLksaMmSEcGbjI5gywpZA	Positive
6990277	R1khUUxidqfaJmcpmGd4aw	Positive
6990278	Rr9kKArrMhSLVE9a53q-aA	Positive
6990279	VAeEXLbEcI9Emt9KGYq9aA	Positive

6990280 rows × 2 columns

In [58]:
```
real_sentiments = pd.merge(business_data,senti, how='inner', on= ['business_id'])
```

In [59]:
```
real_sentiments
```

Out[59]:

	business_id	name	text
0	Pns2l4eNsfO8kk83dixA6A	Abby Rappoport, LAC, CMQ	Positive
1	Pns2l4eNsfO8kk83dixA6A	Abby Rappoport, LAC, CMQ	Positive
2	Pns2l4eNsfO8kk83dixA6A	Abby Rappoport, LAC, CMQ	Positive
3	Pns2l4eNsfO8kk83dixA6A	Abby Rappoport, LAC, CMQ	Positive
4	Pns2l4eNsfO8kk83dixA6A	Abby Rappoport, LAC, CMQ	Positive
...	...	...	...
6990275	jV_XOycEzSITx-65W906pg	Sic Ink	Positive
6990276	jV_XOycEzSITx-65W906pg	Sic Ink	Positive
6990277	jV_XOycEzSITx-65W906pg	Sic Ink	Positive
6990278	jV_XOycEzSITx-65W906pg	Sic Ink	Positive
6990279	jV_XOycEzSITx-65W906pg	Sic Ink	Negative

6990280 rows × 3 columns

In [60]:
```
real_sentiments = real_sentiments.groupby(['business_id','name','text']).value_coun
```

In [61]:
```
real_sentiments_df = real_sentiments.reset_index()
```

In [62]:
```
#Success_Business
success_business = pd.read_sql(F"""
SELECT
    business_id, name,
    AVG(stars) AS avg_rating,
    SUM(review_count) AS review_count,
    COUNT(*) AS restaurant_count
FROM business
WHERE business_id IN {tuple(business_Open_Restaurants['business_id'])}
GROUP BY business_id, name
ORDER BY review_count DESC
""", engine)
```

In [63]:
```
success_business
```

Out[63]:

	business_id	name	avg_rating	review_count	restaurant_count
0	wPQWqLxY6t3-yRBNPPAmkQ	Shallos Antique Restaurant & Brewhouse	4.0	248.0	1
1	y8gjlpJA89qDRCLC0JQaew	Giuseppe & Sons	4.0	248.0	1
2	gkZ6iiEfnO7I2UzOHbkzrA	Ulysses American Gastro Pub	3.5	248.0	1
3	98WBvrn7wzu_93zc7fRfzQ	Vero Amore - Dove	4.0	248.0	1
4	30OhTA38fp8xuqW4O2D6Eg	Homegrown Taproom & Kitchen	4.0	248.0	1
...	...	...	...	...	...
31532	3xoCPDgE5dEretLD4yFpRw	The Juice Pod	2.5	5.0	1
31533	nPruiFveAtUcGrUbFBeQuQ	Bark Busters Dog Home Training	4.0	5.0	1
31534	Hk_EjFDeK5u7rlYEUx6a_g	Jaggie's Restaurant	3.0	5.0	1
31535	qv6TSnK4iXZAXxG13mjv-w	Angelina's Panini Bar	2.0	5.0	1
31536	rwZ-1fH9vdh1KRAowovXOQ	Subway	3.5	5.0	1

31537 rows × 5 columns

In [64]:

```
success_business['Success_score'] = calculate_sucess_metric(success_business)
```

In [65]:

```
success_business
```

Out[65]:

	business_id	name	avg_rating	review_count	restaurant_count
0	wPQWqLxY6t3-yRBNPPAmkQ	Shallos Antique Restaurant & Brewhouse	4.0	248.0	1
1	y8gjlpJA89qDRCLC0JQaew	Giuseppe & Sons	4.0	248.0	1
2	gkZ6iiEfnO7I2UzOHbkzrA	Ulysses American Gastro Pub	3.5	248.0	1
3	98WBvrn7wzu_93zc7fRfzQ	Vero Amore - Dove	4.0	248.0	1
4	30OhTA38fp8xuqW4O2D6Eg	Homegrown Taproom & Kitchen	4.0	248.0	1
...	...	...	...	...	...
31532	3xoCPDgE5dEretLD4yFpRw	The Juice Pod	2.5	5.0	1
31533	nPruiFveAtUcGrUbFBeQuQ	Bark Busters Dog Home Training	4.0	5.0	1
31534	Hk_EjFDeK5u7rlYEUx6a_g	Jaggie's Restaurant	3.0	5.0	1
31535	qv6TSnK4iXZAXxG13mjv-w	Angelina's Panini Bar	2.0	5.0	1
31536	rwZ-1fH9vdh1KRAowovXOQ	Subway	3.5	5.0	1

31537 rows × 6 columns

In [66]:
```
success_business = success_business.sort_values('Success_score',ascending=False)
```

In [67]:
```
success_business
```

Out[67]:

	business_id	name	avg_rating	review_count	restaurant_count
204	bjQrmBSu1A7f5vprEikOKA	Healthy N Fresh Cafe	5.0	238.0	1
399	S5LnH1njwFBlq77tIkjI1g	Yolk White & Associates	5.0	229.0	1
508	emrUsUZvqCkytUu4i3kjLw	Sundae's Ice Cream & Coffee	5.0	225.0	1
557	0I9XZD7JTqY9iTF8nXRnXw	Ali'i Poke Indy	5.0	223.0	1
611	jh8j-DWqgWkbRe_a2XtKFQ	Barrio Bread	5.0	221.0	1
...	...	...	...	...	...
31139	Q4OTSH9DaoeqtYcg6qYtZg	Subway	1.0	5.0	1
30725	F6zk6xPTLQZFdA0hu6nLgA	Hungry Howie's Pizza & Subs	1.0	5.0	1
31157	BJ0Z74sTz9sxRr1R533Inw	Best Rate Home Services	1.0	5.0	1
30715	ogoe6WcXJnW96rvc3NyMfw	Burger King	1.0	5.0	1
31362	2HLZfbL-6Icr9jhriW6GeA	Subway Restaurants	1.0	5.0	1

31537 rows × 6 columns

In [68]:
```
real_sentiments_df
```

Out[68]:

	business_id	name	text	count
0	---kPU91CF4Lq2-WlRu9Lw	Frankie's Raw Bar	Negative	1
1	---kPU91CF4Lq2-WlRu9Lw	Frankie's Raw Bar	Positive	23
2	--0iUa4sNDFiZFrAdIWhZQ	Pupuseria Y Restaurant Melba	Negative	3
3	--0iUa4sNDFiZFrAdIWhZQ	Pupuseria Y Restaurant Melba	Positive	11
4	--30_8IhuyMHbSOcNWd6DQ	Action Karate	Negative	4
...	...	...	...	...
351933	zzu6_r3DxBJuXcjnOYVdTw	Cafe Diblasi	Positive	7
351934	zzw66H6hVjXQEt0Js3Mo4A	Sullivan Farms Christmas Trees	Negative	1
351935	zzw66H6hVjXQEt0Js3Mo4A	Sullivan Farms Christmas Trees	Positive	4
351936	zzyx5x0Z7xXWWvWnZFuxlQ	Walnut Street Pizza	Negative	3
351937	zzyx5x0Z7xXWWvWnZFuxlQ	Walnut Street Pizza	Positive	5

351938 rows × 4 columns

In [69]:

```
main_sentiment_df = pd.merge(real_sentiments_df,success_business, how='inner',on=['
```

In [70]:

```
main_sentiment_df.sort_values('count',ascending=False)
main_sentiment_df
```

Out[70]:

	business_id	name	text	count	avg_rating	review_count	r
50059	bjQrmBSu1A7f5vprEikOKA	Healthy N Fresh Cafe	Positive	234	5.0	238.0	
50058	bjQrmBSu1A7f5vprEikOKA	Healthy N Fresh Cafe	Neutral	1	5.0	238.0	
50057	bjQrmBSu1A7f5vprEikOKA	Healthy N Fresh Cafe	Negative	5	5.0	238.0	
36579	S5LnH1njwFBlq77tlkjl1g	Yolk White & Associates	Positive	219	5.0	229.0	
36577	S5LnH1njwFBlq77tlkjl1g	Yolk White & Associates	Negative	7	5.0	229.0	
...	...	...	...	...	...	...	
31365	NxB8M1wnJQ5xoXDiUgqmIg	Burger King	Neutral	1	1.0	5.0	
34051	Q4OTSH9DaoeqtYcg6qYtZg	Subway	Negative	4	1.0	5.0	
19745	ESUwN81iNYvm0yqYCxSivg	Jack in the Box	Negative	2	1.0	5.0	
19746	ESUwN81iNYvm0yqYCxSivg	Jack in the Box	Positive	3	1.0	5.0	
3476	1dpjXnlEKc-lhgku6CtY5w	Little Caesars Pizza Barataria Blvd	Positive	3	1.0	5.0	

80616 rows × 8 columns

In [71]:

```python
# Top Five Highest Sucess Score Restaurant
main_sentiment_df.head(14)[['name']].drop_duplicates('name').sort_values('name').va
```

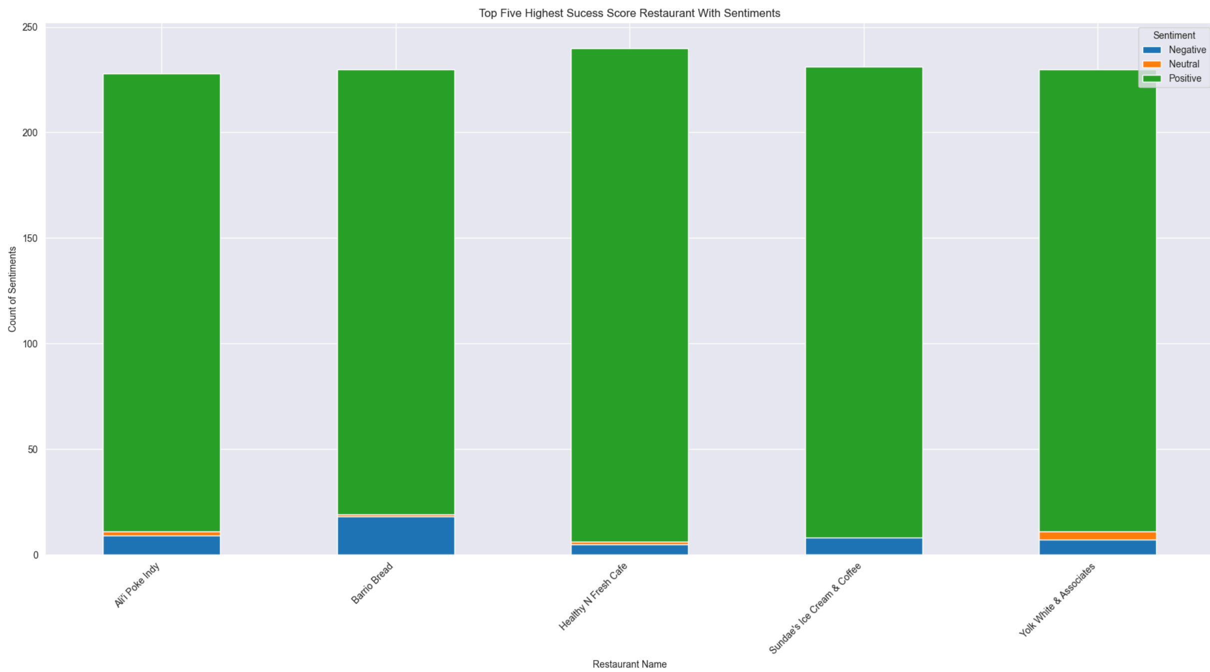
Out[71]:

```
array([["Ali'i Poke Indy"],
       ['Barrio Bread'],
       ['Healthy N Fresh Cafe'],
       ["Sundae's Ice Cream & Coffee"],
       ['Yolk White & Associates']], dtype=object)
```

In [72]:

```python
# Pivot the DataFrame to prepare for stacked bar plot
pivot_df = main_sentiment_df.head(14).pivot_table(index='name', columns='text', val

# Plot Stacked Bar
```

```python
pivot_df.plot(kind='bar', stacked=True, figsize=(18, 10))
plt.title('Top Five Highest Sucess Score Restaurant With Sentiments')
plt.xlabel('Restaurant Name')
plt.ylabel('Count of Sentiments')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Sentiment')
plt.tight_layout()
plt.show()
```



NOTE :

There are top five highest success score restaurants, including not only the higher rating and higher review counts, but also having a higher positive sentiments In compare to the other negative and neutral sentiments. These five restaurant may be the best restaurants in overall comparison.

Distribution Of Data Based On Elite And Non ELite :

Q. Is there any difference in engagement of elite users and non-elite users?

In [73]:
```python
elite_df = pd.read_sql("""
    SELECT
        elite,
        COUNT(*) AS num_users,
```

```
        SUM(review_count) AS total_review_count
    FROM (
        SELECT
            CASE
                WHEN elite = 'None' OR elite = '' OR elite IS NUll THEN 'Not Elite'
                ELSE 'Elite'
            END AS elite,
            user.review_count
        FROM
            user
    ) AS user_elite
    GROUP BY
        elite;
""", engine)
```

In [74]:

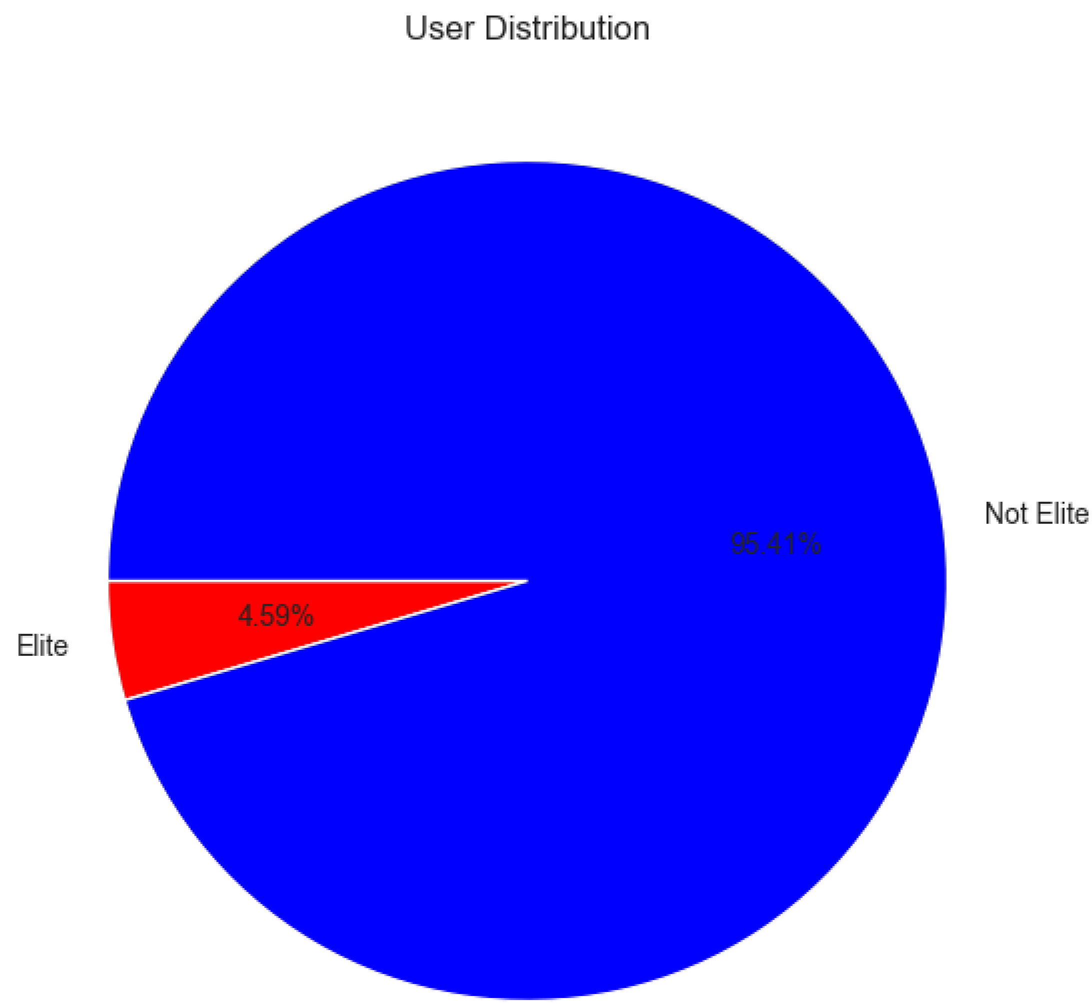
```
# Elite
elite_df
```

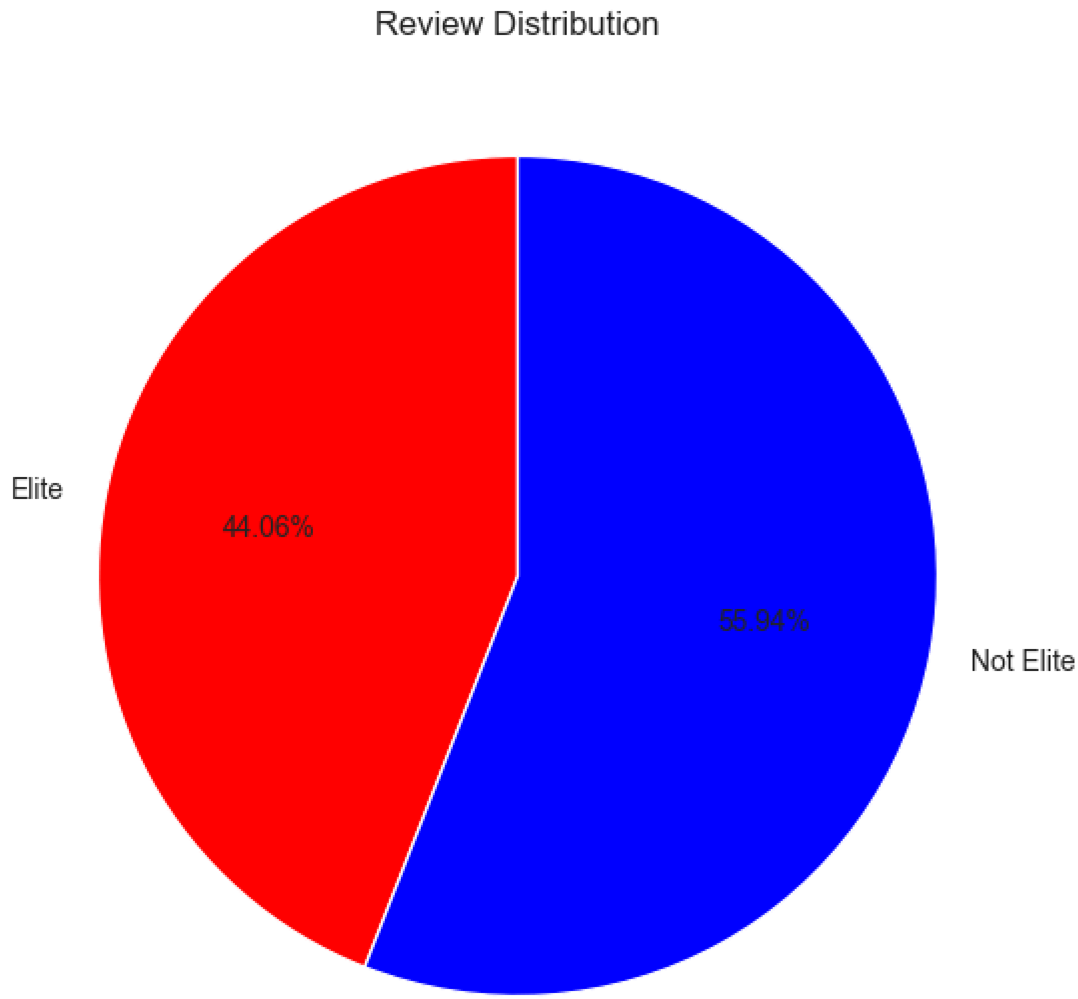
Out[74]:

	elite	num_users	total_review_count
0	Elite	90969	20450520.0
1	Not Elite	1892854	25963868.0

In [75]:

```
# Pie Plot of User Distribution
plt.figure(figsize=(10,15))
plt.subplot(2,1,1)
plt.title('User Distribution')
plt.pie(elite_df['num_users'],labels = elite_df['elite'], autopct = '%0.2f%%', star

# Pie Plot of Review Distribution
plt.figure(figsize=(10,15))
plt.subplot(2,1,2)
plt.title('Review Distribution')
plt.pie(elite_df['total_review_count'],labels = elite_df['elite'], autopct = '%0.2f
plt.show()
```


User Distribution
Not Elite
95.41%
4.59%
Elite



NOTE :

`Elite` users are individual who have been recognized and awarded the `Elite` status by the Yelp and their active and high quality contribution to the platform, such as frequent and detailed review photos and check insurance. Among the others criteria. `Elite` users despite being significant fewer in numbers, contributed a substantial proportion of the total account compared to the non-elite users. `Elite` users often provide detailed and insightful reviews, which can influence other users perceptions and decision regarding a business. review from `Elite` users may receive more attention and visibility on the real platform due to their status potentially leading to the higher exposure through business.Establishing a positive relationship with `Elite` users can lead a repeat visit and loyalty, as they have more likely to continue supporting widgets they have had good experience with.

Time Based Analysis :

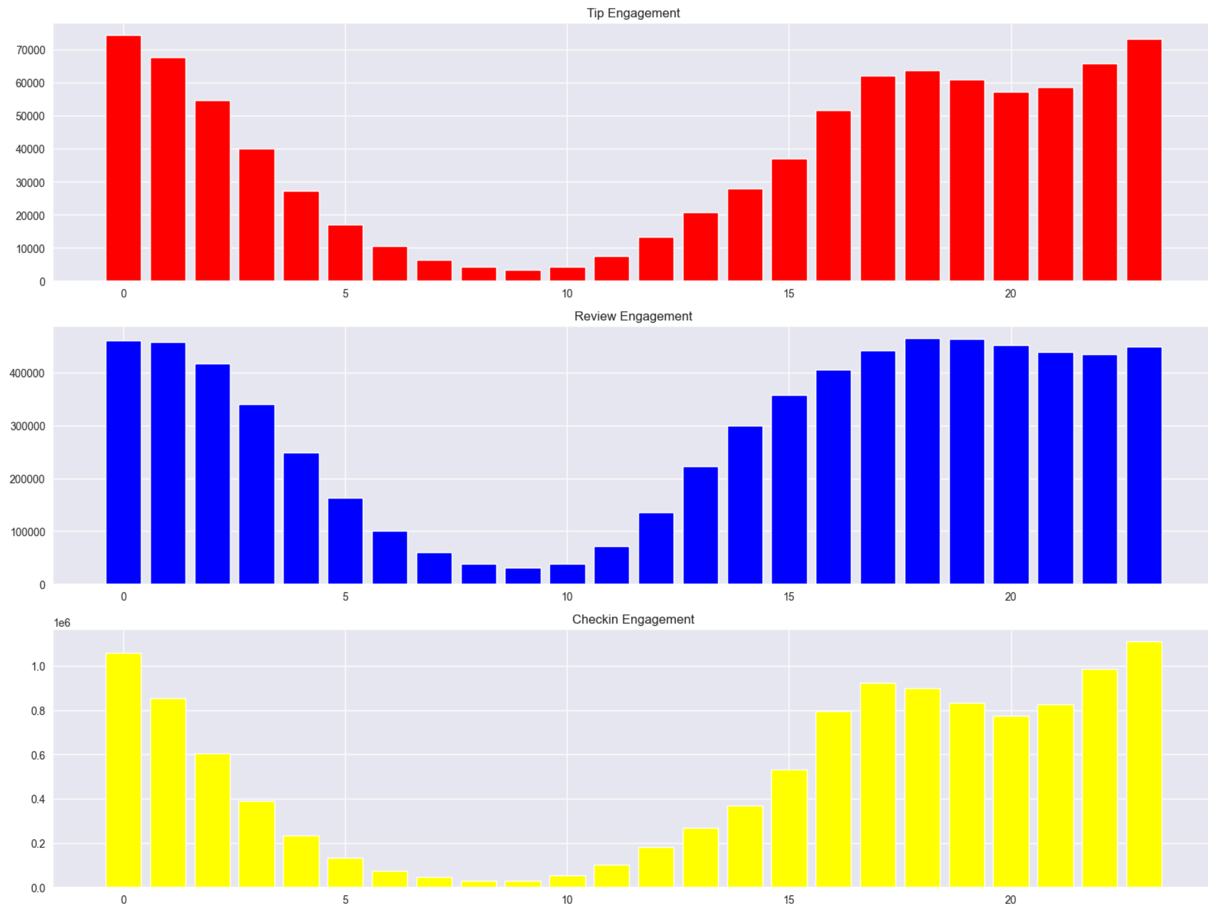
Q. What are the busiest hours for restaurants ?

In [76]:
```
review_engagement = pd.read_sql_query("""
SELECT
    HOUR(STR_TO_DATE(date, '%Y-%m-%d %H:%i:%s')) AS hour,
    COUNT(*) AS review_count
FROM review
GROUP BY hour;
""", engine)

tip_engagement = pd.read_sql_query("""
SELECT
    HOUR(STR_TO_DATE(date, '%Y-%m-%d %H:%i:%s')) AS hour,
    COUNT(*) AS tip_count
FROM tip
GROUP BY hour;
""", engine)

checkin = pd.read_sql_query("SELECT date FROM checkin", engine)
checkin_engagement = []

for i in checkin['date']:
    checkin_engagement.extend(
        [datetime.strptime(j.strip(), "%Y-%m-%d %H:%M:%S").strftime("%H")
         for j in i.split(',')]
    )

checkin_engagement = pd.DataFrame(checkin_engagement).astype(int).groupby(0)[0].cou
```

In [77]:
```
checkin_engagement.name = 'Values'
checkin_engagement
```

Out[77]:
```
0
0     1060009
1      853147
2      602632
3      388609
4      233305
5      133875
6       74652
7       46172
8       30390
9       29614
10      51960
11     100839
12     180490
13     267501
14     368108
15     532775
16     794471
17     924978
18     900055
19     833639
20     775722
21     827312
22     984501
23    1109237
Name: Values, dtype: int64
```

In [78]:
```
# Bar Plot
# tip_engagement
plt.subplot(3,1,1)
plt.title("Tip Engagement")
plt.bar(tip_engagement['hour'], tip_engagement['tip_count'], color = 'RED')

# review_engagement
plt.subplot(3,1,2)
plt.title("Review Engagement")
plt.bar(review_engagement['hour'], review_engagement['review_count'], color = 'BLUE

# Checkin Engagement
plt.subplot(3,1,3)
plt.title("Checkin Engagement")
plt.bar(checkin_engagement.index, checkin_engagement  , color = 'YELLOW')
plt.tight_layout()
plt.show()
```



NOTE :

The busiest R4 restaurants based on user engagement spanned from 3:00 PM to 1:00 AM, Knowing the peak hours allowed businesses to optimize their staffing level and resources accolation allocation during the during this. time to ensure defensiency of resident and quality service delivery. The concentration of user engagement during the evening and night hours suggest a higher demand for dining out, dining these times potentially driven by the factor such as work schedules, Social gathering and leisure activities.

RECOMMENDATIONS :

1. Utilizing inssights from the analysis of various metrics such as user engagement, sentiment of reviews, peak hours, and the impact of eilte users, businesses can make informed decisions to drive sucess.

2. Collaborating With light users and leveraging their influence can amplify proportional efforts, increase brand

awareness and the drive customer acquisition.

3. Businesses can adjust their operating hours or introduce special promotions to capitalize on increased demand during peak hours.

4. Less successful businesses may need to focus on strategies to enhance user engagement over time, such as improving service quality responding to customer feedback.

5. Cities with high success scores presents opportunities for restaurant chains to expand or invest further.

6. Understanding customer preferences behavior and satisfaction level is paramount. Businesses should focus on delivery exponential experience to meet customer expectations.

7. Positive reviews from Elite users and high user engagement can boost a business online visibility and reputation. Maintaining an active engagement with customers and responding promptly to feedback is crucial for building credibility and attracting a new customers.