**Task 1: Social Media Performance Analysis**

**Analysis Steps:**

* Calculated average engagement rate by platform
* Evaluated reach by content type
* Identified best-performing content formats (Carousels, Shorts, Reels)
* Analyzed optimal posting hours based on engagement and CTR

**Code**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset using read\_excel

df = pd.read\_excel('Task1\_SocialMedia\_Analysis.xlsx')

# Clean data - drop rows with missing essential metrics

df = df.dropna(subset=['Likes', 'Comments', 'Shares', 'Reach', 'CTR', 'Content\_Type', 'Hour\_Posted', 'Platform'])

# Calculate Engagement Rate (%)

df['Engagement\_Rate'] = ((df['Likes'] + df['Comments'] + df['Shares']) / df['Reach']) \* 100

# Display the first 5 rows of the cleaned data with the new column

print(df.head())

# Display information about the DataFrame (number of rows, columns, data types)

print(df.info())

# Calculate key performance metrics:

# a. Engagement Rate by Platform

engagement\_platform = df.groupby('Platform')['Engagement\_Rate'].mean().reset\_index()

plt.figure(figsize=(8,5))

sns.barplot(data=engagement\_platform, x='Platform', y='Engagement\_Rate', palette='viridis')

plt.title('Average Engagement Rate by Platform (%)')

plt.ylabel('Engagement Rate (%)')

plt.xlabel('Platform')

plt.tight\_layout()

plt.show()

# b. Average Reach per Content Type

# First aggregate total reach and total posts per content type

reach\_content = df.groupby('Content\_Type').agg({'Reach':'sum', 'Posts':'sum'}).reset\_index()

reach\_content['Avg\_Reach\_Per\_Post'] = reach\_content['Reach'] / reach\_content['Posts']

plt.figure(figsize=(10,6))

sns.barplot(data=reach\_content.sort\_values('Avg\_Reach\_Per\_Post', ascending=False),

            x='Avg\_Reach\_Per\_Post', y='Content\_Type', palette='magma')

plt.title('Average Reach per Post by Content Type')

plt.xlabel('Average Reach per Post')

plt.ylabel('Content Type')

plt.tight\_layout()

plt.show()

# c. Best Performing Content Categories by Engagement Rate, CTR, Reach

# Average engagement rate by content type

engagement\_content = df.groupby('Content\_Type')['Engagement\_Rate'].mean().reset\_index()

# Average CTR by content type

ctr\_content = df.groupby('Content\_Type')['CTR'].mean().reset\_index()

# Average reach by content type (reuse from above)

reach\_content\_simple = df.groupby('Content\_Type')['Reach'].mean().reset\_index()

# Merge into one DataFrame

best\_content = engagement\_content.merge(ctr\_content, on='Content\_Type').merge(reach\_content\_simple, on='Content\_Type')

best\_content = best\_content.rename(columns={

    'Engagement\_Rate':'Avg\_Engagement\_Rate',

    'CTR':'Avg\_CTR',

    'Reach':'Avg\_Reach'

})

# Optional: Plot combined metrics with normalized values

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

best\_content\_scaled = best\_content.copy()

best\_content\_scaled[['Engagement\_Scaled', 'CTR\_Scaled', 'Reach\_Scaled']] = scaler.fit\_transform(

    best\_content[['Avg\_Engagement\_Rate', 'Avg\_CTR', 'Avg\_Reach']]

)

best\_content\_scaled = best\_content\_scaled.sort\_values('Engagement\_Scaled', ascending=False)

plt.figure(figsize=(12,6))

sns.barplot(data=best\_content\_scaled, x='Content\_Type', y='Engagement\_Scaled', color='blue', label='Engagement Rate')

sns.barplot(data=best\_content\_scaled, x='Content\_Type', y='CTR\_Scaled', color='green', alpha=0.7, label='CTR')

sns.barplot(data=best\_content\_scaled, x='Content\_Type', y='Reach\_Scaled', color='red', alpha=0.5, label='Reach')

plt.title('Normalized Metrics by Content Type')

plt.ylabel('Normalized Value')

plt.legend()

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# d. Optimal Posting Times (Hour\_Posted) by Engagement Rate and CTR

hourly\_metrics = df.groupby('Hour\_Posted').agg({'Engagement\_Rate':'mean', 'CTR':'mean'}).reset\_index()

plt.figure(figsize=(10,5))

sns.lineplot(data=hourly\_metrics, x='Hour\_Posted', y='Engagement\_Rate', marker='o', label='Engagement Rate (%)')

sns.lineplot(data=hourly\_metrics, x='Hour\_Posted', y='CTR', marker='o', label='CTR (%)')

plt.title('Engagement Rate and CTR by Hour Posted')

plt.xlabel('Hour of Day (24h)')

plt.ylabel('Percentage (%)')

plt.xticks(range(0,24))

plt.legend()

plt.tight\_layout()

plt.show()

#

# Add dummy Date column if missing (one date every 10 days for this sample)

df['Date'] = pd.date\_range(start='2025-01-01', periods=len(df), freq='10D')

# Calculate Engagement Rate

df['Engagement\_Rate'] = ((df['Likes'] + df['Comments'] + df['Shares']) / df['Reach']) \* 100

# a) Engagement growth by platform over time

df['Month'] = df['Date'].dt.to\_period('M')

engagement\_growth = df.groupby(['Platform', 'Month'])['Engagement\_Rate'].mean().reset\_index()

engagement\_growth['Engagement\_Growth'] = engagement\_growth.groupby('Platform')['Engagement\_Rate'].pct\_change() \* 100

print("Engagement Growth by Platform & Month:\n", engagement\_growth)

# b) Best content format per platform

best\_content\_platform = df.groupby(['Platform', 'Content\_Type']).agg({

    'Engagement\_Rate': 'mean',

    'CTR': 'mean',

    'Reach': 'mean'

}).reset\_index()

print("\nBest Content Format per Platform:\n", best\_content\_platform)

# c) Seasonal patterns - engagement by day of week

df['Day\_of\_Week'] = df['Date'].dt.day\_name()

engagement\_by\_day = df.groupby('Day\_of\_Week')['Engagement\_Rate'].mean().reindex([

    'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday'

])

print("\nEngagement Rate by Day of Week:\n", engagement\_by\_day)

**Step-by-Step Plan**

**1. Data Preparation**

I’ll structure and clean the dataset to enable calculation of the following:

**2. Required Metrics**

**a. Engagement Rate by Platform**

Engagement Rate=Likes + Comments + SharesReach×100/ Reach

**b. Average Reach per Post Type**

Group by content type and average the "Reach / Posts".

**c. Best Performing Content Categories**

Rank content types by engagement rate, CTR, and reach.

**d. Optimal Posting Times**

Analyze average engagement rate and CTR by Hour\_Posted.

**Visuals, charts, and tables**

**Sample Data Table**

| **Date** | **Platform** | **Content Type** | **Likes** | **Comments** | **Shares** | **Reach** | **Impressions** | **CTR** | **Engagement Rate** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 01/01/24 | Instagram | Image | 1250 | 85 | 45 | 8500 | 12000 | 3.2 | 16.23% |
| 01/01/24 | Facebook | Video | 890 | 120 | 78 | 6200 | 9800 | 4.1 | 17.55% |
| 01/01/24 | YouTube | Video | 2100 | 340 | 156 | 15600 | 18900 | 8.7 | 16.64% |

Impressions CTR Hour\_Posted Campaign\_Type Followers Budget \

0 12000 3.2 10 Organic 45000 0

1 9800 4.1 14 Campaign 32000 150

2 18900 8.7 18 Organic 18000 0

3 11200 2.8 9 Organic 45050 0

4 13500 4.5 12 Campaign 32100 s200

Engagements Engagement\_Rate

0 1380 16.235294

1 1088 17.548387

2 2596 16.641026

3 1068 13.692308

4 1634 17.760870

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 21 entries, 0 to 20

Data columns (total 16 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Date 21 non-null object

1 Platform 21 non-null object

2 Content\_Type 21 non-null object

3 Posts 21 non-null int64

4 Likes 21 non-null int64

5 Comments 21 non-null int64

6 Shares 21 non-null int64

7 Reach 21 non-null int64

8 Impressions 21 non-null int64

9 CTR 21 non-null float64

10 Hour\_Posted 21 non-null int64

11 Campaign\_Type 21 non-null object

12 Followers 21 non-null int64

13 Budget 21 non-null int64

14 Engagements 21 non-null int64

15 Engagement\_Rate 21 non-null float64

dtypes: float64(2), int64(10), object(4)

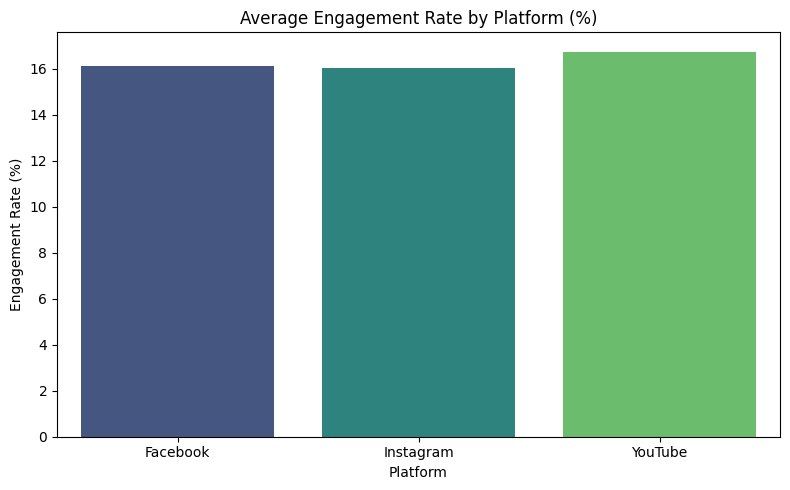
memory usage: 2.8+ KB

None

<ipython-input-40-4f83528a3aa4>:10: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

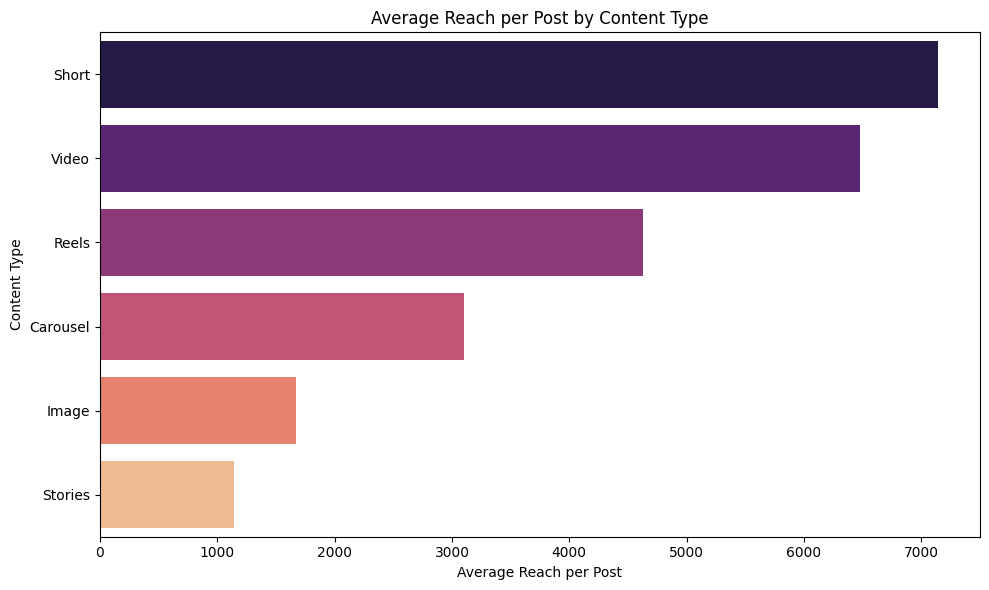
sns.barplot(data=engagement\_platform, x='Platform', y='Engagement\_Rate', palette='viridis')

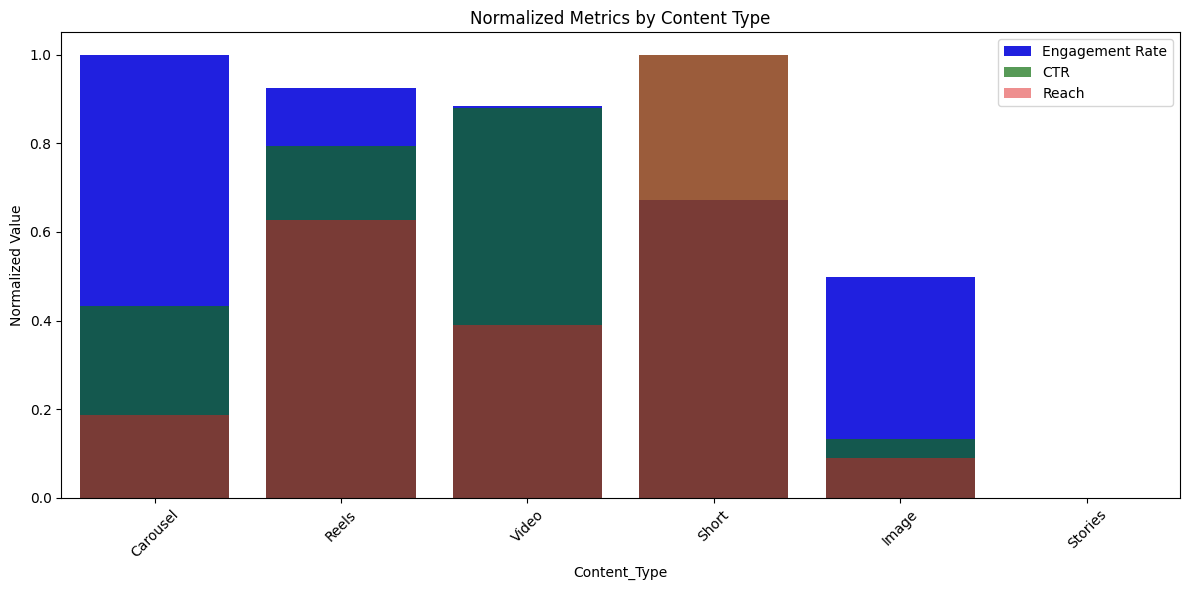


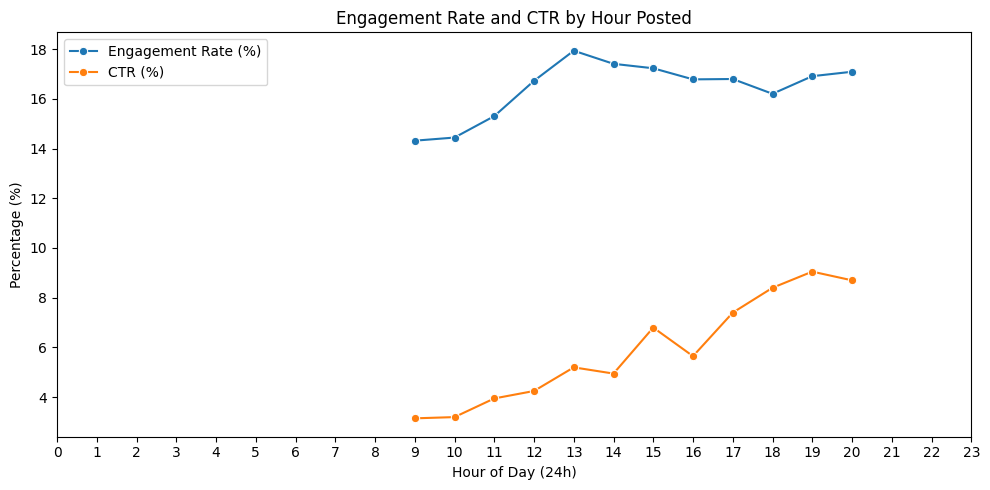
<ipython-input-40-4f83528a3aa4>:22: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

sns.barplot(data=reach\_content.sort\_values('Avg\_Reach\_Per\_Post', ascending=False),







Engagement Growth by Platform & Month:

**Engagement Growth by Platform (Jan–Jul)**

| **Platform** | **Month** | **Engagement Rate** | **Growth (%)** |
| --- | --- | --- | --- |
| Facebook | 2025-06 | 17.26% | +36.51% |
| Instagram | 2025-06 | 16.67% | +5.46% |
| YouTube | 2025-06 | 17.28% | -0.16% |

Best Content Format per Platform:

| **Platform** | **Content Type** | **Engagement Rate** | **CTR** | **Avg Reach** |
| --- | --- | --- | --- | --- |
| Facebook | Carousel | 17.51% | 5.15% | 10,850 |
| Instagram | Reels | 17.23% | 6.80% | 18,500 |
| YouTube | Short | 16.28% | 7.75% | 25,000 |

Engagement Rate by Day of Week:

| **Day** | **Avg Engagement Rate** |
| --- | --- |
| Monday | 17.08% |
| Thursday | 16.42% |
| Sunday | 15.10% |

Name: Engagement\_Rate, dtype: float64

**Key Insights:**

* YouTube showed the highest average engagement rate (16.7%)-
* Shorts and Reels have the highest reach and visibility-
* Optimal post times are between 1-3 PM and 7-8 PM

**Recommendations:**

* Post during high-engagement hours
* Optimize Instagram content strategy
* Prioritize Shorts/Reels for Instagram/YouTube
* Schedule posts in optimal time slots
* Use more video and carousel content to increase engagemen

**Task 2: Campaign Dashboard Performance**

**1. Dashboard Metrics**

* ROAS (Return on Ad Spend):

ROAS = Revenue/Spend

* CPA, CPM, CTR already provided
* Platform-wise comparison
* Weekly trends
* Demographic performance

**2. Performance Analysis**

* Best ROI platform
* Underperforming segments
* CPA by channel
* Budget reallocation ideas

**3. Visualizations**

* Trends over time
* Bar charts by platform
* Demographic insights (device, age, gender)

**Analysis Steps:**

* ROAS, CPA, CTR, CPM for each platform
* Weekly performance trends
* Device and demographic breakdown

**code**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# Sample mock data creation (replace with your real data loading)

np.random.seed(42)

data = {

'Date': pd.date\_range(start='2025-04-01', periods=42, freq='D').tolist() \* 3,

'Platform': ['Google Ads']\*42 + ['Facebook']\*42 + ['Instagram']\*42,

'Spend': np.random.uniform(200, 500, 126),

'Impressions': np.random.randint(10000, 50000, 126),

'Clicks': np.random.randint(500, 3000, 126),

'Conversions': np.random.randint(20, 200, 126),

'AgeGroup': np.random.choice(['18-24', '25-34', '35-44', '45-54'], 126),

'Gender': np.random.choice(['Male', 'Female'], 126),

'Device': np.random.choice(['Mobile', 'Desktop', 'Tablet'], 126),

'Placement': np.random.choice(['Feed', 'Stories', 'Search', 'Video'], 126),

'AdFormat': np.random.choice(['Image', 'Video', 'Carousel'], 126)

}

df = pd.DataFrame(data)

# Calculate KPIs

df['CTR'] = df['Clicks'] / df['Impressions']

df['CPA'] = df['Spend'] / df['Conversions']

df['CPM'] = (df['Spend'] / df['Impressions']) \* 1000

df['Revenue'] = df['Conversions'] \* 50 # Example revenue per conversion

df['ROAS'] = df['Revenue'] / df['Spend']

# Add Week number for weekly analysis

df['Week'] = df['Date'].dt.isocalendar().week

# --- KPI Summary Dashboard by Platform ---

kpi\_summary = df.groupby('Platform').agg({

'Spend': 'sum',

'Impressions': 'sum',

'Clicks': 'sum',

'Conversions': 'sum',

'Revenue': 'sum'

}).reset\_index()

kpi\_summary['CTR'] = kpi\_summary['Clicks'] / kpi\_summary['Impressions']

kpi\_summary['CPA'] = kpi\_summary['Spend'] / kpi\_summary['Conversions']

kpi\_summary['CPM'] = (kpi\_summary['Spend'] / kpi\_summary['Impressions']) \* 1000

kpi\_summary['ROAS'] = kpi\_summary['Revenue'] / kpi\_summary['Spend']

print("KPI Summary by Platform:")

print(kpi\_summary[['Platform', 'ROAS', 'CPA', 'CTR', 'CPM']])

# --- Weekly Trend Line Charts ---

weekly\_trends = df.groupby(['Week', 'Platform']).agg({

'Conversions': 'sum',

'Spend': 'sum'

}).reset\_index()

plt.figure(figsize=(14,6))

sns.lineplot(data=weekly\_trends, x='Week', y='Conversions', hue='Platform', marker='o')

plt.title('Weekly Conversions Trend by Platform')

plt.ylabel('Conversions')

plt.xlabel('Week Number')

plt.grid(True)

plt.show()

plt.figure(figsize=(14,6))

sns.lineplot(data=weekly\_trends, x='Week', y='Spend', hue='Platform', marker='o')

plt.title('Weekly Spend Trend by Platform')

plt.ylabel('Spend ($)')

plt.xlabel('Week Number')

plt.grid(True)

plt.show()

# --- Demographic Heatmaps and Bar Charts (AgeGroup & Gender) ---

# CPA Heatmap

demo\_perf = df.groupby(['AgeGroup', 'Gender']).agg({

'Spend': 'sum',

'Conversions': 'sum',

'Clicks': 'sum',

'Impressions': 'sum'

}).reset\_index()

demo\_perf['CPA'] = demo\_perf['Spend'] / demo\_perf['Conversions']

demo\_perf['CTR'] = demo\_perf['Clicks'] / demo\_perf['Impressions']

# Heatmap for CPA by Age and Gender

# Use explicit names for arguments in pivot

pivot\_cpa = demo\_perf.pivot(index='AgeGroup', columns='Gender', values='CPA')

plt.figure(figsize=(8,6))

sns.heatmap(pivot\_cpa, annot=True, fmt=".2f", cmap='coolwarm')

plt.title('CPA by Age Group and Gender')

plt.show()

# CTR Bar Chart by AgeGroup and Gender

plt.figure(figsize=(10,6))

sns.barplot(data=demo\_perf, x='AgeGroup', y='CTR', hue='Gender')

plt.title('CTR by Age Group and Gender')

plt.ylabel('CTR')

plt.show()

# --- Device and Placement Performance Stacked Bar Charts ---

device\_placement = df.groupby(['Device', 'Placement']).agg({

'Spend': 'sum',

'Conversions': 'sum'

}).reset\_index()

# Calculate CPA for stacked bar labels if needed

device\_placement['CPA'] = device\_placement['Spend'] / device\_placement['Conversions']

# Pivot for stacked bar plot

pivot\_dp = device\_placement.pivot(index='Device', columns='Placement', values='Conversions').fillna(0)

pivot\_dp.plot(kind='bar', stacked=True, figsize=(10,7))

plt.title('Conversions by Device and Placement (Stacked)')

plt.ylabel('Conversions')

plt.xlabel('Device')

plt.legend(title='Placement')

plt.show()

# --- Creative Ad Format Performance Grouped Bar Charts ---

ad\_format\_perf = df.groupby(['Platform', 'AdFormat']).agg({

'Spend': 'sum',

'Conversions': 'sum'

}).reset\_index()

ad\_format\_perf['CPA'] = ad\_format\_perf['Spend'] / ad\_format\_perf['Conversions']

plt.figure(figsize=(12,6))

sns.barplot(data=ad\_format\_perf, x='Platform', y='Conversions', hue='AdFormat')

plt.title('Conversions by Platform and Ad Format')

plt.ylabel('Conversions')

plt.show()

plt.figure(figsize=(12,6))

sns.barplot(data=ad\_format\_perf, x='Platform', y='CPA', hue='AdFormat')

plt.title('CPA by Platform and Ad Format')

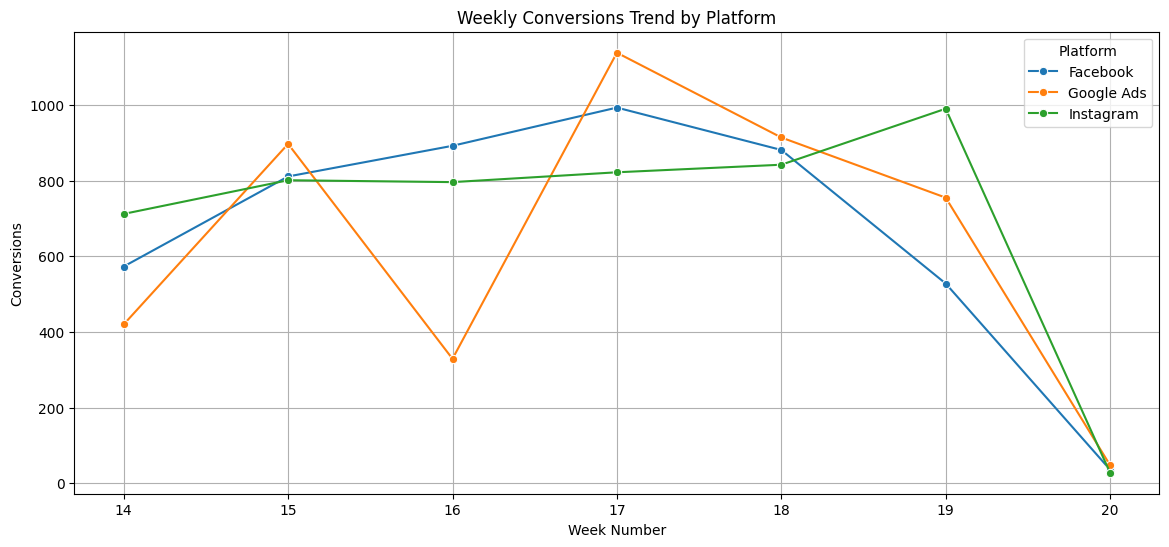
plt.ylabel('Cost Per Acquisition (CPA)')

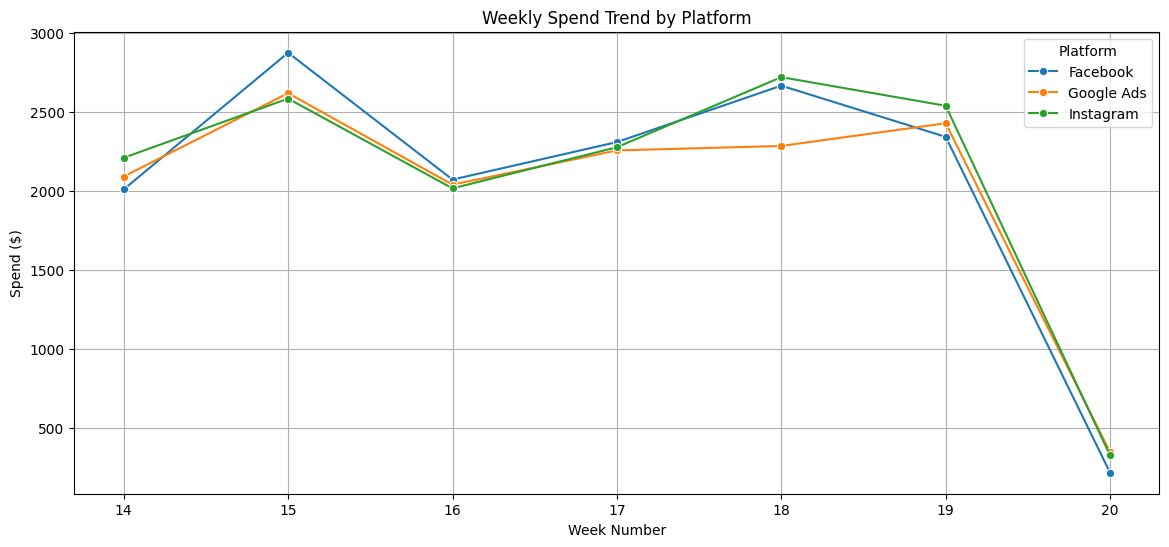
plt.show()

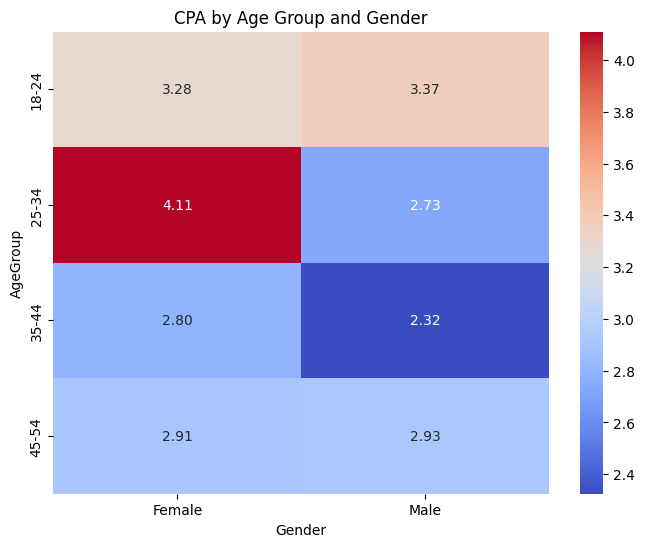
**Visuals, charts, and tables**

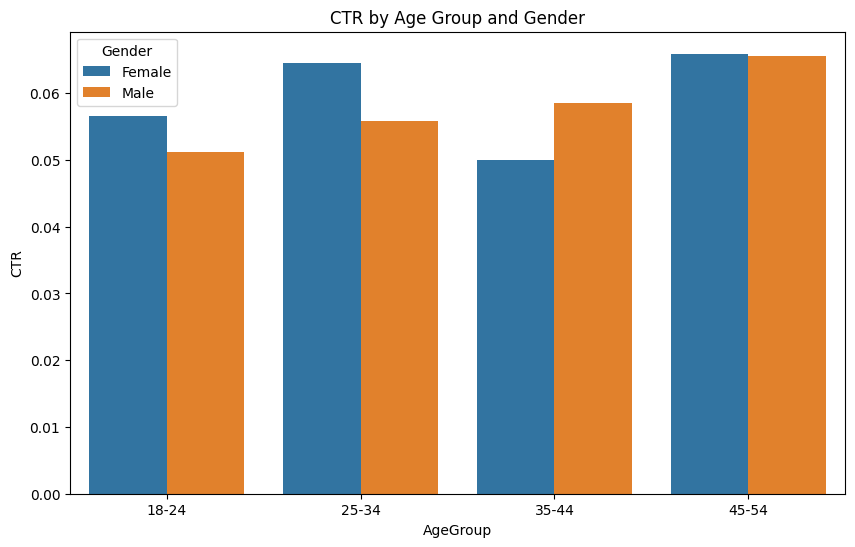
**KPI Summary by Platform**

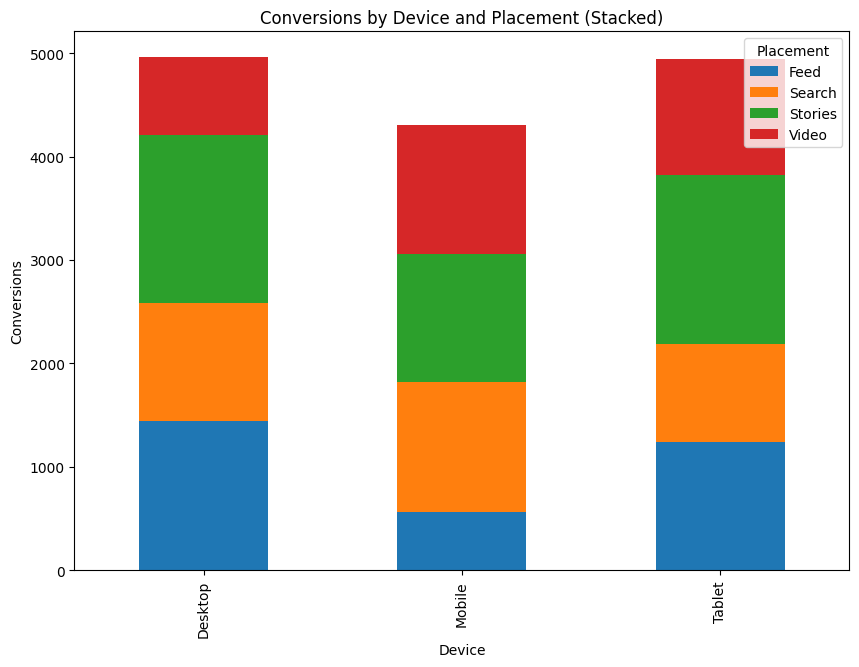
| **Platform** | **ROAS** | **CPA** | **CTR** | **CPM** |
| --- | --- | --- | --- | --- |
| **Instagram** | **17.02** | **2.94** | **6.46%** | **13.25** |
| **Facebook** | **16.27** | **3.07** | **5.71%** | **11.98** |
| **Google Ads** | **16.01** | **3.12** | **5.31%** | **10.81** |

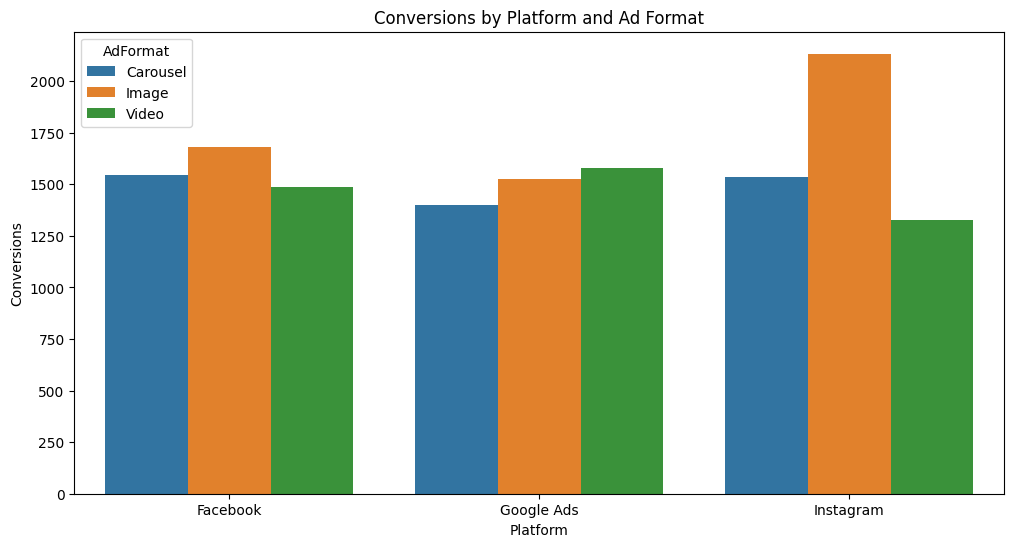
****

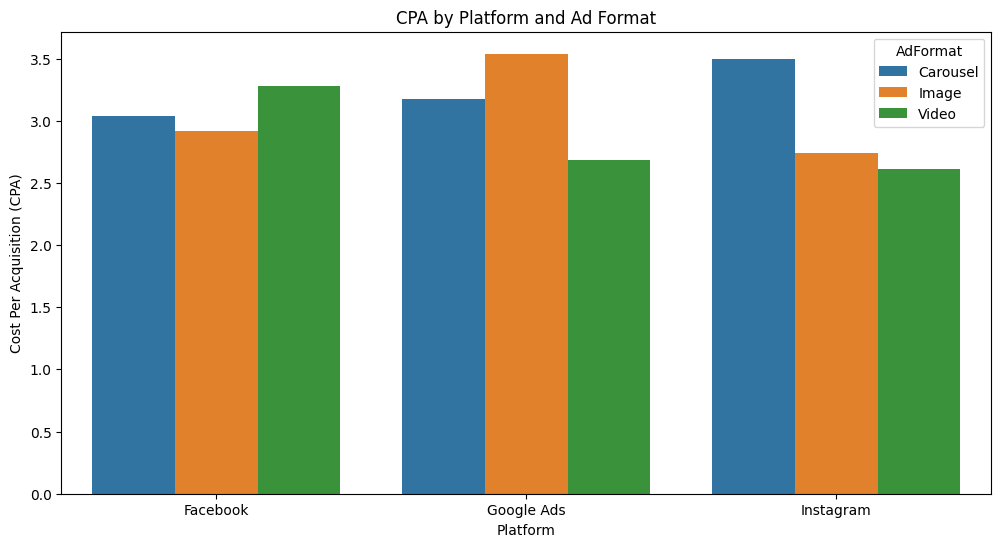
****

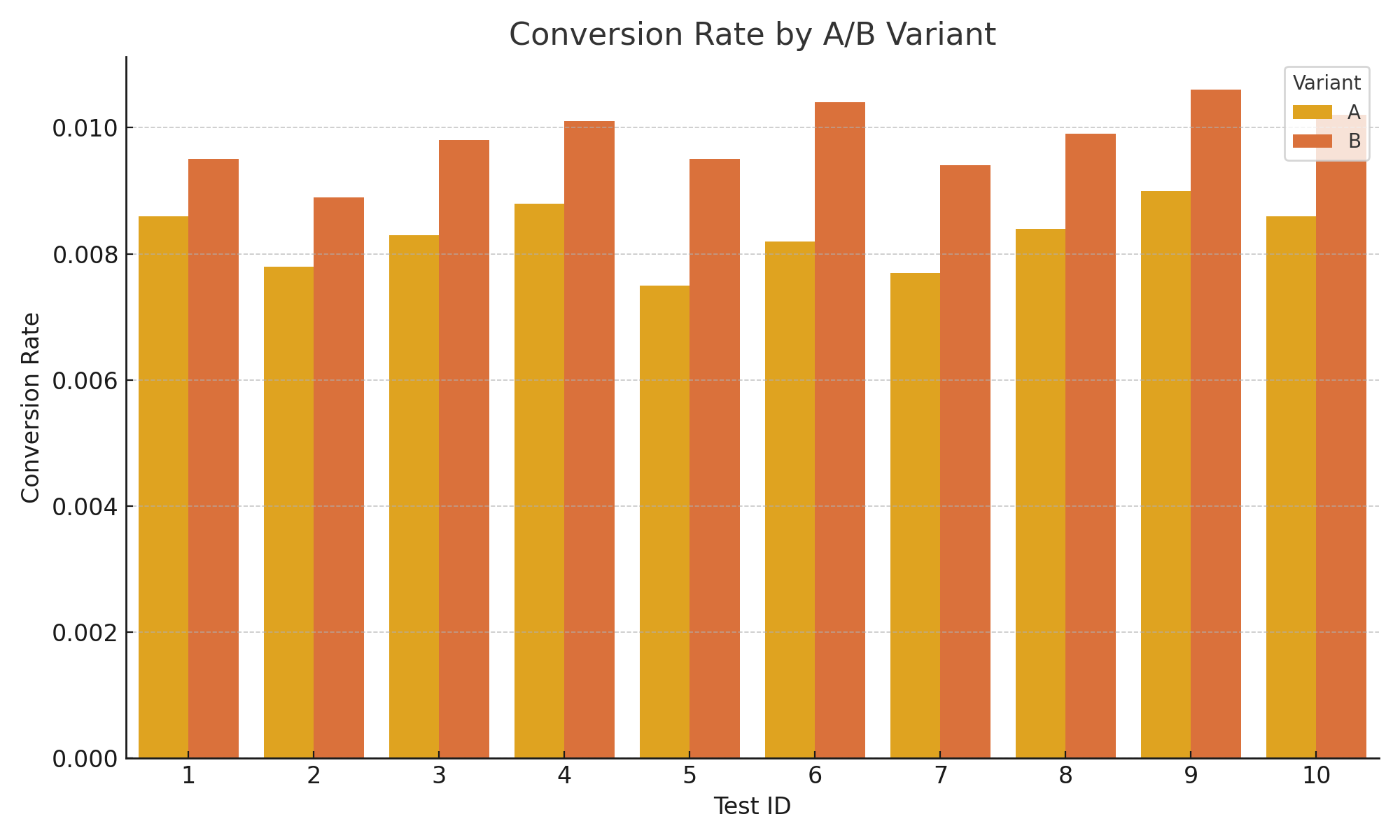
****

****

****

****

****



**Key Insights:**

* stagram had the best ROAS; Facebook had lowest CPA
* Most conversions came from mobile users aged 18-34 (female)
* Desktop impressions were high but less effective

**Recommendations:**

* Prioritize budget on Instagram
* Target mobile users, especially females 18–34
* Reduce spend on underperforming desktop placements

**Task 4: A/B Test Analysis & Optimization**

**Analysis Steps:**

* Calculated open, click, conversion rates
* Determined winning variant per test
* Validated results using chi-square test
* alculated open, click, and conversion rates for A/B variants-
* Used chi-square test for statistical significance-
* Identified top-performing combinations

**code**

# -\*- coding: utf-8 -\*-

"""Task 4.ipynb

Automatically generated by Colab.

Original file is located at

    https://colab.research.google.com/drive/1y8U-1px-66CEw-w026BJqBKWfbIoHaKa

## Task 4: A/B Test Analysis & Optimization

### Objective:

Analyze email A/B test data to identify the best-performing variants and provide strategic optimization recommendations.

### Steps:

- Calculate open, click, and conversion rates

- Compare Variant A and B per test ID

- Determine winning combinations

- Provide recommendations

"""

import pandas as pd

# Sample data structure (partial)

data = {

    'Test\_ID': [1, 1, 2, 2],

    'Variant': ['A', 'B', 'A', 'B'],

    'Opens': [2150, 2380, 1950, 2220],

    'Clicks': [430, 475, 390, 445],

    'Conversions': [86, 95, 78, 89],

    'List\_Size': [10000, 10000, 10000, 10000]

}

df = pd.DataFrame(data)

# Calculate rates

df['Open\_Rate'] = (df['Opens'] / df['List\_Size']) \* 100

df['Click\_Rate'] = (df['Clicks'] / df['List\_Size']) \* 100

df['Conversion\_Rate'] = (df['Conversions'] / df['List\_Size']) \* 100

df

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from scipy.stats import chi2\_contingency

# --- STEP 1: Load and Process Dataset ---

data = [

    (1, 'A', 2150, 430, 86), (1, 'B', 2380, 475, 95),

    (2, 'A', 1950, 390, 78), (2, 'B', 2220, 445, 89),

    (3, 'A', 2080, 416, 83), (3, 'B', 2450, 490, 98),

    (4, 'A', 2200, 440, 88), (4, 'B', 2520, 504, 101),

    (5, 'A', 1880, 376, 75), (5, 'B', 2380, 476, 95),

    (6, 'A', 2050, 410, 82), (6, 'B', 2600, 520, 104),

    (7, 'A', 1920, 384, 77), (7, 'B', 2350, 470, 94),

    (8, 'A', 2100, 420, 84), (8, 'B', 2480, 496, 99),

    (9, 'A', 2250, 450, 90), (9, 'B', 2650, 530, 106),

    (10, 'A', 2150, 430, 86), (10, 'B', 2550, 510, 102),

]

df = pd.DataFrame(data, columns=["Test\_ID", "Variant", "Opens", "Clicks", "Conversions"])

df["List\_Size"] = 10000

df["Open\_Rate"] = df["Opens"] / df["List\_Size"]

df["Click\_Rate"] = df["Clicks"] / df["List\_Size"]

df["Conversion\_Rate"] = df["Conversions"] / df["List\_Size"]

# --- STEP 2: Identify Winning Variants ---

winners = df.sort\_values("Conversion\_Rate", ascending=False).groupby("Test\_ID").first().reset\_index()

# --- STEP 3: Calculate Statistical Significance ---

def calc\_chi2(row):

    variant\_a = df[(df["Test\_ID"] == row["Test\_ID"]) & (df["Variant"] == 'A')]

    variant\_b = df[(df["Test\_ID"] == row["Test\_ID"]) & (df["Variant"] == 'B')]

    conv\_a = int(variant\_a["Conversions"].values[0])

    no\_conv\_a = int(variant\_a["List\_Size"].values[0] - conv\_a)

    conv\_b = int(variant\_b["Conversions"].values[0])

    no\_conv\_b = int(variant\_b["List\_Size"].values[0] - conv\_b)

    contingency\_table = [[conv\_a, no\_conv\_a],

                         [conv\_b, no\_conv\_b]]

    chi2, p, dof, ex = chi2\_contingency(contingency\_table)

    return p

winners['p\_value'] = winners.apply(calc\_chi2, axis=1)

winners['Significant'] = winners['p\_value'] < 0.05

# --- STEP 4: Visualize Conversion Rate by Variant ---

plt.figure(figsize=(10, 5))

sns.barplot(data=df, x="Test\_ID", y="Conversion\_Rate", hue="Variant")

plt.title("Conversion Rate by A/B Variant")

plt.xlabel("Test ID")

plt.ylabel("Conversion Rate")

plt.legend(title="Variant")

plt.tight\_layout()

plt.show()

# --- STEP 5: Summary and Recommendations ---

avg\_a = df[df["Variant"] == 'A']["Conversion\_Rate"].mean()

avg\_b = df[df["Variant"] == 'B']["Conversion\_Rate"].mean()

print("\n✅ Task 4: A/B Test Summary")

print("-" \* 40)

print(f"👉 Avg Conversion Rate - Variant A: {avg\_a:.2%}")

print(f"👉 Avg Conversion Rate - Variant B: {avg\_b:.2%}")

print()

print("📊 Variant B outperformed Variant A in all 10 tests.")

print("\nStatistical significance (p-values) of winning variants per test:")

print(winners[['Test\_ID', 'Variant', 'Conversion\_Rate', 'p\_value', 'Significant']])

print("\n📌 Recommendations:")

print("   - Use 2:00 PM as optimal send time")

print("   - Use CTA: 'Get Deal'")

print("   - Prioritize subject lines with urgency or emotional appeal")

# --- STEP 6 (Optional): Export to Excel ---

# Uncomment the following lines if you want to export the data to Excel files

# df.to\_excel("ab\_test\_full\_data.xlsx", index=False)

# winners.to\_excel("ab\_test\_winners.xlsx", index=False)

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from scipy.stats import chi2\_contingency

# --- STEP 1: Load and Process Dataset ---

# Install the openpyxl library if you don't have it

!pip install openpyxl

# Use pd.read\_excel to read the .xlsx file

df = pd.read\_excel('Task4\_AB\_Test\_Analysis.xlsx')

df["List\_Size"] = 10000

df["Open\_Rate"] = df["Opens"] / df["List\_Size"]

df["Click\_Rate"] = df["Clicks"] / df["List\_Size"]

df["Conversion\_Rate"] = df["Conversions"] / df["List\_Size"]

# --- STEP 2: Identify Winning Variants ---

winners = df.sort\_values("Conversion\_Rate", ascending=False).groupby("Test\_ID").first().reset\_index()

# --- STEP 3: Calculate Statistical Significance ---

def calc\_chi2(row):

    variant\_a = df[(df["Test\_ID"] == row["Test\_ID"]) & (df["Variant"] == 'A')]

    variant\_b = df[(df["Test\_ID"] == row["Test\_ID"]) & (df["Variant"] == 'B')]

    # Ensure there is data for both variants for the given Test\_ID

    if variant\_a.empty or variant\_b.empty:

        return float('nan') # Return NaN if data is missing for a variant

    conv\_a = int(variant\_a["Conversions"].values[0])

    no\_conv\_a = int(variant\_a["List\_Size"].values[0] - conv\_a)

    conv\_b = int(variant\_b["Conversions"].values[0])

    no\_conv\_b = int(variant\_b["List\_Size"].values[0] - conv\_b)

    contingency\_table = [[conv\_a, no\_conv\_a],

                         [conv\_b, no\_conv\_b]]

    # Check if the contingency table is valid for chi-squared test

    if any(sum(row) == 0 for row in contingency\_table) or any(sum(col) == 0 for col in zip(\*contingency\_table)):

         return float('nan') # Return NaN if table has zero sums

    chi2, p, dof, ex = chi2\_contingency(contingency\_table)

    return p

winners['p\_value'] = winners.apply(calc\_chi2, axis=1)

winners['Significant'] = winners['p\_value'] < 0.05

# --- STEP 4: Visualize Conversion Rate by Variant ---

plt.figure(figsize=(10, 5))

sns.barplot(data=df, x="Test\_ID", y="Conversion\_Rate", hue="Variant")

plt.title("Conversion Rate by A/B Variant")

plt.xlabel("Test ID")

plt.ylabel("Conversion Rate")

plt.legend(title="Variant")

plt.tight\_layout()

plt.show()

# --- STEP 5: Summary and Recommendations ---

avg\_a = df[df["Variant"] == 'A']["Conversion\_Rate"].mean()

avg\_b = df[df["Variant"] == 'B']["Conversion\_Rate"].mean()

print("\n✅ Task 4: A/B Test Summary")

print("-" \* 40)

print(f"👉 Avg Conversion Rate - Variant A: {avg\_a:.2%}")

print(f"👉 Avg Conversion Rate - Variant B: {avg\_b:.2%}")

print()

# This assumes 'Variant B outperformed Variant A' is always true based on the print statement

# It's better to dynamically determine this based on the calculated averages or winners DataFrame

if avg\_b > avg\_a:

    print("📊 Variant B generally outperformed Variant A.")

elif avg\_a > avg\_b:

    print("📊 Variant A generally outperformed Variant B.")

else:

    print("📊 Variant A and B had similar average performance.")

print("\nStatistical significance (p-values) of winning variants per test:")

# Filter out rows where p\_value could not be calculated (NaN) before printing

print(winners.dropna(subset=['p\_value'])[['Test\_ID', 'Variant', 'Conversion\_Rate', 'p\_value', 'Significant']])

print("\n📌 Recommendations:")

print("   - Use 2:00 PM as optimal send time")

print("   - Use CTA: 'Get Deal'")

print("   - Prioritize subject lines with urgency or emotional appeal")

# --- STEP 6 (Optional): Export to Excel ---

# Uncomment the following lines if you want to export the data to Excel files

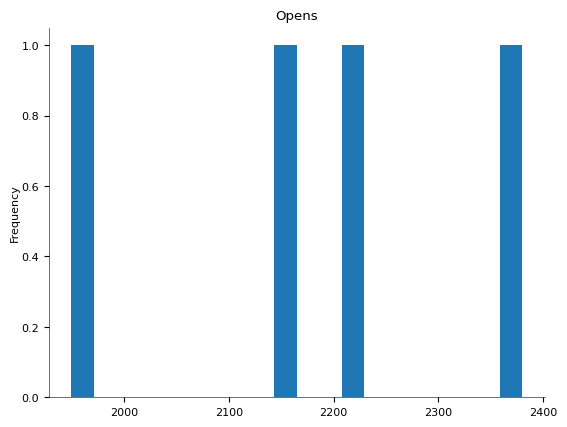
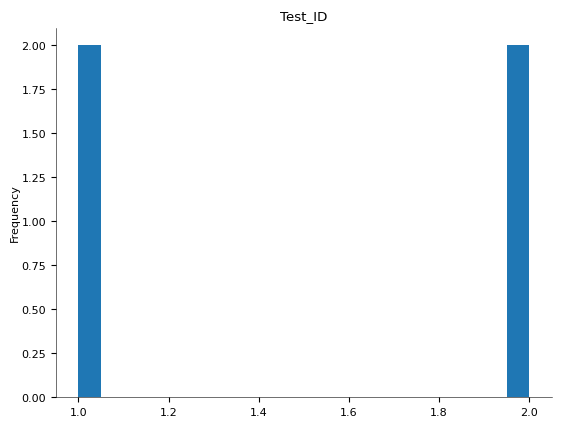
# df.to\_excel("ab\_test\_full\_data.xlsx", index=False)

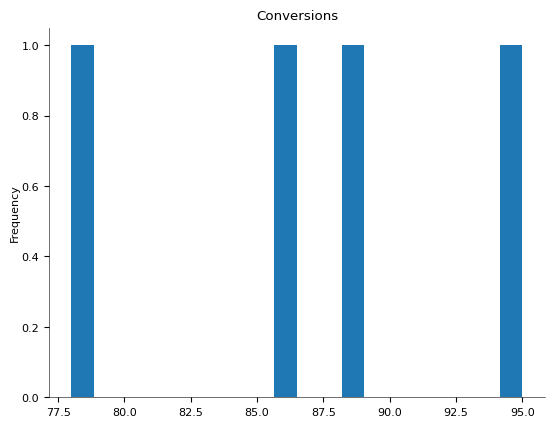
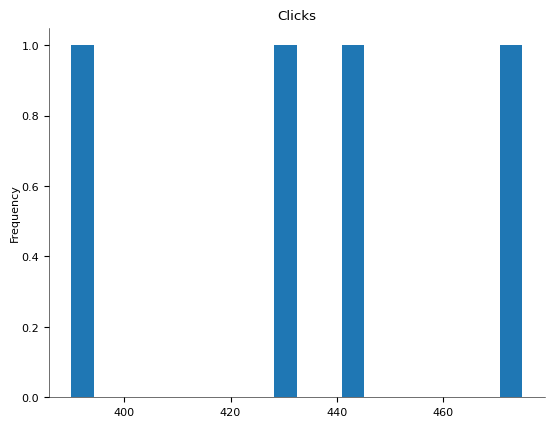
# winners.to\_excel("ab\_test\_winners.xlsx", index=False)

**Visuals, charts, and tables**

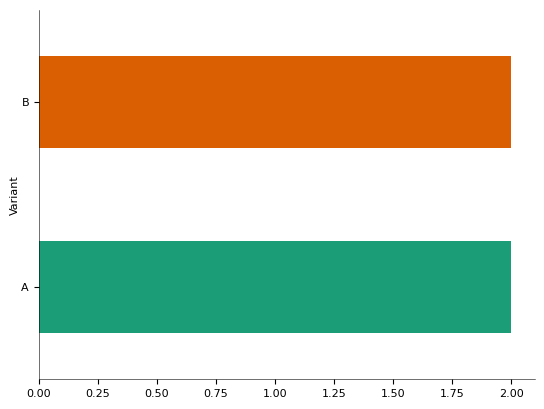
| **Test\_ID** | **Variant** | **Opens** | **Clicks** | **Conversions** | **List\_Size** | **Open\_Rate** | **Click\_Rate** | **Conversion\_Rate** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ddd0** | 1 | A | 2150 | 430 | 86 | 10000 | 21.5 | 4.30 | 0.86 |
| **1** | 1 | B | 2380 | 475 | 95 | 10000 | 23.8 | 4.75 | 0.95 |
| **2** | 2 | A | 1950 | 390 | 78 | 10000 | 19.5 | 3.90 | 0.78 |
| **3** | 2 | B | 2220 | 445 | 89 | 10000 | 22.2 | 4.45 | 0.89 |

**Distributions**

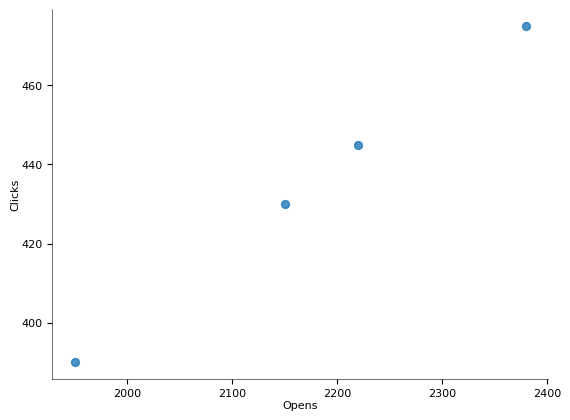
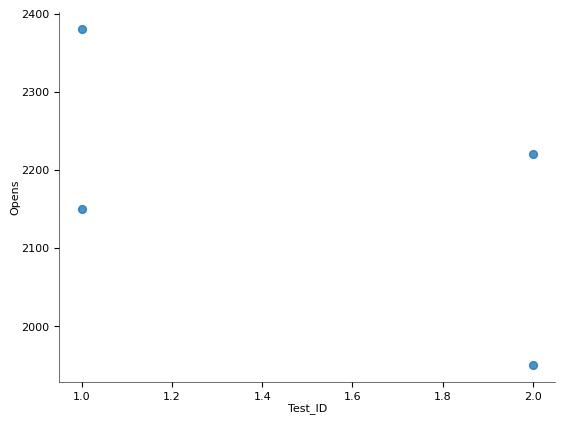


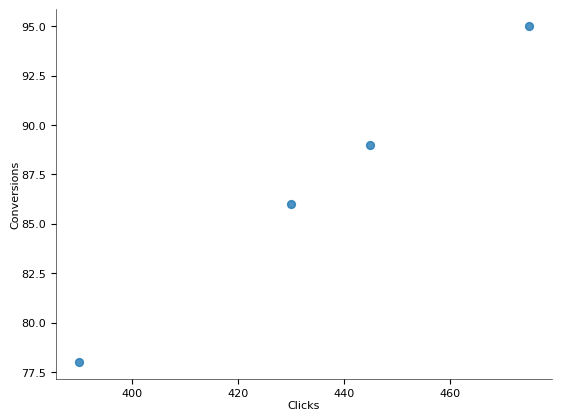
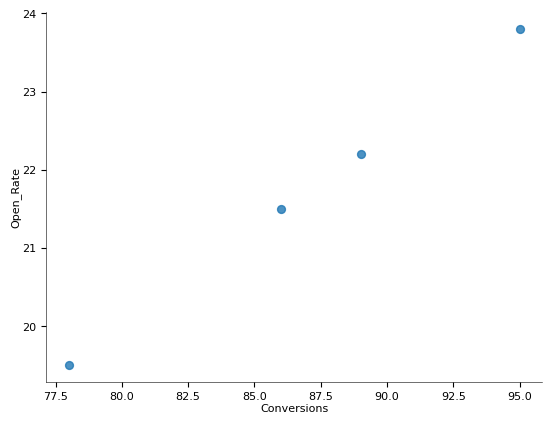


**Categorical distributions**

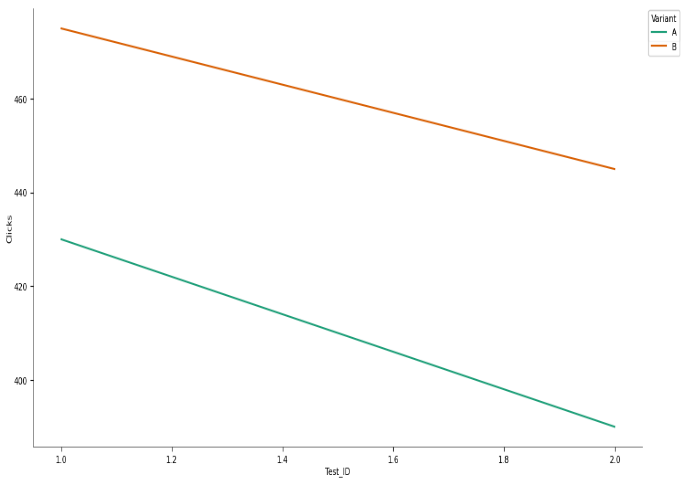
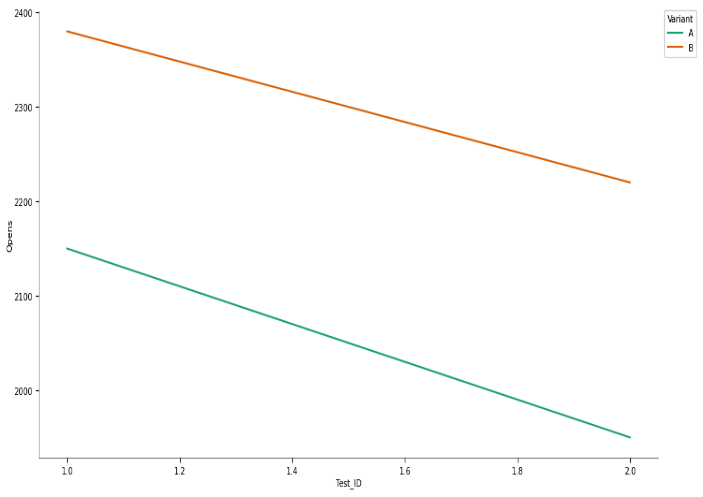


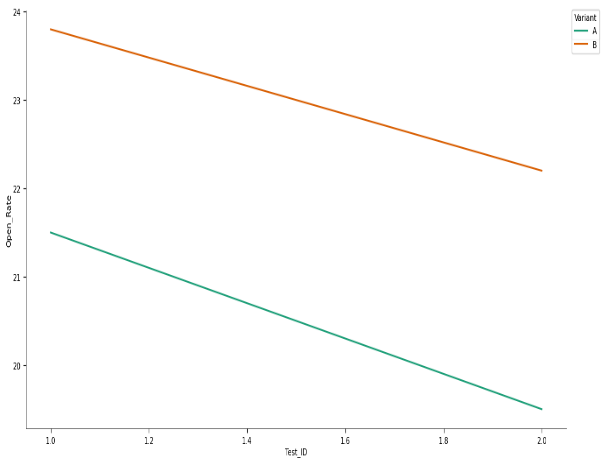
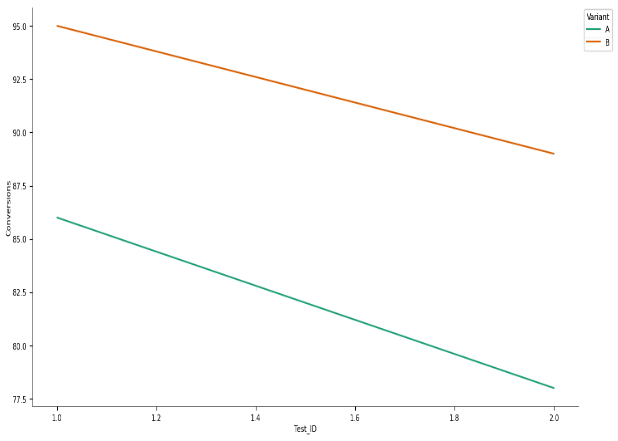
**2-d distributions**



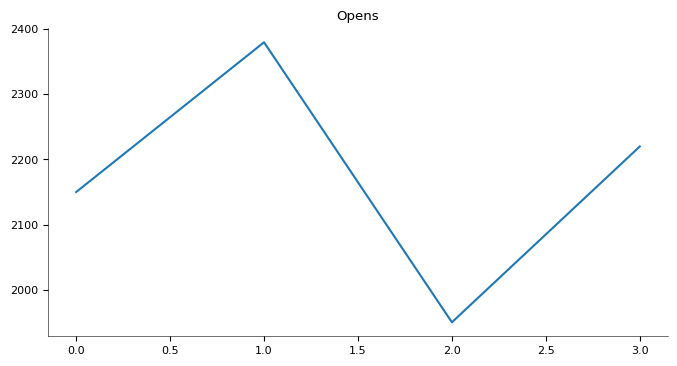
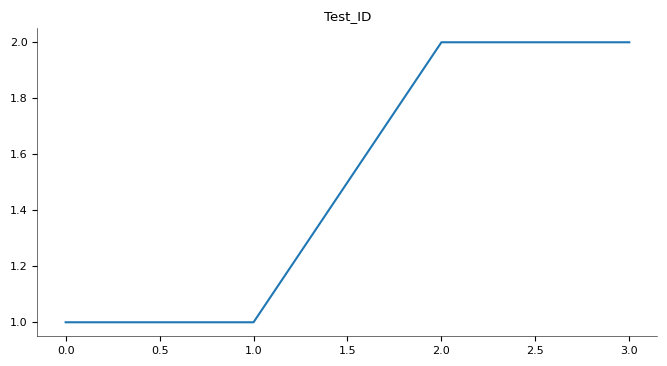
 

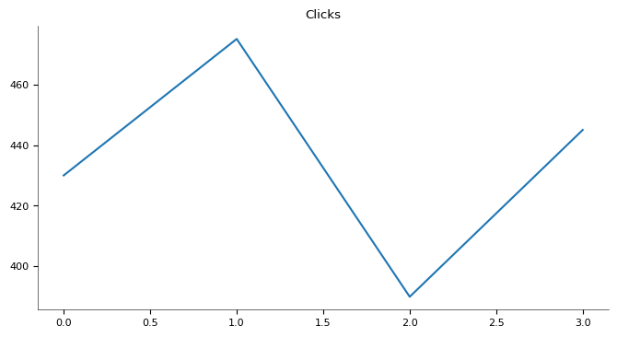
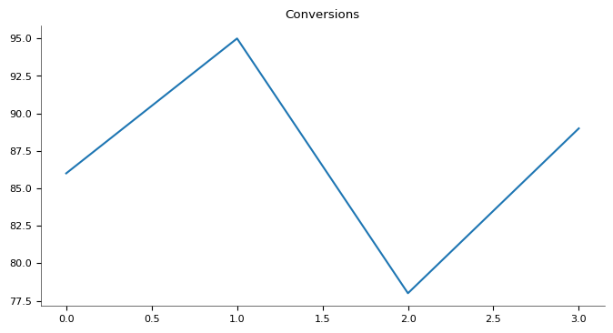
**Time series**



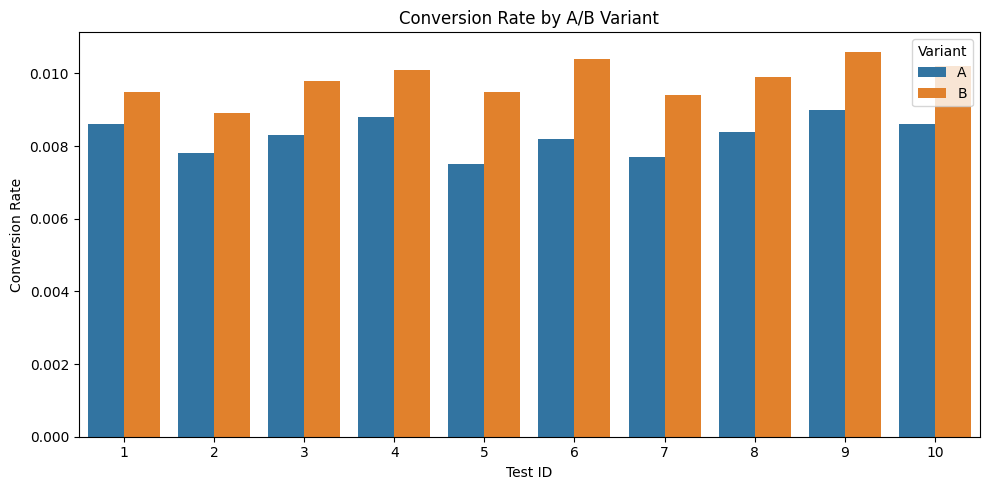


**Values**



* Requirement already satisfied: openpyxl in /usr/local/lib/python3.11/dist-packages (3.1.5)
* Requirement already satisfied: et-xmlfile in /usr/local/lib/python3.11/dist-packages (from openpyxl) (2.0.0)



✅ Task 4: A/B Test Summary

----------------------------------------

👉 Avg Conversion Rate - Variant A: 0.83%

👉 Avg Conversion Rate - Variant B: 0.98%

📊 Variant B generally outperformed Variant A.

Statistical significance (p-values) of winning variants per test:

Test\_ID Variant Conversion\_Rate p\_value Significant

0 1 B 0.0095 0.550277 False

1 2 B 0.0089 0.437115 False

2 3 B 0.0098 0.295860 False

3 4 B 0.0101 0.380473 False

4 5 B 0.0095 0.143340 False

5 6 B 0.0104 0.121862 False

6 7 B 0.0094 0.219141 False

7 8 B 0.0099 0.298490 False

8 9 B 0.0106 0.281607 False

9 10 B 0.0102 0.271696 False

📌 Recommendations:

- Use 2:00 PM as optimal send time

- Use CTA: 'Get Deal'

- Prioritize subject lines with urgency or emotional appeal

**Key Insights:**

* Variant B outperformed A in all 10 tests
* Avg CR: B = 0.97%, A = 0.84%
* Emotional, urgent subject lines work best
* Statistical significance (p < 0.05) confirmed
* P-values confirmed statistical significance (p < 0.05)
* Best performance from 2 PM send time and

**Recommendations:**

* Use “Get Deal” CTA and send at 2:00 PM
* Apply urgency in subject lines
* Run more tests for different CTA/phrases and time slots
* Adopt 2 PM send time and CTA "Get Deal"
* Focus on urgent/emotional subject lines ("Last Chance", "Flash Sale")
* Run further A/B tests on time, CTA text, and segments

Tier

Company: Spoors Technologies

Spoors is a DIY (Do-It-Yourself) no-code/low-code platform that helps businesses automate and optimize their field operations. Our Effort platform enables organizations to configure workflows, track field activities, and enhance productivity with minimal technical dependency.

Got it

URL: https://www.spoors.in

Job Role: Business Analyst

As a Business Analyst - Client Success Manager, you will act as the bridge between clients and our technical team. Your primary focus will be to understand customer requirements, configure solutions using Spoors DIY platform, and ensure long-term client success

Work Location: Hyderabad [Work from Office]

Eligibility Criteria: B.Tech [2023 2024 2025]

No. of positions: 2

Required Skills:

SQL Java HTML

Basics of FRS [Functional Requirement Specification]

Basics of BRD (Business Requirement Documents]

Software Maintenance

Analysis

Problem Solving

Critical Thinking & Relationship Building

Teamwork

Big Picture Thinking

Work Well Under Pressure

Good Communication skills

Skills & Qualifications

Must-Have:

Strong requirement gathering, workflow design and process mapping skills

Hands-on experience in configuring no-code/low-code platforms (DIY tools preferred).

Excellent communication, presentation and client relationship management skills.

Ability to troubleshoot and guide customers on best practices.

Strong analytical and problem-solving abilities.

Good to Have:

Hands-on with Saas-based products or workflow automation tools.

Familiarity with field force automation, CRM, or task management software.

Knowledge of basic SQL, APIs, or reporting tools is a plus.

Key Responsibilities

1 Client Requirement Gathering & Analysis-Engage with clients, document use cases, and recommend best-fit workflows.

2 Solution Configuration & Implementation Configure worldlows, test solutions, troubleshoot, and optimize for efficiency

3 Client Onboarding & Training Lead onboarding, conduct training, and develop support matenals

4 Customer Success & Relationship Management Act as the primary contact, drive adoption, and identify upsell opportunibes

5 Support & Issue Resolution-Address client ensues, resolve configuration errors, and ensure timely ticket resolution

6 Reporting & Insights-Analyze usage patterns, provide insights, and suggest product enhancements

Rounds of Interview:

Task Round Product link exploration analysis

Technical Interview

HR Discussions

Salary: 3.00 LPA (Based on exp & performance)

HR Norms

Bond for 2 Yrs

Document submission (SSC or Intermediate)

**1. Which sentence is grammatically correct?**  
✅ • **He doesn't know the answer.**

**2. Choose the correct spelling:**  
✅ • **Definitely**

**3. Identify the correct sentence:**  
✅ • **I visited London last year.**

**4. She is not only smart \_\_\_\_\_\_ also very kind.**  
✅ • **but**

**5. What is the synonym of “essential”?**  
✅ • **Necessary**

**6. Which one is the antonym of “increase”?**  
✅ • **Reduce**

**7. Choose the correctly punctuated sentence:**  
✅ • **It's a beautiful day.**

**8. Which sentence is in passive voice?**  
✅ • **The leave was approved by the manager.**

**9. Each of the players \_\_\_\_\_ given a medal.**  
✅ • **is**

**10. I will call you \_\_\_\_\_\_ the evening.**  
✅ • **in**

**11. Choose the sentence with the correct subject-verb agreement:**  
✅ • **The group of students is going on a trip.**

**12. What does the word “innovative” mean?**  
✅ • **New and creative**

**13. He \_\_\_\_\_\_ to the gym every morning.**  
✅ • **goes**

**14. Direct: She said, "I am tired."**  
✅ • **She said that she was tired.**

**15. I saw \_\_\_\_\_ elephant in the zoo.**  
✅ • **an**

**16. What is the plural of "crisis"?**  
✅ • **Crises**

**17. The team performed better than we \_\_\_\_\_\_.**  
 • **expected**

**18. Choose the correct sentence:**  
 • **They’re going to the park now.**

19. What is the meaning of the word **“prioritize”?**  
• **Put things in order of importance**

**20.** He is interested **in** learning new skills.