**Analysis Report**

1. **How many unique students are included in the dataset?**

# Counting the total number of students (rows) in the cleaned DataFrame

unique\_students = len(cleaned\_df)

# Displaying the total count of students

print(f"Unique students: {unique\_students}")

Output Cell:

Unique students: 4513

1. **What is the average GPA of the students?**

# Calculating the average GPA (CGPA) from the 'CGPA' column

average\_gpa = cleaned\_df['CGPA'].mean()

# Displaying the average GPA, formatted to 2 decimal places

print(f"Average GPA: {average\_gpa:.2f}")

Output Cell:

Average GPA: 8.05

1. **What is the distribution of students across different graduation years?**

# Getting the distribution of 'Year of Graduation' by counting occurrences and sorting by year

graduation\_year\_distribution = cleaned\_df['Year of Graduation'].value\_counts().sort\_index()

display(graduation\_year\_distribution.reset\_index())

# Creating a bar plot using seaborn with a 'viridis' color palette

ax = sns.barplot(x=graduation\_year\_distribution.index, y=graduation\_year\_distribution.values, palette="viridis")

# Setting the title of the plot

ax.set\_title('Distribution of Students Across Graduation Years')

# Setting the x-axis label

ax.set\_xlabel('Year of Graduation')

# Setting the y-axis label

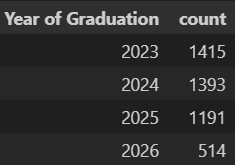
ax.set\_ylabel('Number of Students')

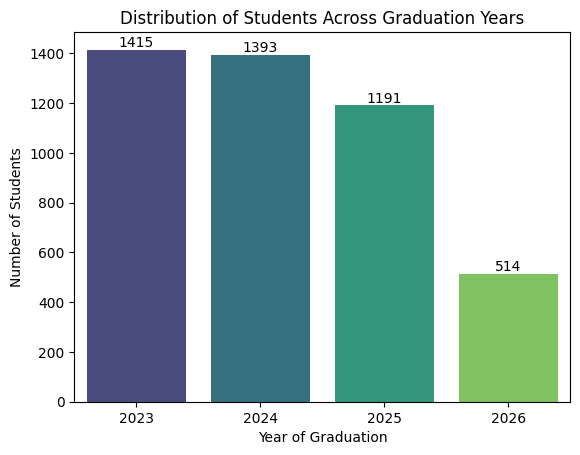
# Adding data labels on top of each bar in the bar plot

for i in ax.containers:

    ax.bar\_label(i)

Output Cell:





1. **What is the distribution of students' experience with Python programming?**

# Extracting the distribution of 'Experience with Python (Months)' from the cleaned dataset

python\_experience\_distribution = cleaned\_df['Experience with python (Months)'].value\_counts().sort\_index()

display(python\_experience\_distribution.reset\_index())

# Creating a bar plot using seaborn with a 'viridis' color palette

ax = sns.barplot(x=python\_experience\_distribution.index, y=python\_experience\_distribution.values, palette="viridis")

# Setting the title of the plot

ax.set\_title('Distribution of Students By Experience with Python')

# Setting the x-axis label

ax.set\_xlabel('Experience with Python (Months)')

# Setting the y-axis label

ax.set\_ylabel('Number of Students')

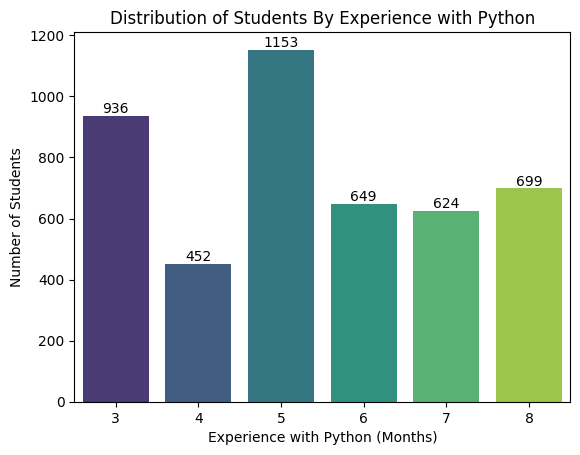
# Adding data labels on top of each bar in the bar plot

for i in ax.containers:

    ax.bar\_label(i)

Output Cell:





1. **What is the average family income of the students?**

# Calculating the average family income from the 'Family Income' column

average\_family\_income = cleaned\_df['Family Income'].mean()

# Printing the average family income rounded to two decimal places, with the unit in Lakhs

print(f"Average Family Income: {average\_family\_income:.2f} Lakh")

Output Cell:

Average Family Income: 1.34 Lakh

1. **How does the GPA vary between different colleges? (Show top 5 results only)**

# Calculate the top 5 colleges by average CGPA

top\_5\_colleges\_gpa = cleaned\_df.groupby('College Name')['CGPA'].mean().sort\_values(ascending=False).head(5)

# Convert the series into a DataFrame for better representation

top\_5\_colleges\_gpa\_df = top\_5\_colleges\_gpa.reset\_index()

# Rename the columns for better readability

top\_5\_colleges\_gpa\_df.columns = ['College Name', 'Average GPA']

# Print the list of top 5 colleges based on GPA

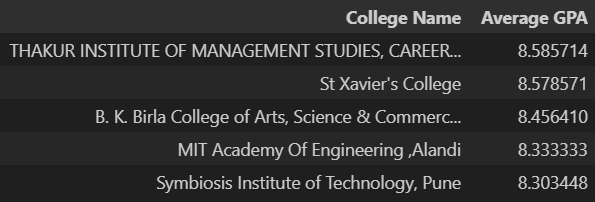
print("Top 5 colleges by GPA:")

# Display the DataFrame in a table format

display(top\_5\_colleges\_gpa\_df)

Output Cell:

Top 5 colleges by GPA:



1. **Are there any outliers in the quantity (number of courses completed) attribute?**

# Create a boxplot for the 'Quantity' column

cleaned\_df.boxplot(column='Quantity')

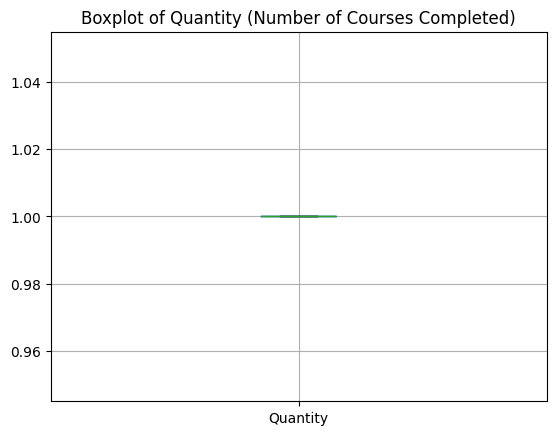
# Add a title to the plot

plt.title('Boxplot of Quantity (Number of Courses Completed)')

# Display the plot

plt.show()

Output Cell:



1. **What is the average GPA of students from each city? (Show top 5 results only)**

# Group by 'City' and calculate the average CGPA, then sort in descending order and take top 5

top\_5\_cities\_gpa = cleaned\_df.groupby('City')['CGPA'].mean().sort\_values(ascending=False).head(10)

print("Top 5 cities by Average GPA:")

# Reset the index and create a DataFrame for easier plotting and display

top\_5\_cities\_gpa\_df = top\_5\_cities\_gpa.reset\_index()

top\_5\_cities\_gpa\_df.columns = ['City', 'Average GPA']

# Display the DataFrame

display(top\_5\_cities\_gpa\_df)

# Plot the data in a bar plot

ax = sns.barplot(x='Average GPA', y='City', data=top\_5\_cities\_gpa\_df, palette="viridis")

# Set the title and axis labels for the plot

ax.set\_title('Top 10 Cities by Average GPA')

ax.set\_xlabel('Average GPA')

ax.set\_ylabel('City')

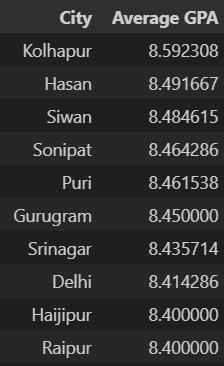
# Adding data labels on top of each bar in the bar plot

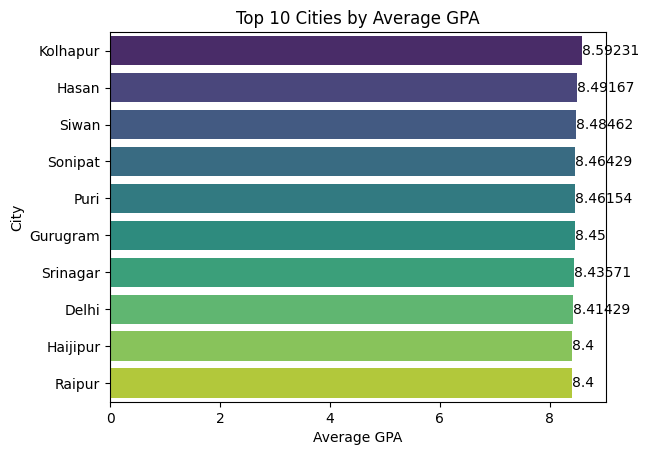
for i in ax.containers:

    ax.bar\_label(i)

Output Cell:

Top 5 cities by Average GPA:





1. **Can we identify any relationship between family income and GPA?**

# Create a boxplot to visualize the distribution of GPA across Family Income

ax = sns.boxplot(x='Family Income', y='CGPA', data=cleaned\_df, palette="viridis")

# Set the title and axis labels for the plot

ax.set\_title('Family Income vs. GPA')

ax.set\_xlabel('Family Income (Lakh)')

ax.set\_ylabel('GPA')

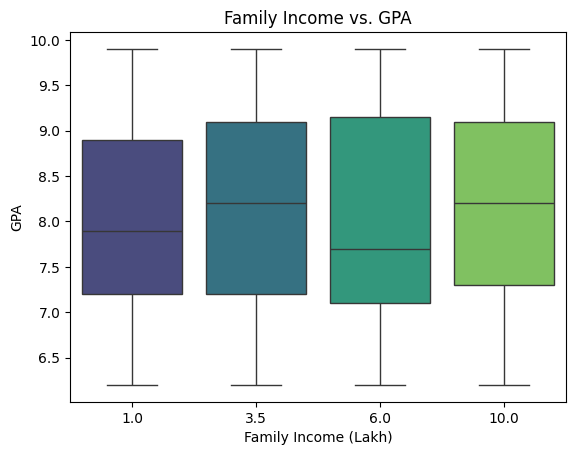
corr = cleaned\_df[['Family Income', 'CGPA']].corr()

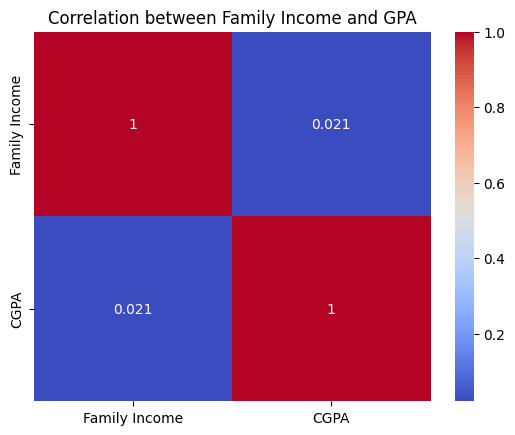
sns.heatmap(corr, annot=True, cmap='coolwarm')

plt.title('Correlation between Family Income and GPA')

plt.show()

Output Cell:





1. **How many students are from various cities? (Show top 5 results only)**

# Top 5 cities by student count

top\_5\_cities\_count = cleaned\_df['City'].value\_counts().head(10)

print("Top 5 cities by student count:")

# Display the top 5 cities in table format

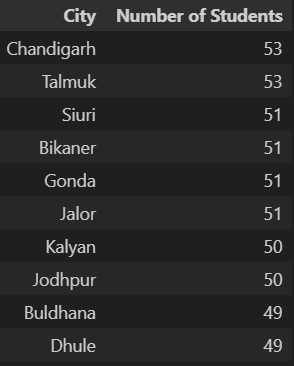
top\_5\_cities\_count\_df = top\_5\_cities\_count.reset\_index()

top\_5\_cities\_count\_df.columns = ['City', 'Number of Students']

display(top\_5\_cities\_count\_df)

Output Cell:

Top 5 cities by student count:



1. **How does the expected salary vary based on factors like 'GPA', 'Family income', 'Experience with Python (Months)'?**

# CGPA vs Expected Salary boxplot

plt.figure(figsize=(15, 6))

ax = sns.boxplot(x='CGPA', y='Expected salary (Lac)', data=cleaned\_df, palette="viridis")

ax.set\_title('GPA vs. Expected Salary')

ax.set\_xlabel('CGPA')

ax.set\_ylabel('Expected Salary (Lac)')

# Show the plot

plt.show()

# Family Income vs Expected Salary boxplot

ax = sns.boxplot(x='Family Income', y='Expected salary (Lac)', data=cleaned\_df, palette="viridis")

ax.set\_title('Family Income vs. Expected Salary')

ax.set\_xlabel('Family Income (Lakh)')

ax.set\_ylabel('Expected Salary (Lac)')

# Show the plot

plt.show()

# Experience with Python vs. Expected Salary

ax = sns.boxplot(x='Experience with python (Months)', y='Expected salary (Lac)', data=cleaned\_df, palette="viridis")

ax.set\_title('Experience with Python vs. Expected Salary')

ax.set\_xlabel('Experience with Python (Months)')

ax.set\_ylabel('Expected Salary (Lac)')

plt.show()

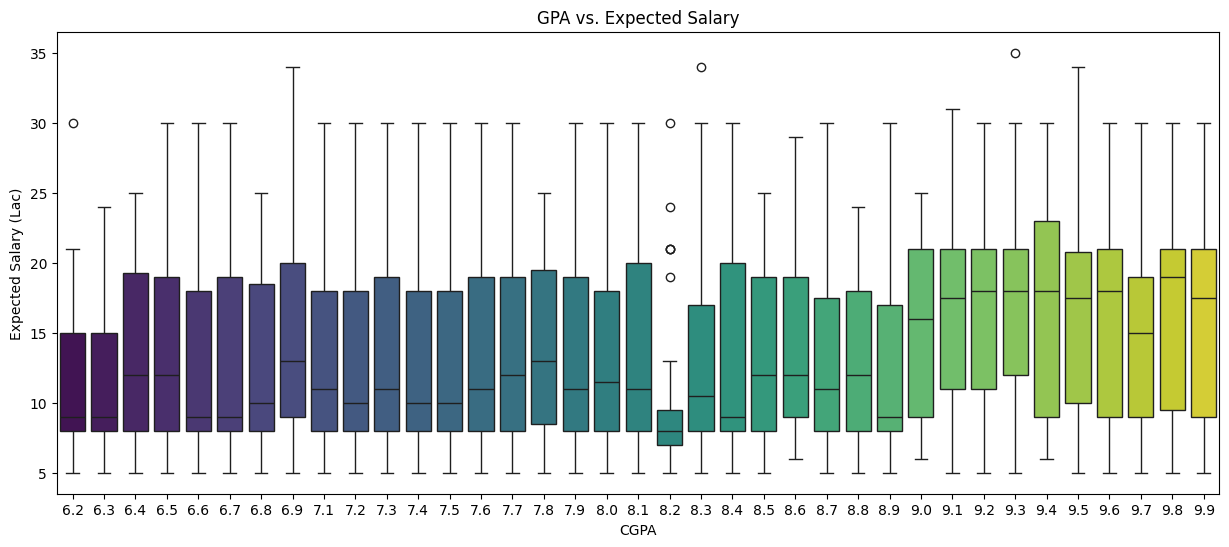
corr = cleaned\_df[['Expected salary (Lac)', 'Family Income', 'CGPA', 'Experience with python (Months)']].corr()

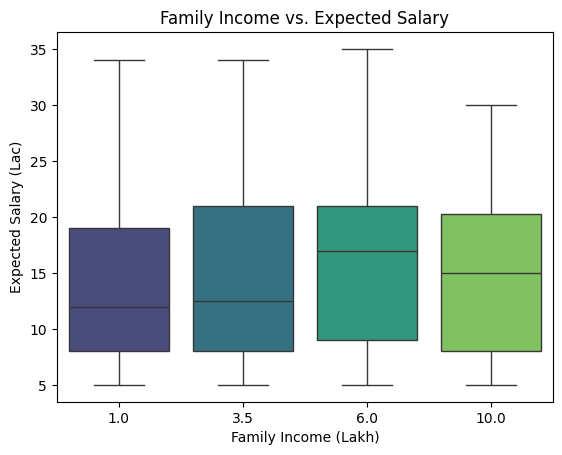
sns.heatmap(corr, annot=True, cmap='coolwarm')

plt.title('Correlation between Family Income and GPA')

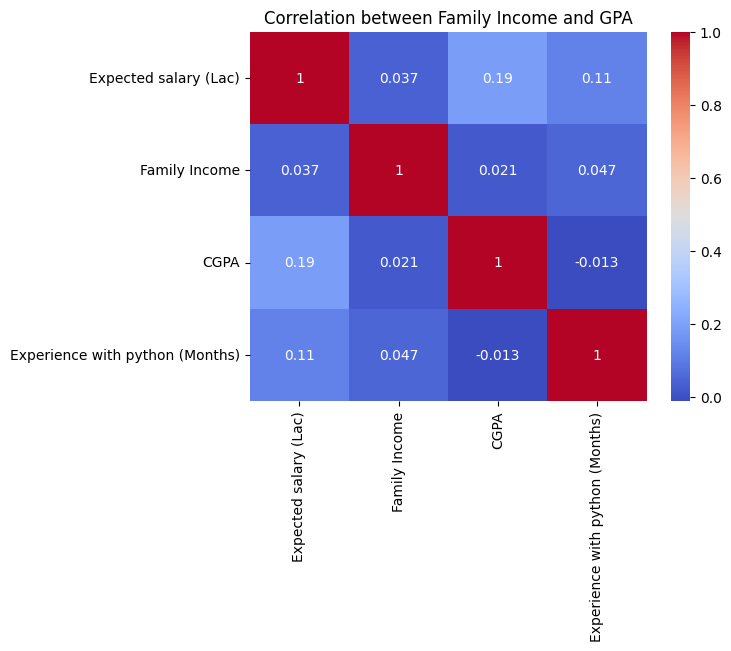
plt.show()

Output Cell:









1. **Which event tends to attract more students from specific fields of study? (Student distribution according to event)**

# Count the number of students for each event

event\_distribution = cleaned\_df['Events'].value\_counts()

print("Student distribution according to event:")

# Convert the distribution to a DataFrame

event\_distribution\_df = event\_distribution.reset\_index()

event\_distribution\_df.columns = ['Events', 'Number of Students']

display(event\_distribution\_df)

# Plot the distribution

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

ax = sns.barplot(x='Number of Students', y='Events', data=event\_distribution\_df, palette="viridis")

ax.set\_title('Student Distribution According to Event')

ax.set\_xlabel('Number of Students')

ax.set\_ylabel('Events')

# Adding data labels on top of each bar in the bar plot

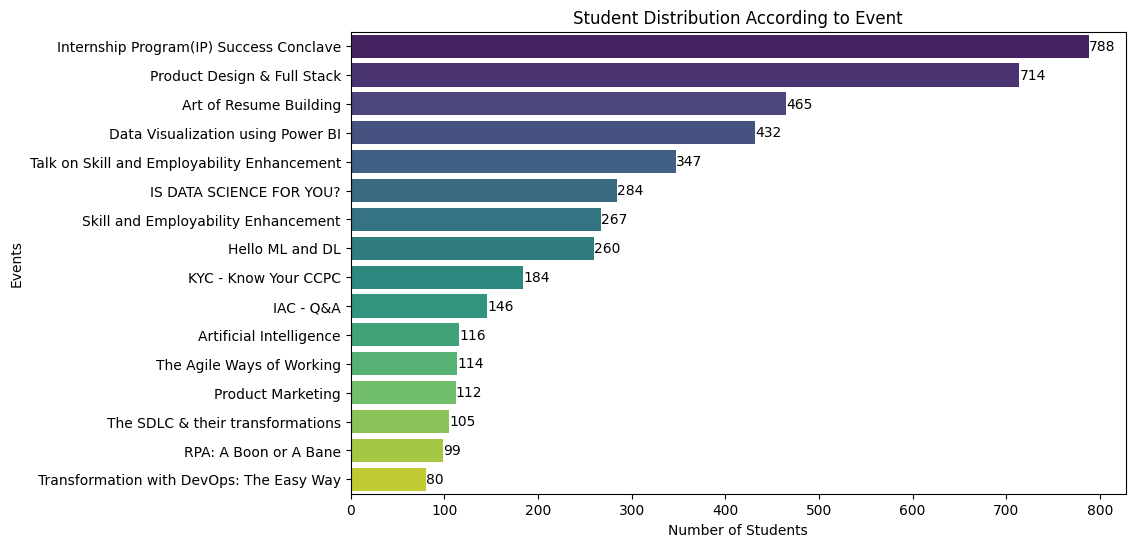
for i in ax.containers:

    ax.bar\_label(i)

Output Cell:

Student distribution according to event:





1. **Do students in leadership positions during their college years tend to have higher GPAs or better expected salary?**

# Box plot for GPA distribution by Leadership Skills

leadership\_gpa = cleaned\_df[cleaned\_df['Leadership- skills'].str.lower() == 'yes']['CGPA']

no\_leadership\_gpa = cleaned\_df[cleaned\_df['Leadership- skills'].str.lower() == 'no']['CGPA']

plt.boxplot([leadership\_gpa, no\_leadership\_gpa], labels=['Leadership', 'No Leadership'])

plt.title('GPA Distribution by Leadership Skills')

plt.ylabel('GPA')

plt.show()

# Box plot for Expected Salary distribution by Leadership Skills

leadership\_salary = cleaned\_df[cleaned\_df['Leadership- skills'].str.lower() == 'yes']['Expected salary (Lac)']

no\_leadership\_salary = cleaned\_df[cleaned\_df['Leadership- skills'].str.lower() == 'no']['Expected salary (Lac)']

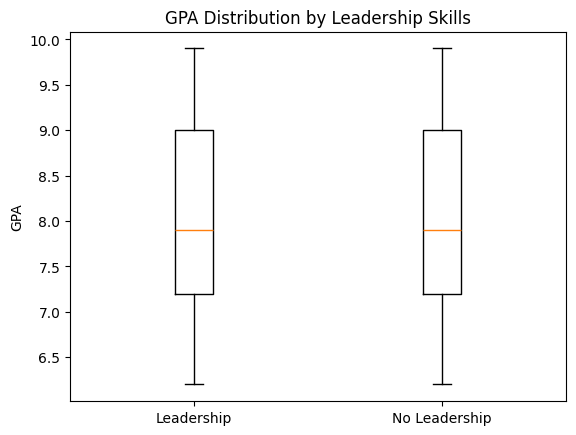
plt.boxplot([leadership\_salary, no\_leadership\_salary], labels=['Leadership', 'No Leadership'])

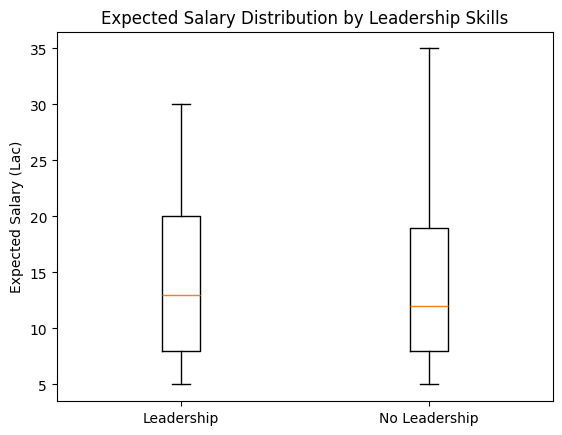
plt.title('Expected Salary Distribution by Leadership Skills')

plt.ylabel('Expected Salary (Lac)')

plt.show()

Output Cell:





1. **How many students are graduating by the end of 2024?**

# Count the number of unique students graduating by the end of 2024

graduating\_by\_2024 = cleaned\_df[cleaned\_df['Year of Graduation'] <= 2024]

print(f"Students graduating by the end of 2024: {len(graduating\_by\_2024)}")

Output Cell:

Students graduating by the end of 2024: 2808

1. **Which promotion channel brings in more student participation for the event?**

# Count the number of students for each promotion channel

promotion\_channel\_distribution = cleaned\_df['How did you come to know about this event?'].value\_counts()

print("Distribution of students according to promotion channel:")

# Create a DataFrame to display the distribution in table format

promotion\_channel\_df = promotion\_channel\_distribution.reset\_index()

promotion\_channel\_df.columns = ['Promotion Channel', 'Number of Students']

display(promotion\_channel\_df)

# Plot the distribution of students by promotion channel

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

ax = sns.barplot(x='Number of Students', y='Promotion Channel', data=promotion\_channel\_df, palette="viridis")

ax.set\_title('Student Participation by Promotion Channel')

ax.set\_xlabel('Number of Students')

ax.set\_ylabel('Promotion Channel')

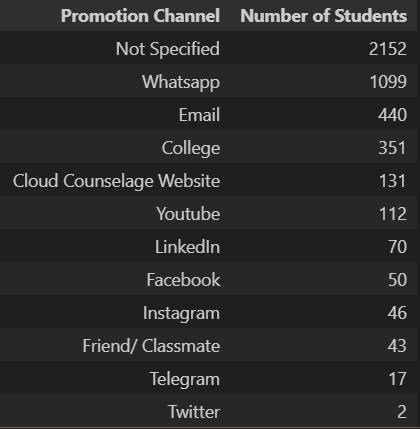
# Adding data labels on top of each bar in the bar plot

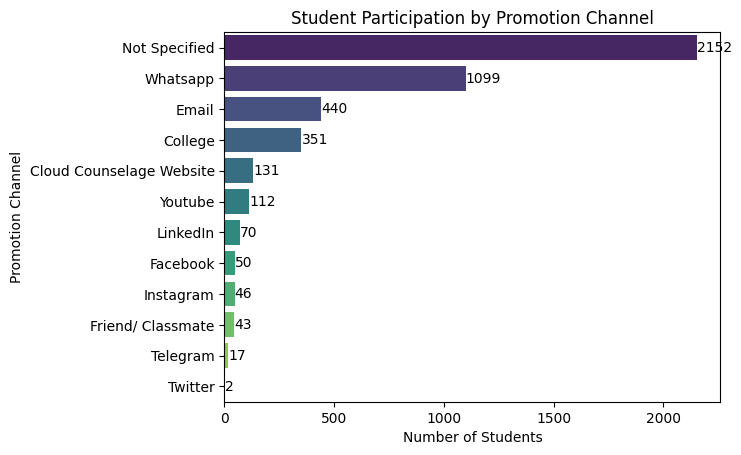
for i in ax.containers:

    ax.bar\_label(i)

Output Cell:

Distribution of students according to promotion channel:





1. **Find the total number of students who attended the events related to Data Science?**

# List of Data Science-related events

data\_science\_events = ['Data Visualization using Power BI', 'Artificial Intelligence', 'Hello ML and DL', 'IS DATA SCIENCE FOR YOU?']

# Filter the DataFrame to include only the rows where the 'Events' column matches any of the Data Science events

data\_science\_students\_df = cleaned\_df[cleaned\_df['Events'].isin(data\_science\_events)]

# Count the number of unique students (based on 'Email ID') who attended the Data Science events

data\_science\_students = len(data\_science\_students\_df)

# Print the total number of unique students who attended Data Science events

print(f"Total number of students who attended Data Science events: {data\_science\_students}")

Output Cell:

Total number of students who attended Data Science events: 1092

1. **Those students who have high CGPA & more experience in Python also have high expectations for salary? (Avg)**

# Filter students with CGPA of 8 or above and Python experience of 8 months or more

high\_cgpa\_experience = cleaned\_df[

    (cleaned\_df['CGPA'] >= 8) & (cleaned\_df['Experience with python (Months)'] >= 8)

]

# Calculate the average expected salary for the filtered students

average\_expected\_salary\_high = high\_cgpa\_experience['Expected salary (Lac)'].mean()

# Print the average expected salary for students with high CGPA and significant Python experience

print(f"Average expected salary for students with high CGPA and more Python experience: {average\_expected\_salary\_high:.2f} Lac")

Output Cell:

Average expected salary for students with high CGPA and more Python experience: 17.06 Lac

1. **How many students know about the event from their colleges? Which of these top 5 colleges?**

# Count the number of unique students who learned about the event from their colleges

college\_promotion\_distribution = cleaned\_df[cleaned\_df['How did you come to know about this event?'].str.contains('college', case=False, na=False)]

print(f"Students who knew about the event from their colleges: {len(college\_promotion\_distribution)}")

# Get the top 5 colleges by the number of students who learned about the event from their colleges

top\_5\_colleges\_promotion = college\_promotion\_distribution['College Name'].value\_counts().head(5)

print("Top 5 colleges by students who knew about the event from their colleges:")

# Create a DataFrame for the top 5 colleges with the number of students

college\_promotion\_df = top\_5\_colleges\_promotion.reset\_index()

college\_promotion\_df.columns = ['College Name', 'Number of Students']

display(college\_promotion\_df)

# Plot the number of students from the top 5 colleges who knew about the event through their colleges

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

ax = sns.barplot(x='Number of Students', y='College Name', data=college\_promotion\_df, palette="viridis")

ax.set\_title('Student Participation by College Promotion')

ax.set\_xlabel('Number of Students')

ax.set\_ylabel('College Name')

plt.show()

Output Cell:

Students who knew about the event from their colleges: 351

Top 5 colleges by students who knew about the event from their colleges:

