Analysis of Startups in 2015

In [1]:
```python
# Importing important liabraries
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```python
# Loading the dataset
df_2015 = pd.read_csv('C:/Users/Snehal/Downloads/2015_data.csv')
df_2015.head()
```

Out[2]:

	Sr.No	Date(dd/mm/yyyy)	Startup Name	Industry Vertical	City / Location	Investors’ Name	InvestmentType	Amount (in USD)	Remarks
0	1	01 September 2015	TOFlo	FinTech Startup Incubation platform	Mumbai	Tania Johny Palathinkal	Seed Funding	1,00,000	NaN
1	2	01 September 2015	FXMartIndia	Payment Services platform	Chandigarh	Flipkart	Private Equity	NaN	Strategic Investment (Majority Stake)
2	3	01 September 2015	Stylecracker	Personalized Styling platform	Mumbai	Group of HNI investors	Private Equity	10,00,000	Series A
3	4	01 September 2015	Luxuryhues	Luxury goods Shopping Platform	Gurgaon	Reliance Capital	Private Equity	9,00,000	Series A
4	5	02 September 2015	HolaChef	Food Delivery Platform	Mumbai	Ratan Tata	Private Equity	NaN	Part of Series A raised inJune 2015

Pre-processing

In [3]:
```python
# Renaming columns for our convinience

def renaming_columns(df_2015):
    df_2015.rename(columns={
    'Date(dd/mm/yyyy)': 'Date',
    'Startup Name': 'Startup_Name',
    'City / Location': 'Location',
    'Investors’ Name': 'Investors',
    'InvestmentType': 'Investment_Type',
    'Amount (in USD)': 'Amount($)',
    'Industry Vertical': 'Sub_Industry'
     }, inplace=True)
renaming_columns(df_2015)
```

In [4]:
```python
# Extracting required columns
df_2015 = df_2015[['Date', 'Startup_Name','Sub_Industry','Location','Investors','Investment_Type','Amount($)']]
```

In [5]:
```python
# Dealing with Date column to extract Year & Month
def date_opertion(df_2015):
    df_2015['Date'] = pd.to_datetime(df_2015['Date'], format="%d %B %Y")
    df_2015['Month'] = df_2015['Date'].dt.strftime('%B')
    df_2015['Year'] = df_2015['Date'].dt.year
date_opertion(df_2015)
```

In [6]:
```python
# Dealing with duplicate rows
def duplicate_rows(data):
    duplicate_rows = data[data.duplicated()]
    if len(duplicate_rows) > 0:
        data = data.drop_duplicates()
        print('Droped',len(duplicate_rows),'Duplicate Rows.')
    else:
        print('No Duplicate Rows.')
duplicate_rows(df_2015)
```

```
Droped 1 Duplicate Rows.
```

In [7]:
```python
# Dealing with Amount column data type
def amount_column(data):
    data['Amount($)'] = data['Amount($)'].fillna(0)
    data['Amount($)'] = data['Amount($)'].astype(str)
    data['Amount($)'] = data['Amount($)'].str.replace(',', '')
    data['Amount($)'] = data['Amount($)'].astype(int)
amount_column(df_2015)
```

In [8]:
```python
# Replacing values from location column which are not location
import numpy as np
df_2015['Location'] = df_2015['Location'].astype(str)
values_to_replace = ['Seed Funding', 'SeedFunding', 'PrivateEquity', 'Crowd funding', 'Crowd Funding','Private Equity']
df_2015['Location'] = df_2015['Location'].replace(values_to_replace, np.nan, regex=True)
```

In [9]:
```python
# Replacing odd values from Investors column
values_to_replace = [ '55,00,000', '1,20,000', '1,50,00,000', '10,00,000', '25,00,000',
       '12,00,000', '5,00,000', '1,50,000', '50,00,000', '2,00,00,000',
       '11,00,00,000', '2,00,000', '1,66,000', '13,00,000', '5,00,00,000',
       '20,00,000', '28,00,000', '30,00,000', '1,80,000', '60,00,000',
       '1,60,000', '2,50,00,000', '17,50,000', '1,00,000', '1,15,000',
       '3,00,00,000', '15,00,000', '1,00,00,000', '7,50,00,000',
       '1,60,00,000','21,50,000',
       '3,15,000', '3,80,000', '1,35,000', '2,85,000', '6,00,000',
       '1,25,000', '30,768', '1,30,00,000', '2,90,000', '6,50,000',
       '1,61,000', '10,00,00,000', '3,30,000', '16,000', '1,10,00,000',
       '1,47,50,000', '3,25,000', '32,50,000', '5,60,00,000',
       '3,10,00,000', '45,00,000', '8,25,000', '1,40,000', '16,600',
       '2,50,000', '2,60,00,000', '5,18,000', '6,00,00,000',
       '1,80,00,000', '8,00,00,000', '4,00,00,000', '1,65,00,000',
       '41,50,000', '8,00,000', '16,00,000', '3,20,000',]

df_2015['Investors'] = df_2015['Investors'].replace(values_to_replace, float('nan'))
```

In [10]:
```python
# Final DataFrame
df_2015.head()
```

Out[10]:

	Date	Startup_Name	Sub_Industry	Location	Investors	Investment_Type	Amount($)	Month	Year
0	2015-09-01	TOFlo	FinTech Startup Incubation platform	Mumbai	Tania Johny Palathinkal	Seed Funding	100000	September	2015
1	2015-09-01	FXMartIndia	Payment Services platform	Chandigarh	Flipkart	Private Equity	0	September	2015
2	2015-09-01	Stylecracker	Personalized Styling platform	Mumbai	Group of HNI investors	Private Equity	1000000	September	2015
3	2015-09-01	Luxuryhues	Luxury goods Shopping Platform	Gurgaon	Reliance Capital	Private Equity	900000	September	2015
4	2015-09-02	HolaChef	Food Delivery Platform	Mumbai	Ratan Tata	Private Equity	0	September	2015

Summary of the year 2015

- Shape = (938, 9)
- Unique Sub_Industry = 874
- Unique Location = 48
- Unique Investment_Type = 24

Graphs

In [11]:
```python
# Total Startup count in 2015
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import pandas as pd

def startup_count(data):
    startup_count = data['Startup_Name'].nunique()
    return go.Indicator(
        mode="number",
        value=startup_count,
        title="Startup Count")

fig = make_subplots(rows=1, cols=1)

fig.add_trace(startup_count(df_2015))

fig.update_layout(title_text="Startup Count in 2015")
fig.show()
```

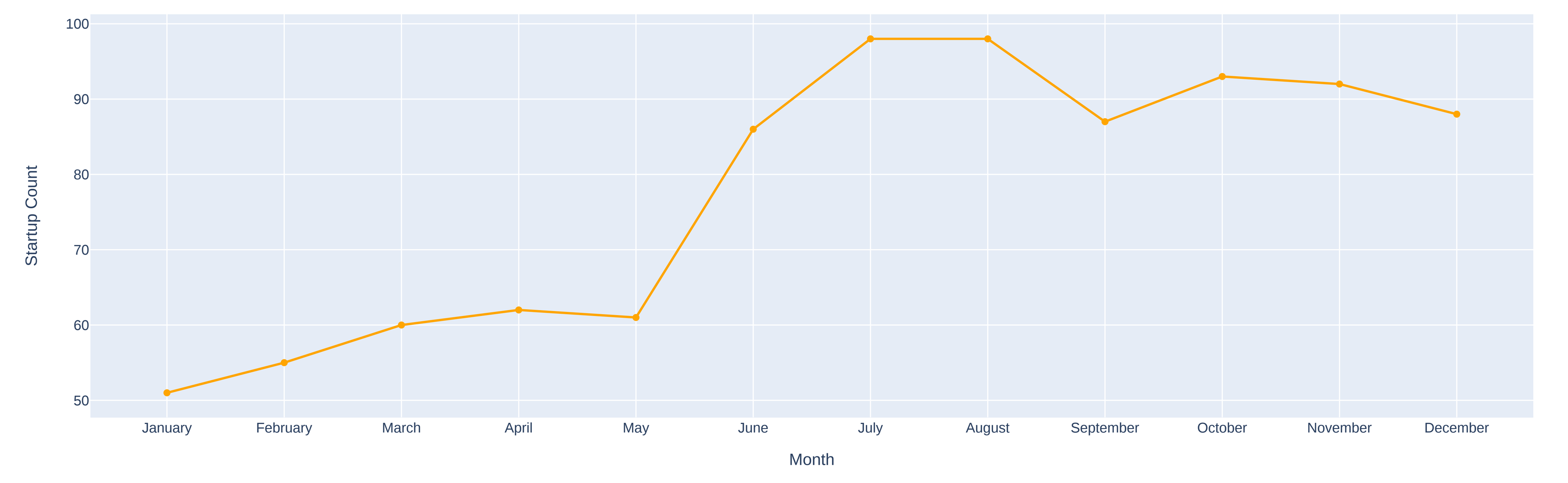
Startup Count in 2015

Startup Count

857

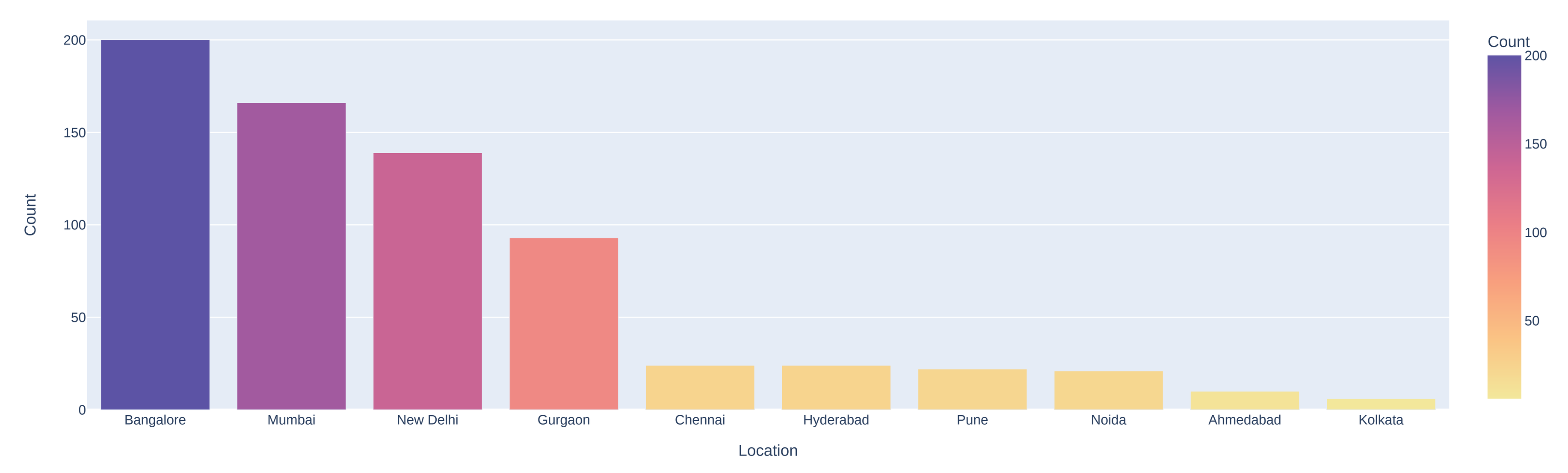
In [12]:
```python
import pandas as pd
import plotly.graph_objects as go

def monthly_startup_count(data):
    months_sequence = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']
    monthly_count = data.groupby('Month')['Startup_Name'].nunique().reindex(months_sequence, fill_value=0)
    fig = go.Figure()
    fig.add_trace(go.Scatter(x=monthly_count.index, y=monthly_count.values, mode='lines+markers', marker=dict(color='orange')))
    fig.update_layout(title='Monthly Startup Count', xaxis_title='Month', yaxis_title='Startup Count', xaxis={'categoryorder': 'array', 'categoryarray': months_sequence})
    fig.show()

monthly_startup_count(df_2015)
```

Monthly Startup Count



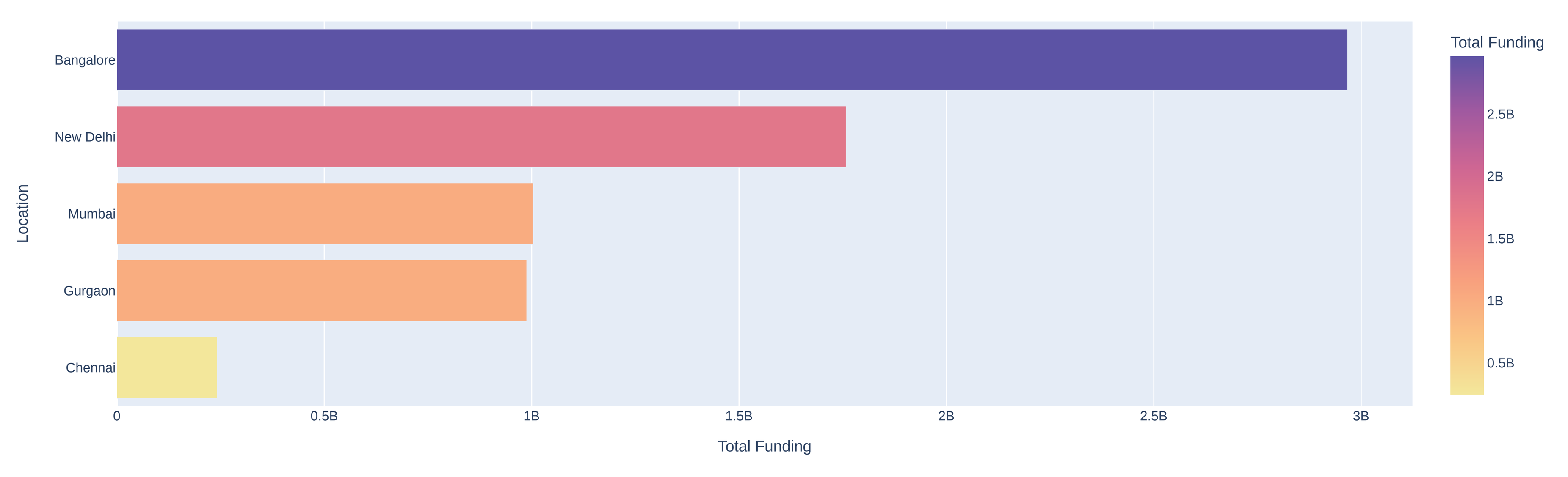
In [13]:
```python
import pandas as pd
import plotly.express as px

def top_10_location(data):
    location_counts = data.groupby('Location').size().reset_index(name='Count')
    top_10_locations = location_counts.sort_values(by='Count', ascending=False).head(10)
    colorscale = px.colors.sequential.Sunset
    fig = px.bar(top_10_locations, x='Location', y='Count',
             title='Top 10 Locations by Startups Count', color='Count', color_continuous_scale=colorscale)
    fig.update_layout(xaxis_title='Location', yaxis_title='Count')
    fig.show()

top_10_location(df_2015)
```

Top 10 Locations by Startups Count



In [14]:
```python
import pandas as pd
import plotly.express as px

def top_locations_funding(data, n=5):
    location_data = data.groupby('Location')['Amount($)'].sum().reset_index()
    top_locations = location_data.nlargest(n, 'Amount($)')
    top_locations = top_locations[::-1]
    colorscale = px.colors.sequential.Sunset

    fig = px.bar(top_locations, y='Location', x='Amount($)', orientation='h',
                 title='Top 5 Locations by Funding',
                 labels={'Location': 'Location', 'Amount($)': 'Total Funding'},
                 color='Amount($)', color_continuous_scale=colorscale)

    fig.show()

top_locations_funding(df_2015, n=5)
```

Top 5 Locations by Funding

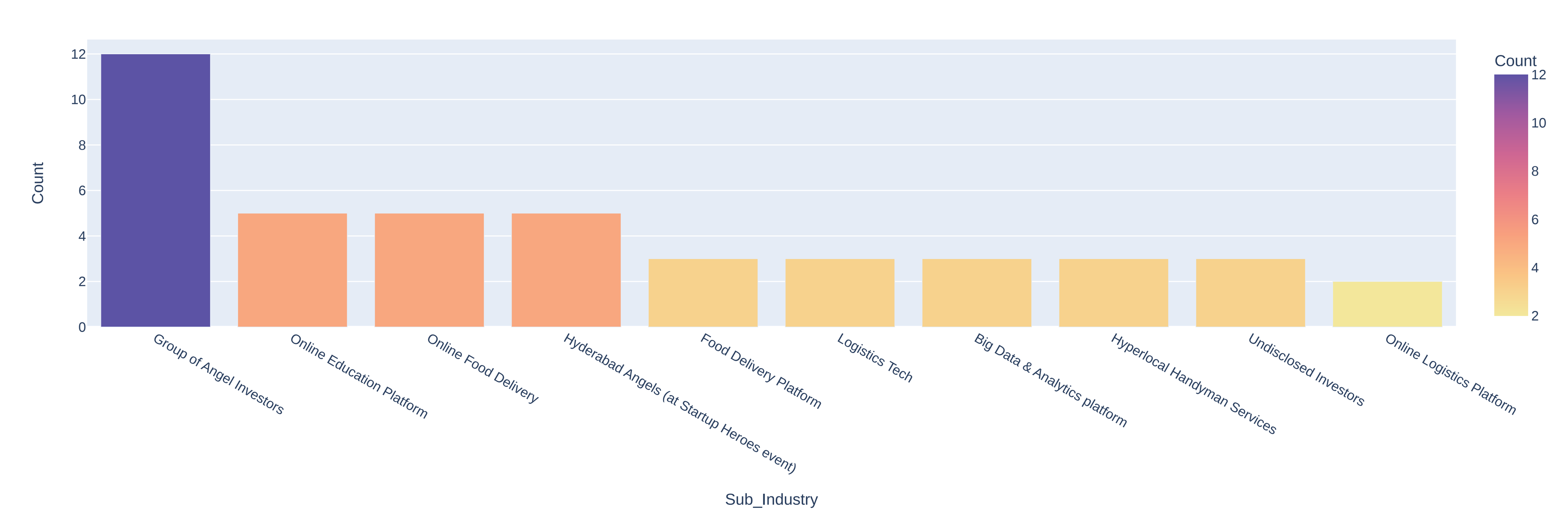


In [15]:
```python
# Location wise Startups

import pandas as pd
import ipywidgets as widgets
from IPython.display import display, HTML

def create_table(selected_location):
    if selected_location == 'All':
        display(HTML("<p>Select an option from the dropdown to view the table.</p>"))
    else:
        filtered_df = df_2015[df_2015['Location'] == selected_location]

        if filtered_df.empty:
            display(HTML("<p>No data available for the selected location.</p>"))
        else:
            trace = go.Table(
                header=dict(values=["Startup Name", "Sub Industry", "Investor", "Investment Type", "Amount($)", "Month"],
                            fill=dict(color='lightblue'),
                            align=['left', 'center']),
                cells=dict(values=[filtered_df['Startup_Name'],
                                   filtered_df['Sub_Industry'],
                                   filtered_df['Investors'],
                                   filtered_df['Investment_Type'],
                                   filtered_df['Amount($)'],
                                   filtered_df['Month']],
                           fill=dict(color=['white', 'lightgray']),
                           align=['left', 'center'])
            )

            layout = dict(width=1000, height=800)
            fig = go.Figure(data=[trace], layout=layout)
            display(fig)

location_dropdown = widgets.Dropdown(options=['All'] + sorted(df_2015['Location'].dropna().unique()), value='All', description='Location:')
widgets.interactive(create_table, selected_location=location_dropdown)
```

Out[15]:
```
interactive(children=(Dropdown(description='Location:', options=('All', 'Ahmedabad', 'Bangalore', 'Bangalore /…
```

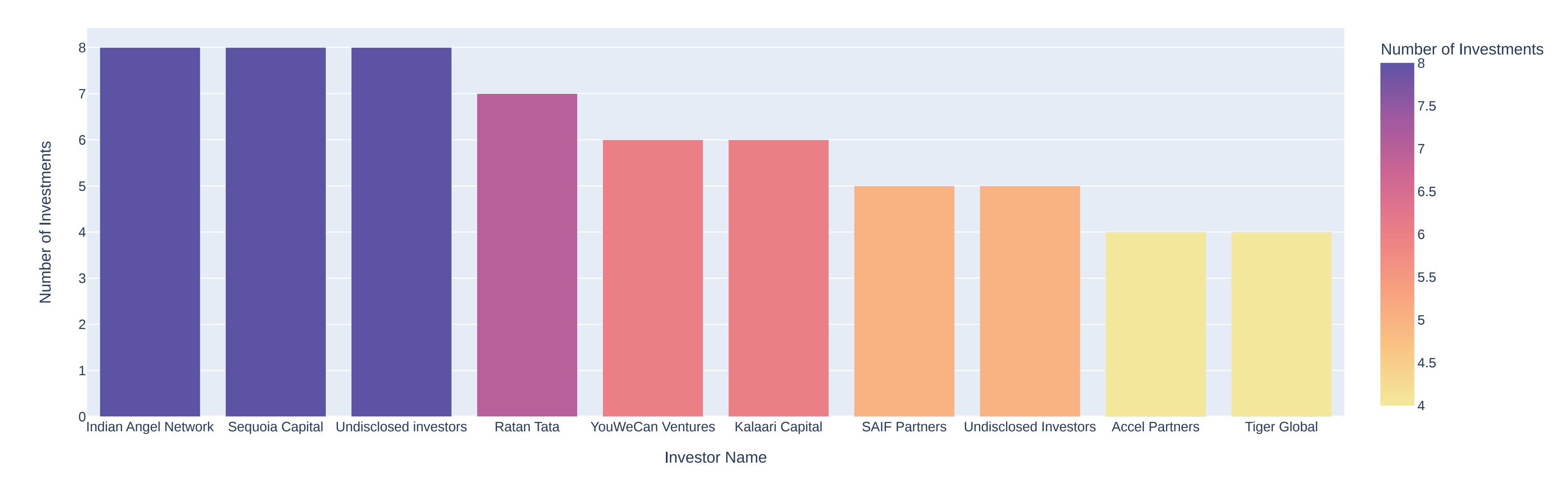
In [16]:
```python
# Top 10 Sub-Industries by Startups Count
def  sub_industry_top_10(data):
    top_sub_industries = data['Sub_Industry'].value_counts().nlargest(10)
    top_sub_df = top_sub_industries.reset_index()
    top_sub_df.columns = ['Sub_Industry', 'Count']
    colorscale = px.colors.sequential.Sunset
    fig = px.bar(top_sub_df, x='Sub_Industry', y='Count', title='Top 10 Sub-Industries by Startups Count',
                color = 'Count' , color_continuous_scale=colorscale)
    fig.update_xaxes(categoryorder='total descending')
    fig.show()
sub_industry_top_10(df_2015)
```

Top 10 Sub-Industries by Startups Count



In [17]:
```python
import pandas as pd
import plotly.express as px

def top_investors(data):
    top_investors = data['Investors'].value_counts().nlargest(10)
    top_investors_df = top_investors.reset_index()
    top_investors_df.columns = ['Investor', 'Investment Count']
    colorscale = px.colors.sequential.Sunset

    fig = px.bar(top_investors_df, x='Investor', y='Investment Count',
                 title='Top 10 Investors by Investment Count',
                 labels={'Investor': 'Investor Name', 'Investment Count': 'Number of Investments'},
                 color='Investment Count', color_continuous_scale=colorscale)
    fig.show()

top_investors(df_2015)
```

Top 10 Investors by Investment Count



In [ ]: