ASSIGNMENT 1

Question 1

Flipkart Sales Data Analysis and Visualization

Load the dataset into a Pandas Data Frame.

In [4]:
```
import pandas as pd
file_path = 'Flipkart-Laptops.xlsx'
df = pd.read_excel(file_path)
print(df)
```

```
                                          Product Name         ProductID  \
0    MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...  COMGZW35W3DSJADN
1    MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...  COMGZW37ZX66DBHF
2    DELL Inspiron 3520 Intel Core i3 12th Gen 1215...  COMGJ75HJGFDJ6JN
3    Acer One (2024) Intel Core i3 11th Gen 1115G4 ...  COMGPF5CQ7VDWDT4
4    Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...  COMGPYKZAWY8UX6C
..                                                 ...               ...
955  Acer Swift Go 14 (2024) AI Powered EVO Intel C...  COMGWKF2VKGAVHDU
956  HP Victus Intel Core i5 12th Gen 12450H - (16 ...  COMH2DYZHMHZ5UPG
957  Infinix X1 Slim Series (2024) Intel Core i3 10...  COMGEHP5EFEGWZW5
958  Lenovo IdeaPad Slim 3 Intel Core i5 12th Gen 1...  COMGYHP5ZB4AGZH6
959  HP (15s-fq5007TU) Intel Core i3 12th Gen 1215U...  COMGYHP5MCEYZHSV

     Product image Actual price Discount price Stars         Rating  \
0              NaN        89990          54990   3.9      7 Ratings
1              NaN        83990          67990   NIL            NIL
2              NaN        49240          35660   4.2  1,805 Ratings
3              NaN        43999          26990   4.2  6,977 Ratings
4              NaN        59400          27989   4.2  1,263 Ratings
..             ...          ...            ...   ...           ...
955            NaN       129999          79990   4.1    108 Ratings
956            NaN          NIL          82414   NIL            NIL
957            NaN        49999          32990   4.3  3,897 Ratings
958            NaN        69890          53390   3.8     53 Ratings
959            NaN        51134          38990   4.2  5,540 Ratings

         Reviews                                        Description  \
0      1 Reviews  Intel Core i5 Processor (12th Gen)16 GB DDR5 R...
1            NIL  Intel Core i7 Processor (12th Gen)8 GB DDR4 RA...
2    143 Reviews  Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...
3    596 Reviews  Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...
4    113 Reviews  AMD Ryzen 3 Quad Core Processor8 GB LPDDR5 RAM...
..           ...                                                ...
955   16 Reviews  Intel Core Ultra 5 Processor16 GB LPDDR5X RAMW...
956          NIL  Intel Core i5 Processor (12th Gen)16 GB DDR4 R...
957  457 Reviews  Intel Core i3 Processor (10th Gen)8 GB LPDDR4X...
958    5 Reviews  Intel Core i5 Processor (12th Gen)16 GB LPDDR5...
959  485 Reviews  Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...

                                                  Link
0    https://www.flipkart.com/msi-cyborg-15-intel-c...
1    https://www.flipkart.com/msi-thin-15-intel-cor...
2    https://www.flipkart.com/dell-inspiron-3520-in...
3    https://www.flipkart.com/acer-one-2024-intel-c...
4    https://www.flipkart.com/lenovo-v15-amd-ryzen-...
..                                                 ...
955  https://www.flipkart.com/acer-swift-go-14-2024...
956  https://www.flipkart.com/hp-victus-intel-core-...
957  https://www.flipkart.com/infinix-x1-slim-serie...
958  https://www.flipkart.com/lenovo-ideapad-slim-3...
959  https://www.flipkart.com/hp-15s-fq5007tu-intel...

[960 rows x 10 columns]
```

Display the first and last 5 rows of the dataset.

In [15]:
```
print(df.head(5))
print(df.tail(5))
```

```
                                        Product Name  ...
Link
0  MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...  ...  https://www.flipkart.com/msi-cyborg-15
-intel-c...
1  MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...  ...  https://www.flipkart.com/msi-thin-15-i
ntel-cor...
2  DELL Inspiron 3520 Intel Core i3 12th Gen 1215...  ...  https://www.flipkart.com/dell-inspiron
-3520-in...
3  Acer One (2024) Intel Core i3 11th Gen 1115G4 ...  ...  https://www.flipkart.com/acer-one-2024
-intel-c...
4  Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...  ...  https://www.flipkart.com/lenovo-v15-am
d-ryzen-...

[5 rows x 10 columns]
                                          Product Name  ...
Link
955  Acer Swift Go 14 (2024) AI Powered EVO Intel C...  ...  https://www.flipkart.com/acer-swift-
go-14-2024...
956  HP Victus Intel Core i5 12th Gen 12450H - (16 ...  ...  https://www.flipkart.com/hp-victus-i
ntel-core-...
957  Infinix X1 Slim Series (2024) Intel Core i3 10...  ...  https://www.flipkart.com/infinix-x1-
slim-serie...
958  Lenovo IdeaPad Slim 3 Intel Core i5 12th Gen 1...  ...  https://www.flipkart.com/lenovo-idea
pad-slim-3...
959  HP (15s-fq5007TU) Intel Core i3 12th Gen 1215U...  ...  https://www.flipkart.com/hp-15s-fq50
07tu-intel...

[5 rows x 10 columns]
```

Calculate the total sales for each Company.*

In [52]:

```
print(df.columns)
# total act - disc
df['Sales'] = df['Actual price'] - df['Discount price']

# print(df)

print("\nTotal Sales by Company:")
print(df[['Product Name','Actual price','Discount price','Sales']])
```

```
Index(['Product Name', 'ProductID', 'Product image', 'Actual price',
       'Discount price', 'Stars', 'Rating', 'Reviews', 'Description', 'Link',
       'Rating Category', 'Sales'],
      dtype='object')

Total Sales by Company:
                                          Product Name  Actual price  \
0    MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...       89990.0
1    MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...       83990.0
2    DELL Inspiron 3520 Intel Core i3 12th Gen 1215...       49240.0
3    Acer One (2024) Intel Core i3 11th Gen 1115G4 ...       43999.0
4    Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...       59400.0
..                                                 ...           ...
955  Acer Swift Go 14 (2024) AI Powered EVO Intel C...      129999.0
956  HP Victus Intel Core i5 12th Gen 12450H - (16 ...           NaN
957  Infinix X1 Slim Series (2024) Intel Core i3 10...       49999.0
958  Lenovo IdeaPad Slim 3 Intel Core i5 12th Gen 1...       69890.0
959  HP (15s-fq5007TU) Intel Core i3 12th Gen 1215U...       51134.0

    Discount price    Sales
0            54990  35000.0
1            67990  16000.0
2            35660  13580.0
3            26990  17009.0
4            27989  31411.0
..             ...      ...
955          79990  50009.0
956          82414      NaN
957          32990  17009.0
958          53390  16500.0
959          38990  12144.0

[960 rows x 4 columns]
```

Display the product name that has price greater than 50000 and less than 80000. *

In [36]:
```
df['Actual price'] = pd.to_numeric(df['Actual price'], errors='coerce')
ranged_products = df[(df['Actual price'] > 50000) & (df['Actual price'] < 80000)]
print(ranged_products['Product Name'],ranged_products['Actual price'])
```

```
4      Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...
5      Lenovo AMD Ryzen 3 Quad Core 7330U - (8 GB/512...
6      HP FQ Series Intel Core i3 12th Gen 1215U - (8...
12     HP AMD Ryzen 5 Hexa Core 5500U - (16 GB/512 GB...
19     HP 2023 Intel Core i3 12th Gen 1215U - (8 GB/5...
                           ...
950    DELL Intel Core i3 13th Gen 1305U - (8 GB/512 ...
951    HP Pavilion AMD Ryzen 5 Hexa Core AMD R5-5600H...
952    HP Intel Core i3 12th Gen 1215U - (16 GB/512 G...
958    Lenovo IdeaPad Slim 3 Intel Core i5 12th Gen 1...
959    HP (15s-fq5007TU) Intel Core i3 12th Gen 1215U...
Name: Product Name, Length: 444, dtype: object 4      59400.0
5      63900.0
6      50843.0
12     59109.0
19     51266.0
        ...
950    51944.0
951    73544.0
952    52721.0
958    69890.0
959    51134.0
Name: Actual price, Length: 444, dtype: float64
```

Count total number of products which has more than 3000 ratings.

In [34]:
```
df['Rating'] = df['Rating'].astype(str)

df['Rating'] = pd.to_numeric(df['Rating'].str.replace(',', ''), errors='coerce')

ratings_above_3000 = df[df['Rating'] > 3000]

print("Ratings greater than 3000:")
print(ratings_above_3000[['Rating','Product Name']])
```

```
Ratings greater than 3000:
     Rating                                       Product Name
3    6977.0  Acer One (2024) Intel Core i3 11th Gen 1115G4 ...
9    4029.0  HP Laptop AMD Ryzen 3 Quad Core 5300U - (8 GB/...
12   3246.0  HP AMD Ryzen 5 Hexa Core 5500U - (16 GB/512 GB...
19   3102.0  HP 2023 Intel Core i3 12th Gen 1215U - (8 GB/5...
22   7151.0  HP AMD Ryzen 5 Hexa Core 5500U - (8 GB/512 GB ...
..      ...                                              ...
939  4238.0  Lenovo IdeaPad Slim 1 (2024) AMD Ryzen 5 Hexa ...
944  4624.0  HP 15s AMD Ryzen 3 Dual Core 3250U - (8 GB/1 T...
952  5540.0  HP Intel Core i3 12th Gen 1215U - (16 GB/512 G...
957  3897.0  Infinix X1 Slim Series (2024) Intel Core i3 10...
959  5540.0  HP (15s-fq5007TU) Intel Core i3 12th Gen 1215U...

[159 rows x 2 columns]
```

Display the product name which has maximum and minimum review count.

In [56]:
```
# df['Reviews'] = df['Reviews'].astype(str)
print(f"\nDisplay the Product having maximum reviews \n{df.loc[df['Reviews'].idxmax()]}")
print(f"\nDisplay the Product having minimum reviews \n{df.loc[df['Reviews'].idxmin()]}")
```

```
Display the Product having maximum reviews
Product Name       realme Book(Slim) Intel Evo Intel Core i5 11th...
ProductID                                              COMG5YDPM8FZZWMQ
Product image                                                   NaN
Actual price                                                69999.0
Discount price                                                47999
Stars                                                           4.3
Rating                                                       7975.0
Reviews                                                      1042.0
Description        Powered by 11th Gen Intel Evo Core i5 Processo...
Link               https://www.flipkart.com/realme-book-slim-inte...
Rating Category                                             Excellent
Sales                                                        22000.0
Name: 419, dtype: object

Display the Product having minimum reviews
Product Name       Lenovo AMD Ryzen 3 Quad Core 7330U - (8 GB/512...
ProductID                                              COMGYG23H3CRABUT
Product image                                                   NaN
Actual price                                                63900.0
Discount price                                                32990
Stars                                                           5.0
Rating                                                          7.0
Reviews                                                         0.0
Description        AMD Ryzen 3 Quad Core Processor8 GB DDR4 RAMWi...
Link               https://www.flipkart.com/lenovo-amd-ryzen-3-qu...
Rating Category                                             Excellent
Sales                                                        30910.0
Name: 5, dtype: object
```

Show the statistical analysis of discount prices.

In [152…]
```
discount_stats = df['Discount price'].describe()
print(discount_stats)
# print(df.columns)
```

```
count       960
unique      294
top       54990
freq         47
Name: Discount price, dtype: int64
```

Show how many product having poor rating(1-2), Average rating (2-3), Good rating (3-4) and Excellent rating (4-5).

In [10]:
```
df['Stars'] = df['Stars'].astype(str)


df['Stars'] = pd.to_numeric(df['Stars'], errors='coerce')


def categorize_rating(rating):
    if pd.isna(rating):
        return 'Unknown'
    elif 1 <= rating < 2:
        return 'Poor'
    elif 2 <= rating < 3:
        return 'Average'
    elif 3 <= rating < 4:
```

```
        return 'Good'
    elif 4 <= rating <= 5:
        return 'Excellent'
    else:
        return 'Unknown'


df['Rating Category'] = df['Stars'].apply(categorize_rating)


rating_counts = df['Rating Category'].value_counts()

print("Product counts by rating category:")
print(rating_counts)
```

```
Product counts by rating category:
Rating Category
Excellent    640
Good         161
Unknown      150
Average        9
Name: count, dtype: int64
```

Treat Nil values as missing values and update it.

In [12]:

```
print("Original DataFrame:")
print(df.head())
print("\nOriginal 'Reviews' column:")
print(df['Reviews'].head())


df['Reviews'] = df['Reviews'].astype(str)


df['Reviews'].replace('NIL', '0 Reviews', inplace=True)


print("\nDataFrame after replacing 'NIL' with '0 Reviews':")
print(df.head())
print("\n'Reviews' column after replacing 'NIL' with '0 Reviews':")
print(df['Reviews'].head())


df['Reviews'] = df['Reviews'].str.replace(' Reviews', '', regex=False).str.replace(',', '', rege


df['Reviews'] = pd.to_numeric(df['Reviews'], errors='coerce')


df['Reviews'].fillna(method='ffill', inplace=True)




print("\nDataFrame after handling missing values:")
print(df.head())
print("\n'Reviews' column after handling missing values:")
print(df['Reviews'].head())
```

```
Original DataFrame:
                                        Product Name         ProductID  \
0  MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...  COMGZW35W3DSJADN
1  MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...  COMGZW37ZX66DBHF
2  DELL Inspiron 3520 Intel Core i3 12th Gen 1215...  COMGJ75HJGFDJ6JN
3  Acer One (2024) Intel Core i3 11th Gen 1115G4 ...  COMGPF5CQ7VDWDT4
4  Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...  COMGPYKZAWY8UX6C

   Product image  Actual price Discount price  Stars         Rating  Reviews  \
0            NaN       89990.0          54990    3.9      7 Ratings      1.0
1            NaN       83990.0          67990    NaN            NIL      NaN
2            NaN       49240.0          35660    4.2  1,805 Ratings    143.0
3            NaN       43999.0          26990    4.2  6,977 Ratings    596.0
4            NaN       59400.0          27989    4.2  1,263 Ratings    113.0

                                         Description  \
0  Intel Core i5 Processor (12th Gen)16 GB DDR5 R...
1  Intel Core i7 Processor (12th Gen)8 GB DDR4 RA...
2  Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...
3  Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...
4  AMD Ryzen 3 Quad Core Processor8 GB LPDDR5 RAM...

                                                Link Rating Category
0  https://www.flipkart.com/msi-cyborg-15-intel-c...            Good
1  https://www.flipkart.com/msi-thin-15-intel-cor...         Unknown
2  https://www.flipkart.com/dell-inspiron-3520-in...       Excellent
3  https://www.flipkart.com/acer-one-2024-intel-c...       Excellent
4  https://www.flipkart.com/lenovo-v15-amd-ryzen-...       Excellent

Original 'Reviews' column:
0      1.0
1      NaN
2    143.0
3    596.0
4    113.0
Name: Reviews, dtype: float64

DataFrame after replacing 'NIL' with '0 Reviews':
                                        Product Name         ProductID  \
0  MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...  COMGZW35W3DSJADN
1  MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...  COMGZW37ZX66DBHF
2  DELL Inspiron 3520 Intel Core i3 12th Gen 1215...  COMGJ75HJGFDJ6JN
3  Acer One (2024) Intel Core i3 11th Gen 1115G4 ...  COMGPF5CQ7VDWDT4
4  Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...  COMGPYKZAWY8UX6C

   Product image  Actual price Discount price  Stars         Rating Reviews  \
0            NaN       89990.0          54990    3.9      7 Ratings     1.0
1            NaN       83990.0          67990    NaN            NIL     nan
2            NaN       49240.0          35660    4.2  1,805 Ratings   143.0
3            NaN       43999.0          26990    4.2  6,977 Ratings   596.0
4            NaN       59400.0          27989    4.2  1,263 Ratings   113.0

                                         Description  \
0  Intel Core i5 Processor (12th Gen)16 GB DDR5 R...
1  Intel Core i7 Processor (12th Gen)8 GB DDR4 RA...
2  Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...
3  Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...
4  AMD Ryzen 3 Quad Core Processor8 GB LPDDR5 RAM...

                                                Link Rating Category
0  https://www.flipkart.com/msi-cyborg-15-intel-c...            Good
1  https://www.flipkart.com/msi-thin-15-intel-cor...         Unknown
```

```
2  https://www.flipkart.com/dell-inspiron-3520-in...       Excellent
3  https://www.flipkart.com/acer-one-2024-intel-c...       Excellent
4  https://www.flipkart.com/lenovo-v15-amd-ryzen-...       Excellent

'Reviews' column after replacing 'NIL' with '0 Reviews':
0      1.0
1      nan
2    143.0
3    596.0
4    113.0
Name: Reviews, dtype: object

DataFrame after handling missing values:
                                        Product Name         ProductID  \
0  MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...  COMGZW35W3DSJADN
1  MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...  COMGZW37ZX66DBHF
2  DELL Inspiron 3520 Intel Core i3 12th Gen 1215...  COMGJ75HJGFDJ6JN
3  Acer One (2024) Intel Core i3 11th Gen 1115G4 ...  COMGPF5CQ7VDWDT4
4  Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...  COMGPYKZAWY8UX6C

   Product image  Actual price Discount price  Stars         Rating  Reviews  \
0            NaN       89990.0          54990    3.9      7 Ratings      1.0
1            NaN       83990.0          67990    NaN           NIL      1.0
2            NaN       49240.0          35660    4.2  1,805 Ratings    143.0
3            NaN       43999.0          26990    4.2  6,977 Ratings    596.0
4            NaN       59400.0          27989    4.2  1,263 Ratings    113.0

                                         Description  \
0  Intel Core i5 Processor (12th Gen)16 GB DDR5 R...
1  Intel Core i7 Processor (12th Gen)8 GB DDR4 RA...
2  Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...
3  Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...
4  AMD Ryzen 3 Quad Core Processor8 GB LPDDR5 RAM...

                                            Link Rating Category
0  https://www.flipkart.com/msi-cyborg-15-intel-c...           Good
1  https://www.flipkart.com/msi-thin-15-intel-cor...        Unknown
2  https://www.flipkart.com/dell-inspiron-3520-in...       Excellent
3  https://www.flipkart.com/acer-one-2024-intel-c...       Excellent
4  https://www.flipkart.com/lenovo-v15-amd-ryzen-...       Excellent

'Reviews' column after handling missing values:
0      1.0
1      1.0
2    143.0
3    596.0
4    113.0
Name: Reviews, dtype: float64
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_6596\3961750355.py:10: FutureWarning: A value is tryi
ng to be set on a copy of a DataFrame or Series through chained assignment using an inplace metho
d.
The behavior will change in pandas 3.0. This inplace method will never work because the intermedi
ate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value},
inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on th
e original object.


  df['Reviews'].replace('NIL', '0 Reviews', inplace=True)
C:\Users\admin\AppData\Local\Temp\ipykernel_6596\3961750355.py:25: FutureWarning: A value is tryi
ng to be set on a copy of a DataFrame or Series through chained assignment using an inplace metho
d.
The behavior will change in pandas 3.0. This inplace method will never work because the intermedi
ate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value},
inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on th
e original object.


  df['Reviews'].fillna(method='ffill', inplace=True)
C:\Users\admin\AppData\Local\Temp\ipykernel_6596\3961750355.py:25: FutureWarning: Series.fillna w
ith 'method' is deprecated and will raise in a future version. Use obj.ffill() or obj.bfill() ins
tead.
  df['Reviews'].fillna(method='ffill', inplace=True)
```

Question 2

Olympic Data Analysis and Visualization

Perform any 10-12 operation on data using Pandas and display the result.

In [77]:
```
import pandas as pd
file_path = 'Olympic_Results.csv'
dff = pd.read_csv(file_path)
print(dff)
```

```
      result_id                              event_title  \
0         30359  Super-Heavyweight (>105 kilograms), Men
1          1626                     Giant Slalom, Women1
2            76                            Singles, Men
3           962                       1,500 metres, Men
4        258824            Canadian Singles, Slalom, Men
...         ...                                     ...
7389   19001250                       Basketball, Men
7390      84835                      Sabre, Team, Men
7391      72031                  Foil, Individual, Men
7392     258676                Beach Volleyball, Women
7393      48121                            Doubles, Men

                   edition  edition_id             sport  \
0     2004 Summer Olympics          26     Weightlifting
1     1998 Winter Olympics          46      Snowboarding
2     1976 Winter Olympics          40              Luge
3     1928 Winter Olympics          30     Speed Skating
4     2008 Summer Olympics          53      Canoe Slalom
...                    ...         ...               ...
7389  2020 Summer Olympics          61        Basketball
7390  1936 Summer Olympics          11           Fencing
7391  1900 Summer Olympics           2           Fencing
7392  2008 Summer Olympics          53  Beach Volleyball
7393  2000 Summer Olympics          25      Table Tennis

                    sport_url                             result_date  \
0     /editions/26/sports/WLF  25 August 2004 — 16:30 (B), 20:00 (A)
1     /editions/46/sports/SBD                        9 February 1998
2     /editions/40/sports/LUG                   4 –  7 February 1976
3     /editions/30/sports/SSK                14 February 1928 — 9:00
4     /editions/53/sports/CSL                    11 – 12 August 2008
...                       ...                                   ...
7389  /editions/61/sports/BKB               25 July –  7 August 2021
7390  /editions/11/sports/FEN                    12 – 13 August 1936
7391   /editions/2/sports/FEN                       14 – 21 May 1900
7392  /editions/53/sports/VBV                     9 – 21 August 2008
7393  /editions/25/sports/TTE                 16 – 23 September 2000

                                      result_location  \
0     Olympiako Gymnastirio Arsis Varon Nikaias, Nikaia
1                 Mt. Yakebitai, Shiga Kogen, Yamanouchi
2                      Kunsteis-Bob- und Rodelbahn, Igls
3           Olympia-Eisstadion Badrutts Park, St. Moritz
4     Shunyi Aolinpike Shuishang Gongyuan, Mapo, Shunyi
...                                                 ...
7389  Saitama Super Arena, Chūō-ku, Saitama, Saitama...
7390  Sportforum, Turnhalle, Reichssportfeld, Berlin...
7391  La Grande Salle des Fêtes de l'Exposition, Cha...
7392     Chaoyang Gongyuan Shatan Paiqiu Chang, Beijing
7393  State Sports Centre, Olympic Park, Sydney, New...

        result_participants  \
0      17 from 15 countries
1      31 from 14 countries
2      43 from 15 countries
3      30 from 14 countries
4      16 from 16 countries
...                     ...
7389  143 from 12 countries
7390  107 from 21 countries
7391    53 from 9 countries
```

```
7392   48 from 17 countries
7393   72 from 29 countries

                                        result_format  \
0     Total of best lifts in snatch and clean & jerk...
1            Two runs, total time determined placement.
2           Four runs, total time determined placement.
3                                                   na
4                                                   na
...                                                ...
7389  Round-robin pools advance teams to classificat...
7390                                                na
7391                                                na
7392  Top 16 teams from round-robin pools advanced t...
7393  Round-robin qualifying pools, followed by sing...

                                        result_detail  \
0                                                   na
1     Gates: 38 / 36Length: 936 mStart Altitude: 196...
2     Curves: 14Length: 1220 mStart Altitude: ?Verti...
3                                                   na
4                                                   na
...                                                ...
7389                                                na
7390                                                na
7391                                                na
7392                                                na
7393                                                na

                                    result_description
0     Not so much a competition as a coronation, the...
1     The women’s giant slalom was postponed one day...
2     Once more, the competitors from East and West ...
3     There was little doubt that the Olympic 1500 m...
4     Two former Olympic champions in the C-1 slalom...
...                                                ...
7389  All the games took place at Saitama Super Aren...
7390  On the middle of an era of total domination of...
7391  There were 53 fencers from 9 nations but 39 of...
7392  After winning the 2004 gold medal, Misty May a...
7393  Kong Linghui and Liu Guoliang had won the 1996...

[7394 rows x 12 columns]
```

In [99]:
```
# print(dff.columns)
print("Information about the file\n")
print(dff.info())
```

```
Information about the file

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7394 entries, 0 to 7393
Data columns (total 12 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   result_id            7394 non-null   int64 
 1   event_title          7394 non-null   object
 2   edition              7394 non-null   object
 3   edition_id           7394 non-null   int64 
 4   sport                7394 non-null   object
 5   sport_url            7394 non-null   object
 6   result_date          7394 non-null   object
 7   result_location      7393 non-null   object
 8   result_participants  7394 non-null   object
 9   result_format        7394 non-null   object
 10  result_detail        7394 non-null   object
 11  result_description   7394 non-null   object
dtypes: int64(2), object(10)
memory usage: 693.3+ KB
None
```

In [103…

```
print("\n. Missing values in each column:")
print(dff.isnull().sum())
```

```
. Missing values in each column:
result_id              0
event_title            0
edition                0
edition_id             0
sport                  0
sport_url              0
result_date            0
result_location        1
result_participants    0
result_format          0
result_detail          0
result_description     0
dtype: int64
```

In [125…

```
df_sorted = dff.sort_values('result_date')
# print(df_sorted.head())
print("\nDataFrame sorted by 'result_date':")
print(df_sorted[['result_date', 'event_title', 'edition', 'result_location']].head())
```

```
DataFrame sorted by 'result_date':
                result_date                        event_title  \
6370          1 August 1900  Free Pistol, 50 metres, Team, Men
5585    1 August 1900 – 8:00        Free Pistol, 50 metres, Men
619           1 August 1928                   Pole Vault, Men
1649          1 August 1928                 Discus Throw, Men
1181   1 August 1932 – 14:30                 Hammer Throw, Men

                  edition                                  result_location
6370  1900 Summer Olympics                      Camp de Satory, Versailles
5585  1900 Summer Olympics                      Camp de Satory, Versailles
619   1928 Summer Olympics                    Olympisch Stadion, Amsterdam
1649  1928 Summer Olympics                    Olympisch Stadion, Amsterdam
1181  1932 Summer Olympics  Los Angeles Memorial Coliseum, Los Angeles, Ca...
```

In [129…

```
sport_counts = dff.groupby('sport').size()
print("\n Number of occurrences per sport:")
```

```
print(sport_counts)
```

```
 Number of occurrences per sport:
sport
3x3 Basketball         2
Aeronautics            1
Alpine Skiing        170
Alpinism               3
American Football      2
                    ...
Waterskiing            6
Weightlifting        227
Winter Pentathlon      1
Wrestling            428
Wushu                 15
Length: 112, dtype: int64
```

In [145…
```
unique_sports = dff['sport'].nunique()
print("\n Number of unique sports:", unique_sports)
print(dff.columns)
```

```
 Number of unique sports: 112
Index(['result_id', 'event_title', 'edition', 'edition_id', 'sport',
       'sport_url', 'result_date', 'result_location', 'result_participants',
       'result_format', 'result_detail', 'result_description'],
      dtype='object')
```

In [159…
```
dff['result_date'] = pd.to_datetime(dff['result_date'], errors='coerce')

dff['event_year'] = dff['result_date'].dt.year

events_2022 = dff[dff['event_year'] == 2022].shape[0]
print("\n Number of events in 2022:", events_2022)
```

```
Number of events in 2022: 20
```

In [167…
```
dff['description_length'] = dff['result_description'].str.len()
avg_description_length = dff['description_length'].mean()
print("\n Average length of 'result_description':", avg_description_length)
```

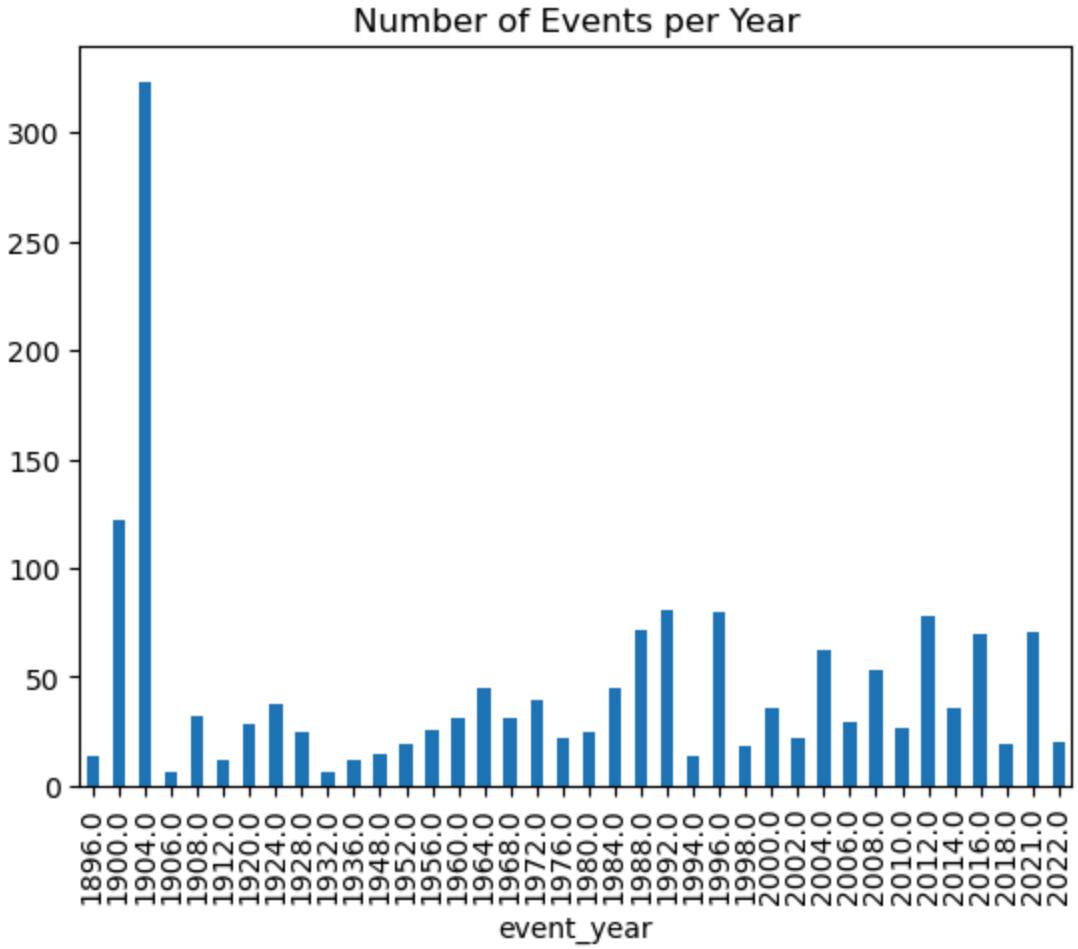
```
Average length of 'result_description': 1115.2412767108467
```

In [165…
```
sport_edition_counts = dff.groupby('sport')['edition_id'].count()


max_sport = sport_edition_counts.idxmax()
max_count = sport_edition_counts.max()


max_sport_df = dff[dff['sport'] == max_sport][['edition_id', 'sport']]

print(f"\nSport with the maximum number of 'edition_id': {max_sport} ({max_count} editions)")
print(f"\nDetails of the sport with the most editions:\n{max_sport_df}")
```

```
Sport with the maximum number of 'edition_id': Athletics (1335 editions)

Details of the sport with the most editions:
      edition_id      sport
6             11  Athletics
7             26  Athletics
11             3  Athletics
14            23  Athletics
22            59  Athletics
...          ...        ...
7360           3  Athletics
7362          54  Athletics
7368          54  Athletics
7384          19  Athletics
7385          19  Athletics

[1335 rows x 2 columns]
```

In [181…
```
events_per_year = dff['event_year'].value_counts().sort_index()
events_per_year.plot(kind='bar', title='Number of Events per Year')
```

Out[181…
```
<Axes: title={'center': 'Number of Events per Year'}, xlabel='event_year'>
```



In [209…
```
filtered_df = dff[(dff['sport'] == 'Shooting') & (dff['event_year'] >= 2021)]
print(filtered_df)
```

```
      result_id                                        event_title  \
1049   18000754  Small-Bore Rifle, Three Positions, 50 metres, ...
2148   18000731  Small-Bore Rifle, Three Positions, 50 metres, Men
2898   18000728                          Air Rifle, 10 metres, Men
2985   18000769                Air Rifle, 10 metres, Team, Mixed
3723   18000772                                     Trap, Team, Mixed
5226   18000720                       Air Pistol, 10 metres, Men
5392   18000743                     Air Pistol, 10 metres, Women
6111   18000751                      Air Rifle, 10 metres, Women
6758   18000766               Air Pistol, 10 metres, Team, Mixed

                  edition  edition_id     sport               sport_url  \
1049  2020 Summer Olympics          61  Shooting  /editions/61/sports/SHO
2148  2020 Summer Olympics          61  Shooting  /editions/61/sports/SHO
2898  2020 Summer Olympics          61  Shooting  /editions/61/sports/SHO
2985  2020 Summer Olympics          61  Shooting  /editions/61/sports/SHO
3723  2020 Summer Olympics          61  Shooting  /editions/61/sports/SHO
5226  2020 Summer Olympics          61  Shooting  /editions/61/sports/SHO
5392  2020 Summer Olympics          61  Shooting  /editions/61/sports/SHO
6111  2020 Summer Olympics          61  Shooting  /editions/61/sports/SHO
6758  2020 Summer Olympics          61  Shooting  /editions/61/sports/SHO

     result_date                     result_location   result_participants  \
1049  2021-07-31  Asaka Shooting Range, Nerima, Tokyo  37 from 29 countries
2148  2021-08-02  Asaka Shooting Range, Nerima, Tokyo  39 from 27 countries
2898  2021-07-25  Asaka Shooting Range, Nerima, Tokyo  47 from 32 countries
2985  2021-07-27  Asaka Shooting Range, Nerima, Tokyo  58 from 20 countries
3723  2021-07-31  Asaka Shooting Range, Nerima, Tokyo  32 from 12 countries
5226  2021-07-24  Asaka Shooting Range, Nerima, Tokyo  36 from 29 countries
5392  2021-07-25  Asaka Shooting Range, Nerima, Tokyo  53 from 37 countries
6111  2021-07-24  Asaka Shooting Range, Nerima, Tokyo  50 from 39 countries
6758  2021-07-27  Asaka Shooting Range, Nerima, Tokyo  40 from 15 countries

     result_format result_detail  \
1049            na            na
2148            na            na
2898            na            na
2985            na            na
3723            na            na
5226            na            na
5392            na            na
6111            na            na
6758            na            na

                                  result_description  event_year  \
1049  The women’s 50 m small-bore rifle three positi...      2021.0
2148  The men’s 50 m small-bore rifle three position...      2021.0
2898  The men’s 10 m air rifle was held for the 10th...      2021.0
2985  On the afternoon of the fourth day of shooting...      2021.0
3723  The last of the three new mixed team shooting ...      2021.0
5226  The first day of the shooting schedule of the ...      2021.0
5392  The second day of the shooting events started ...      2021.0
6111  Since 1988, the women’s 10 m air rifle event h...      2021.0
6758  The 10 m air pistol event was the first ever m...      2021.0

      description_length
1049                2105
2148                2182
2898                1854
2985                1263
3723                2317
5226                2390
```

5392	2677
6111	2203
6758	1878

ASSIGNMENT 2

In [3]:
```
import pandas as pd
import matplotlib.pyplot as plt
file_path = 'employee_data.csv'
df = pd.read_csv(file_path)
df
```

Out[3]:

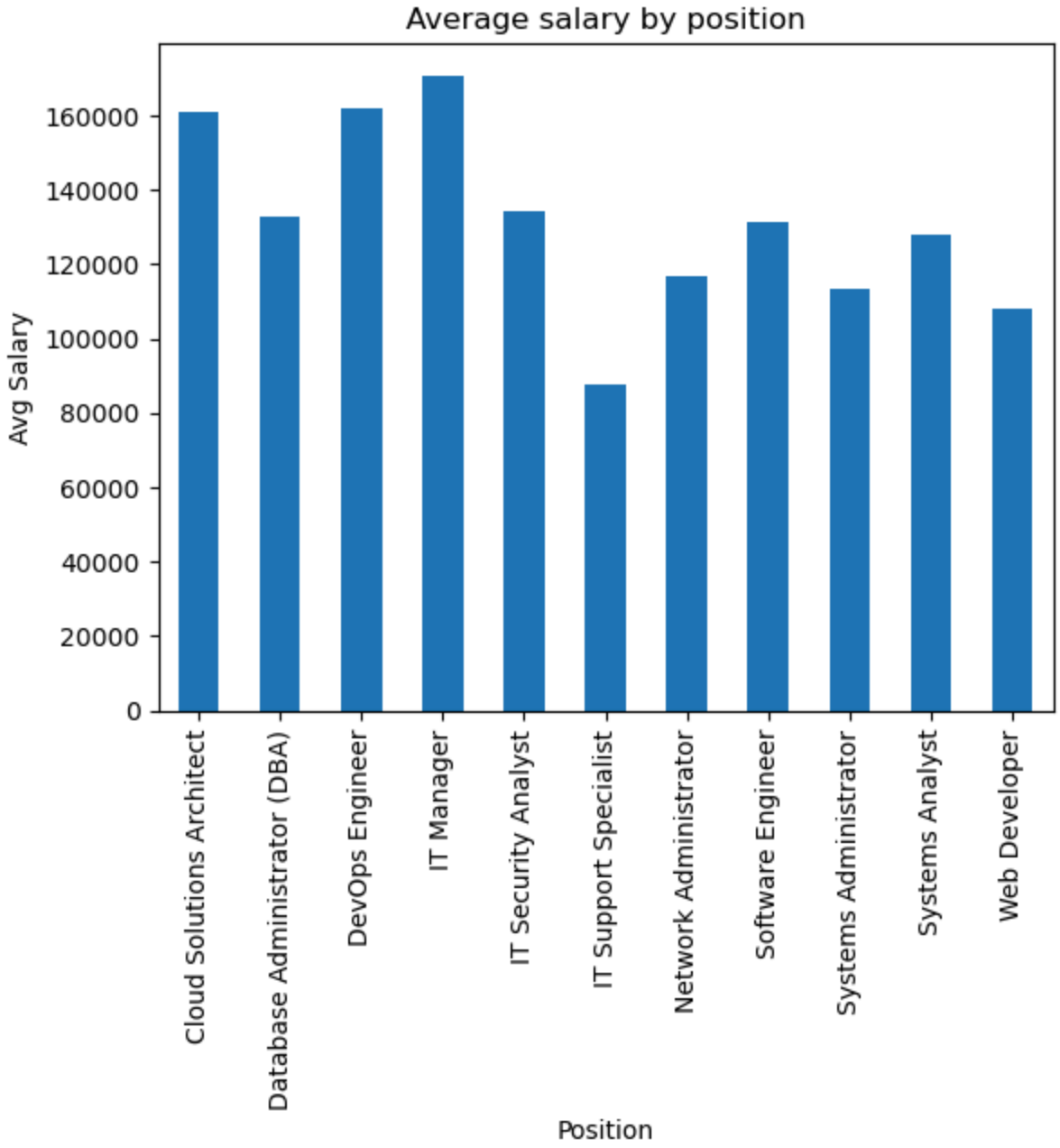
	ID	Gender	Experience (Years)	Position	Salary
0	1	F	4	DevOps Engineer	109976
1	2	M	6	DevOps Engineer	120088
2	3	M	17	Web Developer	181301
3	4	M	7	Systems Administrator	77530
4	5	F	13	Systems Administrator	152397
...	...	...	...	...	...
395	396	F	19	Cloud Solutions Architect	236045
396	397	F	20	Web Developer	182770
397	398	F	9	Network Administrator	85550
398	399	M	18	Database Administrator (DBA)	129996
399	400	F	11	IT Security Analyst	169058

400 rows × 5 columns

In [ ]:
```
```

In [44]:
```
# avg_salary = df.groupby('Position')['Salary'].mean()
# print(avg_salary)

# avg_salary.plot(kind='bar', title='Average Salary by Position')
# plt.xlabel('Position')
# plt.ylabel('Average Salary')
# plt.show()
avg_salary = df.groupby('Position')['Salary'].mean()
print(avg_salary)
avg_salary.plot(kind='bar',title='Average salary by position')
plt.xlabel('Position')
plt.ylabel('Avg Salary')
plt.show()
# avg_salary.dtype
# avg_salary.shape
```

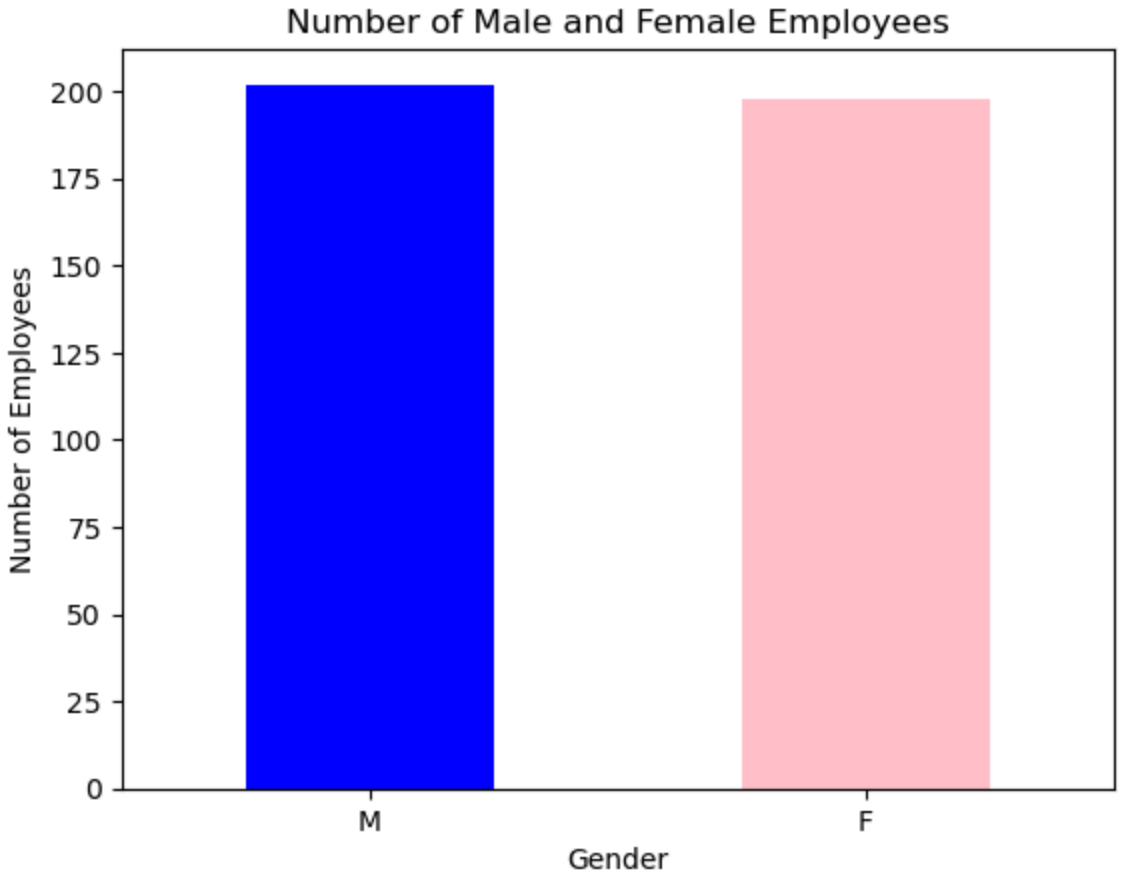
```
Position
Cloud Solutions Architect       160841.633333
Database Administrator (DBA)    132864.552632
DevOps Engineer                 161859.081081
IT Manager                      170711.550000
IT Security Analyst             134440.820513
IT Support Specialist            87683.806452
Network Administrator           116865.064516
Software Engineer               131357.416667
Systems Administrator           113117.447368
Systems Analyst                 127658.189189
Web Developer                   108238.116279
Name: Salary, dtype: float64
```



In [ ]:
```
# 1. Take employee_data Dataset and Perform following task:
# • Display Average Salary of each Position also plot same on graph.(Graph you
# may take according to your requirement).
# • Display total number of Male and Female employees and plot same on Chart.
# • Show using chart, how much salary is earn by the employee who having
# experience between 10 to 15 years.
# • Display a bar char for number of positions in company.
# • Analys which position is better in terms of salary.
```

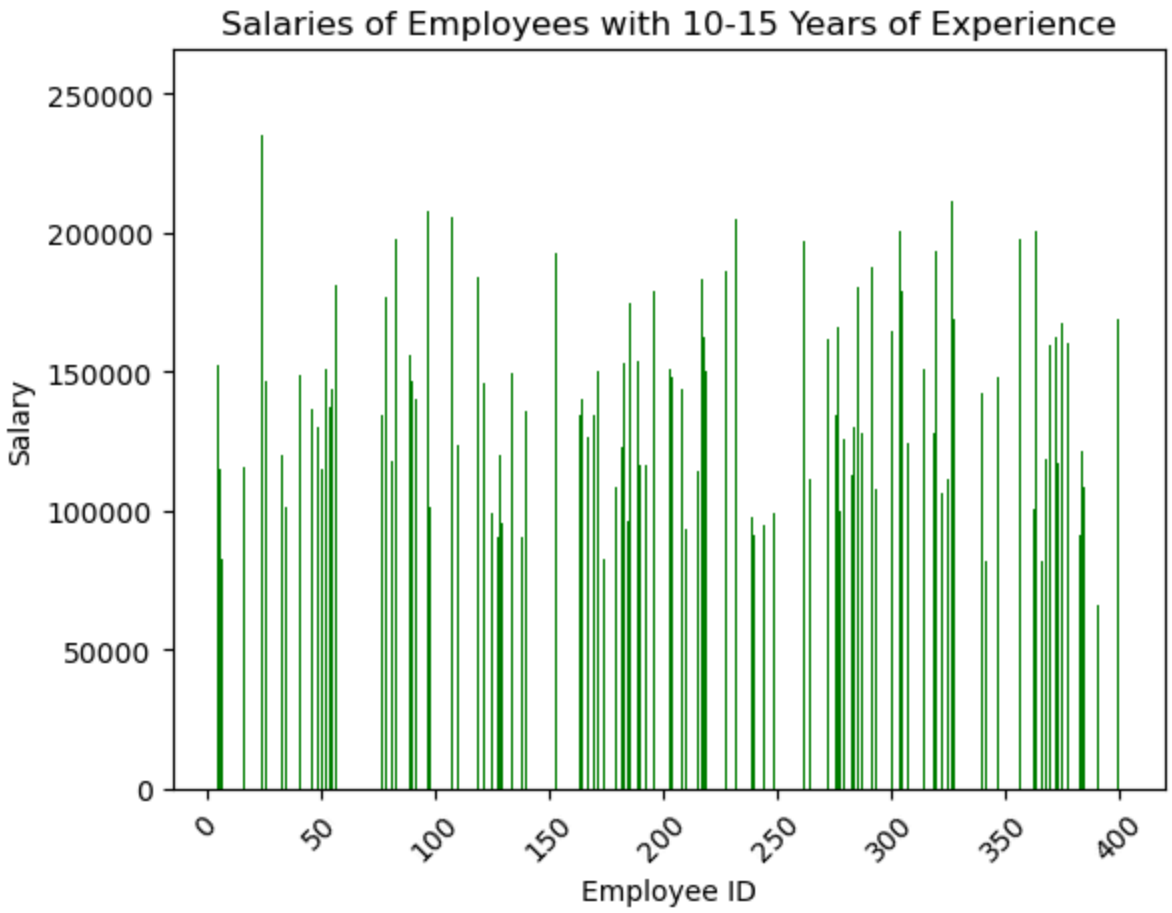
In [7]:
```
# Assuming your DataFrame is named df and it has a 'Gender' column
# Count the number of Male and Female employees
gender_counts = df['Gender'].value_counts()

# Display the total number of Male and Female employees
print(gender_counts)

# Plotting the result using a bar chart
gender_counts.plot(kind='bar', color=['blue', 'pink'], title='Number of Male and Female Employees
plt.xlabel('Gender')
plt.ylabel('Number of Employees')
plt.xticks(rotation=0)  # To make sure the labels (Male, Female) are horizontal
plt.show()
```

```
Gender
M    202
F    198
Name: count, dtype: int64
```



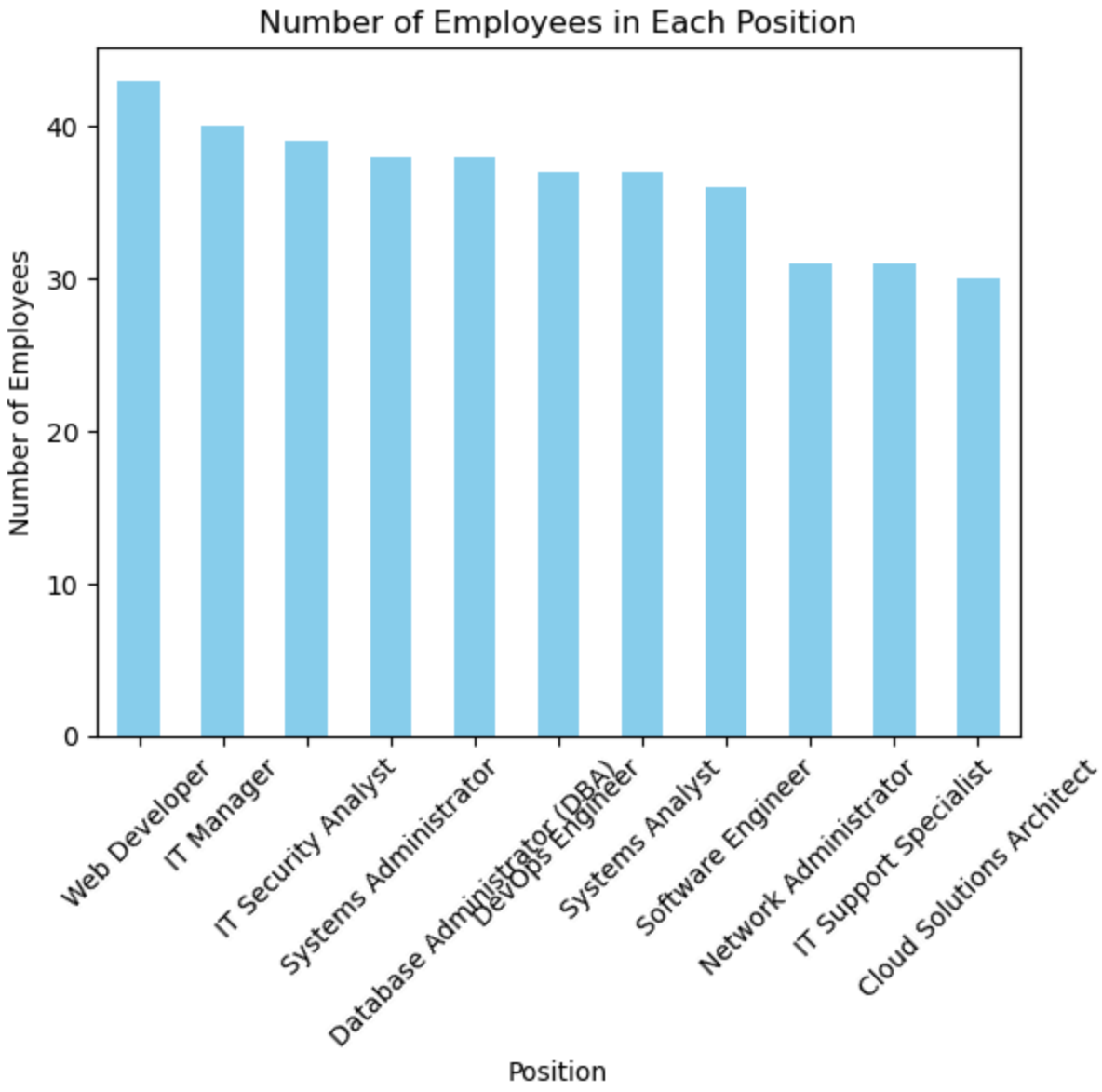
In [13]:
```
filtered_df = df[(df['Experience (Years)'] >= 10) & (df['Experience (Years)'] <= 15)]

# Display the filtered data (optional)
print(filtered_df[['Experience (Years)', 'Salary']])

# Plot the salary of these employees using a bar chart
plt.bar(filtered_df['ID'], filtered_df['Salary'], color='green')
plt.title('Salaries of Employees with 10-15 Years of Experience')
plt.xlabel('Employee ID')
plt.ylabel('Salary')
plt.xticks(rotation=45)
plt.show()
```

```
     Experience (Years)  Salary
4                    13  152397
5                    13  114998
6                    11   82328
15                   13  115698
23                   13  235235
..                  ...     ...
382                  10   91542
383                  14  121650
384                  14  108497
390                  11   66076
399                  11  169058

[116 rows x 2 columns]
```



In [15]:
```python
position_counts = df['Position'].value_counts()

# Display the counts (optional)
print(position_counts)

# Plotting the number of positions using a bar chart
position_counts.plot(kind='bar', color='skyblue', title='Number of Employees in Each Position')
plt.xlabel('Position')
plt.ylabel('Number of Employees')
plt.xticks(rotation=45)  # Rotate the labels on the x-axis for better readability
plt.show()
```

```
Position
Web Developer                   43
IT Manager                      40
IT Security Analyst             39
Systems Administrator           38
Database Administrator (DBA)    38
DevOps Engineer                 37
Systems Analyst                 37
Software Engineer               36
Network Administrator           31
IT Support Specialist           31
Cloud Solutions Architect       30
Name: count, dtype: int64
```

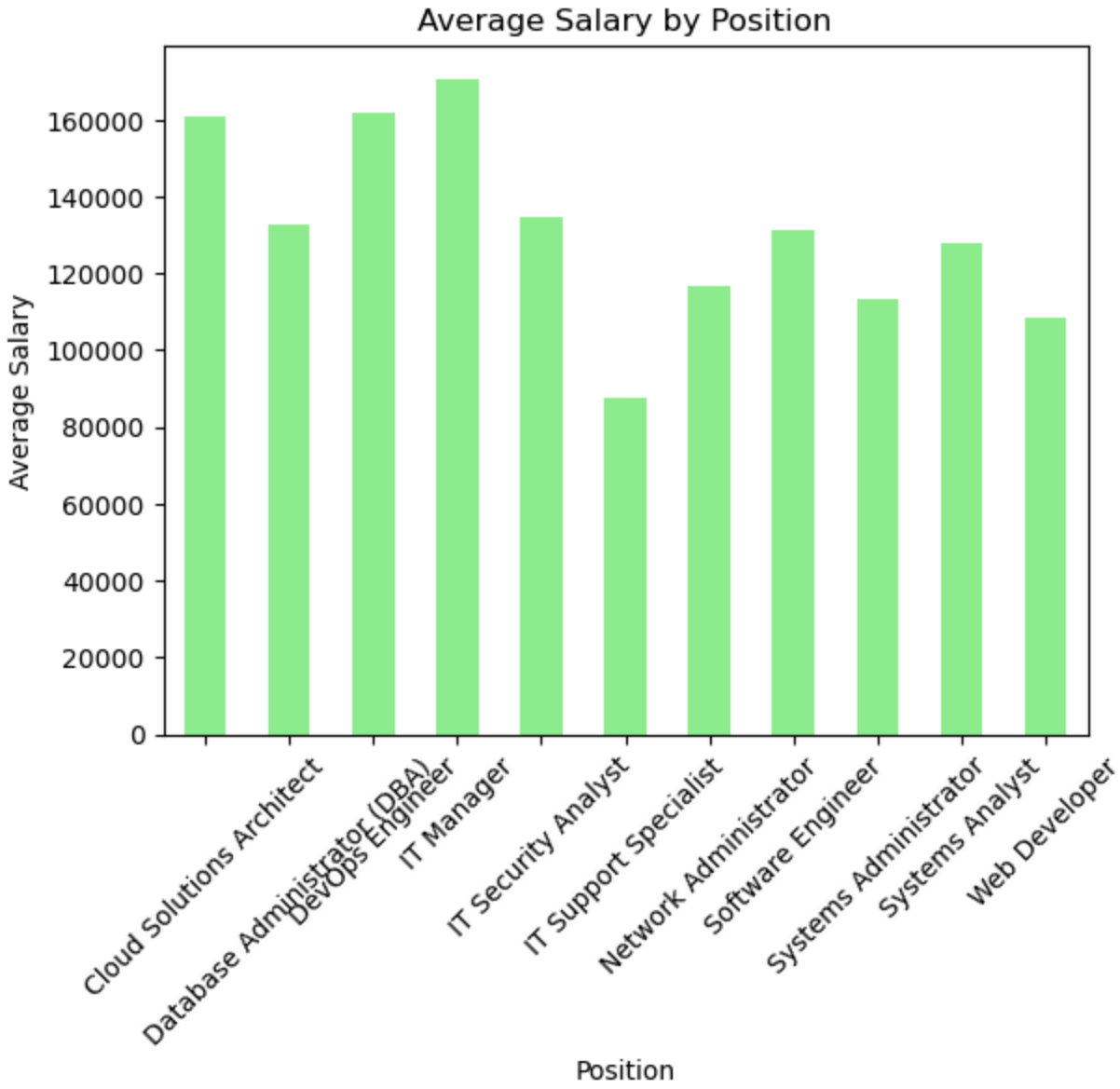


In [17]:

```python
avg_salary_by_position = df.groupby('Position')['Salary'].mean()

# Display the average salary for each position
print("Average salary by position:")
print(avg_salary_by_position)

# Identify the position with the highest average salary
best_position = avg_salary_by_position.idxmax()
highest_avg_salary = avg_salary_by_position.max()

print(f"The best position in terms of salary is: {best_position} with an average salary of {high
```

```python
# Plotting the average salary by position
avg_salary_by_position.plot(kind='bar', color='lightgreen', title='Average Salary by Position')
plt.xlabel('Position')
plt.ylabel('Average Salary')
plt.xticks(rotation=45)
plt.show()
```

```
Average salary by position:
Position
Cloud Solutions Architect       160841.633333
Database Administrator (DBA)    132864.552632
DevOps Engineer                 161859.081081
IT Manager                      170711.550000
IT Security Analyst             134440.820513
IT Support Specialist            87683.806452
Network Administrator           116865.064516
Software Engineer               131357.416667
Systems Administrator           113117.447368
Systems Analyst                 127658.189189
Web Developer                   108238.116279
Name: Salary, dtype: float64
The best position in terms of salary is: IT Manager with an average salary of 170711.55
```



Question 2

In [6]:
```
import pandas as pd
file_path = 'Flipkart-Laptops.xlsx'
df = pd.read_excel(file_path)
print(df)
```

```
                                          Product Name         ProductID  \
0    MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...  COMGZW35W3DSJADN
1    MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...  COMGZW37ZX66DBHF
2    DELL Inspiron 3520 Intel Core i3 12th Gen 1215...  COMGJ75HJGFDJ6JN
3    Acer One (2024) Intel Core i3 11th Gen 1115G4 ...  COMGPF5CQ7VDWDT4
4    Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...  COMGPYKZAWY8UX6C
..                                                 ...               ...
955  Acer Swift Go 14 (2024) AI Powered EVO Intel C...  COMGWKF2VKGAVHDU
956  HP Victus Intel Core i5 12th Gen 12450H - (16 ...  COMH2DYZHMHZ5UPG
957  Infinix X1 Slim Series (2024) Intel Core i3 10...  COMGEHP5EFEGWZW5
958  Lenovo IdeaPad Slim 3 Intel Core i5 12th Gen 1...  COMGYHP5ZB4AGZH6
959  HP (15s-fq5007TU) Intel Core i3 12th Gen 1215U...  COMGYHP5MCEYZHSV

     Product image Actual price Discount price Stars         Rating  \
0              NaN        89990          54990   3.9      7 Ratings
1              NaN        83990          67990   NIL            NIL
2              NaN        49240          35660   4.2  1,805 Ratings
3              NaN        43999          26990   4.2  6,977 Ratings
4              NaN        59400          27989   4.2  1,263 Ratings
..             ...          ...            ...   ...           ...
955            NaN       129999          79990   4.1    108 Ratings
956            NaN          NIL          82414   NIL            NIL
957            NaN        49999          32990   4.3  3,897 Ratings
958            NaN        69890          53390   3.8     53 Ratings
959            NaN        51134          38990   4.2  5,540 Ratings

         Reviews                                        Description  \
0      1 Reviews  Intel Core i5 Processor (12th Gen)16 GB DDR5 R...
1            NIL  Intel Core i7 Processor (12th Gen)8 GB DDR4 RA...
2    143 Reviews  Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...
3    596 Reviews  Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...
4    113 Reviews  AMD Ryzen 3 Quad Core Processor8 GB LPDDR5 RAM...
..           ...                                                ...
955   16 Reviews  Intel Core Ultra 5 Processor16 GB LPDDR5X RAMW...
956          NIL  Intel Core i5 Processor (12th Gen)16 GB DDR4 R...
957  457 Reviews  Intel Core i3 Processor (10th Gen)8 GB LPDDR4X...
958    5 Reviews  Intel Core i5 Processor (12th Gen)16 GB LPDDR5...
959  485 Reviews  Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...

                                                  Link
0    https://www.flipkart.com/msi-cyborg-15-intel-c...
1    https://www.flipkart.com/msi-thin-15-intel-cor...
2    https://www.flipkart.com/dell-inspiron-3520-in...
3    https://www.flipkart.com/acer-one-2024-intel-c...
4    https://www.flipkart.com/lenovo-v15-amd-ryzen-...
..                                                 ...
955  https://www.flipkart.com/acer-swift-go-14-2024...
956  https://www.flipkart.com/hp-victus-intel-core-...
957  https://www.flipkart.com/infinix-x1-slim-serie...
958  https://www.flipkart.com/lenovo-ideapad-slim-3...
959  https://www.flipkart.com/hp-15s-fq5007tu-intel...

[960 rows x 10 columns]
```

In [14]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 960 entries, 0 to 959
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Product Name    960 non-null    object
 1   ProductID       960 non-null    object
 2   Product image   0 non-null      float64
 3   Actual price    960 non-null    object
 4   Discount price  960 non-null    object
 5   Stars           960 non-null    object
 6   Rating          960 non-null    object
 7   Reviews         960 non-null    object
 8   Description     960 non-null    object
 9   Link            960 non-null    object
dtypes: float64(1), object(9)
memory usage: 75.1+ KB
```

In [12]:
```
print(df.head())
```

```
                                        Product Name         ProductID  \
0  MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...  COMGZW35W3DSJADN
1  MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...  COMGZW37ZX66DBHF
2  DELL Inspiron 3520 Intel Core i3 12th Gen 1215...  COMGJ75HJGFDJ6JN
3  Acer One (2024) Intel Core i3 11th Gen 1115G4 ...  COMGPF5CQ7VDWDT4
4  Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...  COMGPYKZAWY8UX6C

   Product image Actual price Discount price Stars         Rating  \
0            NaN        89990          54990   3.9      7 Ratings
1            NaN        83990          67990   NIL            NIL
2            NaN        49240          35660   4.2  1,805 Ratings
3            NaN        43999          26990   4.2  6,977 Ratings
4            NaN        59400          27989   4.2  1,263 Ratings

       Reviews                                        Description  \
0    1 Reviews  Intel Core i5 Processor (12th Gen)16 GB DDR5 R...
1          NIL  Intel Core i7 Processor (12th Gen)8 GB DDR4 RA...
2  143 Reviews  Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...
3  596 Reviews  Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...
4  113 Reviews  AMD Ryzen 3 Quad Core Processor8 GB LPDDR5 RAM...

                                                Link
0  https://www.flipkart.com/msi-cyborg-15-intel-c...
1  https://www.flipkart.com/msi-thin-15-intel-cor...
2  https://www.flipkart.com/dell-inspiron-3520-in...
3  https://www.flipkart.com/acer-one-2024-intel-c...
4  https://www.flipkart.com/lenovo-v15-amd-ryzen-...
```

In [10]:
```
print(df.describe())
```

```
       Product image
count            0.0
mean             NaN
std              NaN
min              NaN
25%              NaN
50%              NaN
75%              NaN
max              NaN
```

In [44]:
```
print(df.head())
```

```
                                        Product Name         ProductID  \
0  MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...  COMGZW35W3DSJADN
1  MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...  COMGZW37ZX66DBHF
2  DELL Inspiron 3520 Intel Core i3 12th Gen 1215...  COMGJ75HJGFDJ6JN
3  Acer One (2024) Intel Core i3 11th Gen 1115G4 ...  COMGPF5CQ7VDWDT4
4  Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...  COMGPYKZAWY8UX6C

   Product image Actual price Discount price Stars         Rating  \
0            NaN        89990          54990   3.9      7 Ratings
1            NaN        83990          67990   NIL            NIL
2            NaN        49240          35660   4.2  1,805 Ratings
3            NaN        43999          26990   4.2  6,977 Ratings
4            NaN        59400          27989   4.2  1,263 Ratings

       Reviews                                        Description  \
0    1 Reviews  Intel Core i5 Processor (12th Gen)16 GB DDR5 R...
1          NIL  Intel Core i7 Processor (12th Gen)8 GB DDR4 RA...
2  143 Reviews  Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...
3  596 Reviews  Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...
4  113 Reviews  AMD Ryzen 3 Quad Core Processor8 GB LPDDR5 RAM...

                                                Link
0  https://www.flipkart.com/msi-cyborg-15-intel-c...
1  https://www.flipkart.com/msi-thin-15-intel-cor...
2  https://www.flipkart.com/dell-inspiron-3520-in...
3  https://www.flipkart.com/acer-one-2024-intel-c...
4  https://www.flipkart.com/lenovo-v15-amd-ryzen-...
```

In [48]:

```python
df.replace('NIL', pd.NA, inplace=True)

# Drop rows where 'Actual price' or 'Discount price' is missing
df.dropna(subset=['Actual price', 'Discount price'], inplace=True)
df.head()
```

Out[48]:

	Product Name	ProductID	Product image	Actual price	Discount price	Stars	Rating	Reviews	Description	
0	MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...	COMGZW35W3DSJADN	NaN	89990	54990	3.9	7 Ratings	1 Reviews	Intel Core i5 Processor (12th Gen)16 GB DDR5 R...	h
1	MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...	COMGZW37ZX66DBHF	NaN	83990	67990	<NA>	<NA>	<NA>	Intel Core i7 Processor (12th Gen)8 GB DDR4 RA...	h
2	DELL Inspiron 3520 Intel Core i3 12th Gen 1215...	COMGJ75HJGFDJ6JN	NaN	49240	35660	4.2	1,805 Ratings	143 Reviews	Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...	h
3	Acer One (2024) Intel Core i3 11th Gen 1115G4 ...	COMGPF5CQ7VDWDT4	NaN	43999	26990	4.2	6,977 Ratings	596 Reviews	Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...	ht
4	Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...	COMGPYKZAWY8UX6C	NaN	59400	27989	4.2	1,263 Ratings	113 Reviews	AMD Ryzen 3 Quad Core Processor8 GB LPDDR5 RAM...	http

In [52]:
```
print(df.dtypes)
```

```
Product Name       object
ProductID          object
Product image     float64
Actual price       object
Discount price     object
Stars              object
Rating             object
Reviews            object
Description        object
Link               object
dtype: object
```

In [18]:
```
df['Actual price'] = df['Actual price'].astype(str)
df['Discount price'] = df['Discount price'].astype(str)
df['Stars'] = df['Stars'].astype(str)
df['Rating'] = df['Rating'].astype(str)
df['Reviews'] = df['Reviews'].astype(str)
```

In [20]:
```
df['Actual price'] = pd.to_numeric(df['Actual price'].str.replace(',', ''), errors='coerce')
df['Discount price'] = pd.to_numeric(df['Discount price'].str.replace(',', ''), errors='coerce')

df['Stars'] = pd.to_numeric(df['Stars'], errors='coerce')

df['Rating'] = pd.to_numeric(df['Rating'].str.extract(r'(\d+)')[0].str.replace(',', ''), errors=
df['Reviews'] = pd.to_numeric(df['Reviews'].str.extract(r'(\d+)')[0], errors='coerce')
```

In [22]:
```
df['Discount Percentage'] = ((df['Actual price'] - df['Discount price']) / df['Actual price']) *
df.head()
```

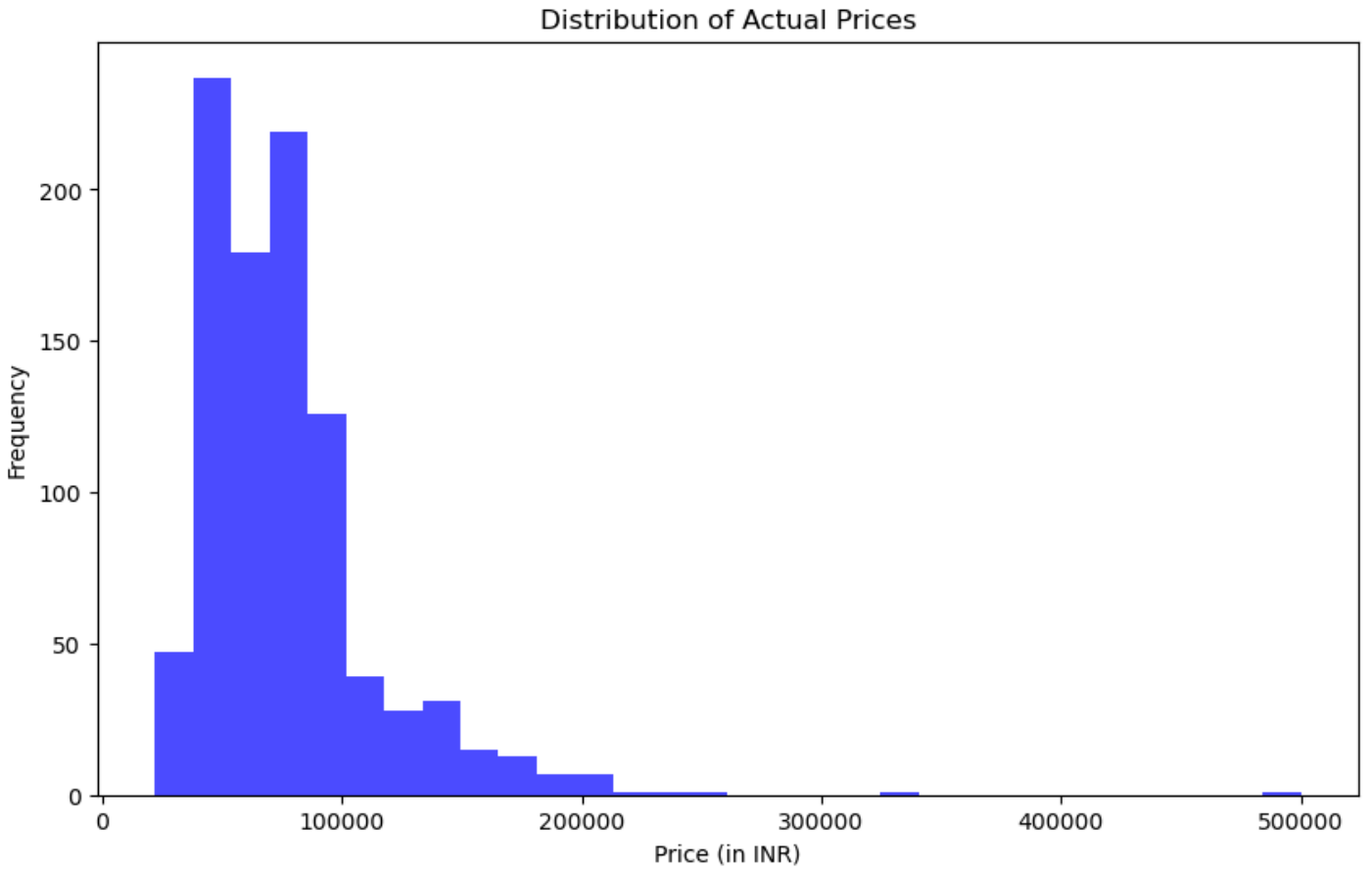
Out[22]:

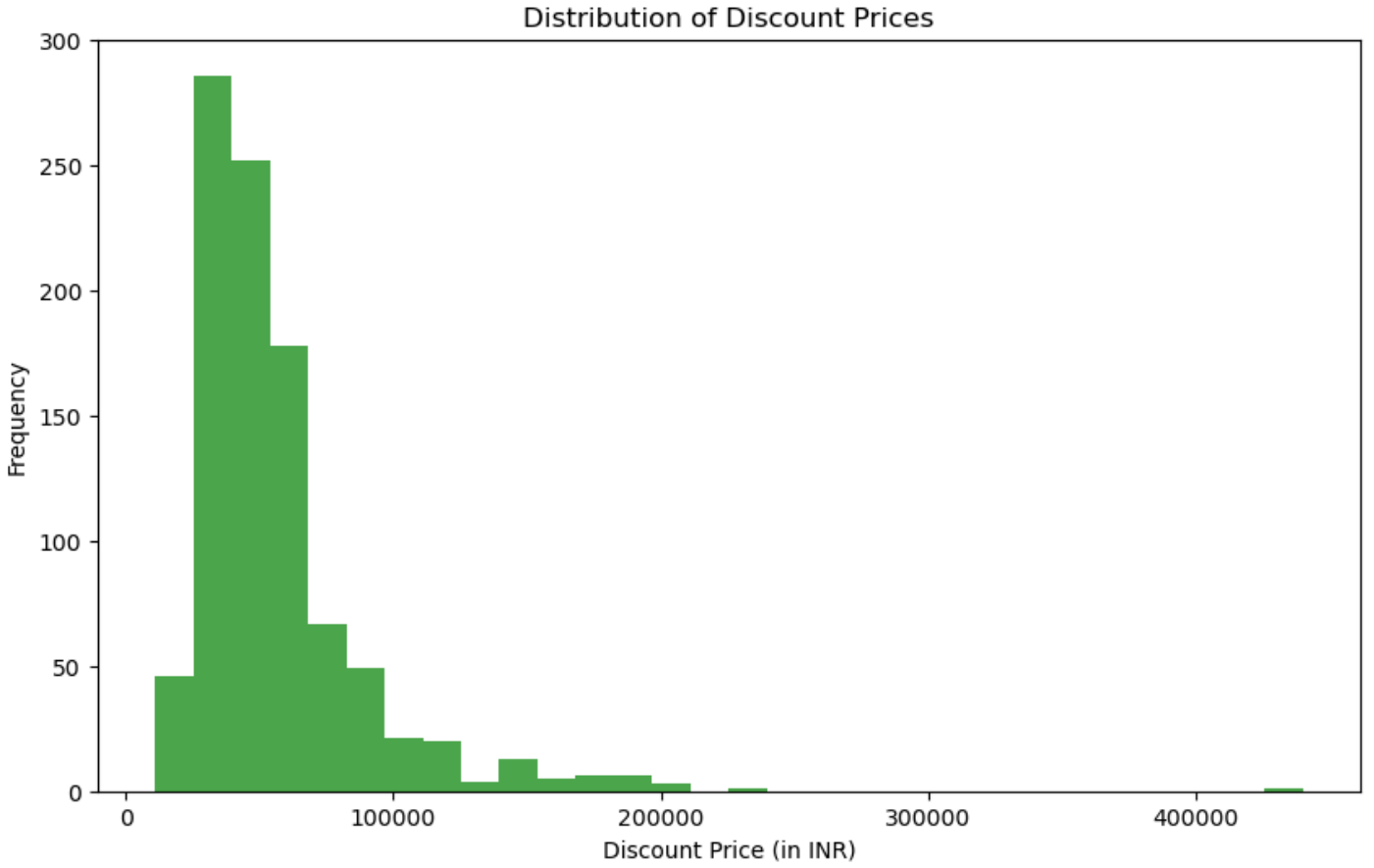
	Product Name	ProductID	Product image	Actual price	Discount price	Stars	Rating	Reviews	Description	
0	MSI Cyborg 15 Intel Core i5 12th Gen 12450H - ...	COMGZW35W3DSJADN	NaN	89990.0	54990.0	3.9	7.0	1.0	Intel Core i5 Processor (12th Gen)16 GB DDR5 R...	h
1	MSI Thin 15 Intel Core i7 12th Gen 12650H - (8...	COMGZW37ZX66DBHF	NaN	83990.0	67990.0	NaN	NaN	NaN	Intel Core i7 Processor (12th Gen)8 GB DDR4 RA...	h
2	DELL Inspiron 3520 Intel Core i3 12th Gen 1215...	COMGJ75HJGFDJ6JN	NaN	49240.0	35660.0	4.2	1.0	143.0	Intel Core i3 Processor (12th Gen)8 GB DDR4 RA...	h
3	Acer One (2024) Intel Core i3 11th Gen 1115G4 ...	COMGPF5CQ7VDWDT4	NaN	43999.0	26990.0	4.2	6.0	596.0	Intel Core i3 Processor (11th Gen)8 GB DDR4 RA...	ht
4	Lenovo V15 AMD Ryzen 3 Quad Core 7320U - (8 GB...	COMGPYKZAWY8UX6C	NaN	59400.0	27989.0	4.2	1.0	113.0	AMD Ryzen 3 Quad Core Processor8 GB LPDDR5 RAM...	https

In [24]:

```
# distribution of actual prices
plt.figure(figsize=(10, 6))
plt.hist(df['Actual price'], bins=30, color='blue', alpha=0.7)
plt.title('Distribution of Actual Prices')
plt.xlabel('Price (in INR)')
plt.ylabel('Frequency')
plt.show()

# distribution of discount prices
plt.figure(figsize=(10, 6))
plt.hist(df['Discount price'], bins=30, color='green', alpha=0.7)
```

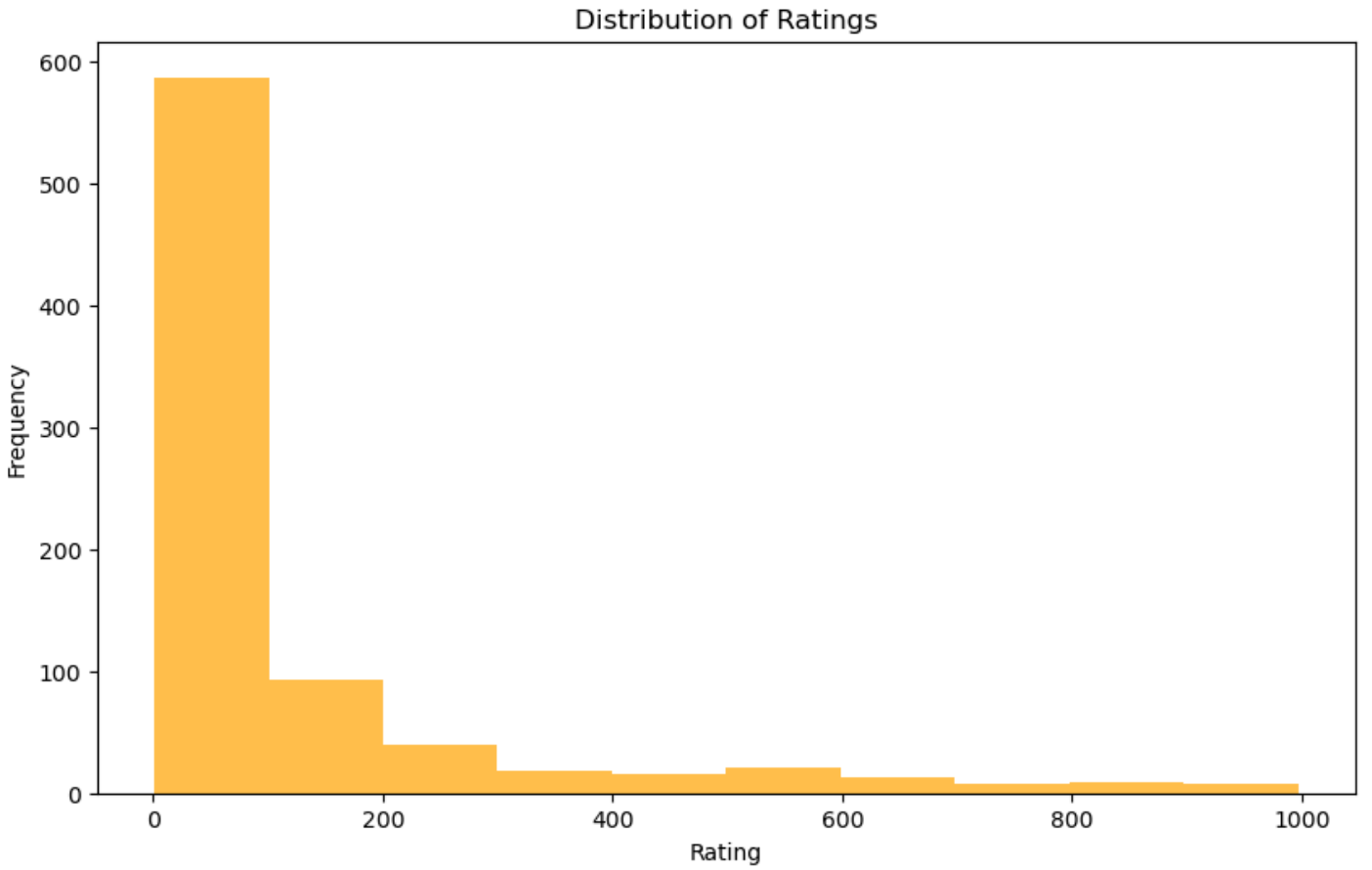
```
plt.title('Distribution of Discount Prices')
plt.xlabel('Discount Price (in INR)')
plt.ylabel('Frequency')
plt.show()
```



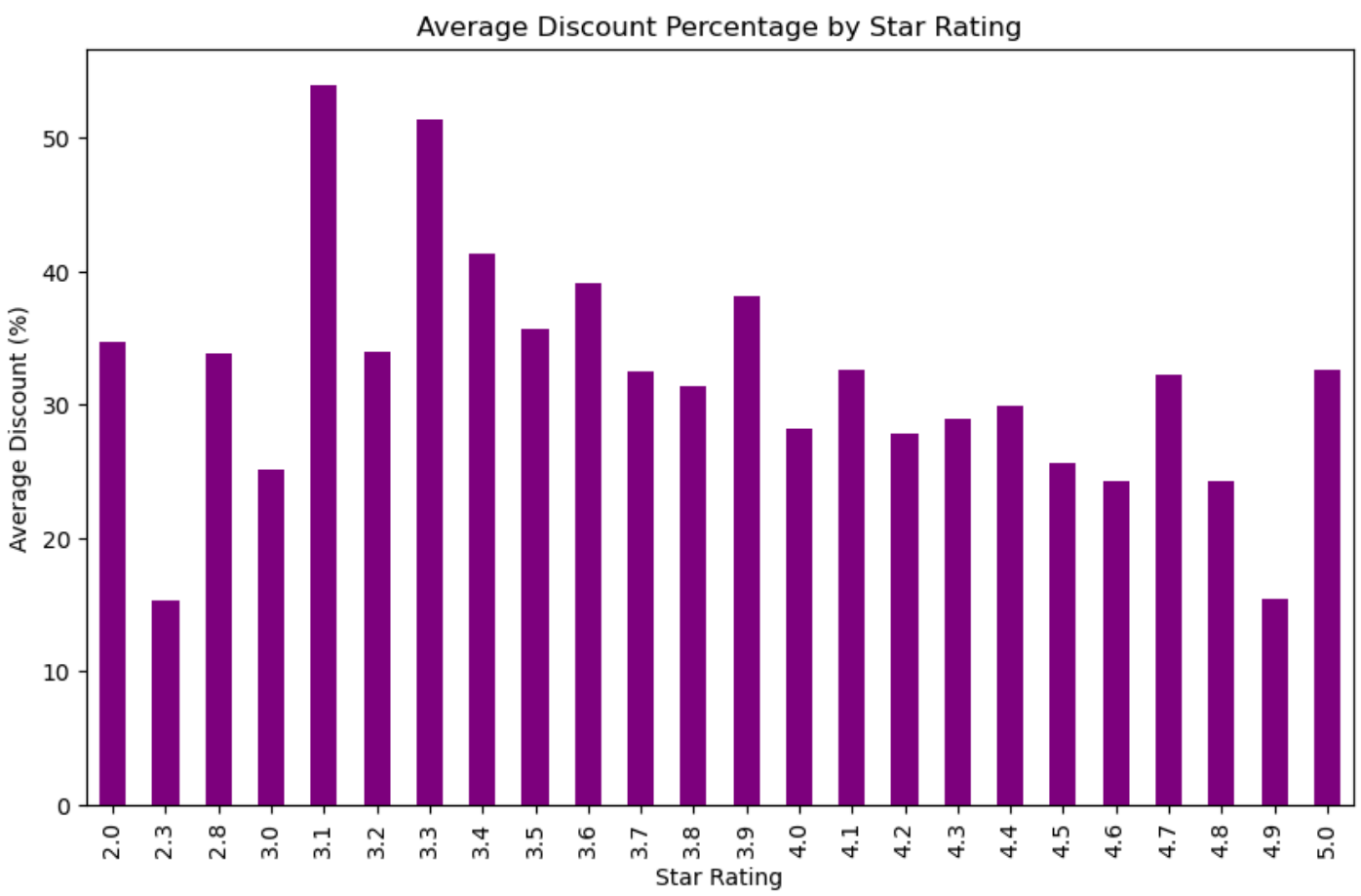


In [26]:
```
# distribution of laptop ratings
plt.figure(figsize=(10, 6))
```

```
plt.hist(df['Rating'], bins=10, color='orange', alpha=0.7)
plt.title('Distribution of Ratings')
plt.xlabel('Rating')
plt.ylabel('Frequency')
plt.show()
```



In [28]:
```
# The average discount percentage for each star rating
avg_discount_by_stars = df.groupby('Stars')['Discount Percentage'].mean()

# Plot the average discount percentage by star ratings
plt.figure(figsize=(10, 6))
avg_discount_by_stars.plot(kind='bar', color='purple')
plt.title('Average Discount Percentage by Star Rating')
plt.xlabel('Star Rating')
plt.ylabel('Average Discount (%)')
plt.show()
```



In [30]:
```
# Group by 'Product Name' and calculate the average rating
average_ratings = df.groupby('Product Name')['Rating'].mean().sort_values(ascending=False)

# Display the top products with the highest ratings
print(average_ratings.head())
```

```
Product Name
ASUS Chromebook Intel Celeron Dual Core N4500 - (4 GB/64 GB EMMC Storage/Chrome OS) CX1500CKA-NJ0
393 C...    997.0
ASUS Chromebook Intel Celeron Dual Core N4500 - (8 GB/128 GB EMMC Storage/Chrome OS) CX1500CKA-NJ
0395 ...    997.0
Lenovo ThinkBook 15 G3 AMD Ryzen 3 Quad Core 5300U - (8 GB/512 GB SSD/Windows 11 Home) TB15 G3 AC
L Thi...    997.0
MSI Modern 14 Intel Core i5 12th Gen 1235U - (8 GB/512 GB SSD/Windows 11 Home) Modern 14 C12M-440
IN / ...    912.0
MSI Modern 14 Intel Core i5 12th Gen 1235U - (16 GB/512 GB SSD/Windows 11 Home) Modern 14 C12M-43
9IN /...    912.0
Name: Rating, dtype: float64
```

ASSIGNMENT 3

CODE:

F:\Tej\Rollwala GU\SEM 7\DATA VIZUALISATION\assignment flask\1A.py:

```
from flask import Flask, render_template
import io
import pandas as p
import matplotlib.pyplot as plt
import base64


app = Flask(__name__)


@app.route('/')
def index():
    img1A = io.BytesIO()
    df = p.read_csv('employee_data.csv')
    best_salary_postion = df.groupby('Position')['Salary'].max().reset_index()
    best_salary_postion = best_salary_postion.sort_values(by = 'Salary', ascending = True)
    res = best_salary_postion.tail(1)
    plt.barh(best_salary_postion['Position'], best_salary_postion['Salary'], color= 'blue')
    plt.savefig(img1A, format='png')
    plt.close()
    img1A.seek(0)
    plot_url1A = base64.b64encode(img1A.getvalue()).decode('utf8')
    # data = {
    #     'plot_url1A' : 'plot_url1A',
```

```
    #     'text' : 'res',
    # }
    df = p.DataFrame({'plot_url1A':['plot_url1A'],
       'res1':['res']})
    print(df)
    print(best_salary_postion.tail(1))
    return render_template('1A.html', data = df)


if __name__ == '__main__':
    app.run(debug=True)
```

F:\Tej\Rollwala GU\SEM 7\DATA VIZUALISATION\assignment flask\index.py:

```
from flask import Flask, render_template

from flask import Flask, render_template

import io

import pandas as p

import matplotlib.pyplot as plt

import base64

import matplotlib

import pandas as pd

matplotlib.use('Agg')


app = Flask(__name__)




@app.route('/')

def index():

    return render_template('index.html')


@app.route('/1A')

def first():


    img1A = io.BytesIO()

    file_path = 'employee_data.csv'

    df = pd.read_csv(file_path)
```

```
    avg_salary = df.groupby('Position')['Salary'].mean()
    print(avg_salary)
    dat="Average salary by position"
    avg_salary.plot(kind='bar',title='Average salary by position')
    plt.xlabel('Position')
    plt.ylabel('Avg Salary')
    plt.title("Average salary by position")
    plt.savefig(img1A, format='png')
    plt.close()
    img1A.seek(0)
    plot_url1A= base64.b64encode(img1A.getvalue()).decode('utf8')
    return render_template('1A.html',plot_url1A=plot_url1A,dat=dat)


@app.route('/1B')
def second():


    img1A = io.BytesIO()
    file_path = 'employee_data.csv'
    df = pd.read_csv(file_path)
    gender_counts = df['Gender'].value_counts()
    print(gender_counts)
    dat="Number of Male and Female Employees"
    gender_counts.plot(kind='bar', color=['blue', 'pink'], title='Number of Male and Female Employees')
    plt.xlabel('Gender')
    plt.ylabel('Number of Employees')
```

```
    plt.xticks(rotation=0)
    plt.show()
    plt.title('Gender count')
    plt.savefig(img1A, format='png')
    plt.close()
    img1A.seek(0)
    plot_url1A = base64.b64encode(img1A.getvalue()).decode('utf8')
    return render_template('1A.html',plot_url1A=plot_url1A,dat=dat)


@app.route('/1C')
def third():
    img1A = io.BytesIO()
    df = p.read_csv('employee_data.csv')
    q3 = df[(df['Experience (Years)']>=10) & (df['Experience (Years)']<=15)]
    group_by_years = q3.groupby('Experience (Years)')['Salary'].mean().reset_index()
    print(group_by_years)
    dat="Salary earned by employees with 10 to 15 years of experience"
    x = group_by_years['Experience (Years)']
    y = group_by_years['Salary']
    plt.bar(x,y,color= 'brown')
    plt.xlabel('No. of Years')
    plt.ylabel('Salary')
    plt.title('Avg. Salary By Exp. Year')
    plt.savefig(img1A, format='png')
    plt.close()
```

```
    img1A.seek(0)
    plot_url1A = base64.b64encode(img1A.getvalue()).decode('utf8')
    return render_template('1A.html',plot_url1A=plot_url1A,dat=dat)


@app.route('/1D')
def fourth():
    img1A = io.BytesIO()
    file_path = 'employee_data.csv'
    df = pd.read_csv(file_path)
    dat="Bar chart for number of positions in company"
    position_counts = df['Position'].value_counts()
    position_counts.plot(kind='bar', color='skyblue', title='Number of Employees in Each Position')
    plt.xlabel('Position')
    plt.ylabel('Number of Employees')
    plt.xticks(rotation=45)
    plt.show()
    plt.title('bar char for number of positions in company')
    plt.savefig(img1A, format='png')
    plt.close()
    img1A.seek(0)
    plot_url1A = base64.b64encode(img1A.getvalue()).decode('utf8')
    return render_template('1A.html',plot_url1A=plot_url1A,dat=dat)


@app.route('/1E')
def fifth():
```

```
    img1A = io.BytesIO()
    file_path = 'employee_data.csv'
    df = pd.read_csv(file_path)
    dat="Best Position in Terms of Salary"
    avg_salary_by_position = df.groupby('Position')['Salary'].mean()
    avg_salary_by_position.plot(kind='bar', color='lightgreen', title='Average Salary by Position')
    plt.xlabel('Position')
    plt.ylabel('Average Salary')
    plt.xticks(rotation=45)
    plt.show()
    plt.title('Best Position in Terms of Salary')
    plt.tight_layout()
    plt.savefig(img1A, format='png')
    plt.close()

    img1A.seek(0)
    plot_url1A = base64.b64encode(img1A.getvalue()).decode('utf8')

    return render_template('1A.html', plot_url1A=plot_url1A,dat=dat)

@app.route('/dashboard')
def dashboard():
     five=fifth()
     return render_template('dashboard.html',five=five)
```

```
@app.route('/2A')
def question():
    img2A = io.BytesIO()
    img2B = io.BytesIO()

    file_path = 'Flipkart-Laptops.csv'
    df = pd.read_csv(file_path)

    df['Actual price'] = pd.to_numeric(df['Actual price'].str.replace(',', ''), errors='coerce')
    df['Discount price'] = pd.to_numeric(df['Discount price'].str.replace(',', ''), errors='coerce')
    df['Stars'] = pd.to_numeric(df['Stars'], errors='coerce')
    df['Rating'] = pd.to_numeric(df['Rating'].str.extract(r'(\d+)')[0].str.replace(',', ''), errors='coerce')
    df['Reviews'] = pd.to_numeric(df['Reviews'].str.extract(r'(\d+)')[0], errors='coerce')


    df['Discount Percentage'] = ((df['Actual price'] - df['Discount price']) / df['Actual price']) * 100


    plt.figure(figsize=(10, 6))
    plt.hist(df['Actual price'].dropna(), bins=30, color='blue', alpha=0.7)
    plt.title('Distribution of Actual Prices')
    plt.xlabel('Price (in INR)')
    plt.ylabel('Frequency')
    plt.tight_layout()
    plt.savefig(img2A, format='png')
```

```
    plt.close()
    img2A.seek(0)
    plot_url1A = base64.b64encode(img2A.getvalue()).decode('utf8')



    plt.figure(figsize=(10, 6))
    plt.hist(df['Discount price'].dropna(), bins=30, color='green', alpha=0.7)
    plt.title('Distribution of Discount Prices')
    plt.xlabel('Discount Price (in INR)')
    plt.ylabel('Frequency')
    plt.tight_layout()
    plt.savefig(img2B, format='png')
    plt.close()
    img2B.seek(0)
    plot_url2B = base64.b64encode(img2B.getvalue()).decode('utf8')

    return render_template('2A.html', plot_url1A=plot_url1A, plot_url2B=plot_url2B)




if __name__ == '__main__':
    app.run(debug=True)
```

F:\Tej\Rollwala GU\SEM 7\DATA VIZUALISATION\assignment flask\templates\1A.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Employee Salary Analysis</title>
   <link rel="stylesheet" href="styles.css">
</head>
<body>
   <div class="container">

      <div class="chart">
         <h2>{{dat}}</h2>
         <img src="{{ plot_url1A }}" alt="Bar chart">
      </div>




   </div>
</body>
</html>
```

F:\Tej\Rollwala GU\SEM 7\DATA VIZUALISATION\assignment flask\templates\2A.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Graphs</title>
</head>
<body>
    <h1>Graphs from Flipkart Laptops Data</h1>
    <h2>Distribution of Actual Prices</h2>
    <img src=" {{ plot_url1A }}" alt="Distribution of Actual Prices">

    <h2>Distribution of Discount Prices</h2>
    <img src=" {{ plot_url2B }}" alt="Distribution of Discount Prices">
</body>
</html>
```

F:\Tej\Rollwala GU\SEM 7\DATA VIZUALISATION\assignment flask\templates\index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Data Analysis - Home</title>
    <link rel="stylesheet" href="styles.css">
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f4f4f4;
        }

        .container {
            width: 80%;
            margin: auto;
            padding: 20px;
            background: white;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }

        h1 {
```

```
    text-align: center;
    color: #333;
}

h2 {
    text-align: center;
    color: #555;
    margin-bottom: 30px;
}

ul {
    list-style-type: none;
    padding: 0;
}

li {
    margin-bottom: 20px;
}

h3 {
    color: #444;
    margin: 0 0 10px 0;
    text-align: left;
    font-size: 1.2em;
}
```

```
section {
    margin-bottom: 30px;
    padding: 10px;
    border-bottom: 1px solid #ccc;
}

button {
    display: inline-block;
    padding: 10px 20px;
    background-color: #007bff;
    color: white;
    border: none;
    cursor: pointer;
    font-size: 16px;
    text-align: center;
    transition: background-color 0.3s ease;
}

button:hover {
    background-color: #0056b3;
}

button a {
    color: white;
```

```
        text-decoration: none;
        display: block;
    }

    img {
        display: block;
        margin: 20px auto;
        max-width: 100%;
        height: auto;
    }

    p {
        font-size: 1.2em;
        color: #555;
    }

    .hidden {
        display: none;
    }
  </style>
</head>
<body>
  <div class="container">
    <h1>Data Analysis</h1>
```

```
<ul>
  <li>
    <h3>Q1: Display Average Salary of each Position and plot it on a graph.</h3>
    <button><a href="{{url_for('first')}}">Click</a></button>
  </li>
  <li>
    <h3>Q2: Display total number of Male and Female employees and plot it on a chart.</h3>
    <button><a href="{{url_for('second')}}">Click</a></button>
  </li>
  <li>
    <h3>Q3: Show a chart displaying the salary earned by employees with 10 to 15 years of experience.</h3>
    <button><a href="{{url_for('third')}}">Click</a></button>
  </li>
  <li>
    <h3>Q4: Display a bar chart of the number of positions in the company.</h3>
    <button><a href="{{url_for('fourth')}}">Click</a></button>
  </li>
  <li>
    <h3>Q5: Analyze which position is better in terms of salary.</h3>
    <button><a href="{{url_for('fifth')}}">Click</a></button>
  </li>
  <li>
    <h3>Flipkart Laptops Analysis: Price Distribution</h3>
    <button><a href="{{url_for('question')}}">Click</a></button>
```

```
            </li>
        </ul>
    </div>
</body>
</html>
```

Output:

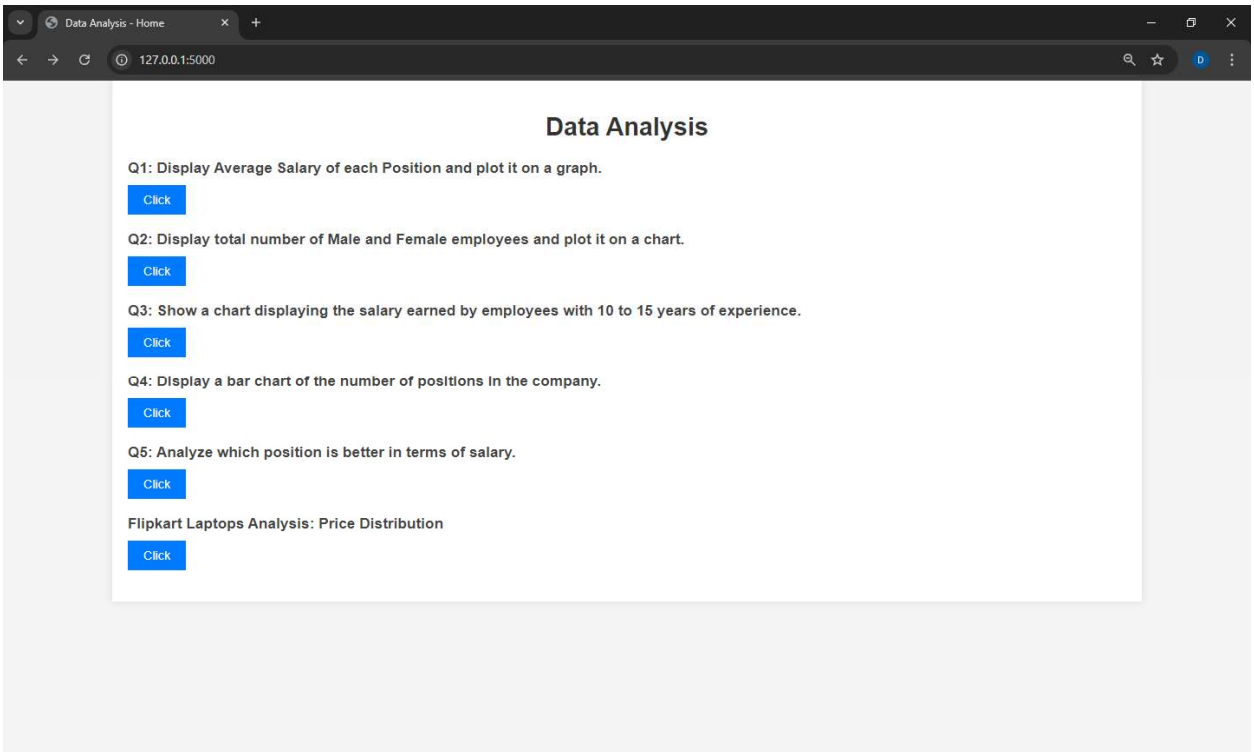


Figure 1: Home Page

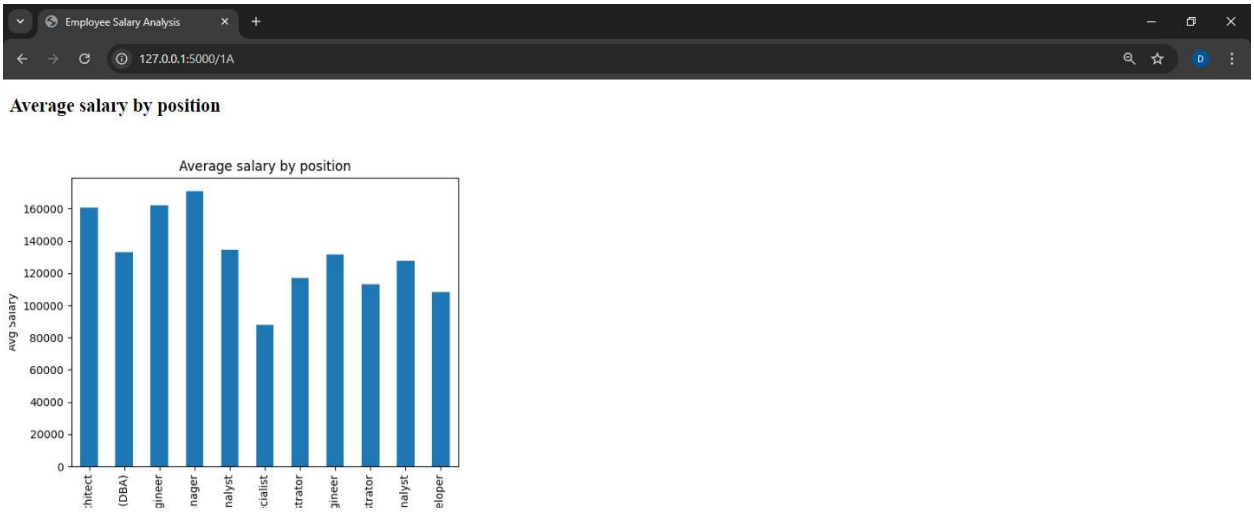


Figure 2:Q1

Figure 3:Q2

Figure 4:Q3

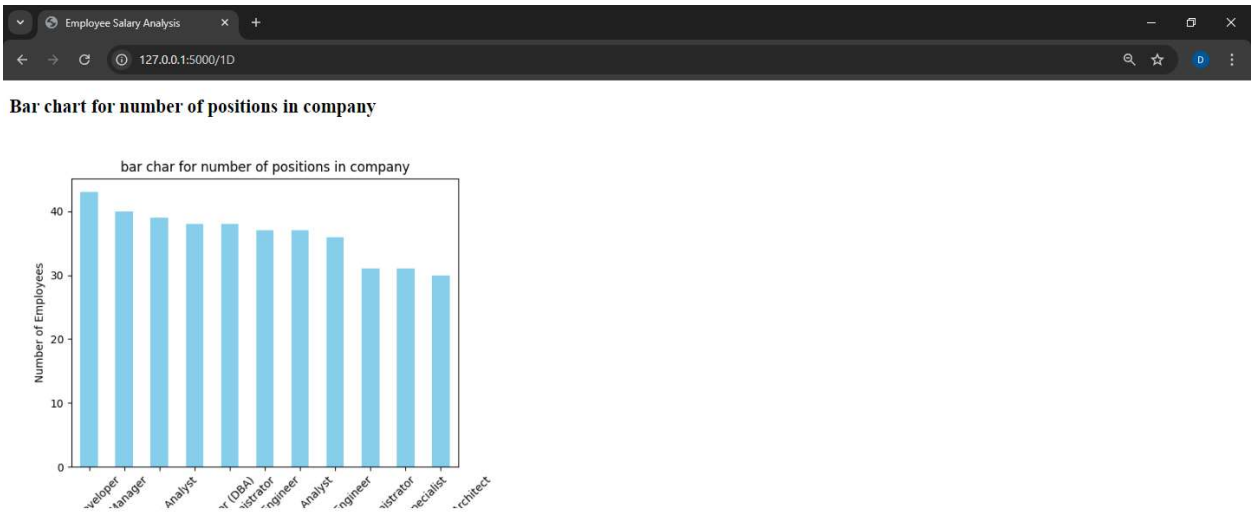


Figure 5:Q4

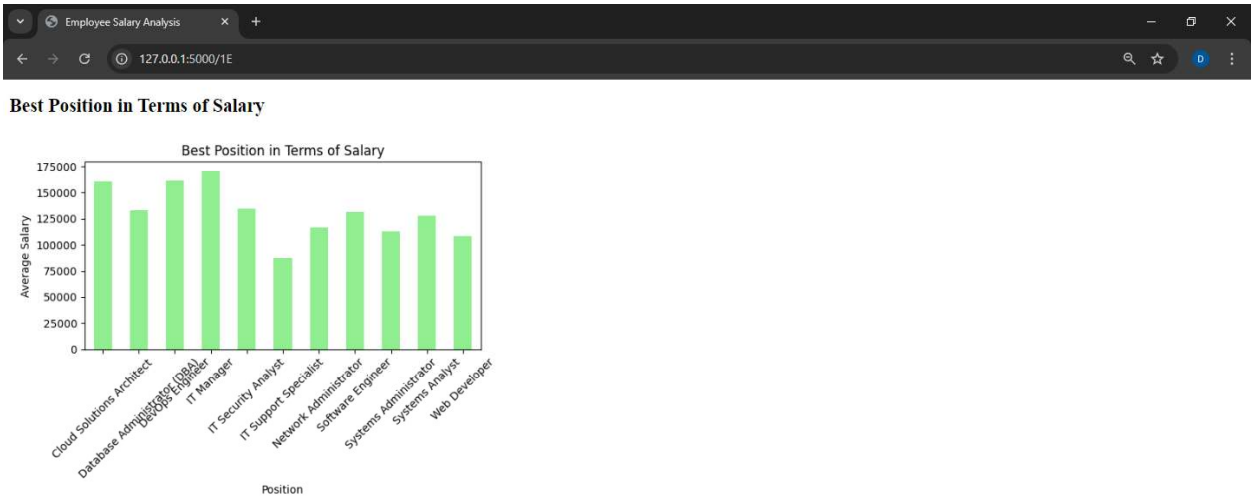


Figure 6:Q5

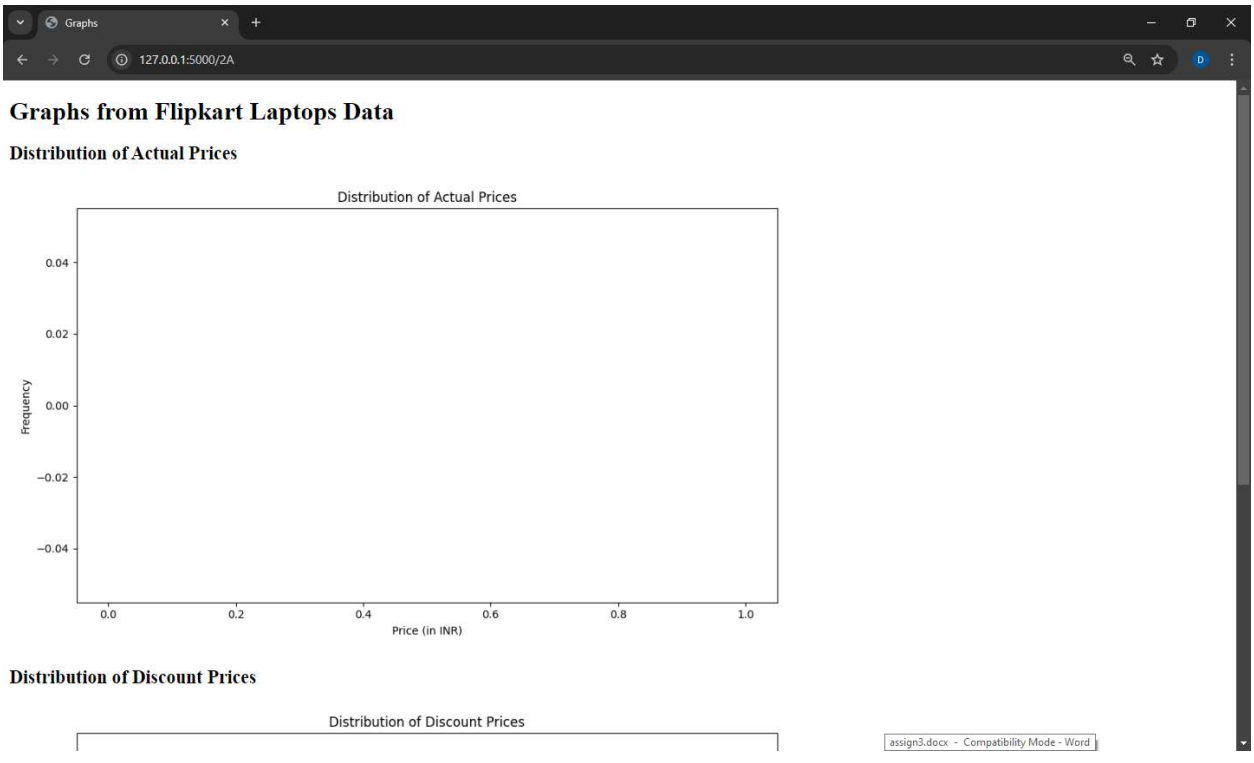
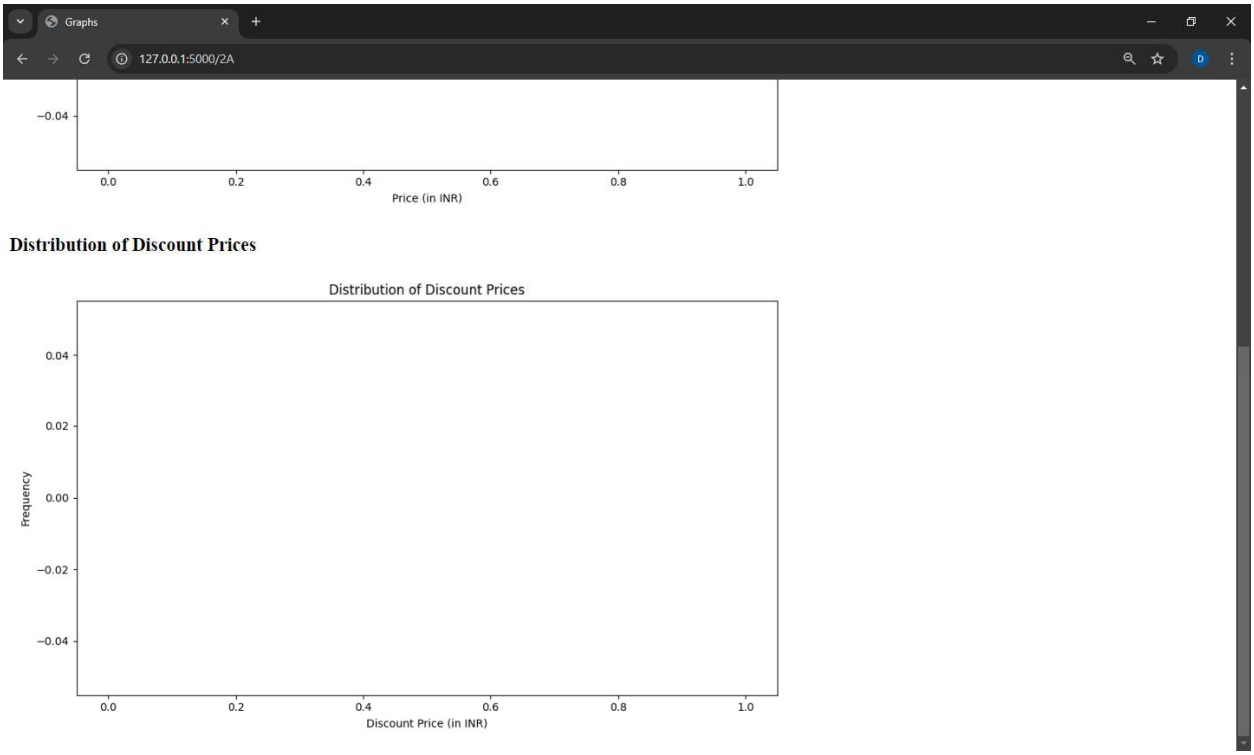


Figure 7: FLIPKART QUESTION



Distribution of Discount Prices

ASSIGNMENT 4

ASSIGNMENT 4

QUESTION:

Use StudentsPerformance Dataset and Perform following Task:

Generate a dashboard on webpage which represent following

1. Pie Chart for How many students got more than 70 score, between 60 to 70, between 40 to 60 and below 40 in math. When cursor is above part of chart it should display DIST for above 70, First for below 70 and above 60, and so on.

2. Pie Chart for How many students got more than 70 score, between 60 to 70, between 40 to 60 and below 40 in reading. When cursor is above part of chart it should display DIST for above 70, First for below 70 and above 60, and so on.

3. Pie Chart for How many students got more than 70 score, between 60 to 70, between 40 to 60 and below 40 in writing. When cursor is above part of chart it should display DIST for above 70, First for below 70 and above 60, and so on.

4. Bar chart for how many student completed test preparation course and how many student not completed test preparation course.

5. Line chart for math score of 20 to 30 roll nos.

6. Display the count with proper design that how many students parent having bachelor and master degree.

7. In dataset, replace data of lunch for free/reduced to premium and display the count of no of students who choose standard lunch and premium lunch.

Code:

F:\Tej\Rollwala GU\SEM 7\DATA VIZUALISATION\assign4\app.py:

```
from flask import Flask, render_template

import io

import pandas as pd
```

```
import matplotlib.pyplot as plt

import base64

import matplotlib

matplotlib.use('Agg')


app = Flask(__name__)



student_data = pd.read_csv("./StudentsPerformance.csv")



def plot_to_base64(img_io):

    img_io.seek(0)

    return base64.b64encode(img_io.getvalue()).decode('utf8')


@app.route('/')

def dashboard():

    # 1

    img1 = io.BytesIO()

    more_than_70 = student_data[student_data['math score'] > 70].count()[0]

    between_60_70 = student_data[(student_data['math score'] >= 60) & (student_data['math score'] <=
70)].count()[0]

    between_40_60 = student_data[(student_data['math score'] >= 40) & (student_data['math score'] <
60)].count()[0]

    below_40 = student_data[student_data['math score'] < 40].count()[0]
```

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

    plt.pie([more_than_70, between_60_70, between_40_60, below_40], labels=['Distinction', 'First Class', 'Second Class', 'Third Class'], autopct='%1.1f%%')

    plt.title('Distribution of Math Scores')

    plt.savefig(img1, format='png')

    plt.close()

    math_pie = plot_to_base64(img1)


    # 2

    img2 = io.BytesIO()

    more_than_70 = student_data[student_data['reading score'] > 70].count()[0]

    between_60_70 = student_data[(student_data['reading score'] >= 60) & (student_data['reading score'] <= 70)].count()[0]

    between_40_60 = student_data[(student_data['reading score'] >= 40) & (student_data['reading score'] < 60)].count()[0]

    below_40 = student_data[student_data['reading score'] < 40].count()[0]


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

    plt.pie([more_than_70, between_60_70, between_40_60, below_40], labels=['Distinction', 'First Class', 'Second Class', 'Third Class'], autopct='%1.1f%%')

    plt.title('Distribution of Reading Scores')

    plt.savefig(img2, format='png')

    plt.close()

    reading_pie = plot_to_base64(img2)


    # 3

    img3 = io.BytesIO()
```

```
    more_than_70 = student_data[student_data['writing score'] > 70].count()[0]

    between_60_70 = student_data[(student_data['writing score'] >= 60) & (student_data['writing score'] <= 70)].count()[0]

    between_40_60 = student_data[(student_data['writing score'] >= 40) & (student_data['writing score'] < 60)].count()[0]

    below_40 = student_data[student_data['writing score'] < 40].count()[0]


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

    plt.pie([more_than_70, between_60_70, between_40_60, below_40], labels=['Distinction', 'First Class', 'Second Class', 'Third Class'], autopct='%1.1f%%')

    plt.title('Distribution of Writing Scores')

    plt.savefig(img3, format='png')

    plt.close()

    writing_pie = plot_to_base64(img3)


    # 4

    img4 = io.BytesIO()

    completed = student_data[student_data['test preparation course'] == 'completed'].count()[0]

    not_completed = student_data[student_data['test preparation course'] == 'none'].count()[0]


    plt.figure(figsize=(6, 4))

    plt.bar(['Completed', 'Not Completed'], [completed, not_completed], color=['green', 'red'])

    plt.title('Test Preparation Course Completion')

    plt.ylabel('Count')

    plt.savefig(img4, format='png')

    plt.close()

    test_preparation_bar = plot_to_base64(img4)
```

```
    # 5
    img5 = io.BytesIO()
    filtered_data = student_data[(student_data['Roll No'] >= 20) & (student_data['Roll No'] <= 30)]
    plt.figure(figsize=(6, 4))
    plt.plot(filtered_data['Roll No'], filtered_data['math score'], marker='o', linestyle='-', color='blue')
    plt.title('Math Scores (Roll Nos 20-30)')
    plt.xlabel('Roll No')
    plt.ylabel('Math Score')
    plt.savefig(img5, format='png')
    plt.close()
    roll_no_line = plot_to_base64(img5)

    # 6
    img6 = io.BytesIO()
    bachelor_count = student_data[student_data['parental level of education'] == 'bachelor\'s degree'].count()[0]
    master_count = student_data[student_data['parental level of education'] == 'master\'s degree'].count()[0]

    plt.figure(figsize=(6, 4))
    plt.bar(['Bachelor\'s Degree', 'Master\'s Degree'], [bachelor_count, master_count], color=['blue', 'orange'])
    plt.title('Count of Students by Parental Education Level')
    plt.ylabel('Count')
    plt.savefig(img6, format='png')
    plt.close()
```

```
    parental_education_bar = plot_to_base64(img6)


    # 7
    img7 = io.BytesIO()
    student_data['lunch'] = student_data['lunch'].replace('free/reduced', 'premium')
    standard_count = student_data[student_data['lunch'] == 'standard'].count()[0]
    premium_count = student_data[student_data['lunch'] == 'premium'].count()[0]


    plt.figure(figsize=(6, 4))
    plt.bar(['Standard Lunch', 'Premium Lunch'], [standard_count, premium_count], color=['green',
'purple'])
    plt.title('Count of Students by Lunch Type')
    plt.ylabel('Count')
    plt.savefig(img7, format='png')
    plt.close()
    lunch_bar = plot_to_base64(img7)


    return render_template('index.html',
                           math_pie=math_pie,
                           reading_pie=reading_pie,
                           writing_pie=writing_pie,
                           test_preparation_bar=test_preparation_bar,
                           roll_no_line=roll_no_line,
                           parental_education_bar=parental_education_bar,
                           lunch_bar=lunch_bar)
```

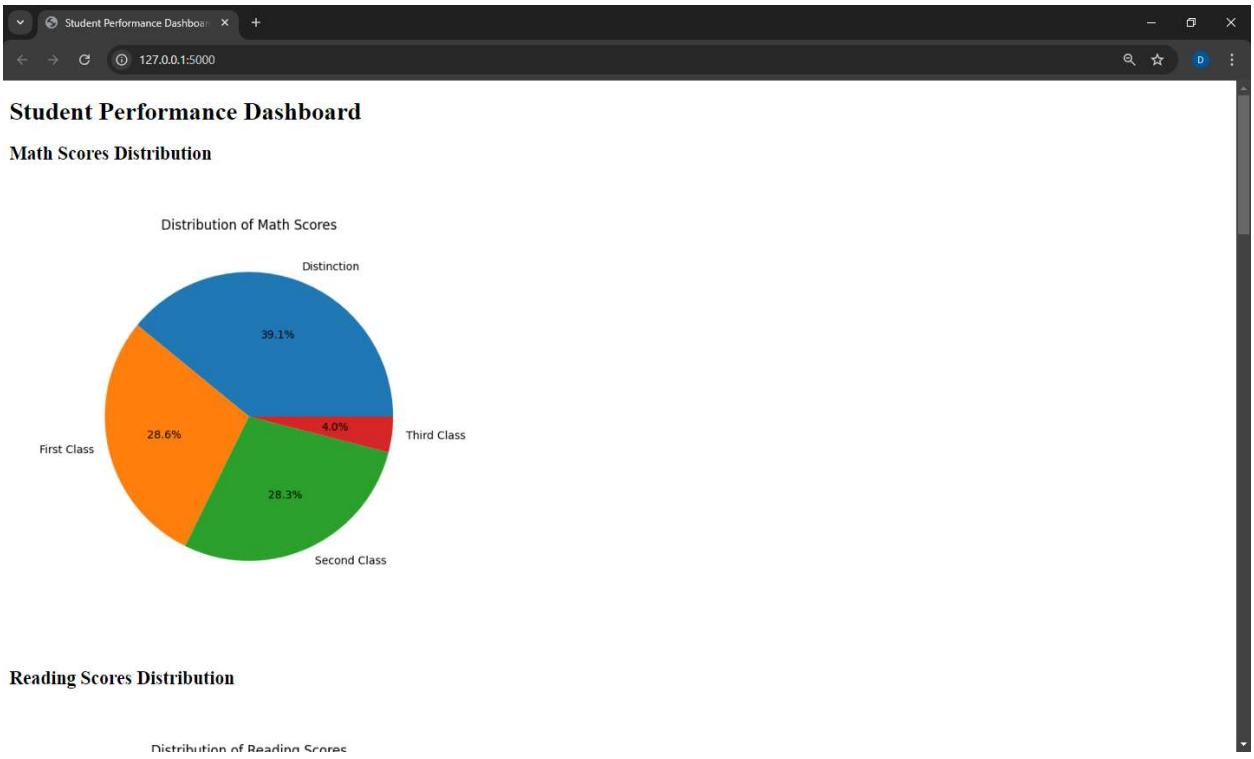
```
if __name__ == '__main__':
    app.run(debug=True)
```

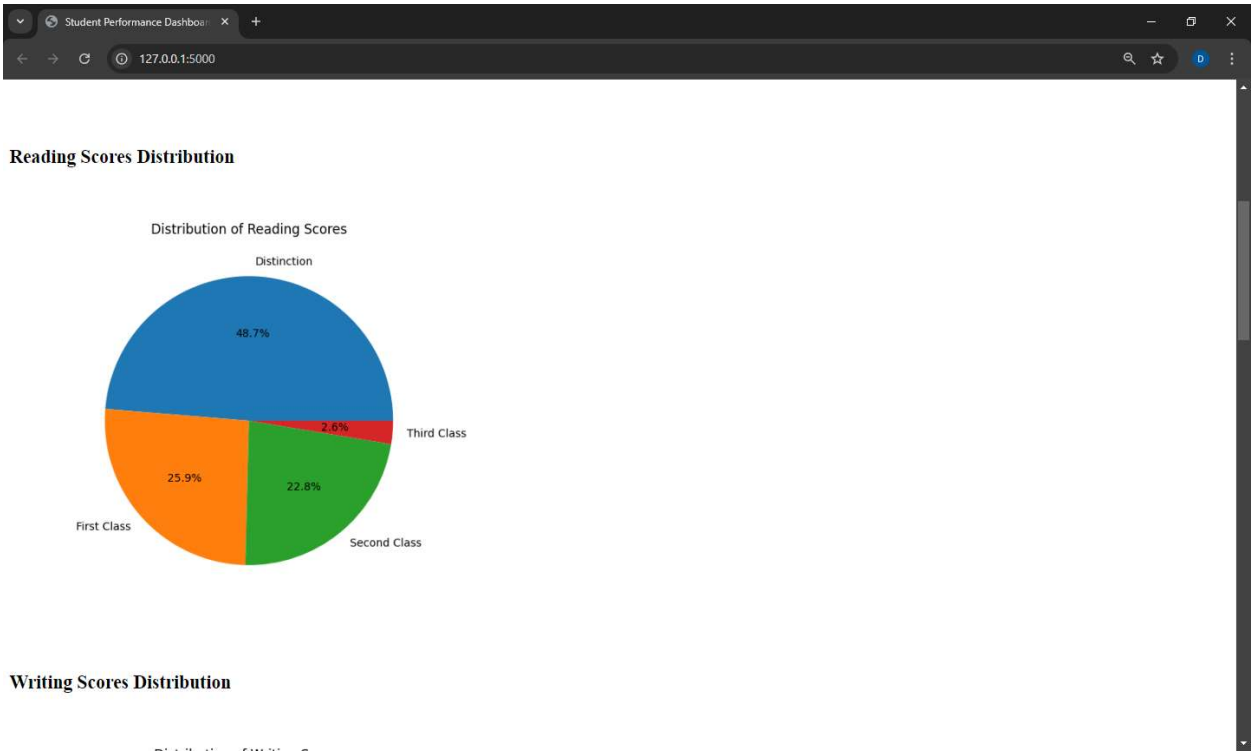
F:\Tej\Rollwala GU\SEM 7\DATA VIZUALISATION\assign4\templates\index.html:

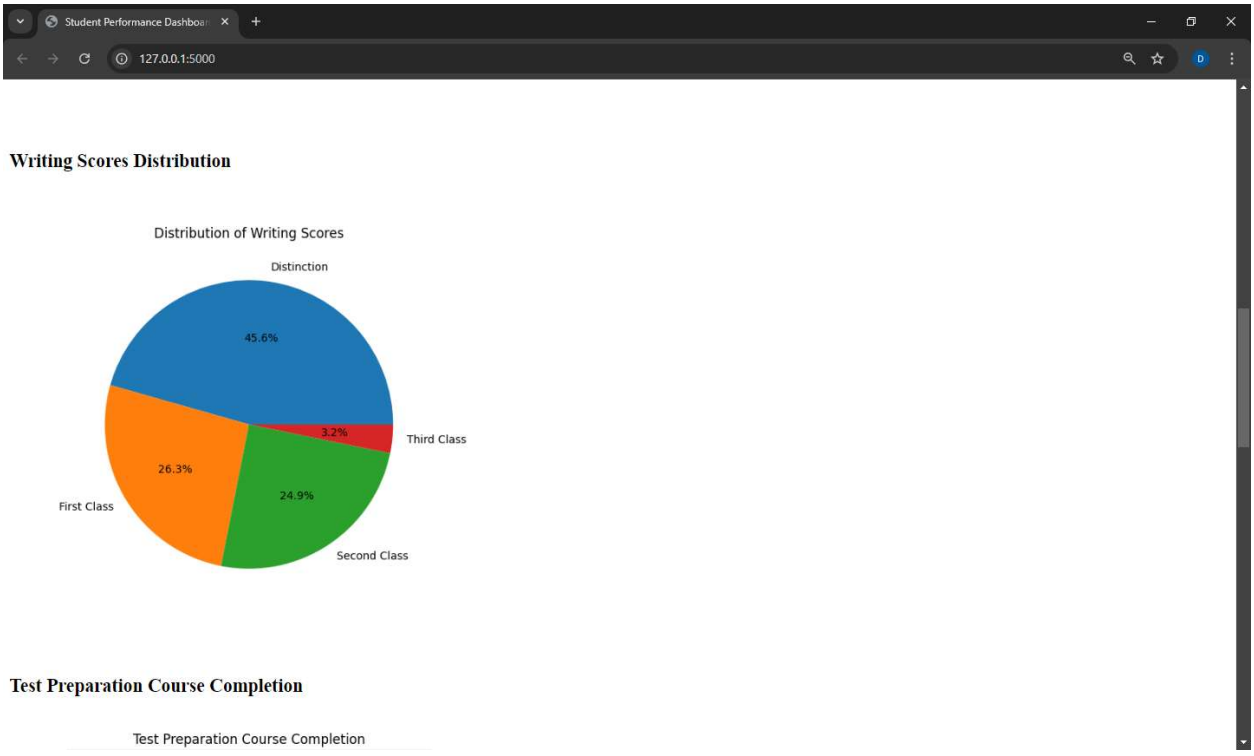
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Performance Dashboard</title>
</head>
<body>
    <h1>Student Performance Dashboard</h1>

    <h2>Math Scores Distribution</h2>
    <img src="{{ math_pie }}" alt="Math Pie Chart">

    <h2>Reading Scores Distribution</h2>
    <img src="{{ reading_pie }}" alt="Reading Pie Chart">

    <h2>Writing Scores Distribution</h2>
    <img src="{{ writing_pie }}" alt="Writing Pie Chart">

    <h2>Test Preparation Course Completion</h2>
```

```
    <img src="{{ test_preparation_bar }}" alt="Test Preparation Bar Chart">

    <h2>Math Scores (Roll Nos 20-30)</h2>
    <img src="{{ roll_no_line }}" alt="Math Scores Line Chart">

    <h2>Parental Education Level</h2>
    <img src="{{ parental_education_bar }}" alt="Parental Education Bar Chart">

    <h2>Lunch Type Distribution</h2>
    <img src="{{ lunch_bar }}" alt="Lunch Type Bar Chart">
</body>
</html>
```

Output:

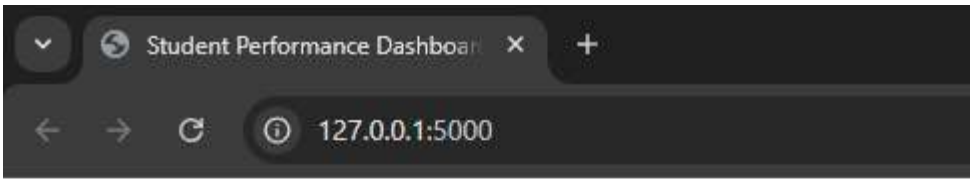



Student Performance Dashboa
127.0.0.1:5000
Reading Scores Distribution
Distribution of Reading Scores
Distinction
48.7%
2.6%
Third Class
25.9%
22.8%
First Class
Second Class
Writing Scores Distribution

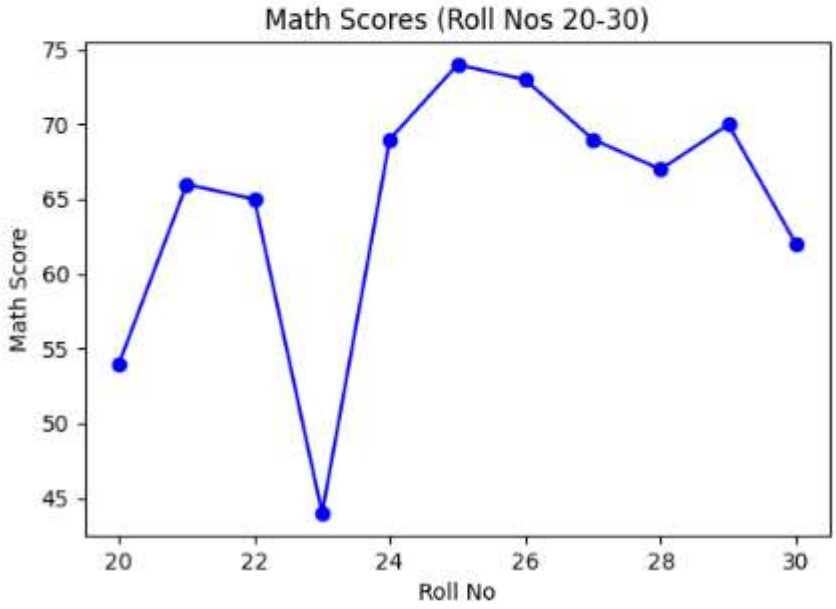

Student Performance Dashboa
127.0.0.1:5000
Writing Scores Distribution
Distribution of Writing Scores
Distinction
45.6%
3.2%
Third Class
26.3%
24.9%
First Class
Second Class
Test Preparation Course Completion
Test Preparation Course Completion

Test Preparation Course Completion

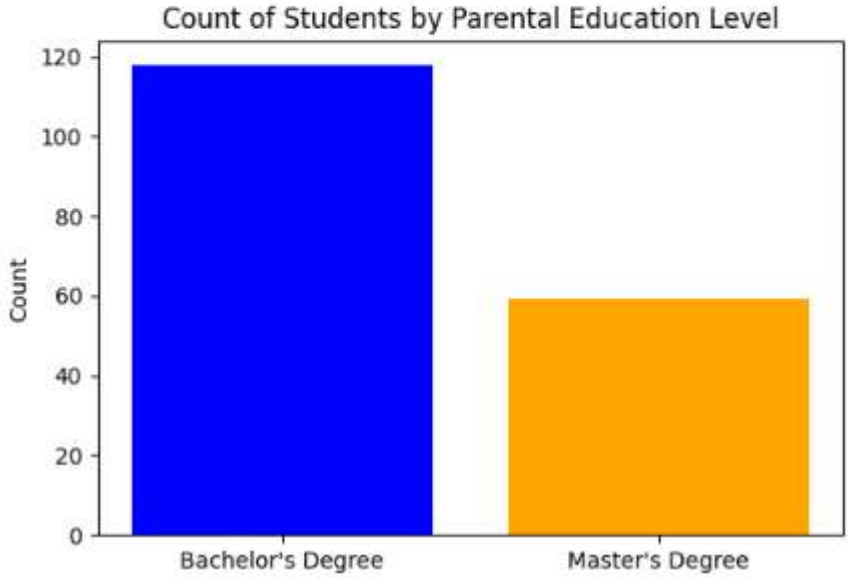
Math Scores (Roll Nos 20-30)



Math Scores (Roll Nos 20-30)



Parental Education Level



Lunch Type Distribution