Analyzing Billionaires Statistics Dataset (2023) with Python

Problem Statement This dataset contains statistics on the world's billionaires, including information about their businesses, industries, and personal details. It provides insights into the wealth distribution, business sectors, and demographics of billionaires worldwide.

Import Library

In [1]:
```
import pandas as pd
```

In [2]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import seaborn as sns
```

```
C:\Users\Syed Arif\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWar
ning: A NumPy version >=1.16.5 and <1.23.0 is required for this version of Sc
iPy (detected version 1.25.1
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

Uploading Csv fle

In [3]:
```
df = pd.read_csv(r"Billionaires.csv")
```

Data Preprocessing

.head()

head is used show to the By default = 5 rows in the dataset

In [4]:
```
df.head()
```

Out[4]:

	rank	finalWorth	category	personName	age	country	city	source	industries	cou
0	1	211000	Fashion & Retail	Bernard Arnault & family	74.0	France	Paris	LVMH	Fashion & Retail	
1	2	180000	Automotive	Elon Musk	51.0	United States	Austin	Tesla, SpaceX	Automotive	
2	3	114000	Technology	Jeff Bezos	59.0	United States	Medina	Amazon	Technology	
3	4	107000	Technology	Larry Ellison	78.0	United States	Lanai	Oracle	Technology	
4	5	106000	Finance & Investments	Warren Buffett	92.0	United States	Omaha	Berkshire Hathaway	Finance & Investments	

5 rows × 35 columns

.tail()

tail is used to show rows by Descending order

In [5]:
```
df.tail()
```

Out[5]:

	rank	finalWorth	category	personName	age	country	city	source	
2635	2540	1000	Healthcare	Yu Rong	51.0	China	Shanghai	Health clinics	
2636	2540	1000	Food & Beverage	Richard Yuengling, Jr.	80.0	United States	Pottsville	Beer	
2637	2540	1000	Manufacturing	Zhang Gongyun	60.0	China	Gaomi	Tyre manufacturing machinery	Mar
2638	2540	1000	Real Estate	Zhang Guiping & family	71.0	China	Nanjing	Real estate	F
2639	2540	1000	Diversified	Inigo Zobel	66.0	Philippines	Makati	Diversified	

5 rows × 35 columns

.shape

It show the total no of rows & Column in the dataset

In [6]:
```
df.shape
```

Out[6]:
```
(2640, 35)
```

.Columns

It show the no of each Column

In [7]:
```
df.columns
```

Out[7]:
```
Index(['rank', 'finalWorth', 'category', 'personName', 'age', 'country',
       'city', 'source', 'industries', 'countryOfCitizenship', 'organizatio
n',
       'selfMade', 'status', 'gender', 'birthDate', 'lastName', 'firstName',
       'title', 'date', 'state', 'residenceStateRegion', 'birthYear',
       'birthMonth', 'birthDay', 'cpi_country', 'cpi_change_country',
       'gdp_country', 'gross_tertiary_education_enrollment',
       'gross_primary_education_enrollment_country', 'life_expectancy_countr
y',
       'tax_revenue_country_country', 'total_tax_rate_country',
       'population_country', 'latitude_country', 'longitude_country'],
      dtype='object')
```

.dtypes

This Attribute show the data type of each column

In [8]:
```
df.dtypes
```

Out[8]:
```
rank                                          int64
finalWorth                                    int64
category                                     object
personName                                   object
age                                         float64
country                                      object
city                                         object
source                                       object
industries                                   object
countryOfCitizenship                         object
organization                                 object
selfMade                                       bool
status                                       object
gender                                       object
birthDate                                    object
lastName                                     object
firstName                                    object
title                                        object
date                                         object
state                                        object
residenceStateRegion                         object
birthYear                                   float64
birthMonth                                  float64
birthDay                                    float64
cpi_country                                 float64
cpi_change_country                          float64
gdp_country                                  object
gross_tertiary_education_enrollment         float64
gross_primary_education_enrollment_country  float64
life_expectancy_country                     float64
tax_revenue_country_country                 float64
total_tax_rate_country                      float64
population_country                          float64
latitude_country                            float64
longitude_country                           float64
dtype: object
```

.unique()

In a column, It show the unique value of specific column.

In [9]:
```
df["category"].unique()
```

Out[9]:
```
array(['Fashion & Retail', 'Automotive', 'Technology',
       'Finance & Investments', 'Media & Entertainment', 'Telecom',
       'Diversified', 'Food & Beverage', 'Logistics',
       'Gambling & Casinos', 'Manufacturing', 'Real Estate',
       'Metals & Mining', 'Energy', 'Healthcare', 'Service',
       'Construction & Engineering', 'Sports'], dtype=object)
```

.nuique()

It will show the total no of unque value from whole data frame

In [10]:
```
df.nunique()
```

Out[10]:
```
rank                                          219
finalWorth                                    219
category                                       18
personName                                   2638
age                                            79
country                                        78
city                                          741
source                                        906
industries                                     18
countryOfCitizenship                           77
organization                                  294
selfMade                                        2
status                                          6
gender                                          2
birthDate                                    2060
lastName                                     1736
firstName                                    1770
title                                          97
date                                            2
state                                          45
residenceStateRegion                            5
birthYear                                      77
birthMonth                                     12
birthDay                                       31
cpi_country                                    63
cpi_change_country                             44
gdp_country                                    68
gross_tertiary_education_enrollment            63
gross_primary_education_enrollment_country     60
life_expectancy_country                        54
tax_revenue_country_country                    57
total_tax_rate_country                         63
population_country                             68
latitude_country                               68
longitude_country                              68
dtype: int64
```

.describe()

It show the Count, mean , median etc

In [11]:
```
df.describe()
```

Out[11]:

	rank	finalWorth	age	birthYear	birthMonth	birthDay	cpi_cou
count	2640.000000	2640.000000	2575.000000	2564.000000	2564.000000	2564.000000	2456.00(
mean	1289.159091	4623.787879	65.140194	1957.183307	5.740250	12.099844	127.75
std	739.693726	9834.240939	13.258098	13.282516	3.710085	9.918876	26.45
min	1.000000	1000.000000	18.000000	1921.000000	1.000000	1.000000	99.55(
25%	659.000000	1500.000000	56.000000	1948.000000	2.000000	1.000000	117.24(
50%	1312.000000	2300.000000	65.000000	1957.000000	6.000000	11.000000	117.24(
75%	1905.000000	4200.000000	75.000000	1966.000000	9.000000	21.000000	125.08(
max	2540.000000	211000.000000	101.000000	2004.000000	12.000000	31.000000	288.57(

.value_counts

It Shows all the unique values with their count

In [12]:
```
df["category"].value_counts()
```

Out[12]:
```
Finance & Investments          372
Manufacturing                  324
Technology                     314
Fashion & Retail               266
Food & Beverage                212
Healthcare                     201
Real Estate                    193
Diversified                    187
Energy                         100
Media & Entertainment           91
Metals & Mining                 74
Automotive                      73
Service                         53
Construction & Engineering      45
Logistics                       40
Sports                          39
Telecom                         31
Gambling & Casinos              25
Name: category, dtype: int64
```

.isnull()

It shows the how many null values

In [13]:
```
df.isnull()
```

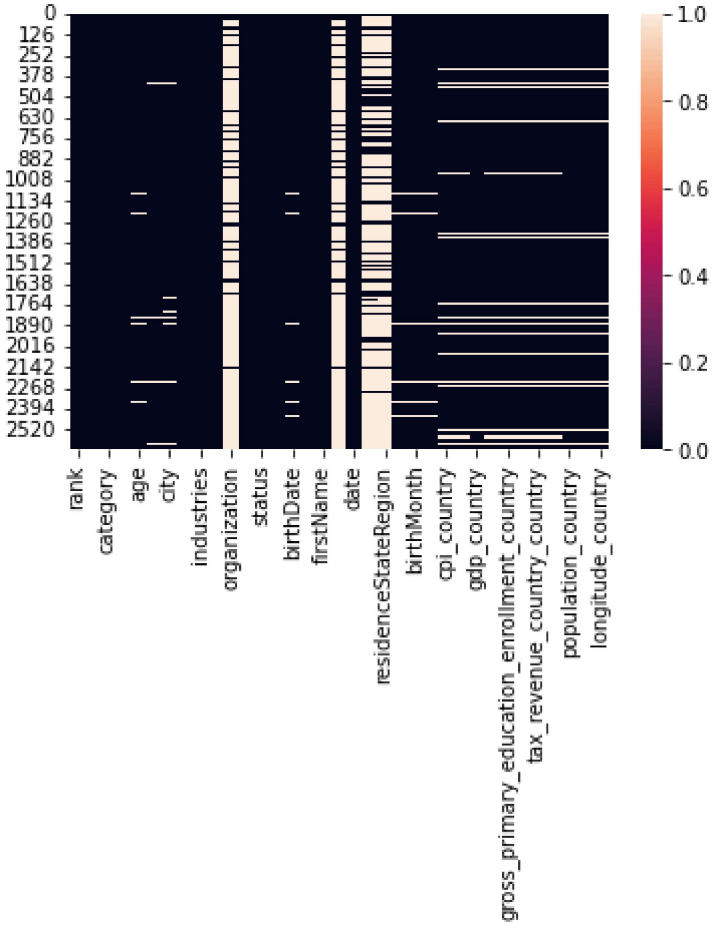
Out[13]:

	rank	finalWorth	category	personName	age	country	city	source	industries	country
0	False	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	
2635	False	False	False	False	False	False	False	False	False	
2636	False	False	False	False	False	False	False	False	False	
2637	False	False	False	False	False	False	False	False	False	
2638	False	False	False	False	False	False	False	False	False	
2639	False	False	False	False	False	False	False	False	False	

2640 rows × 35 columns

In [14]:
```
sns.heatmap(df.isnull())
```

Out[14]:
```
<AxesSubplot:>
```



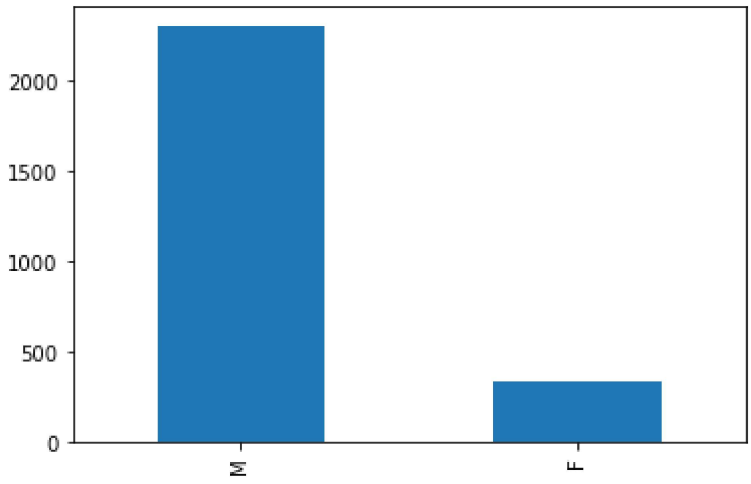
Count Of Gender

In [58]:
```
gender_diversity = df['gender'].value_counts()
gender_diversity
```

Out[58]:
```
M    2303
F     337
Name: gender, dtype: int64
```

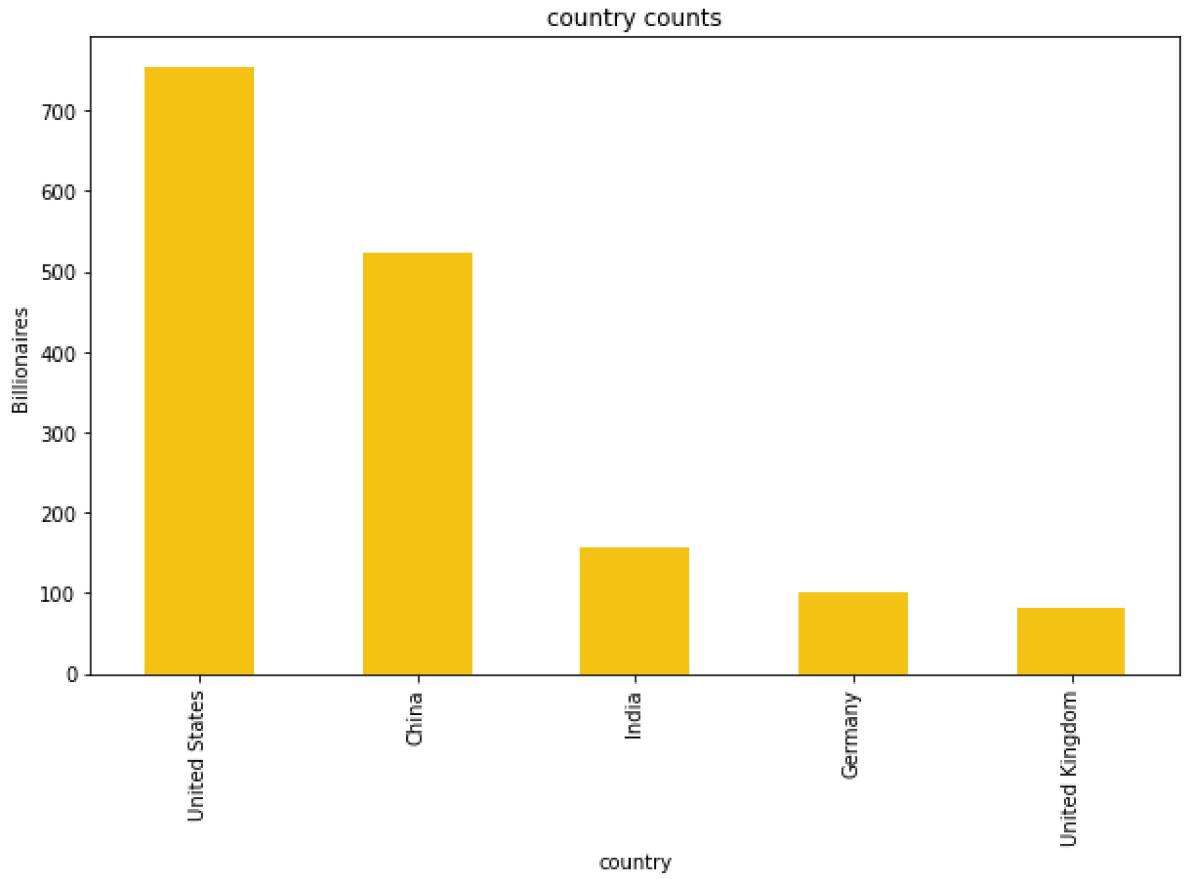
In [77]:
```
df.gender.value_counts().plot(kind= 'bar')
```

Out[77]:
```
<AxesSubplot:>
```



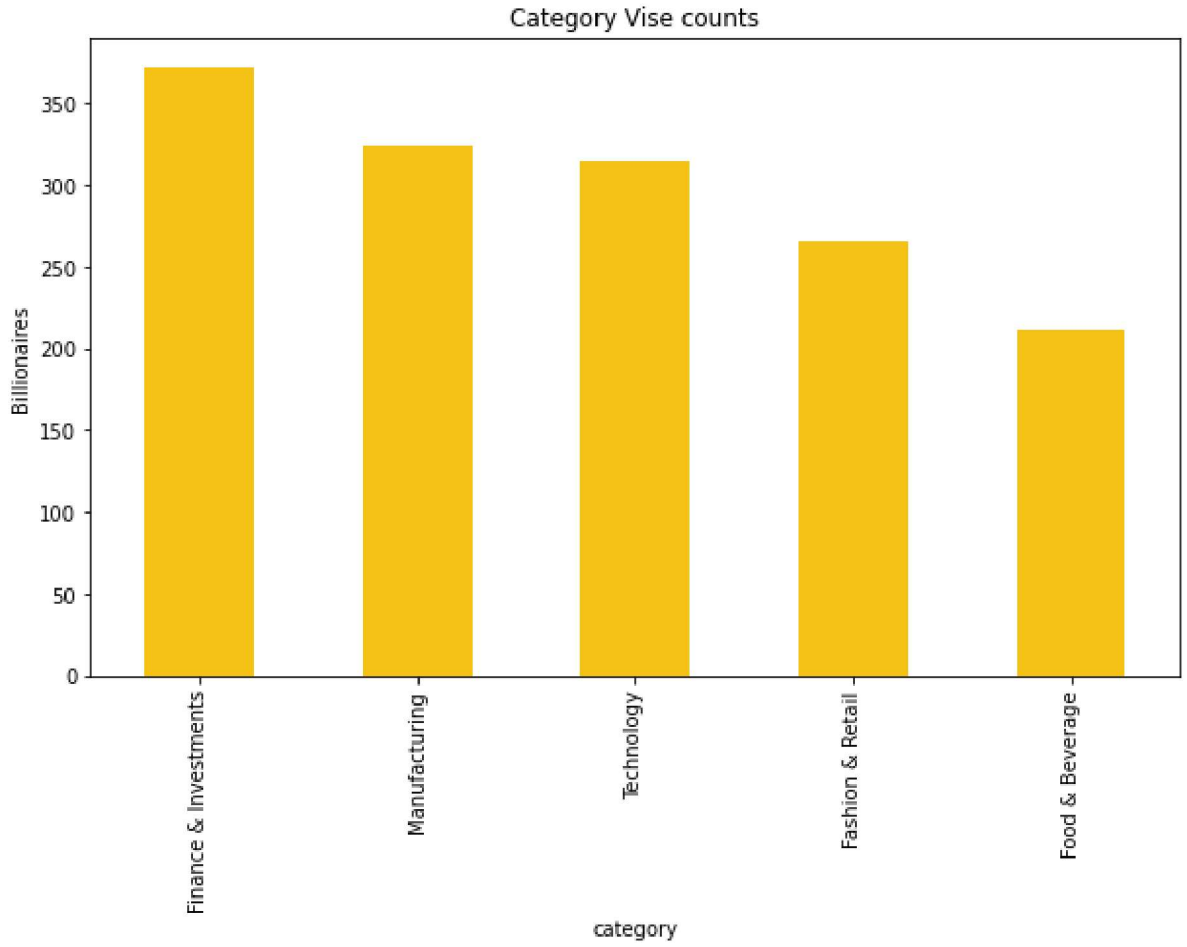
Country Vise Billionaires

In [15]:
```
country_counts = df["country"].value_counts().head()
plt.figure(figsize=(10, 6))
country_counts.plot(kind='bar', color = "#F4C314")
plt.title('country counts')
plt.xlabel('country')
plt.ylabel('Billionaires')
plt.xticks(rotation=90)
plt.show()
```



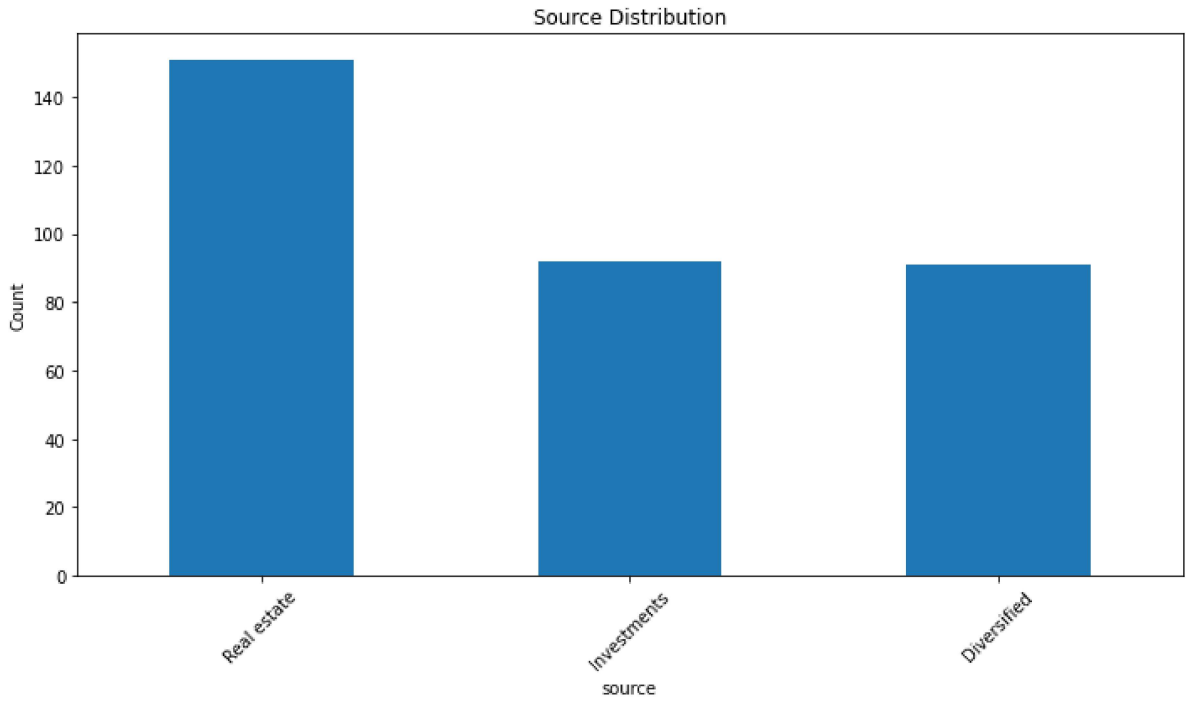
Category Vise counts

In [16]:
```
category_counts = df["category"].value_counts().head()
plt.figure(figsize=(10, 6))
category_counts.plot(kind='bar', color = "#F4C214")
plt.title('Category Vise counts')
plt.xlabel('category')
plt.ylabel('Billionaires')
plt.xticks(rotation=90)
plt.show()
```



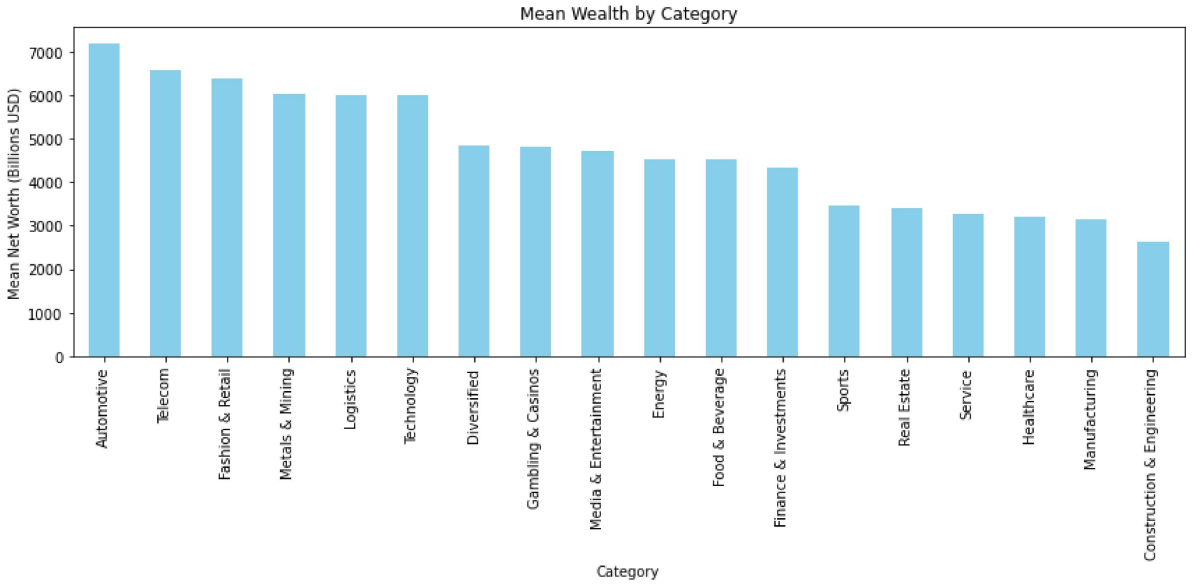
Source Distribution

In [17]:
```
# Count the occurrences of each currency
source_counts = df['source'].value_counts().head(3)

# Create a bar plot
plt.figure(figsize=(10, 6))
source_counts.plot(kind='bar')
plt.title('Source Distribution')
plt.xlabel('source')
plt.ylabel('Count')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.tight_layout()

# Show the plot
plt.show()
```



Wealth by Category

In [22]:
```
category_mean_worth = df.groupby('category')['finalWorth'].mean().sort_values(

# Create a bar plot
plt.figure(figsize=(12, 6))
category_mean_worth.plot(kind='bar', color='skyblue')
plt.xlabel('Category')
plt.ylabel('Mean Net Worth (Billions USD)')
plt.title('Mean Wealth by Category')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

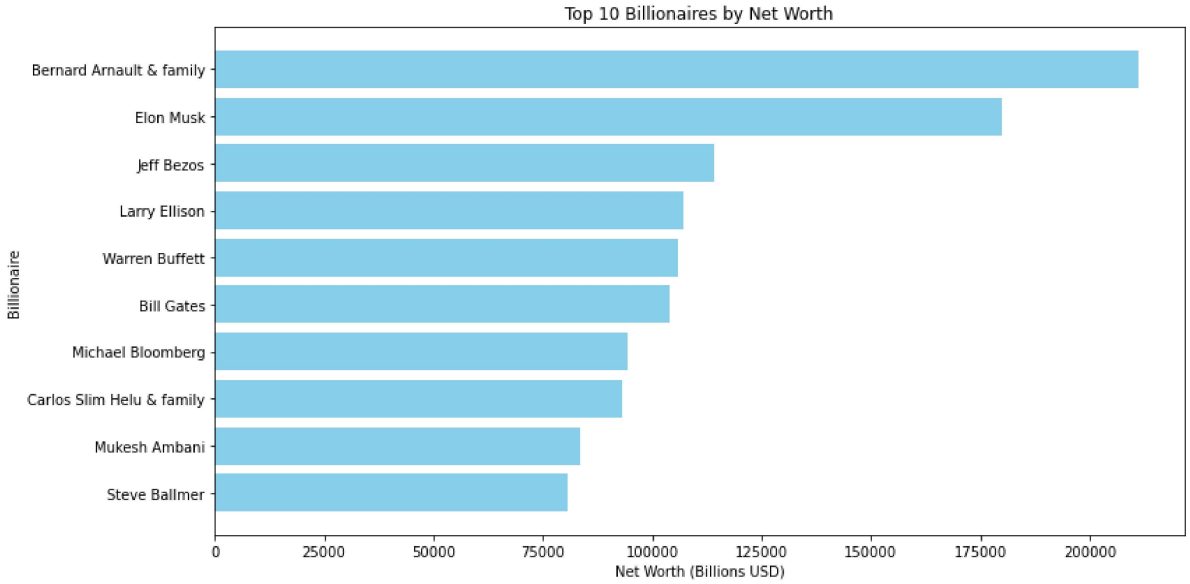


Top Billionaires by Net Worth and Category

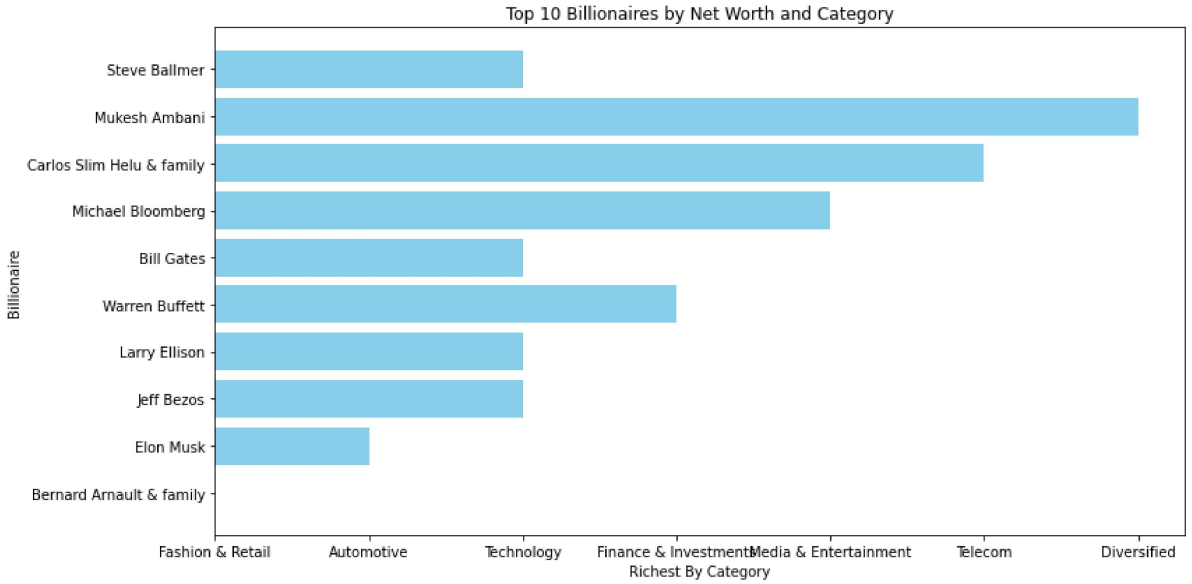
In [27]:
```
top_billionaires = df.sort_values(by='finalWorth', ascending=False).head(10).r
print("Top Billionaires by Net Worth and Category:")
print(top_billionaires[['personName', 'finalWorth', 'category']])
```

```
Top Billionaires by Net Worth and Category:
                  personName  finalWorth              category
0   Bernard Arnault & family      211000       Fashion & Retail
1                 Elon Musk      180000             Automotive
2                Jeff Bezos      114000             Technology
3             Larry Ellison      107000             Technology
4            Warren Buffett      106000  Finance & Investments
5                Bill Gates      104000             Technology
6          Michael Bloomberg       94500  Media & Entertainment
7  Carlos Slim Helu & family       93000                Telecom
8              Mukesh Ambani       83400            Diversified
9              Steve Ballmer       80700             Technology
```

In [56]:
```
top_billionaires = df.sort_values(by='finalWorth', ascending=False).head(10).r

# Create a bar plot to display the top billionaires
plt.figure(figsize=(12, 6))
plt.barh(top_billionaires['personName'], top_billionaires['finalWorth'], color
plt.xlabel('Net Worth (Billions USD)')
plt.ylabel('Billionaire')
plt.title('Top 10 Billionaires by Net Worth')
plt.gca().invert_yaxis()  # Invert the y-axis to display the richest at the to
plt.tight_layout()
plt.show()
```



In [49]:
```
top_billionaires = df.sort_values(by='finalWorth', ascending=False).head(10).r

# Create a bar plot to display the top billionaires
plt.figure(figsize=(12, 6))
plt.barh(top_billionaires['personName'], top_billionaires['category'], color='
plt.xlabel('Richest By Category')
plt.ylabel('Billionaire')
plt.title('Top 10 Billionaires by Net Worth and Category')
plt.tight_layout()
plt.show()
```



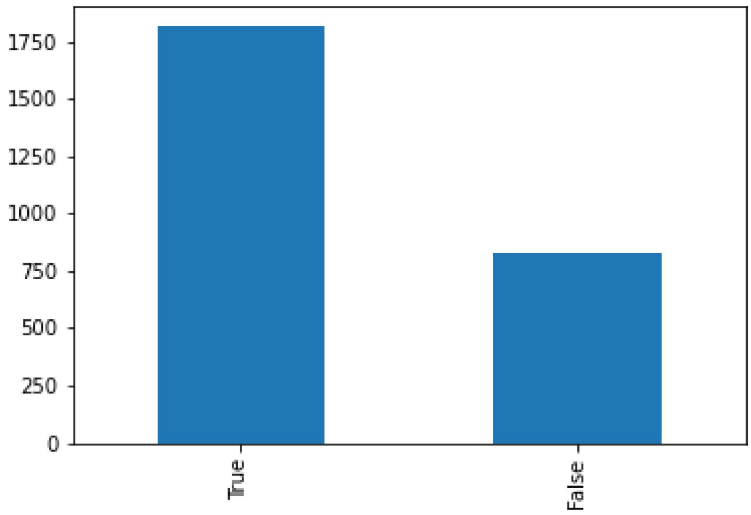
Self-Made vs. Inherited Billionaires

In [78]:
```
self_made_counts = df['selfMade'].value_counts()
print("Self-Made vs. Inherited Billionaires:")
print(self_made_counts)
```

```
Self-Made vs. Inherited Billionaires:
True     1812
False     828
Name: selfMade, dtype: int64
```

In [62]:
```
df.selfMade.value_counts().plot(kind = 'bar')
```

Out[62]: <AxesSubplot:>



Top Billionaires by Category and Gender

In [79]:
```
top_billionaires = df.sort_values(by='finalWorth', ascending=False).head(50).r

# Create a bar plot to display the top billionaires
plt.figure(figsize=(15, 9))
plt.barh(top_billionaires['gender'], top_billionaires['category'])
plt.xlabel('Richest By Category')
plt.ylabel('Billionaire')
plt.title('Top 50 Billionaires by Category and Gender')
plt.tight_layout()
plt.show()
```