```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

```
df = pd.read_csv('/content/Banking.csv')
df.head(5)
```

	Client ID	Name	Age	Location ID	Joined Bank	Banking Contact	Nationality	Occupation	Fee Structure	Loyalty Classification	...	Bank Deposits	Checking Accounts	A
0	IND81288	Raymond Mills	24	34324	06-05-2019	Anthony Torres	American	Safety Technician IV	High	Jade	...	1485828.64	603617.88	6(
1	IND65833	Julia Spencer	23	42205	10-12-2001	Jonathan Hawkins	African	Software Consultant	High	Jade	...	641482.79	229521.37	3⁴
2	IND47499	Stephen Murray	27	7314	25-01-2010	Anthony Berry	European	Help Desk Operator	High	Gold	...	1033401.59	652674.69	2(
3	IND72498	Virginia Garza	40	34594	28-03-2019	Steve Diaz	American	Geologist II	Mid	Silver	...	1048157.49	1048157.49	2⁃
4	IND60181	Melissa Sanders	46	41269	20-07-2012	Shawn Long	American	Assistant Professor	Mid	Platinum	...	487782.53	446644.25	1⁄

5 rows × 25 columns

```
df.shape
```

```
(3000, 25)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 25 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Client ID                 3000 non-null   object
 1   Name                      3000 non-null   object
 2   Age                       3000 non-null   int64
 3   Location ID               3000 non-null   int64
 4   Joined Bank               3000 non-null   object
 5   Banking Contact           3000 non-null   object
 6   Nationality               3000 non-null   object
 7   Occupation                3000 non-null   object
 8   Fee Structure             3000 non-null   object
 9   Loyalty Classification    3000 non-null   object
 10  Estimated Income          3000 non-null   float64
 11  Superannuation Savings    3000 non-null   float64
 12  Amount of Credit Cards    3000 non-null   int64
 13  Credit Card Balance       3000 non-null   float64
 14  Bank Loans                3000 non-null   float64
 15  Bank Deposits             3000 non-null   float64
 16  Checking Accounts         3000 non-null   float64
 17  Saving Accounts           3000 non-null   float64
 18  Foreign Currency Account  3000 non-null   float64
 19  Business Lending          3000 non-null   float64
 20  Properties Owned          3000 non-null   int64
 21  Risk Weighting            3000 non-null   int64
 22  BRId                      3000 non-null   int64
 23  GenderId                  3000 non-null   int64
 24  IAId                      3000 non-null   int64
dtypes: float64(9), int64(8), object(8)
memory usage: 586.1+ KB
```

```
# statistics for the dataframe
df.describe()
```

	Age	Location ID	Estimated Income	Superannuation Savings	Amount of Credit Cards	Credit Card Balance	Bank Loans	Bank Deposits	Checking Accounts	Ac
count	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3.000000e+03	3.000000e+03	3.000000e+03	3.0000
mean	51.039667	21563.323000	171305.034263	25531.599673	1.463667	3176.206943	5.913862e+05	6.715602e+05	3.210929e+05	2.3290
std	19.854760	12462.273017	111935.808209	16259.950770	0.676387	2497.094709	4.575570e+05	6.457169e+05	2.820796e+05	2.3000
min	17.000000	12.000000	15919.480000	1482.030000	1.000000	1.170000	0.000000e+00	0.000000e+00	0.000000e+00	0.0000
25%	34.000000	10803.500000	82906.595000	12513.775000	1.000000	1236.630000	2.396281e+05	2.044004e+05	1.199475e+05	7.4794
50%	51.000000	21129.500000	142313.480000	22357.355000	1.000000	2560.805000	4.797934e+05	4.633165e+05	2.428157e+05	1.6408
75%	69.000000	32054.500000	242290.305000	35464.740000	2.000000	4522.632500	8.258130e+05	9.427546e+05	4.348749e+05	3.1557
max	85.000000	43369.000000	522330.260000	75963.900000	3.000000	13991.990000	2.667557e+06	3.890598e+06	1.969923e+06	1.7241

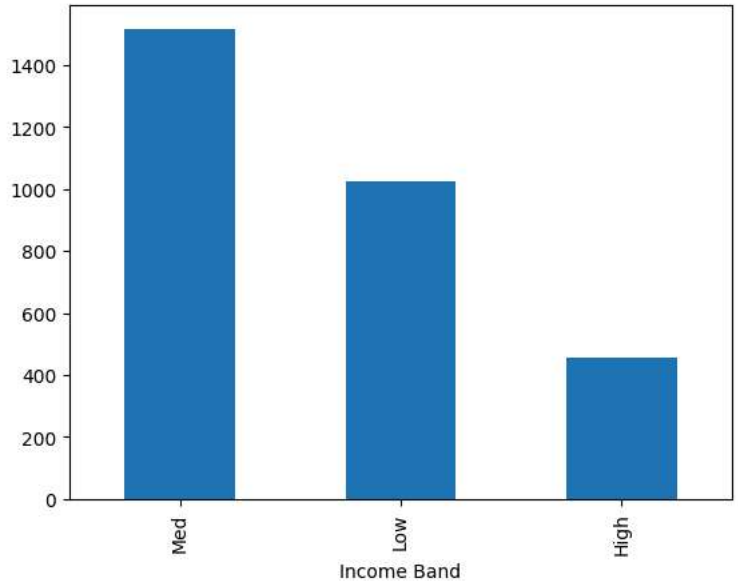
Start coding or generate with AI.

```
bins = [0, 100000, 300000, float('inf')]
labels = ['Low', 'Med', 'High']

df['Income Band'] = pd.cut(df['Estimated Income'], bins=bins, labels=labels, right=False)


df['Income Band'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Income Band'>
```




```
# Examine the distribution of unique cataegories in categorical columns
categorical_cols = df[["BRId", "GenderId", "IAId", "Amount of Credit Cards", "Nationality", "Occupation", "Fee Structure", "Loyalty Classifi

for col in categorical_cols:
  print(f"Value Counts for '{col}':")
  display(df[col].value_counts())
```

```
Value Counts for 'BRId':
```

BRId	count
3	1352
1	660
2	495
4	493

dtype: int64

```
Value Counts for 'GenderId':
```

GenderId	count
2	1512
1	1488

dtype: int64

```
Value Counts for 'IAId':
```

IAId	count
1	177
2	177
3	177
4	177
8	177
9	176
13	176
12	176
10	176
11	176
14	176
15	176
6	89
5	89
7	89
16	88
17	88
18	88
21	88
22	88

dtype: int64

```
Value Counts for 'Amount of Credit Cards':
```

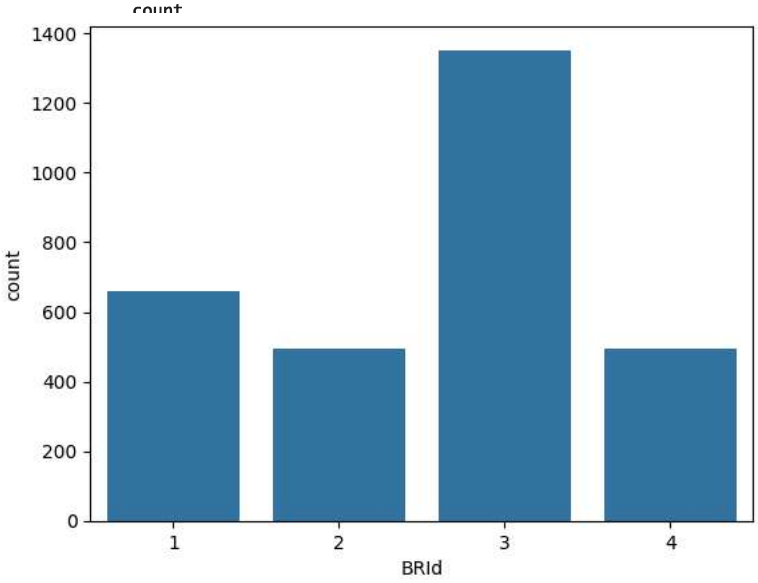
Amount of Credit Cards	count
1	1922
2	765
3	313

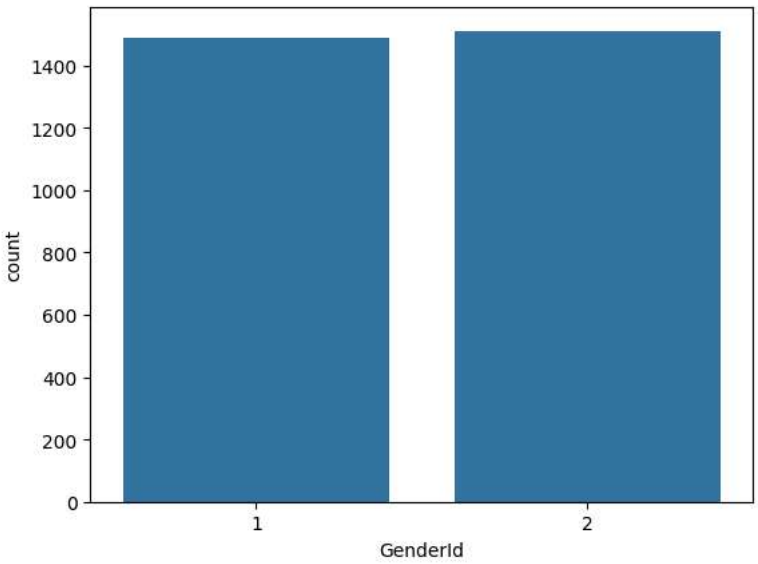
dtype: int64

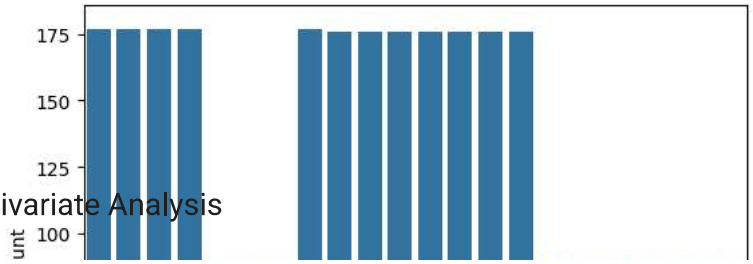
```
Value Counts for 'Nationality':
```

Univariate Analysis

```
for i, predictor in enumerate(df[["BRId", "GenderId", "IAId", "Amount of Credit Cards", "Nationality", "Occupation", "Fee Structure", "Loyal
    plt.figure(i)
    sns.countplot(data=df, x=predictor)
```

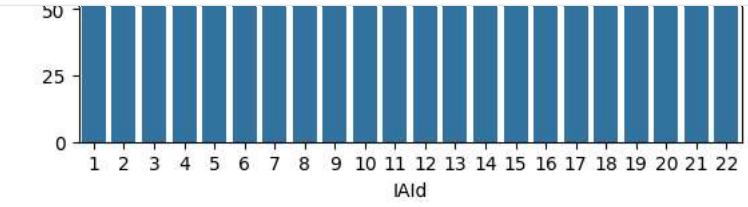


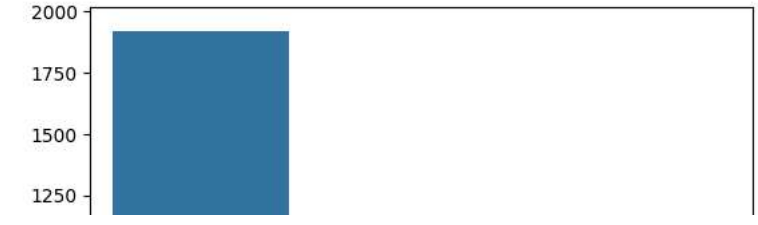


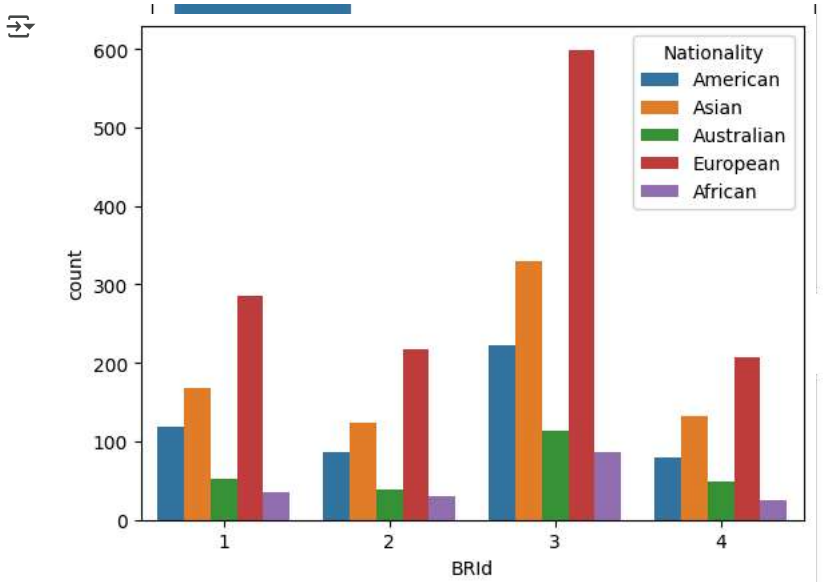


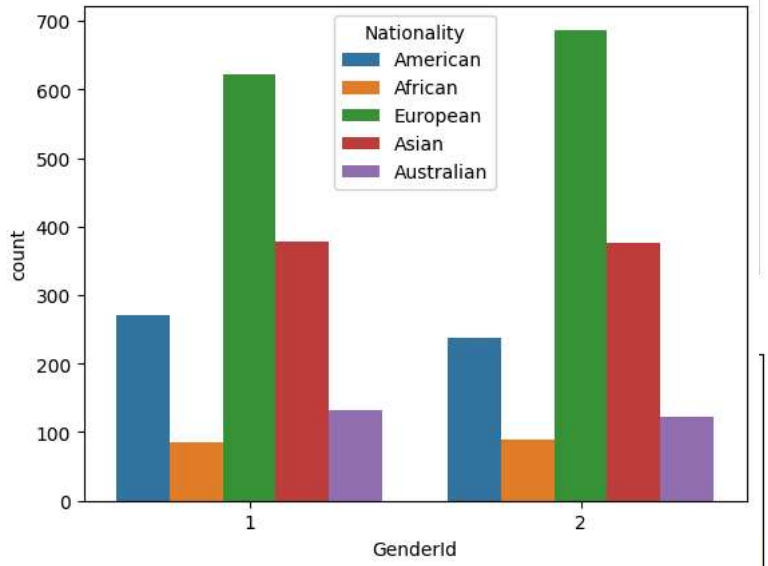
Bivariate Analysis

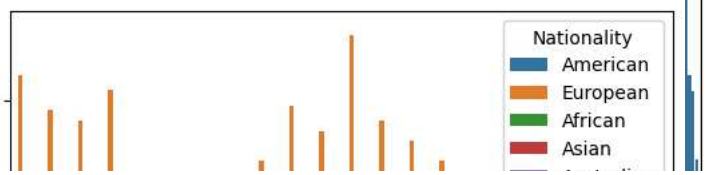
```
for i, predictor in enumerate(df[["BRId", "GenderId", "IAId", "Amount of Credit Cards", "Nationality", "Occupation", "Fee Structure", "Loyal
    plt.figure(i)
    sns.countplot(data=df, x=predictor, hue='Nationality')
```





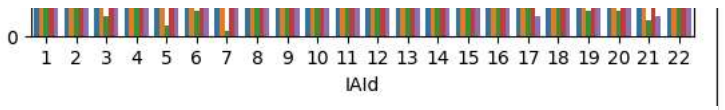


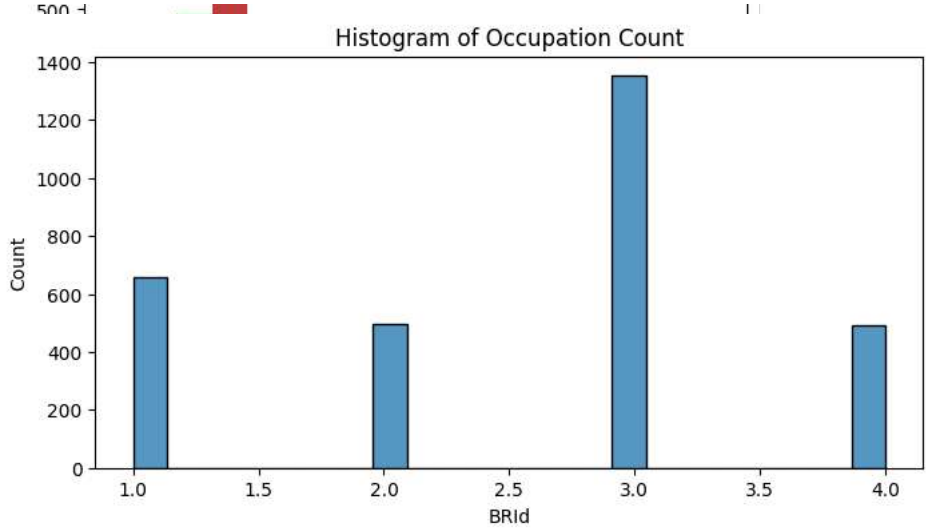


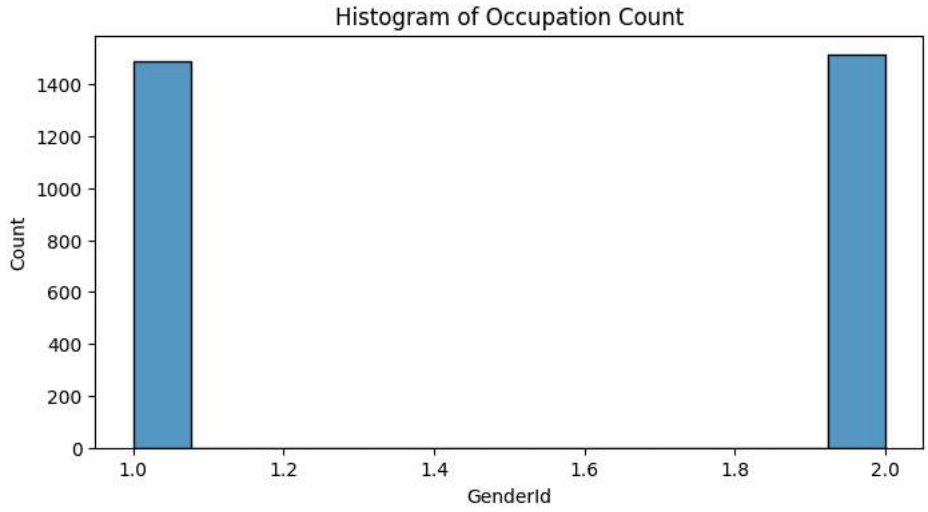


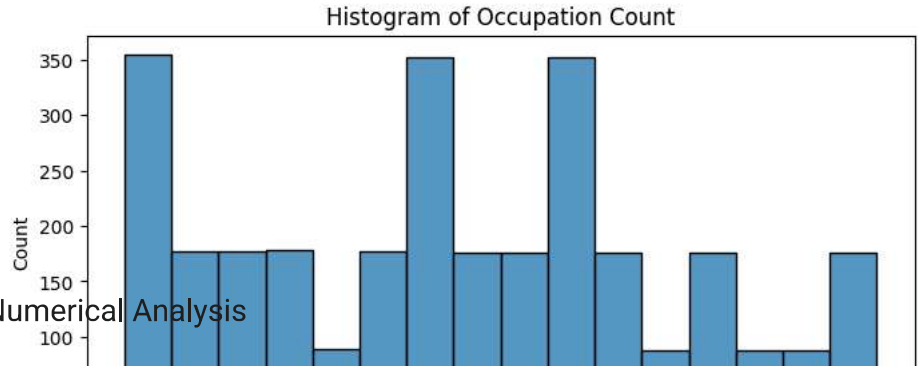

```
# HIstplot of value counts for different Occupation

for col in categorical_cols:
  if col == "Occupation":
    continue
  plt.figure(figsize=(8,4))
  sns.histplot(df[col])
  plt.title('Histogram of Occupation Count')
  plt.xlabel(col)
  plt.ylabel("Count")
  plt.show()
```





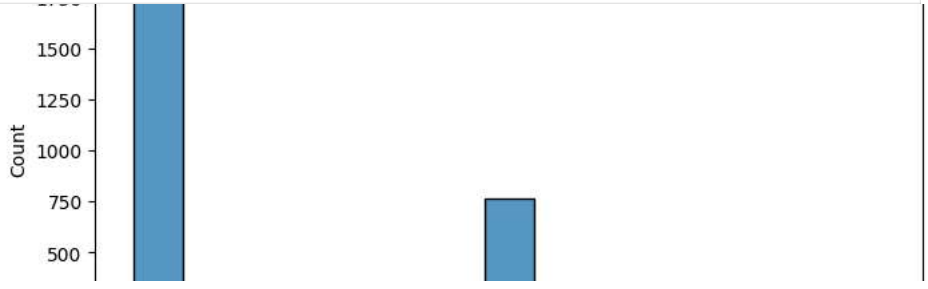


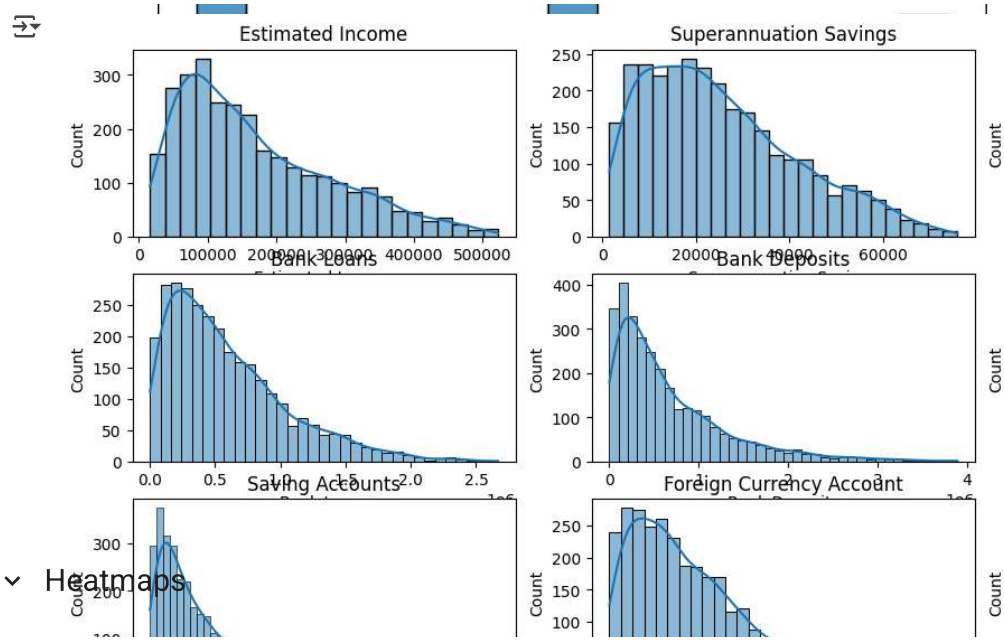


Numerical Analysis

```
numerical_cols = ['Estimated Income', 'Superannuation Savings', 'Credit Card Balance', 'Bank Loans', 'Bank Deposits', 'Checking Accounts', '

# Univariate analysis and visualization
plt.figure(figsize=(15,10))
for i,col in enumerate(numerical_cols):
  plt.subplot(4,3,i+1)
  sns.histplot(df[col],kde=True)
  plt.title(col)
plt.show()
```





Heatmaps

```
numerical_cols = ['Estimated Income', 'Superannuation Savings', 'Credit Card Balance', 'Bank Loans', 'Bank Deposits', 'Checking Accounts', '

correlation_matrix = df[numerical_cols].corr()

plt.figure(figsize=(12,12))
sns.heatmap(correlation_matrix, annot=True, cmap='crest', fmt=".2f")
plt.title("Correlation Matrix")
plt.show()
```