Understanding the Gender Pay Gap in Hourly Earnings: An Analysis of Trends and Factors

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px # is a high-level interface for creating various types of interactive
import plotly.graph_objects as go # is a lower-level interface that offers more control and custo
```

In [2]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [3]:

```
df = pd.read_csv("female_hourly_earnings.csv")
```

In [4]:

```
df.shape
```

Out[4]:

```
(916, 7)
```

In [5]:

```
df.columns
```

Out[5]:

```
Index(['Unnamed: 0', 'country_id', 'country', 'gender_code', 'gender', 'year',
       'amount_local_currency'],
      dtype='object')
```

In [6]:

```
df = df.drop('Unnamed: 0', axis = 1)
```

In [7]:

```
df.duplicated().sum()
```

Out[7]:

```
0
```

In [8]:

```
df.isnull().sum()
```

Out[8]:

```
country_id               0
country                  0
gender_code              0
gender                   0
year                     0
amount_local_currency    0
dtype: int64
```

In [9]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 916 entries, 0 to 915
Data columns (total 6 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   country_id             916 non-null    int64  
 1   country                916 non-null    object 
 2   gender_code            916 non-null    object 
 3   gender                 916 non-null    object 
 4   year                   916 non-null    int64  
 5   amount_local_currency  916 non-null    float64
dtypes: float64(1), int64(2), object(3)
memory usage: 43.1+ KB
```

In [10]:

```
df.describe()
```

Out[10]:

	country_id	year	amount_local_currency
count	916.000000	916.000000	916.000000
mean	477.310044	2015.340611	350.170753
std	263.137563	3.331868	1502.006191
min	8.000000	2009.000000	-46.200000
25%	250.000000	2013.000000	9.227500
50%	498.000000	2015.000000	29.060000
75%	724.000000	2018.000000	57.627500
max	860.000000	2022.000000	20370.200000

In [11]:

```
object_columns = df.select_dtypes(include='object').columns.tolist()
numerical_columns = df.select_dtypes(include=['int', 'float']).columns.tolist()

print("Object columns:", object_columns)
print("Numerical columns:", numerical_columns)
```

```
Object columns: ['country', 'gender_code', 'gender']
Numerical columns: ['country_id', 'year', 'amount_local_currency']
```

In [12]:

```
df.nunique()
```

Out[12]:

```
country_id                45
country                   45
gender_code                4
gender                     4
year                      14
amount_local_currency    829
dtype: int64
```

In [13]:

```
for i in object_columns:
    print(i)
    print(df[i].unique())
    print('\n')
```

```
country
['Armenia' 'Belgium' 'Germany' 'Bulgaria' 'Italy' 'Republic of Moldova'
 'Denmark' 'United Kingdom' 'Slovakia' 'United States' 'Israel' 'Serbia'
 'Iceland' 'Bosnia and Herzegovina' 'Croatia' 'Luxembourg' 'Estonia'
 'Spain' 'Sweden' 'Latvia' 'Austria' 'Belarus' 'Albania' 'Türkiye' 'Malta'
 'Russian Federation' 'Uzbekistan' 'Norway' 'Ukraine' 'North Macedonia'
 'Poland' 'Hungary' 'Finland' 'Lithuania' 'Greece' 'Cyprus' 'Ireland'
 'Montenegro' 'Slovenia' 'Czechia' 'Romania' 'Portugal' 'Switzerland'
 'Netherlands' 'France']


gender_code
['F' 'M' 'T' 'G']


gender
['Female' 'Male' 'Total' 'Gender gap']
```

In [14]:

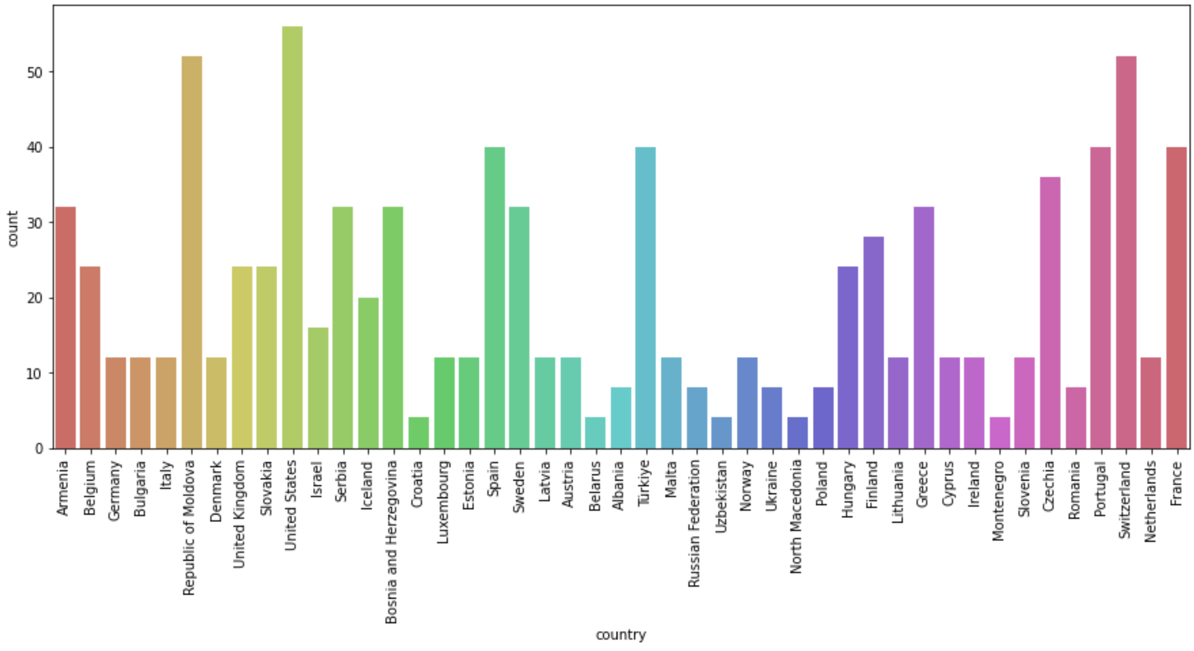
```
for i in object_columns:
    print(i)
    print(df[i].value_counts())
    print('\n')
```

```
country
United States             56
Switzerland               52
Republic of Moldova       52
France                    40
Portugal                  40
Türkiye                   40
Spain                     40
Czechia                   36
Bosnia and Herzegovina    32
Greece                    32
Sweden                    32
Armenia                   32
Serbia                    32
Finland                   28
Slovakia                  24
Hungary                   24
Belgium                   24
United Kingdom            24
Iceland                   20
Israel                    16
Norway                    12
Denmark                   12
Lithuania                 12
Slovenia                  12
Italy                     12
Malta                     12
Bulgaria                  12
Ireland                   12
Germany                   12
Austria                   12
Latvia                    12
Estonia                   12
Luxembourg                12
Netherlands               12
Cyprus                    12
Romania                    8
Ukraine                    8
Poland                     8
Russian Federation         8
Albania                    8
Montenegro                 4
North Macedonia            4
Uzbekistan                 4
Belarus                    4
Croatia                    4
Name: country, dtype: int64


gender_code
F    229
M    229
T    229
G    229
Name: gender_code, dtype: int64


gender
Female        229
Male          229
Total         229
Gender gap    229
Name: gender, dtype: int64
```

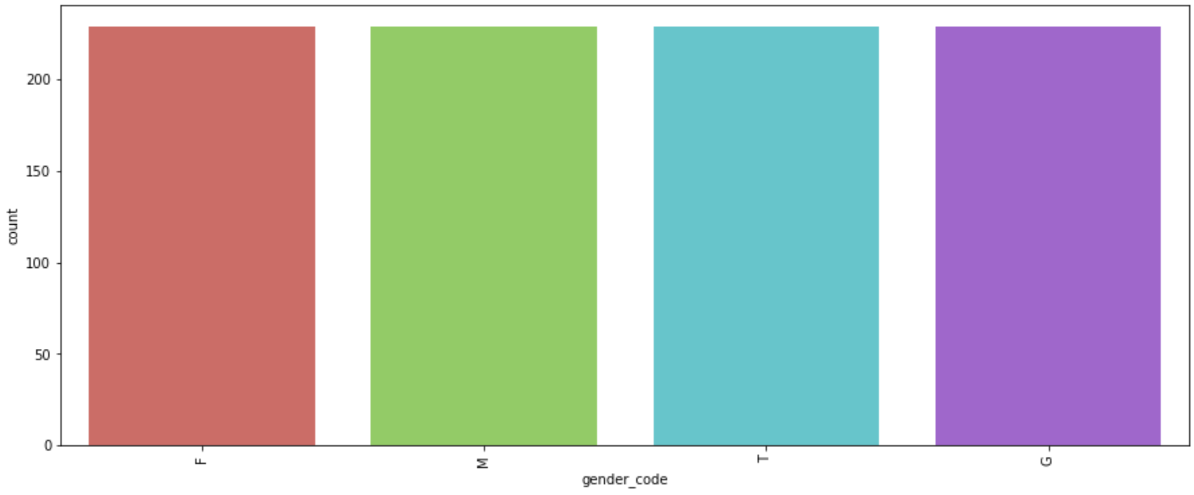
```
In [15]:
```

```
for i in object_columns:
    print('Countplot for:', i)
    plt.figure(figsize=(15,6))
    sns.countplot(df[i], data = df, palette = 'hls')
    plt.xticks(rotation = 90)
    plt.show()
    print('\n')
```

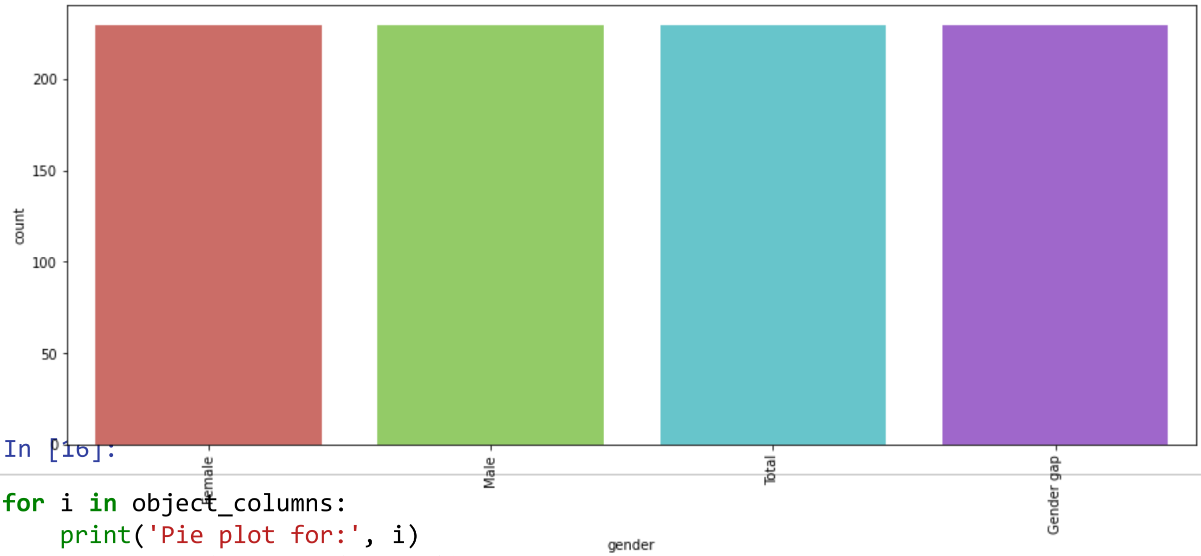
```
Countplot for: country
```



```
Countplot for: gender_code
```



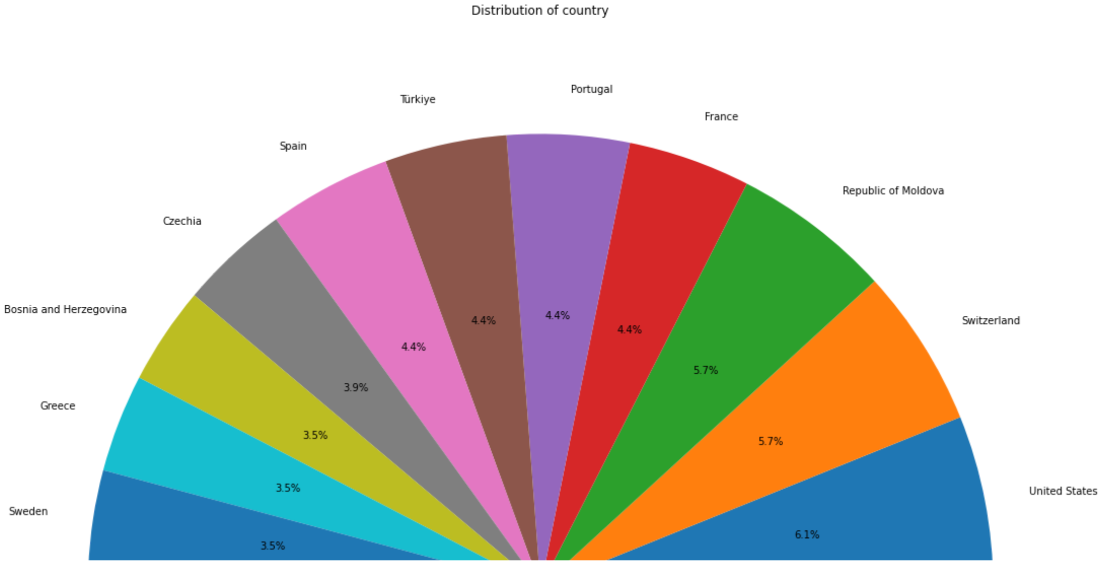
```
Countplot for: gender
```



In [16]:

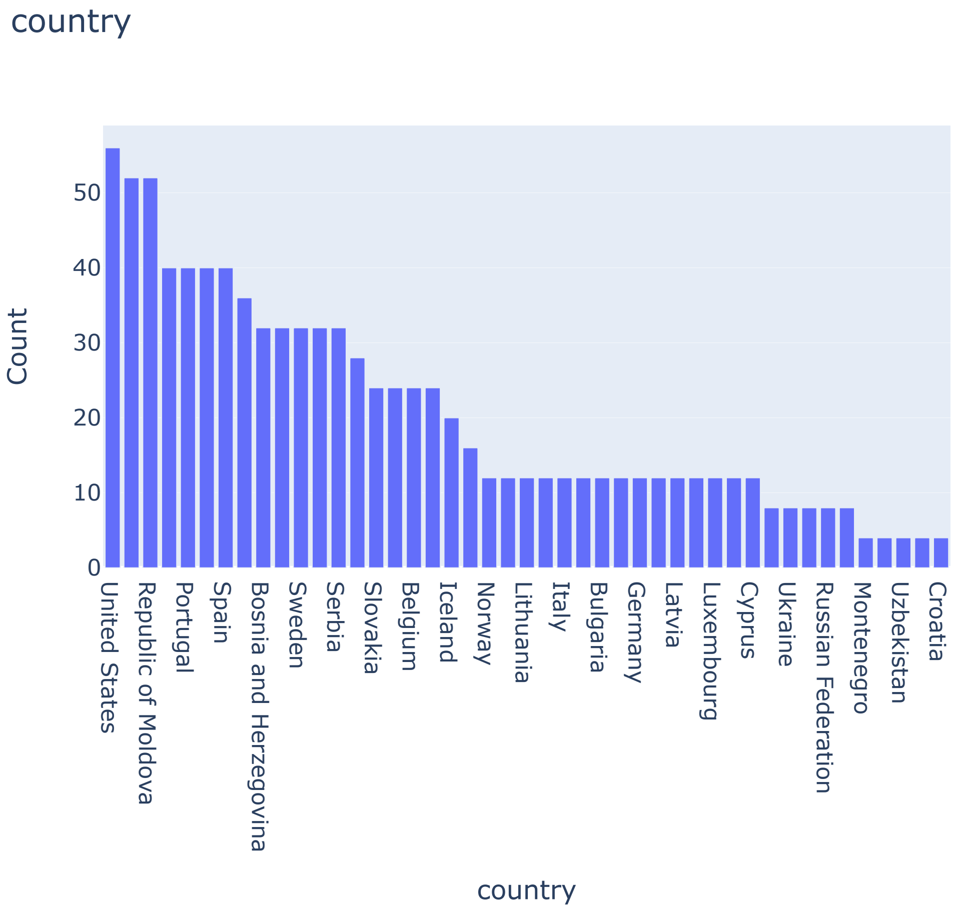
```python
for i in object_columns:
    print('Pie plot for:', i)
    plt.figure(figsize=(20, 30))
    df[i].value_counts().plot(kind='pie', autopct='%1.1f%%')
    plt.title('Distribution of ' + i)
    plt.ylabel('')
    plt.show()
    print('\n')
```

```
Pie plot for: country
```

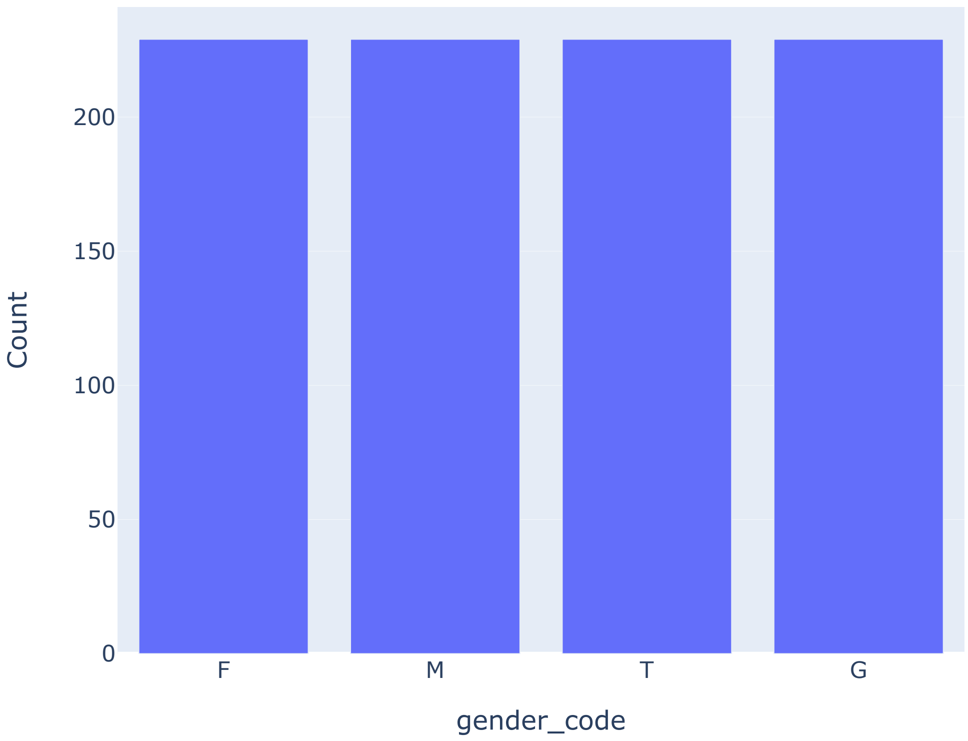


In [17]:

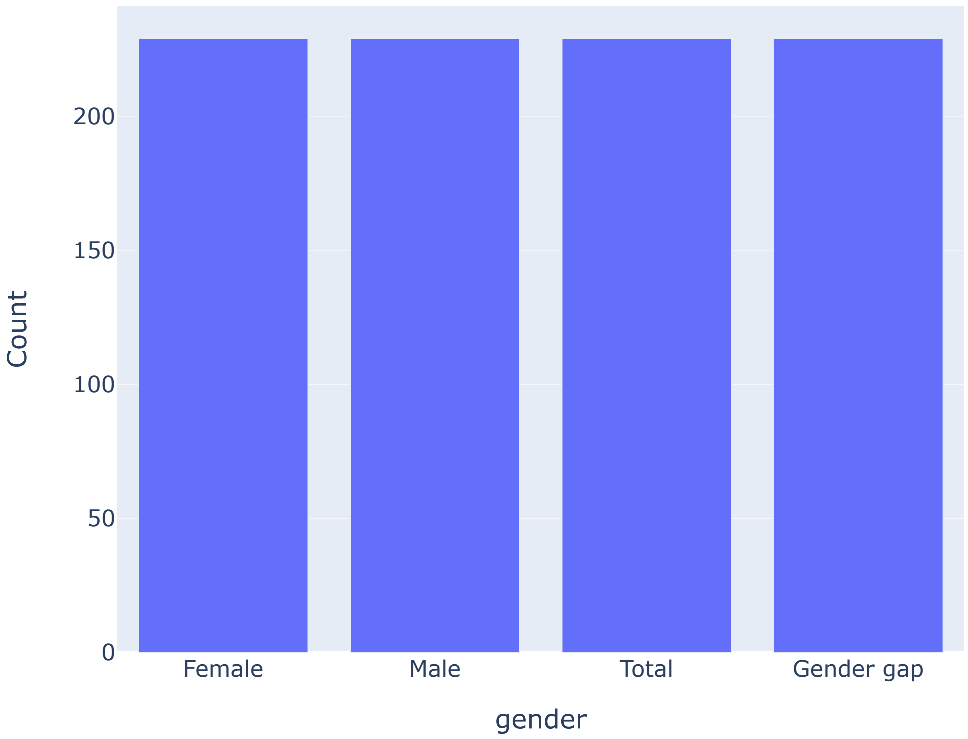
```
for i in object_columns:
    fig = go.Figure(data=[go.Bar(x=df[i].value_counts().index, y=df[i].value_counts())])
    fig.update_layout(
        title=i,
        xaxis_title=i,
        yaxis_title="Count")
    fig.show()
```



gender_code



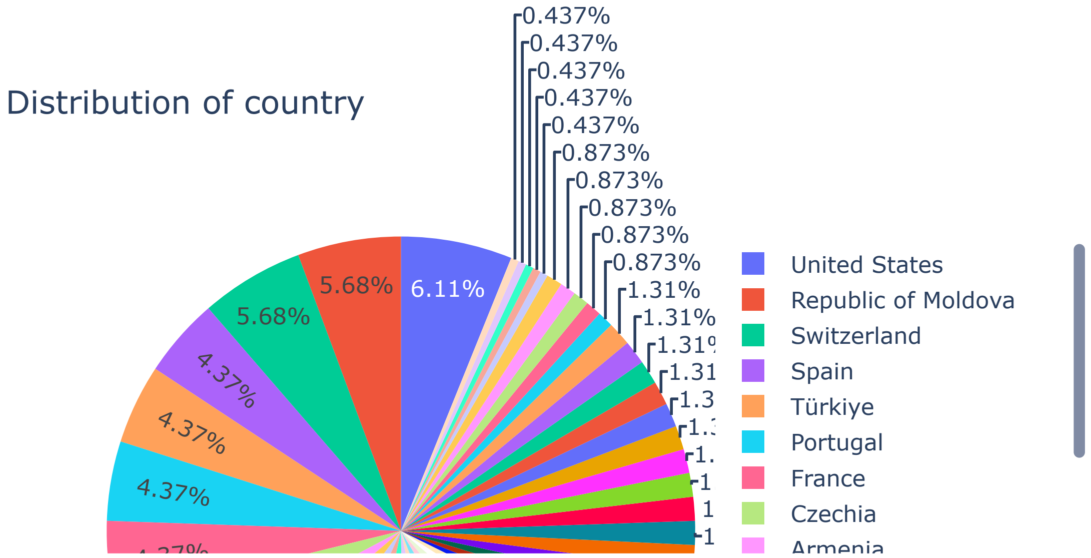
gender



In [18]:

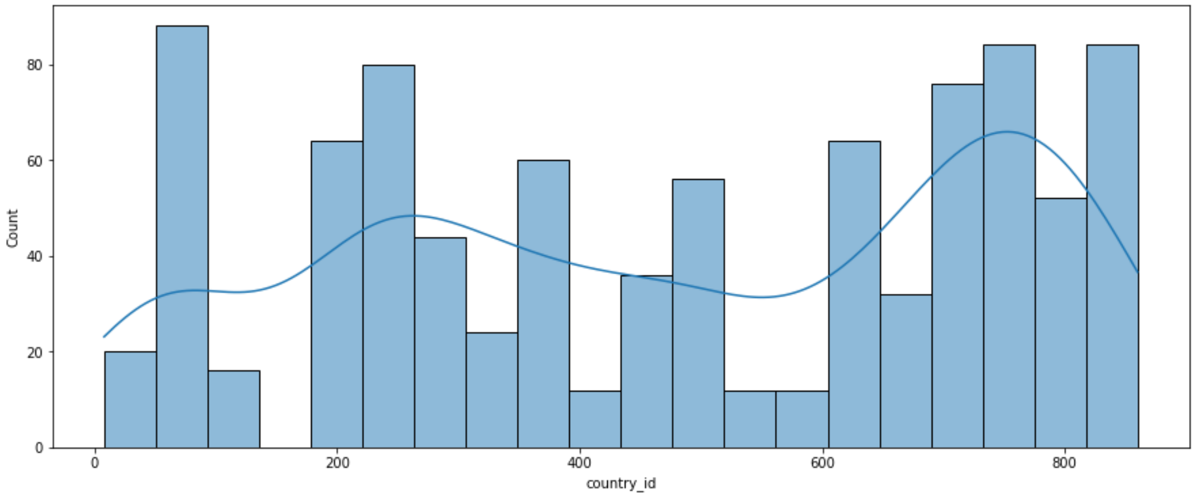
```
for i in object_columns:
    print('Pie plot for:', i)
    fig = px.pie(df, names=i, title='Distribution of ' + i)
    fig.show()
    print('\n')
```

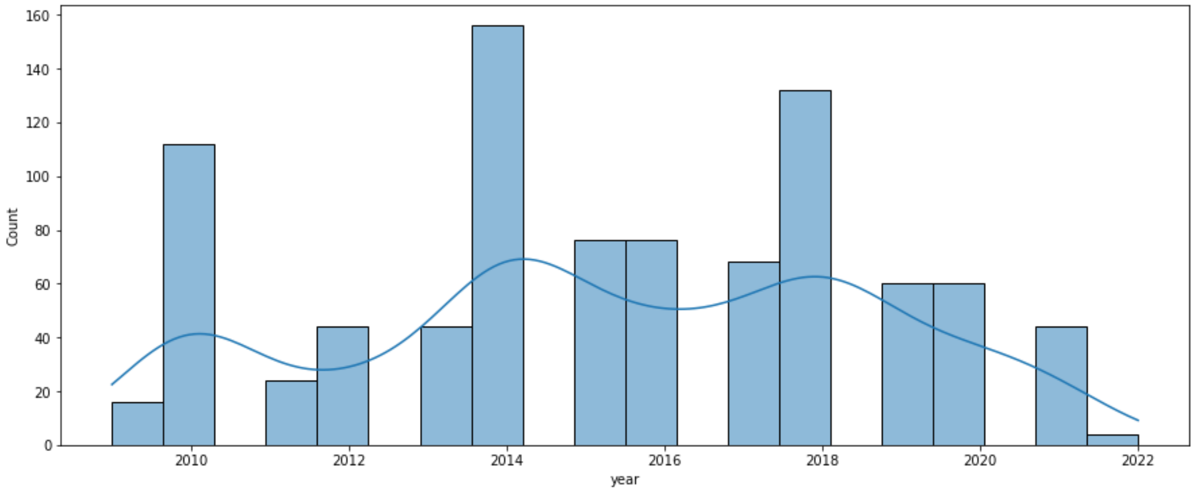
```
Pie plot for: country
```

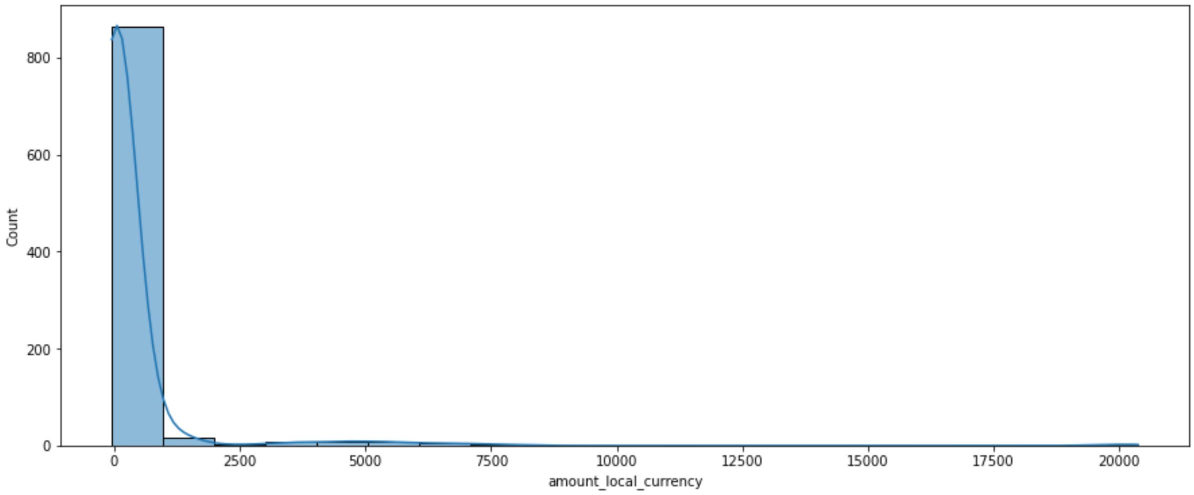


In [19]:

```
for i in numerical_columns:
    plt.figure(figsize=(15,6))
    sns.histplot(df[i], kde = True, bins = 20, palette = 'hls')
    plt.xticks(rotation = 0)
    plt.show()
```

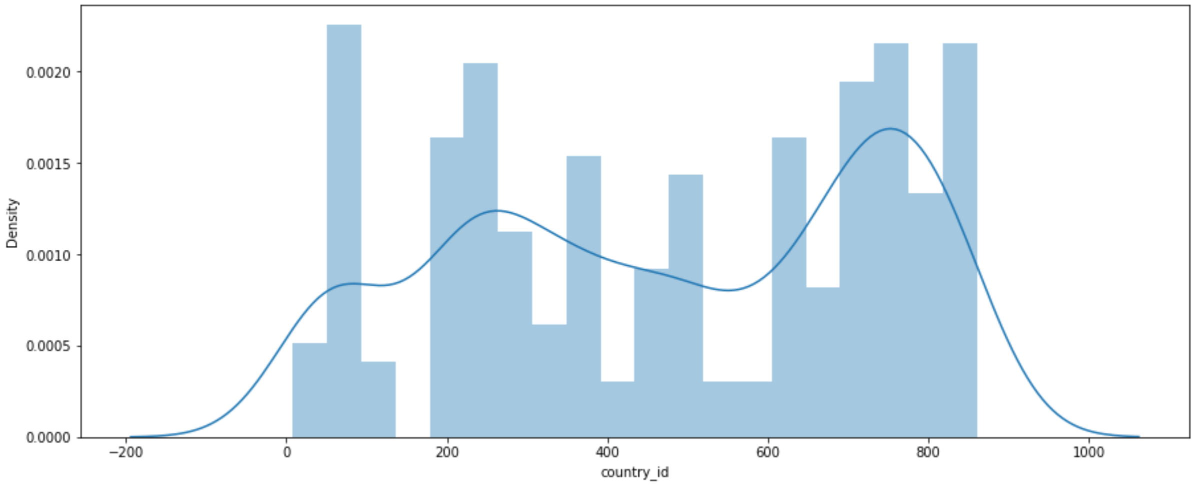


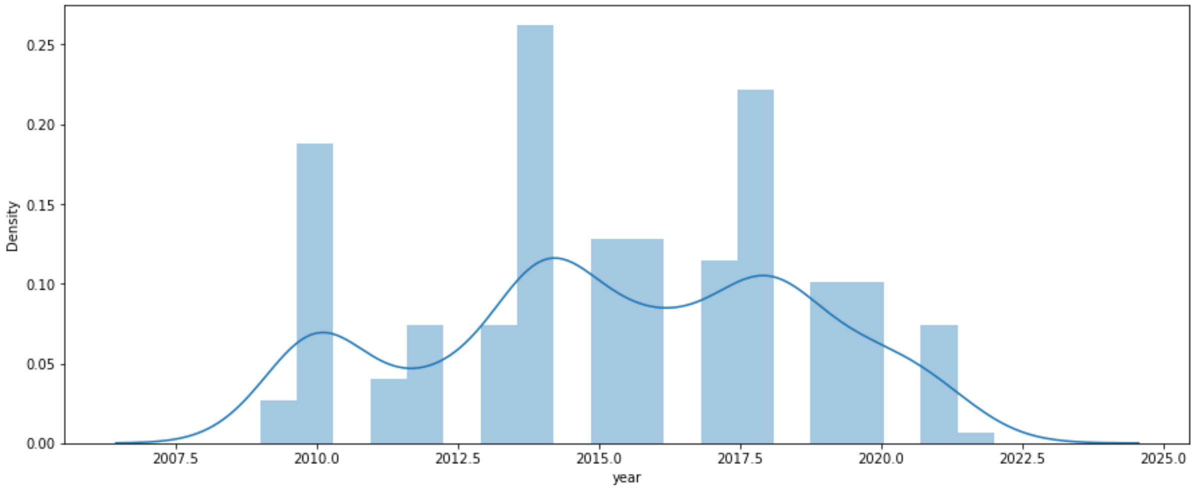


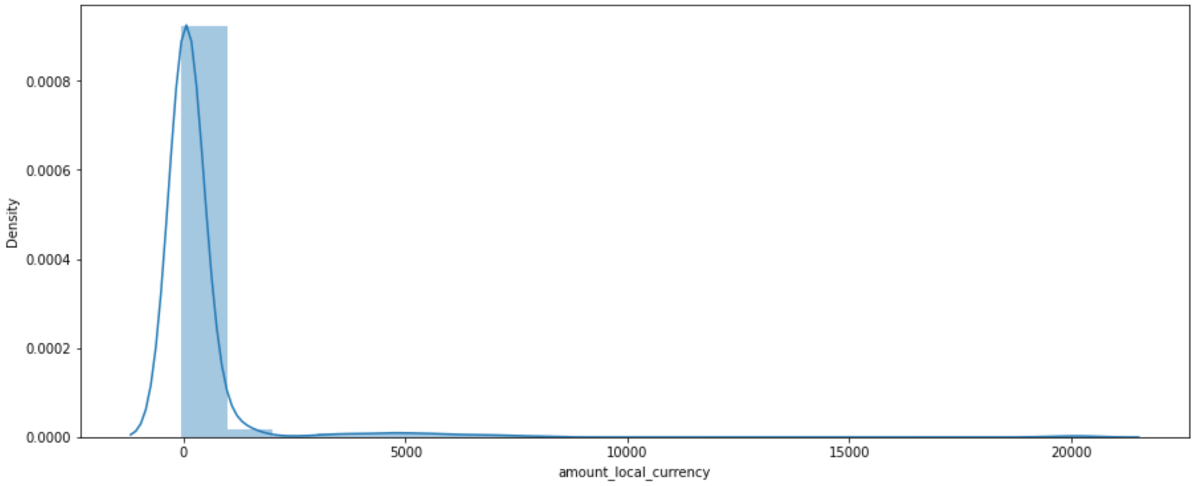


In [20]:

```
for i in numerical_columns:
    plt.figure(figsize=(15,6))
    sns.distplot(df[i], kde = True, bins = 20)
    plt.xticks(rotation = 0)
    plt.show()
```

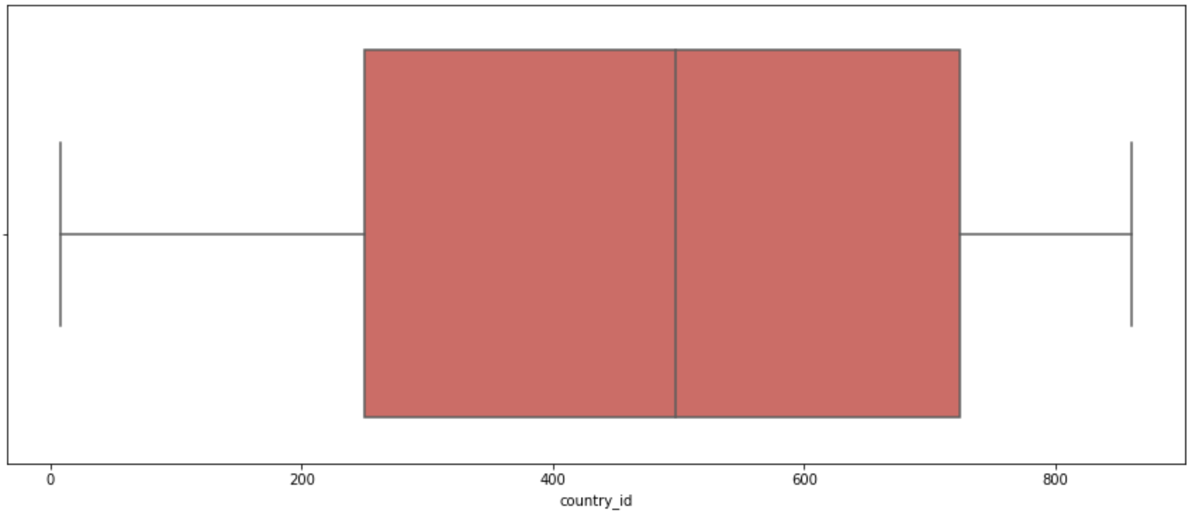


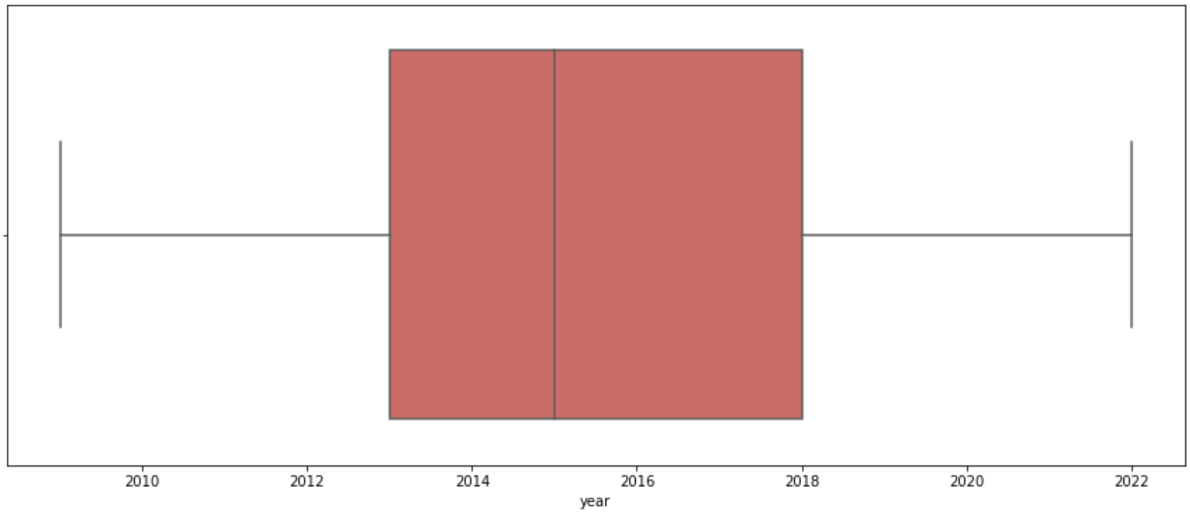


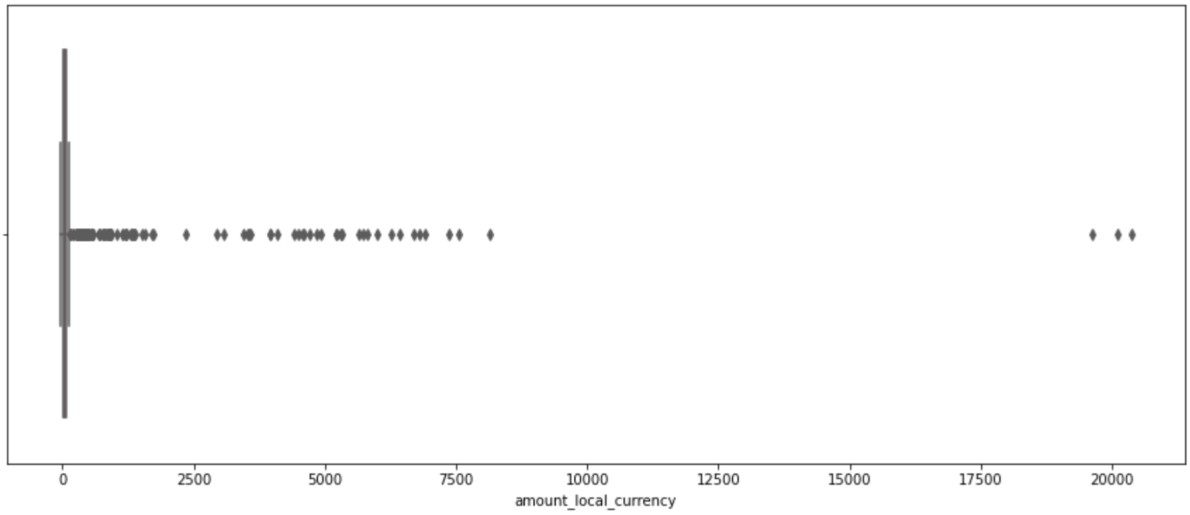


In [21]:

```
for i in numerical_columns:
    plt.figure(figsize=(15,6))
    sns.boxplot(df[i], data=df, palette='hls')
    plt.xticks(rotation = 0)
    plt.show()
```

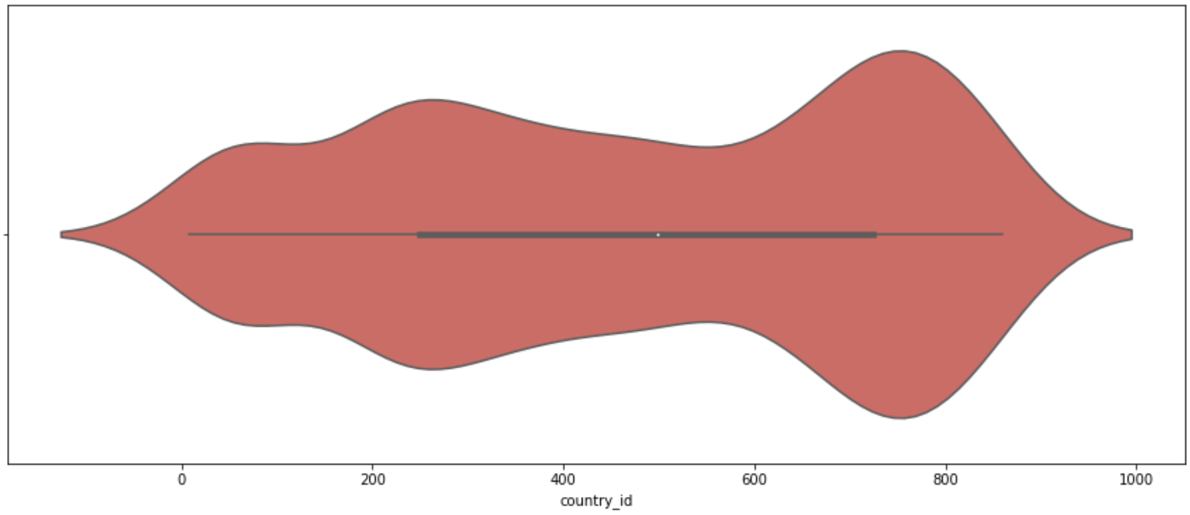


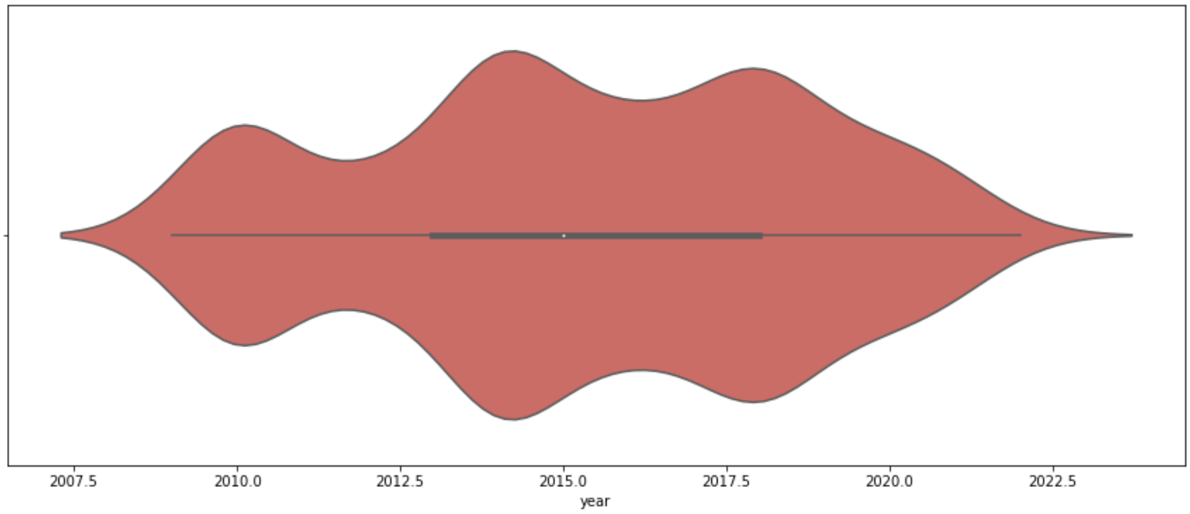


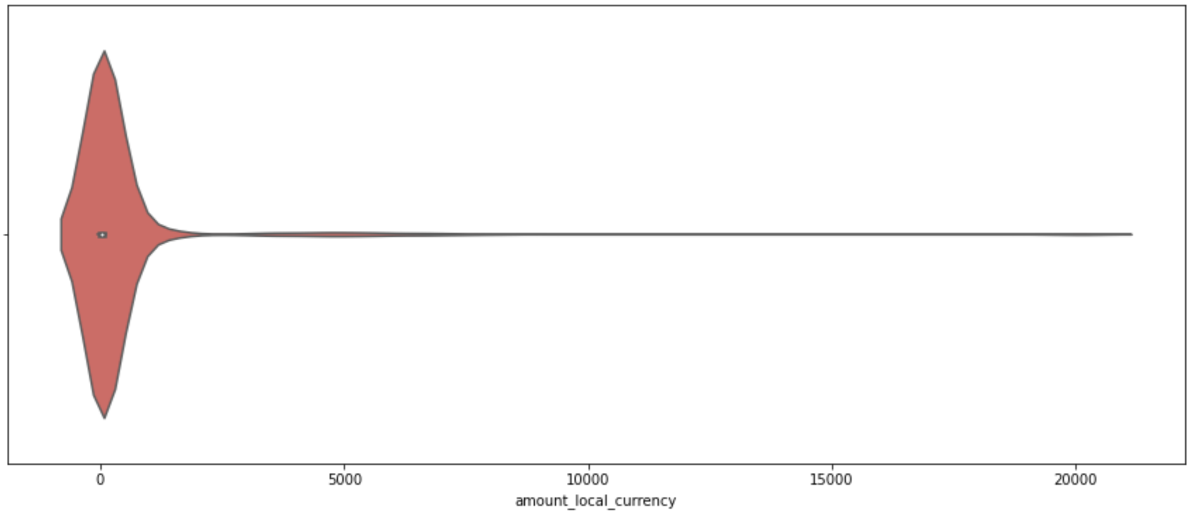


In [22]:

```
for i in numerical_columns:
    plt.figure(figsize=(15,6))
    sns.violinplot(df[i], data=df, palette='hls')
    plt.xticks(rotation = 0)
    plt.show()
```

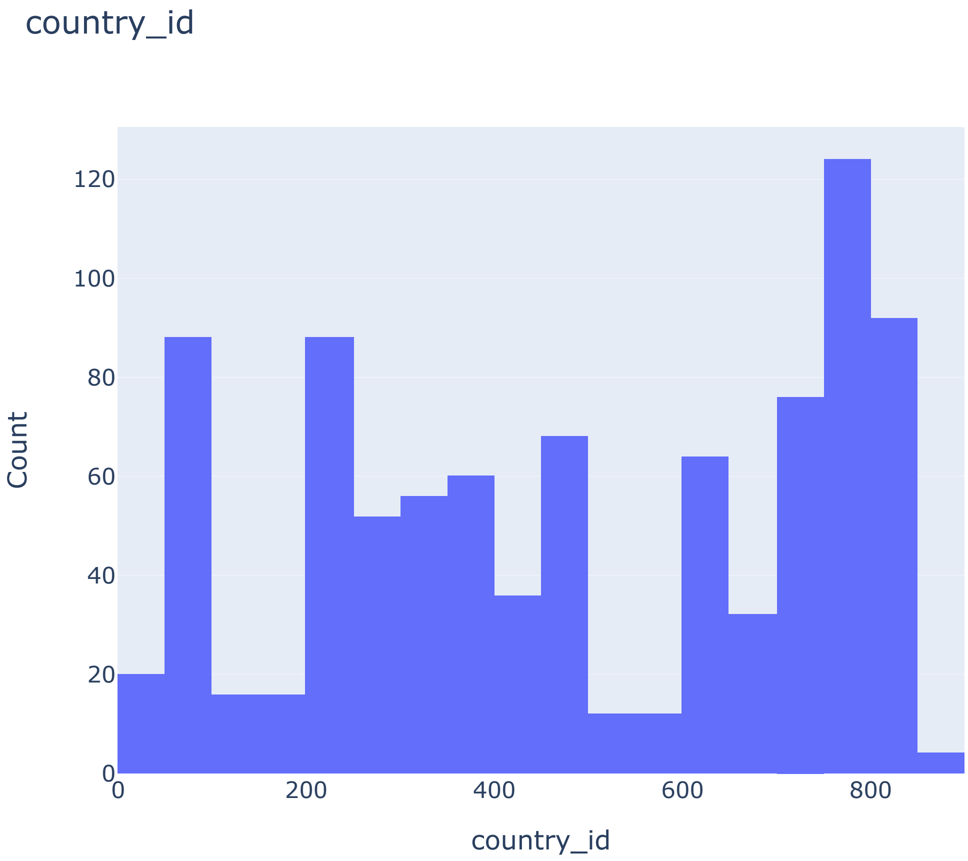




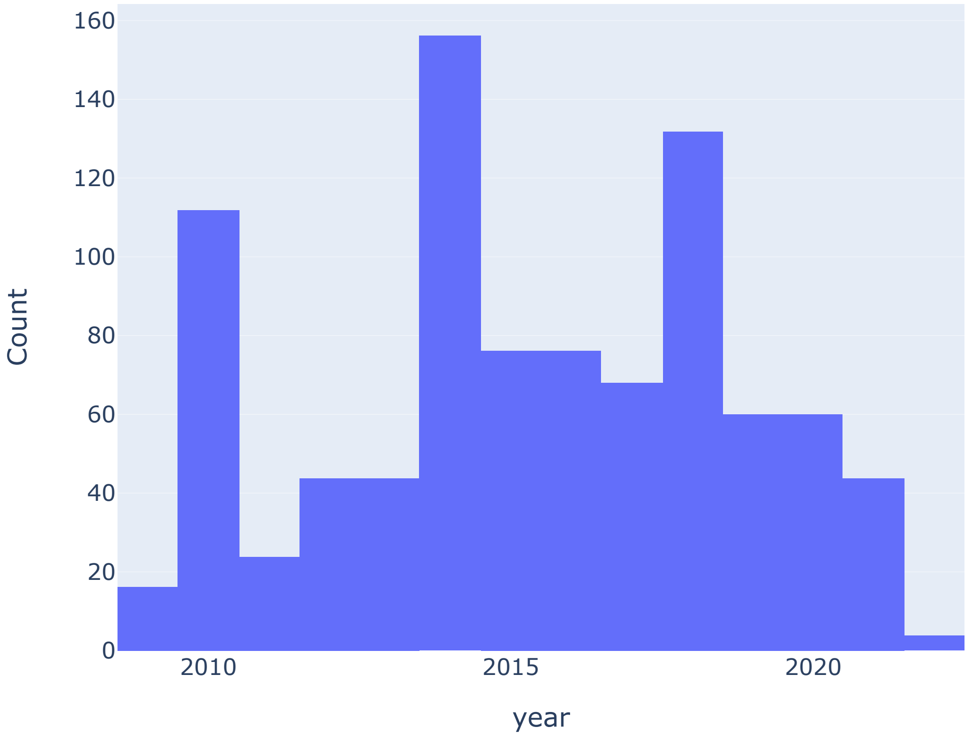


In [23]:

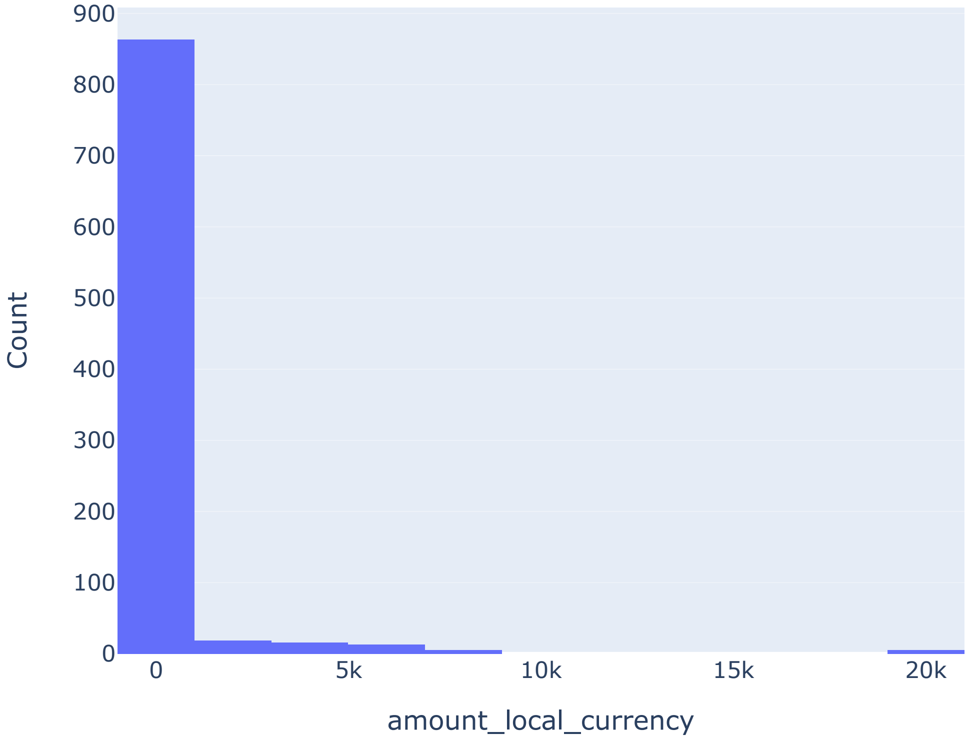
```
for i in numerical_columns:
    fig = go.Figure(data=[go.Histogram(x=df[i], nbinsx=20)])
    fig.update_layout(
        title=i,
        xaxis_title=i,
        yaxis_title="Count")
    fig.show()
```



year

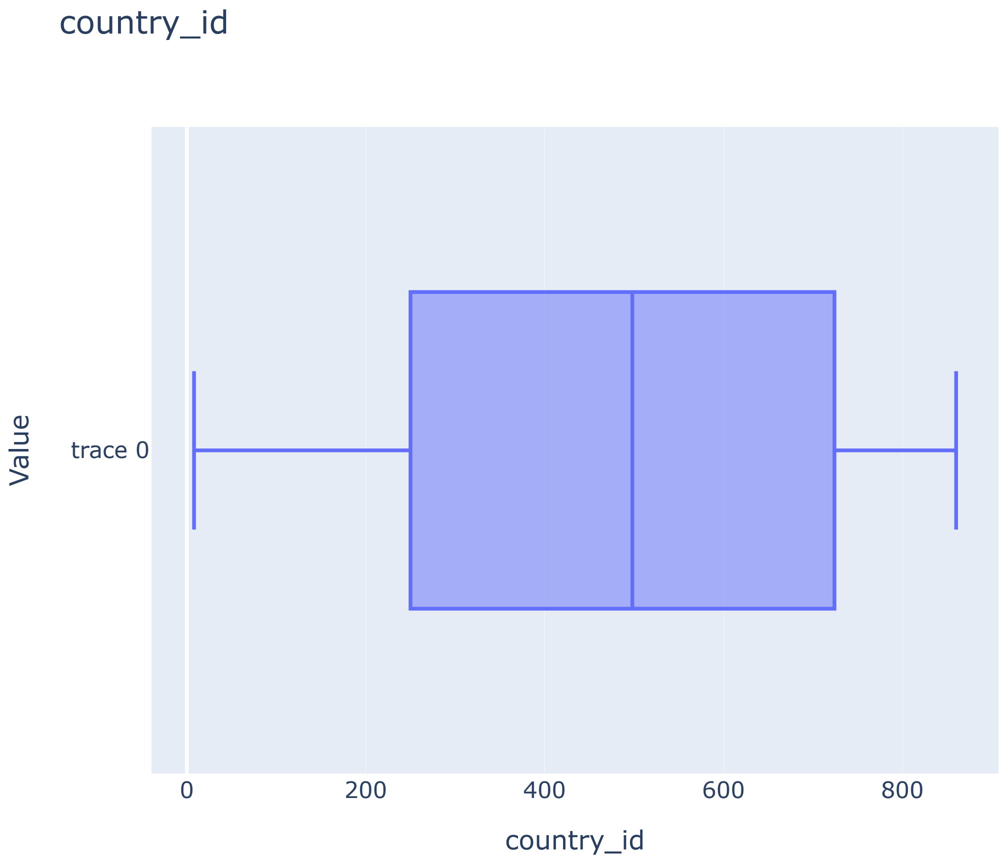


amount_local_currency

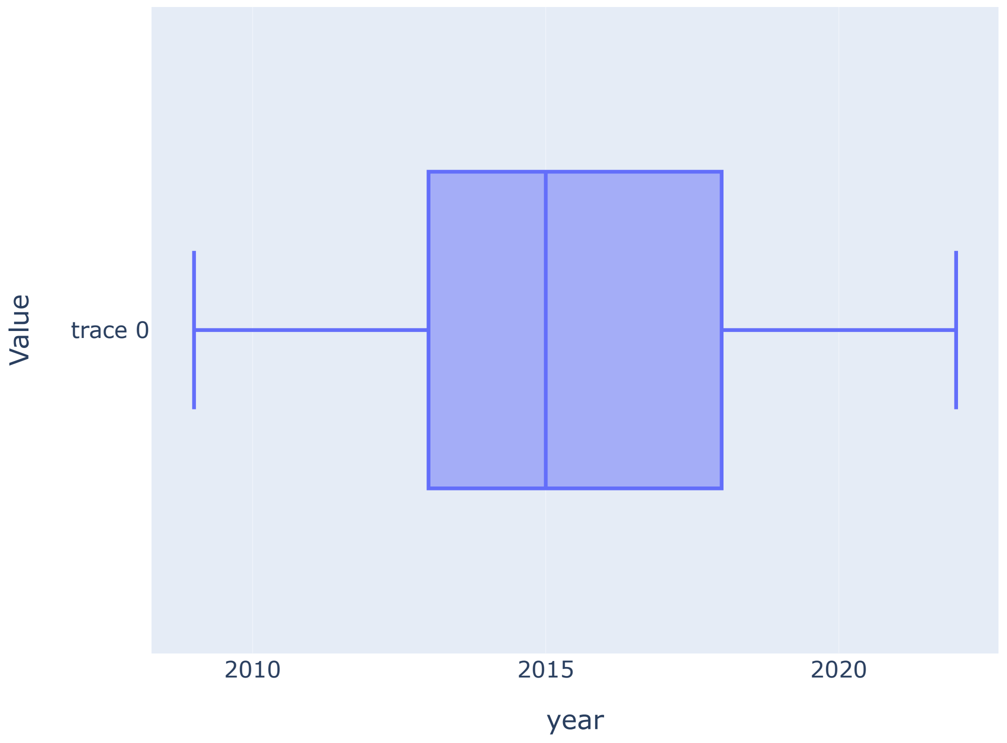


In [24]:

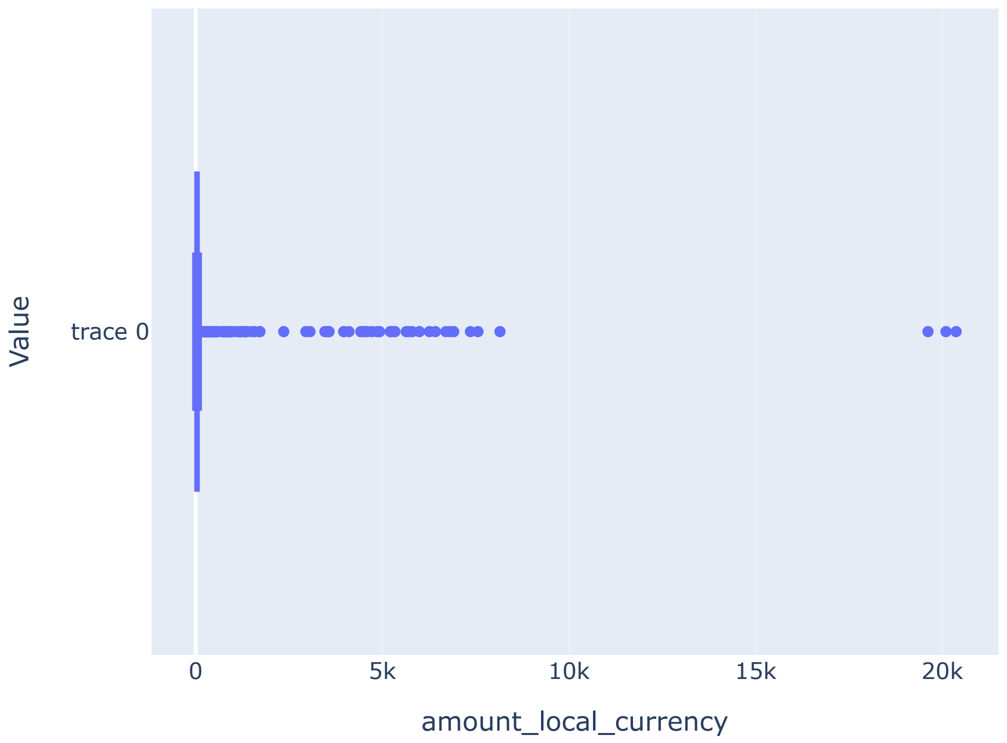
```
for i in numerical_columns:
    fig = go.Figure(data=[go.Box(x=df[i])])
    fig.update_layout(
        title=i,
        xaxis_title=i,
        yaxis_title="Value")
    fig.show()
```



year

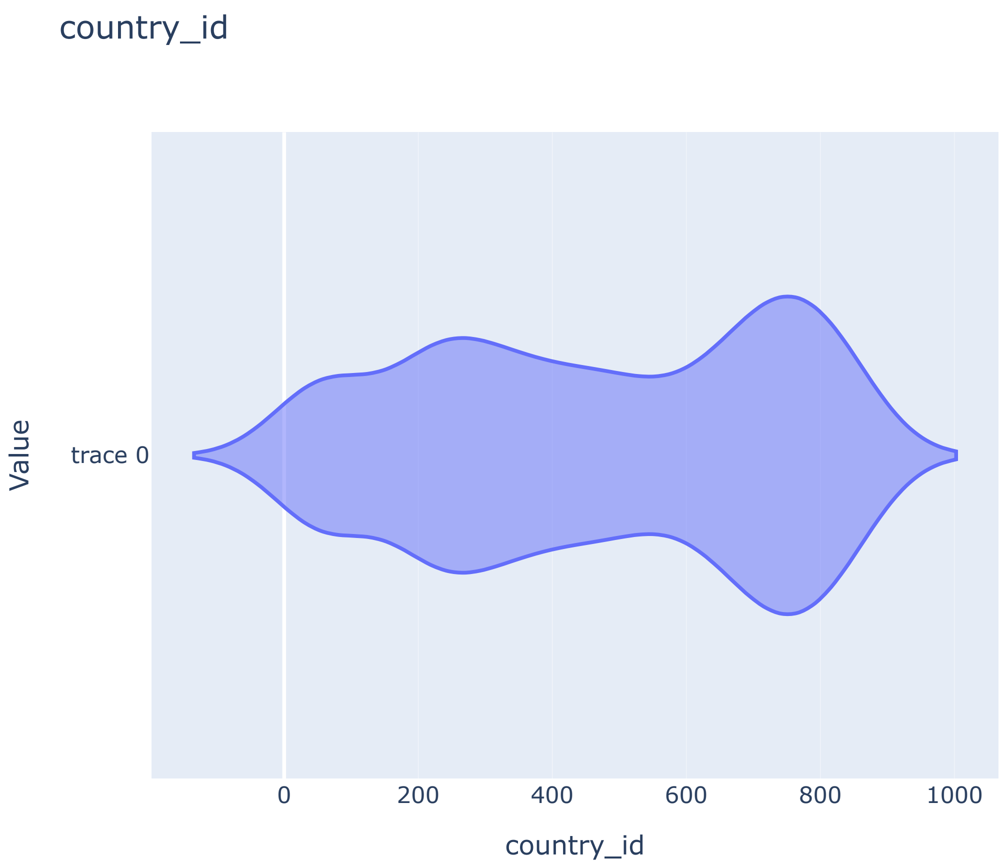


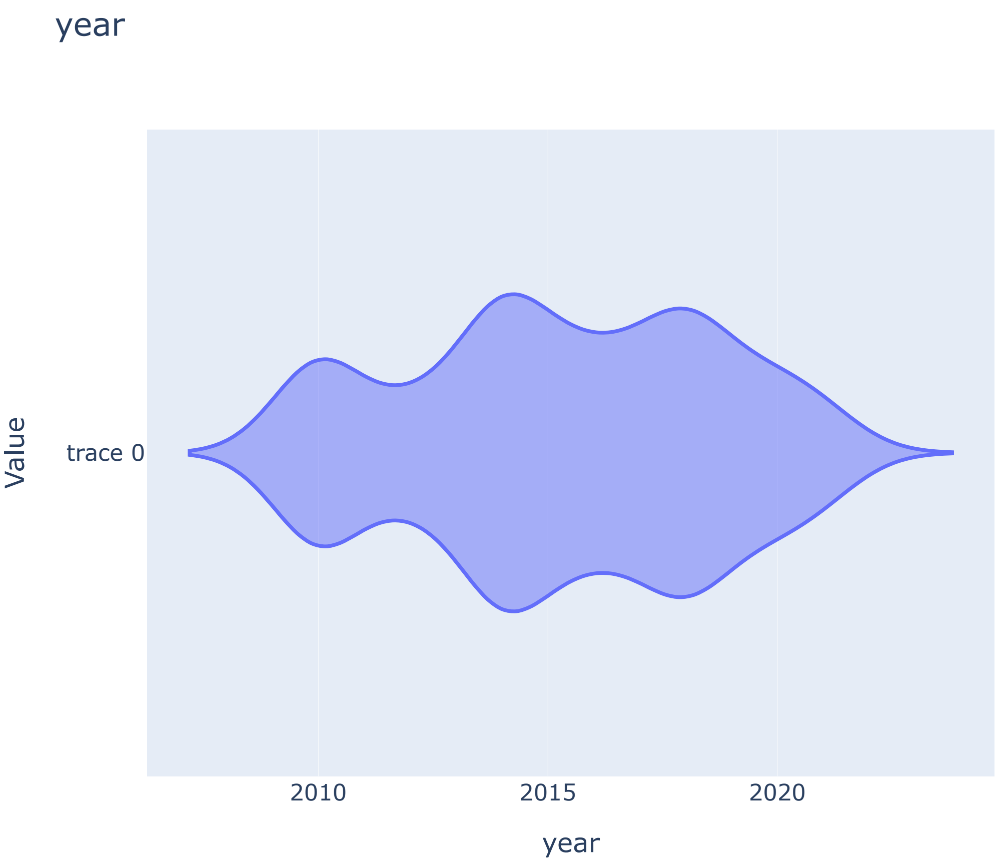
amount_local_currency

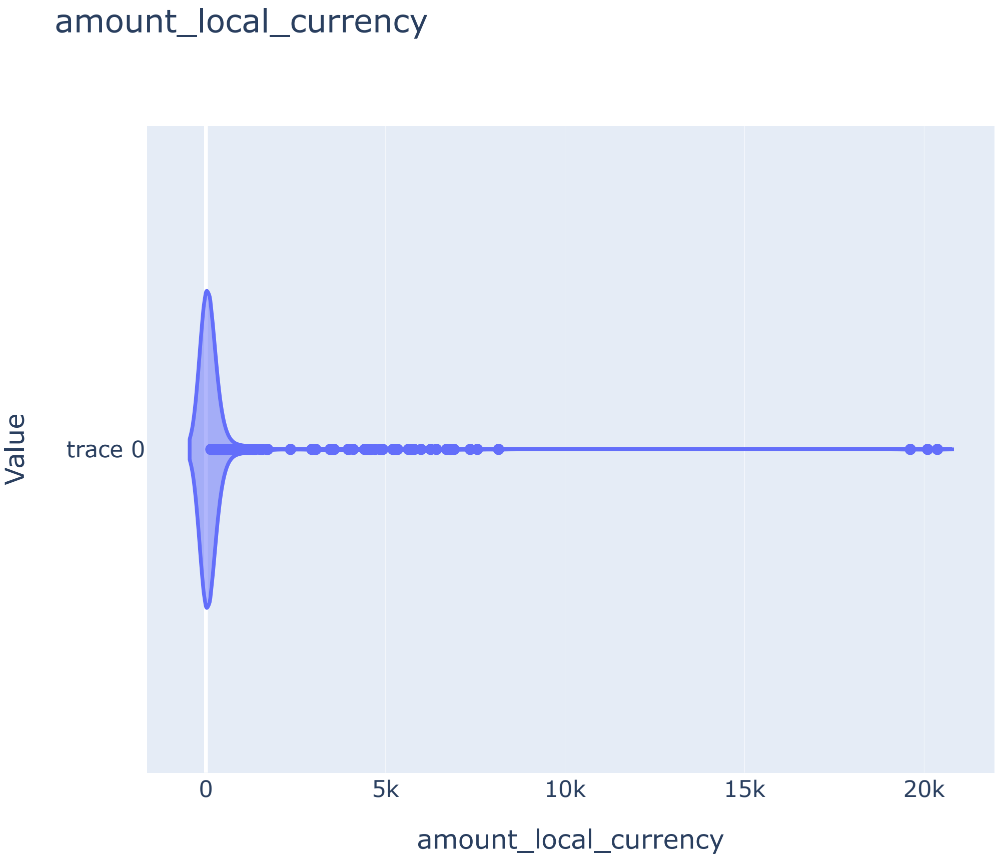


In [25]:

```
for i in numerical_columns:
    fig = go.Figure(data=[go.Violin(x=df[i])])
    fig.update_layout(
        title=i,
        xaxis_title=i,
        yaxis_title="Value")
    fig.show()
```

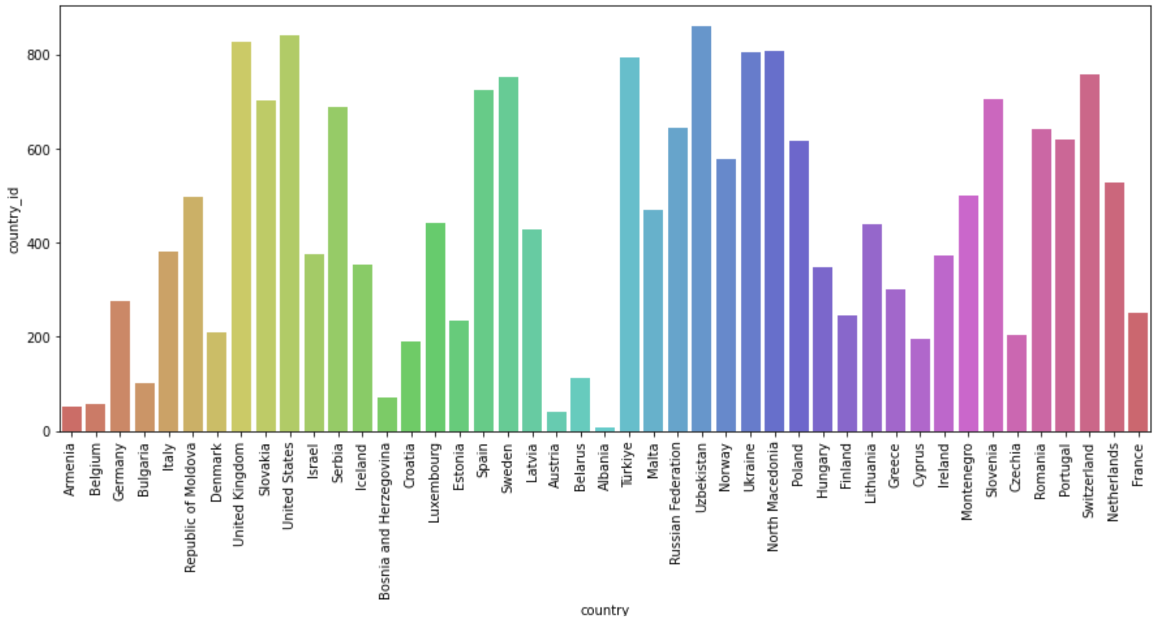



year
Value
trace 0
2010
2015
2020
year


amount_local_currency
Value
trace 0
0
5k
10k
15k
20k
amount_local_currency

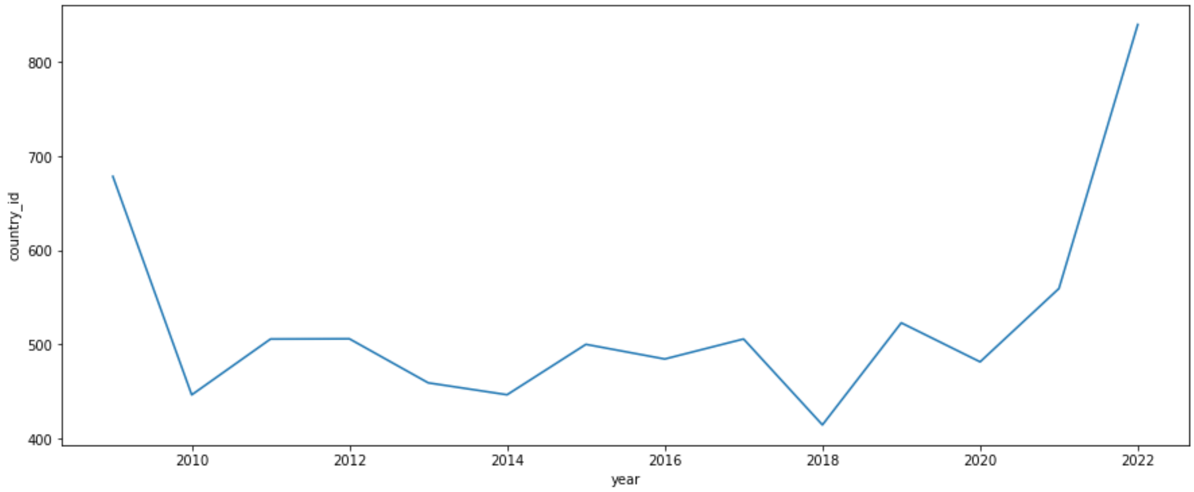
In [26]:

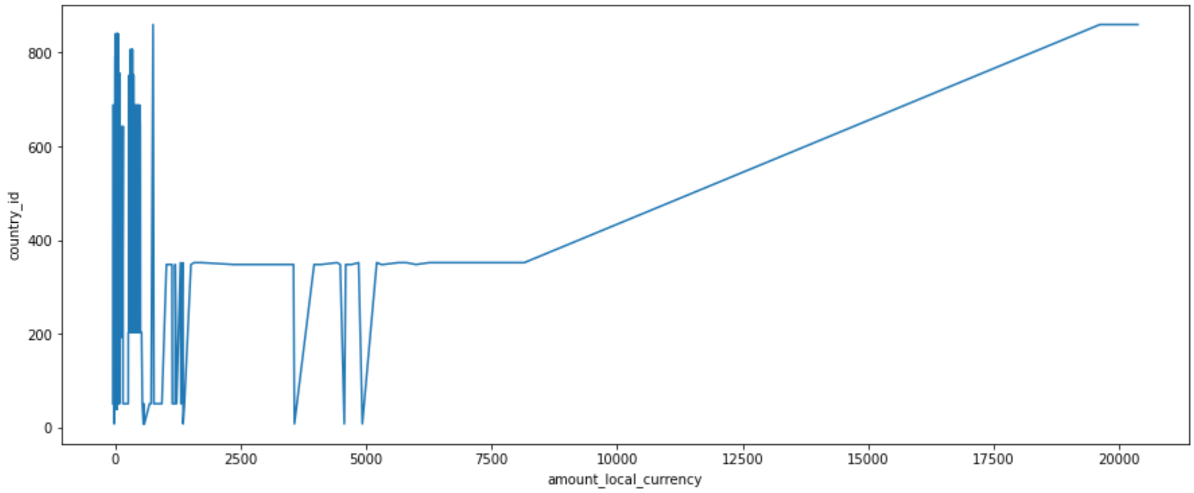
```
for i in numerical_columns:
    for j in object_columns:
        plt.figure(figsize=(15,6))
        sns.barplot(x = df[j], y = df[i], data = df, ci = None, palette = 'hls')
        plt.xticks(rotation = 90)
        plt.show()
```

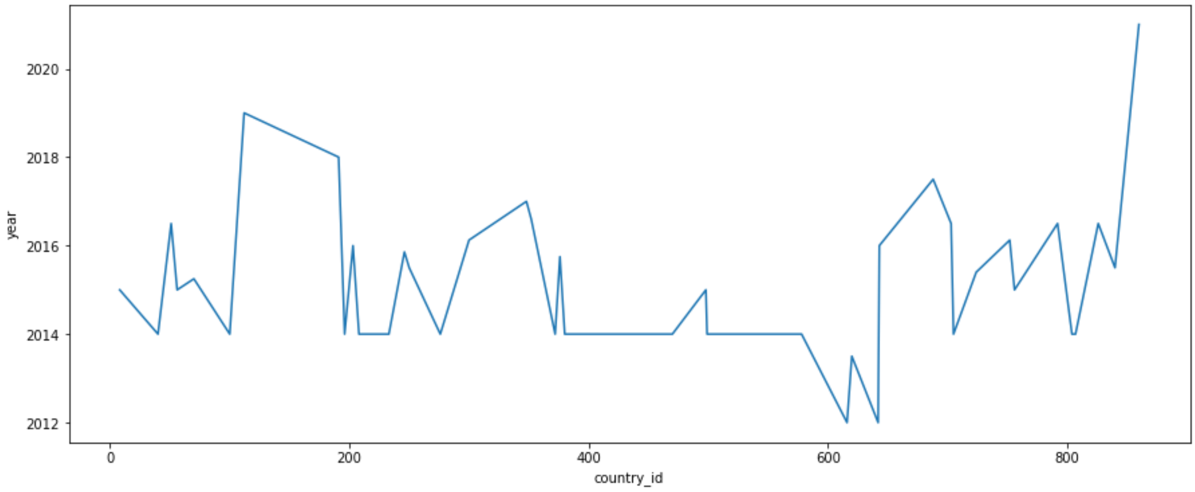


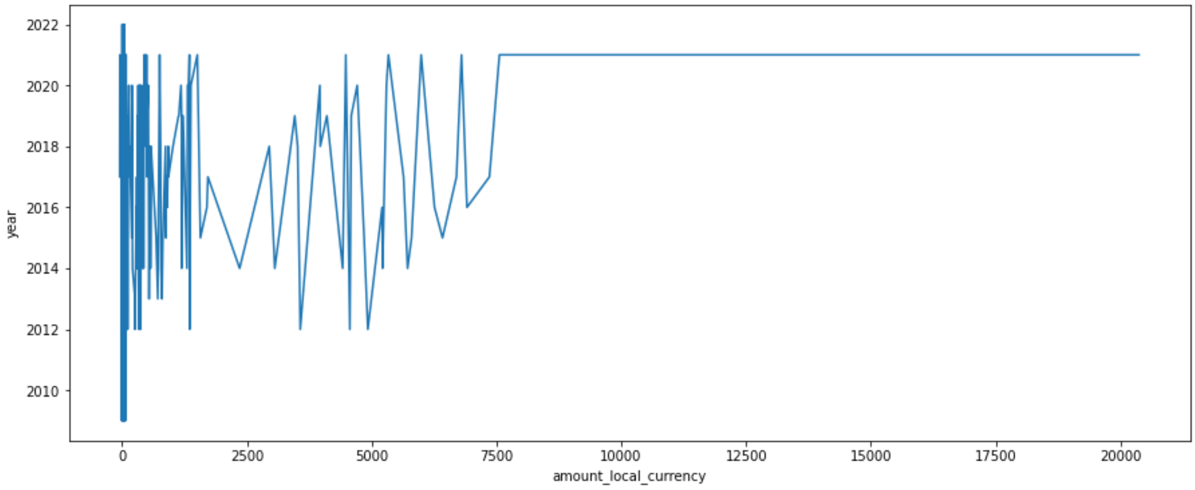
In [27]:

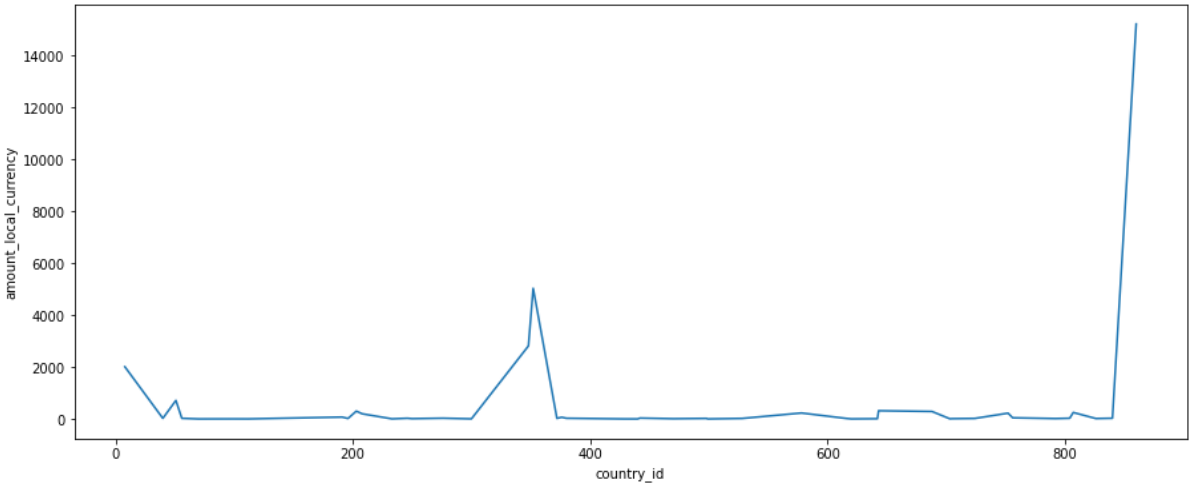
```
for i in numerical_columns:
    for j in numerical_columns:
        if i != j:
            plt.figure(figsize=(15,6))
            sns.lineplot(x = df[j], y = df[i], data = df, ci = None, palette = 'hls')
            plt.show()
```

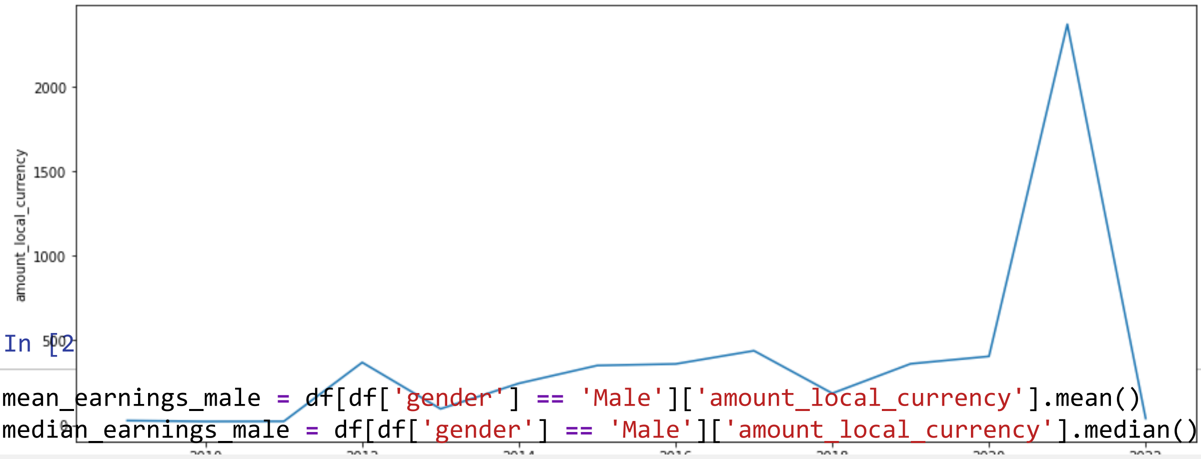












In [2

```
mean_earnings_male = df[df['gender'] == 'Male']['amount_local_currency'].mean()
median_earnings_male = df[df['gender'] == 'Male']['amount_local_currency'].median()
```

In [29]:

```
mean_earnings_female = df[df['gender'] == 'Female']['amount_local_currency'].mean()
median_earnings_female = df[df['gender'] == 'Female']['amount_local_currency'].median()
```

In [30]:

```
print("Mean Earnings (Male):", mean_earnings_male)
print("Median Earnings (Male):", median_earnings_male)
print("Mean Earnings (Female):", mean_earnings_female)
print("Median Earnings (Female):", median_earnings_female)
```

```
Mean Earnings (Male): 478.3003930131005
Median Earnings (Male): 38.66
Mean Earnings (Female): 389.76903930131
Median Earnings (Female): 31.0
```

In [31]:

```
yearly_earnings = df.groupby(['year', 'gender'])['amount_local_currency'].mean().unstack()
```

In [32]:

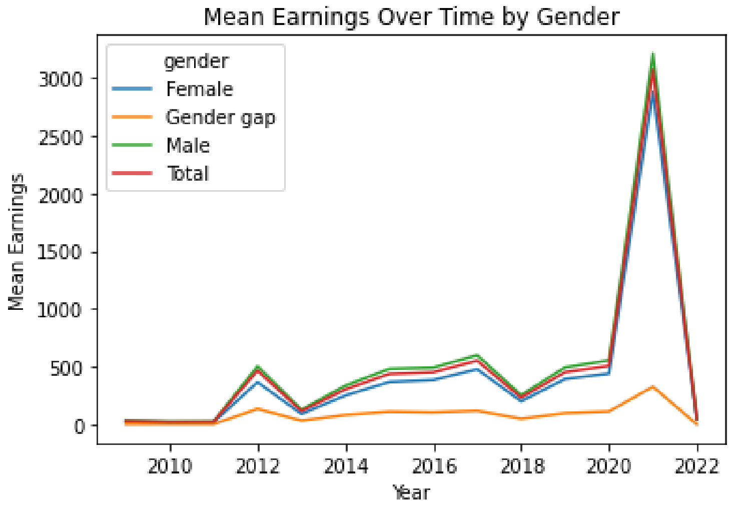
```
yearly_earnings
```

Out[32]:

gender	Female	Gender gap	Male	Total
year				
2009	28.487500	5.712500	34.200000	32.125000
2010	21.190000	6.166429	27.356429	25.420714
2011	22.651667	5.926667	28.578333	26.475000
2012	367.716364	136.923636	504.640000	467.432727
2013	93.605455	36.262727	129.868182	118.183636
2014	253.120769	84.890256	338.011026	305.427949
2015	369.781053	113.021053	482.802105	439.235789
2016	386.647895	107.155263	493.803158	453.517895
2017	479.412353	120.770588	600.182941	553.461765
2018	203.024545	52.804242	255.828788	234.472424
2019	395.278667	100.317333	495.596000	454.092667
2020	441.477333	114.839333	556.316667	508.332000
2021	2877.045455	328.789091	3205.834545	3068.010000
2022	46.870000	3.820000	50.690000	48.990000

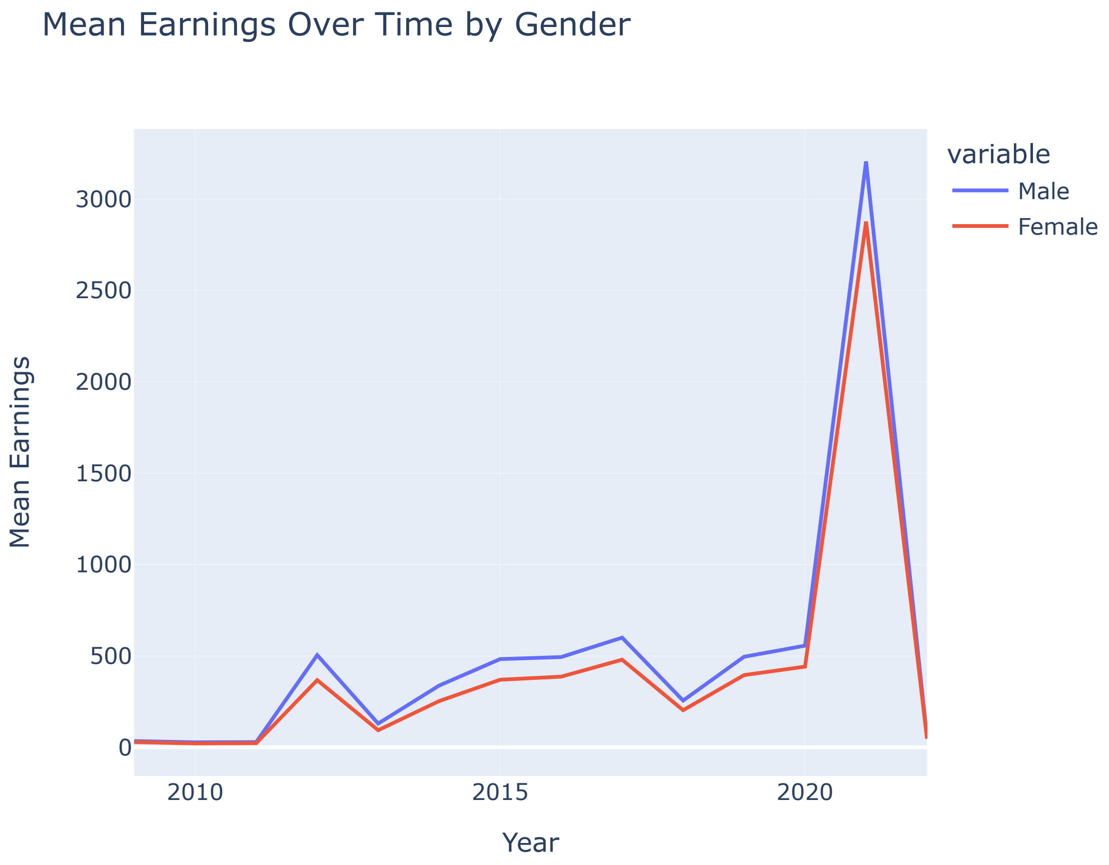
In [33]:

```
yearly_earnings.plot(kind='line')
plt.xlabel('Year')
plt.ylabel('Mean Earnings')
plt.title('Mean Earnings Over Time by Gender')
plt.show()
```



In [34]:

```
fig = px.line(yearly_earnings, x=yearly_earnings.index, y=['Male', 'Female'],
              title='Mean Earnings Over Time by Gender')
fig.update_xaxes(title='Year')
fig.update_yaxes(title='Mean Earnings')
fig.show()
```

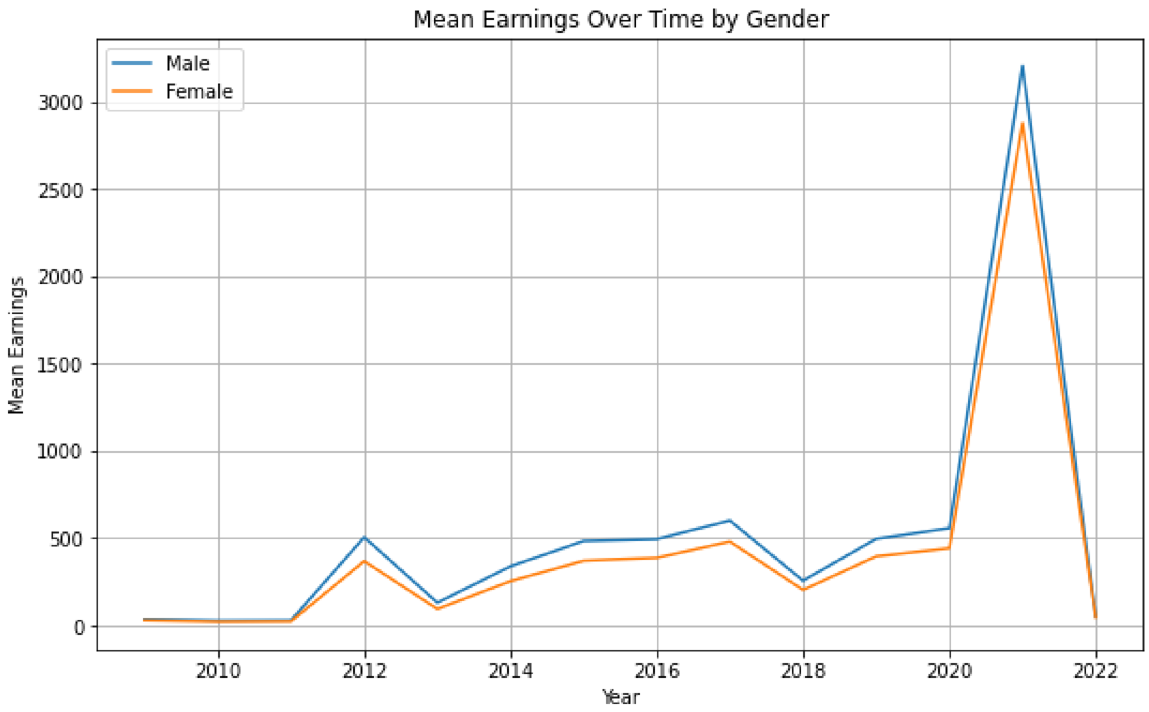


In [35]:

```
earnings_by_year_gender = df.groupby(['year', 'gender'])['amount_local_currency'].mean().unstack(
```

In [36]:

```
plt.figure(figsize=(10, 6))
plt.plot(earnings_by_year_gender.index, earnings_by_year_gender['Male'], label='Male')
plt.plot(earnings_by_year_gender.index, earnings_by_year_gender['Female'], label='Female')
plt.xlabel('Year')
plt.ylabel('Mean Earnings')
plt.title('Mean Earnings Over Time by Gender')
plt.legend()
plt.grid(True)
plt.show()
```



In [37]:

```
gender_pay_gap = mean_earnings_male - mean_earnings_female
print("Gender Pay Gap:", gender_pay_gap)
```

```
Gender Pay Gap: 88.53135371179053
```

In [38]:

```
country_earnings = df.groupby(['country', 'gender'])['amount_local_currency'].mean().unstack()
```

In [39]:

```
country_earnings
```

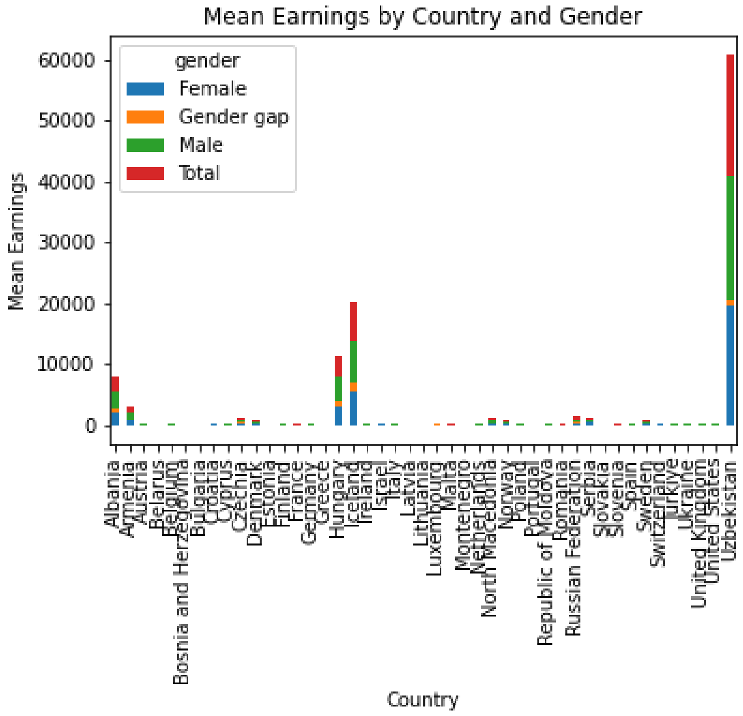
Out[39]:

gender	Female	Gender gap	Male	Total
country				
Albania	2075.785000	667.200000	2742.985000	2566.040000
Armenia	831.622500	140.851250	972.473750	924.191250
Austria	26.943333	10.360000	37.303333	34.863333
Belarus	7.240000	2.740000	9.980000	8.460000
Belgium	33.866667	5.493333	39.360000	37.766667
Bosnia and Herzegovina	7.650000	0.692500	8.342500	8.178750
Bulgaria	9.153333	1.823333	10.976667	10.290000
Croatia	75.000000	31.000000	106.000000	92.000000
Cyprus	27.316667	3.463333	30.780000	29.993333
Czechia	303.611111	114.228889	417.840000	382.307778
Denmark	216.540000	58.783333	275.323333	257.520000
Estonia	9.270000	2.953333	12.223333	10.780000
Finland	33.975714	8.275714	42.251429	39.265714
France	19.231000	2.539000	21.770000	20.848000
Germany	32.916667	12.226667	45.143333	42.006667
Greece	10.491250	1.158750	11.650000	11.257500
Hungary	2870.078333	1032.506667	3902.585000	3456.608333
Iceland	5384.000000	1528.000000	6912.000000	6308.000000
Ireland	27.126667	8.676667	35.803333	32.006667
Israel	78.325000	15.900000	94.225000	89.350000
Italy	30.026667	13.093333	43.120000	40.200000
Latvia	7.680000	1.610000	9.290000	8.473333
Lithuania	6.420000	1.200000	7.620000	7.123333
Luxembourg	39.556667	11.636667	51.193333	48.890000
Malta	14.623333	4.160000	18.783333	17.396667
Montenegro	8.300000	0.530000	8.830000	8.680000
Netherlands	25.840000	7.653333	33.493333	31.293333
North Macedonia	302.120000	40.620000	342.740000	329.640000
Norway	247.400000	72.940000	320.340000	293.436667
Poland	23.220000	8.915000	32.135000	28.215000
Portugal	8.154000	1.954000	10.108000	9.394000
Republic of Moldova	29.875385	3.775385	33.650769	31.985385
Romania	17.785000	0.790000	18.575000	18.245000
Russian Federation	314.500000	135.000000	449.500000	385.000000
Serbia	371.841250	19.646250	391.487500	384.760000
Slovakia	11.220000	3.918333	15.138333	13.653333
Slovenia	18.140000	3.140000	21.280000	19.963333
Spain	26.991000	6.011000	33.002000	31.134000
Sweden	265.636250	42.762500	308.398750	291.610000

gender	Female	Gender gap	Male	Total
country				
Switzerland	51.126154	15.586154	66.712308	61.667692
Türkiye	26.958000	-1.256000	25.702000	25.958000
Ukraine	34.205000	8.980000	43.185000	38.840000
United Kingdom	21.915000	6.211667	28.126667	25.881667
United States	36.442857	7.015714	43.458571	40.382857
Uzbekistan	19616.000000	754.200000	20370.200000	20098.000000

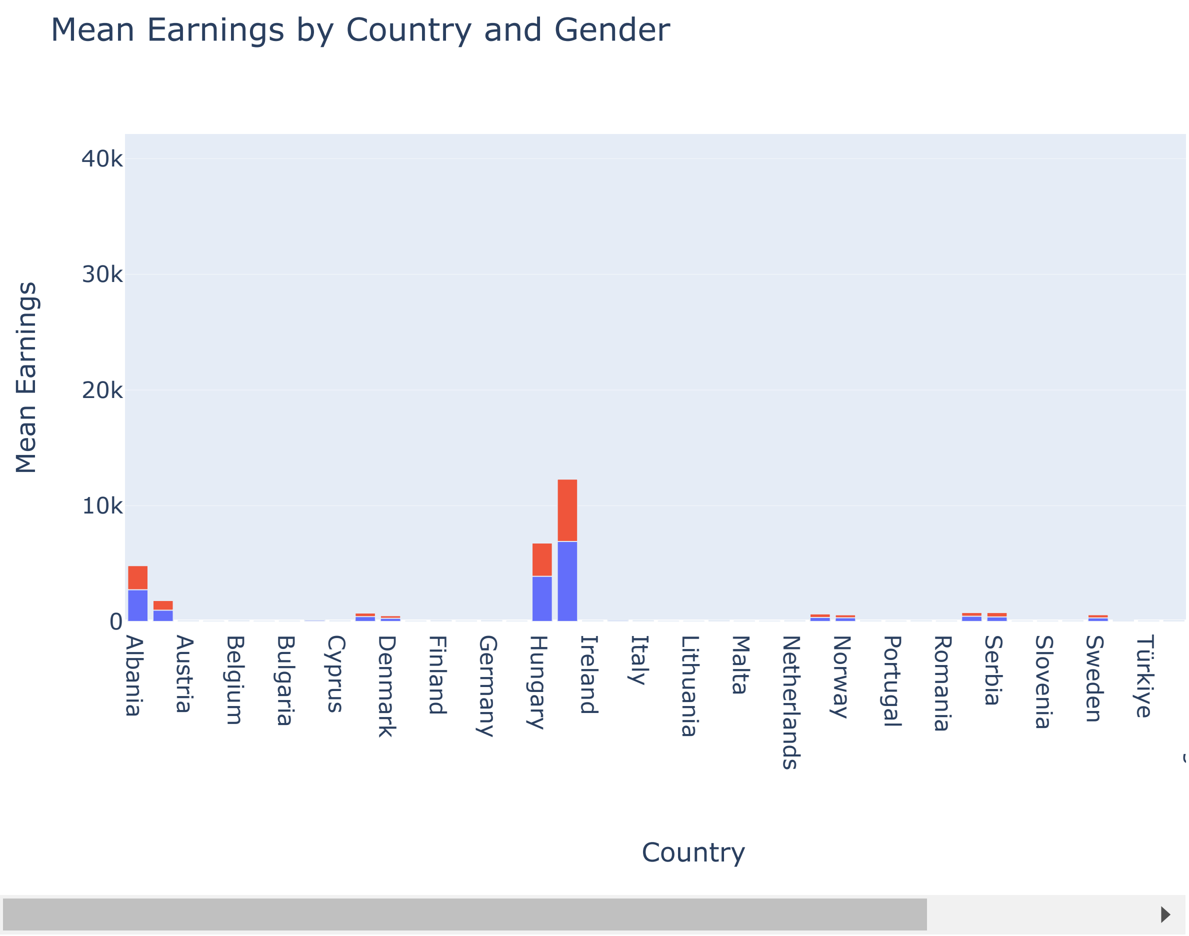
In [40]:

```
country_earnings.plot(kind='bar', stacked=True)
plt.xlabel('Country')
plt.ylabel('Mean Earnings')
plt.title('Mean Earnings by Country and Gender')
plt.show()
```



In [41]:

```
fig = px.bar(country_earnings, x=country_earnings.index, y=['Male', 'Female'],
             title='Mean Earnings by Country and Gender')
fig.update_layout(width=800, height=500)
fig.update_xaxes(title_text='Country')
fig.update_yaxes(title_text='Mean Earnings')
fig.show()
```



In [42]:

```
import statsmodels.api as sm
```

In [43]:

```
df_male = df[df['gender'] == 'Male']
df_female = df[df['gender'] == 'Female']
```

In [44]:

```
X_male = df_male[['year', 'country_id']]
X_male = sm.add_constant(X_male)
y_male = df_male['amount_local_currency']
```

In [45]:

```
model_male = sm.OLS(y_male, X_male).fit()
```

In [46]:

```
X_female = df_female[['year', 'country_id']]
X_female = sm.add_constant(X_female)
y_female = df_female['amount_local_currency']
```

In [47]:

```
model_female = sm.OLS(y_female, X_female).fit()
```

In [48]:

```
male_coefficients = model_male.params
female_coefficients = model_female.params
```

In [49]:

```
explained_gap = (male_coefficients - female_coefficients) @ X_female.mean()
unexplained_gap = (y_male.mean() - y_female.mean()) - explained_gap
```

In [50]:

```
print("Explained Gap:", explained_gap)
print("Unexplained Gap:", unexplained_gap)
```

```
Explained Gap: 88.53135371184908
Unexplained Gap: -5.854872142663226e-11
```

In [51]:

```
from scipy.stats import mannwhitneyu
```

In [52]:

```
gender_pay_gap_df = pd.DataFrame(columns=['Country', 'Year', 'p-value', 'Significant'])
```

In [53]:

```
for country in df['country'].unique():
    for year in df['year'].unique():
        data_country_year = df[(df['country'] == country) & (df['year'] == year)]
        male_earnings = data_country_year[data_country_year['gender'] == 'Male']['amount_local_cu
        female_earnings = data_country_year[data_country_year['gender'] == 'Female']['amount_loca
        if len(male_earnings) > 0 and len(female_earnings) > 0:
            stat, p_value = mannwhitneyu(male_earnings, female_earnings, alternative='two-sided')
            significance = 'Yes' if p_value < 0.05 else 'No'
            gender_pay_gap_df = gender_pay_gap_df.append({
                'Country': country,
                'Year': year,
                'p-value': p_value,
                'Significant': significance
            }, ignore_index=True)
```

In [54]:

```
print(gender_pay_gap_df)
```

```
     Country  Year  p-value Significant
0    Armenia  2013      1.0          No
1    Armenia  2014      1.0          No
2    Armenia  2015      1.0          No
3    Armenia  2016      1.0          No
4    Armenia  2017      1.0          No
..       ...   ...      ...         ...
224   France  2018      1.0          No
225   France  2019      1.0          No
226   France  2020      1.0          No
227   France  2011      1.0          No
228   France  2012      1.0          No

[229 rows x 4 columns]
```