In [28]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [29]:
```
df=pd.read_csv("18_world-data-2023.csv")
df.fillna(0,inplace=True)
df
```

Out[29]:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code	Capital/Major City	Co2-Emissions	...	Out of pocket health expenditure	Physicians per thousand	Populat
0	Afghanistan	60	AF	58.10%	652,230	323,000	32.49	93.0	Kabul	8,672	...	78.40%	0.28	38,041,
1	Albania	105	AL	43.10%	28,748	9,000	11.78	355.0	Tirana	4,536	...	56.90%	1.20	2,854,
2	Algeria	18	DZ	17.40%	2,381,741	317,000	24.28	213.0	Algiers	150,006	...	28.10%	1.72	43,053,
3	Andorra	164	AD	40.00%	468	0	7.20	376.0	Andorra la Vella	469	...	36.40%	3.33	77,
4	Angola	26	AO	47.50%	1,246,700	117,000	40.73	244.0	Luanda	34,693	...	33.40%	0.21	31,825,
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
190	Venezuela	32	VE	24.50%	912,050	343,000	17.88	58.0	Caracas	164,175	...	45.80%	1.92	28,515,
191	Vietnam	314	VN	39.30%	331,210	522,000	16.75	84.0	Hanoi	192,668	...	43.50%	0.82	96,462,
192	Yemen	56	YE	44.60%	527,968	40,000	30.45	967.0	Sanaa	10,609	...	81.00%	0.31	29,161,
193	Zambia	25	ZM	32.10%	752,618	16,000	36.19	260.0	Lusaka	5,141	...	27.50%	1.19	17,861,
194	Zimbabwe	38	ZW	41.90%	390,757	51,000	30.68	263.0	Harare	10,983	...	25.80%	0.21	14,645,

195 rows × 35 columns

In [30]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Data columns (total 35 columns):
 #   Column                                    Non-Null Count  Dtype  
---  ------                                    --------------  -----  
 0   Country                                   195 non-null    object 
 1   Density
(P/Km2)                          195 non-null    object 
 2   Abbreviation                              195 non-null    object 
 3   Agricultural Land( %)                     195 non-null    object 
 4   Land Area(Km2)                            195 non-null    object 
 5   Armed Forces size                         195 non-null    object 
 6   Birth Rate                                195 non-null    float64
 7   Calling Code                              195 non-null    float64
 8   Capital/Major City                        195 non-null    object 
 9   Co2-Emissions                             195 non-null    object 
 10  CPI                                       195 non-null    object 
 11  CPI Change (%)                            195 non-null    object 
 12  Currency-Code                             195 non-null    object 
 13  Fertility Rate                            195 non-null    float64
 14  Forested Area (%)                         195 non-null    object 
 15  Gasoline Price                            195 non-null    object 
 16  GDP                                       195 non-null    object 
 17  Gross primary education enrollment (%)    195 non-null    object 
 18  Gross tertiary education enrollment (%)   195 non-null    object 
 19  Infant mortality                          195 non-null    float64
 20  Largest city                              195 non-null    object 
 21  Life expectancy                           195 non-null    float64
 22  Maternal mortality ratio                  195 non-null    float64
 23  Minimum wage                              195 non-null    object 
 24  Official language                         195 non-null    object 
 25  Out of pocket health expenditure          195 non-null    object 
 26  Physicians per thousand                   195 non-null    float64
 27  Population                                195 non-null    object 
 28  Population: Labor force participation (%) 195 non-null    object 
 29  Tax revenue (%)                           195 non-null    object 
 30  Total tax rate                            195 non-null    object 
 31  Unemployment rate                         195 non-null    object 
 32  Urban_population                          195 non-null    object 
 33  Latitude                                  195 non-null    float64
 34  Longitude                                 195 non-null    float64
dtypes: float64(9), object(26)
memory usage: 53.4+ KB
```

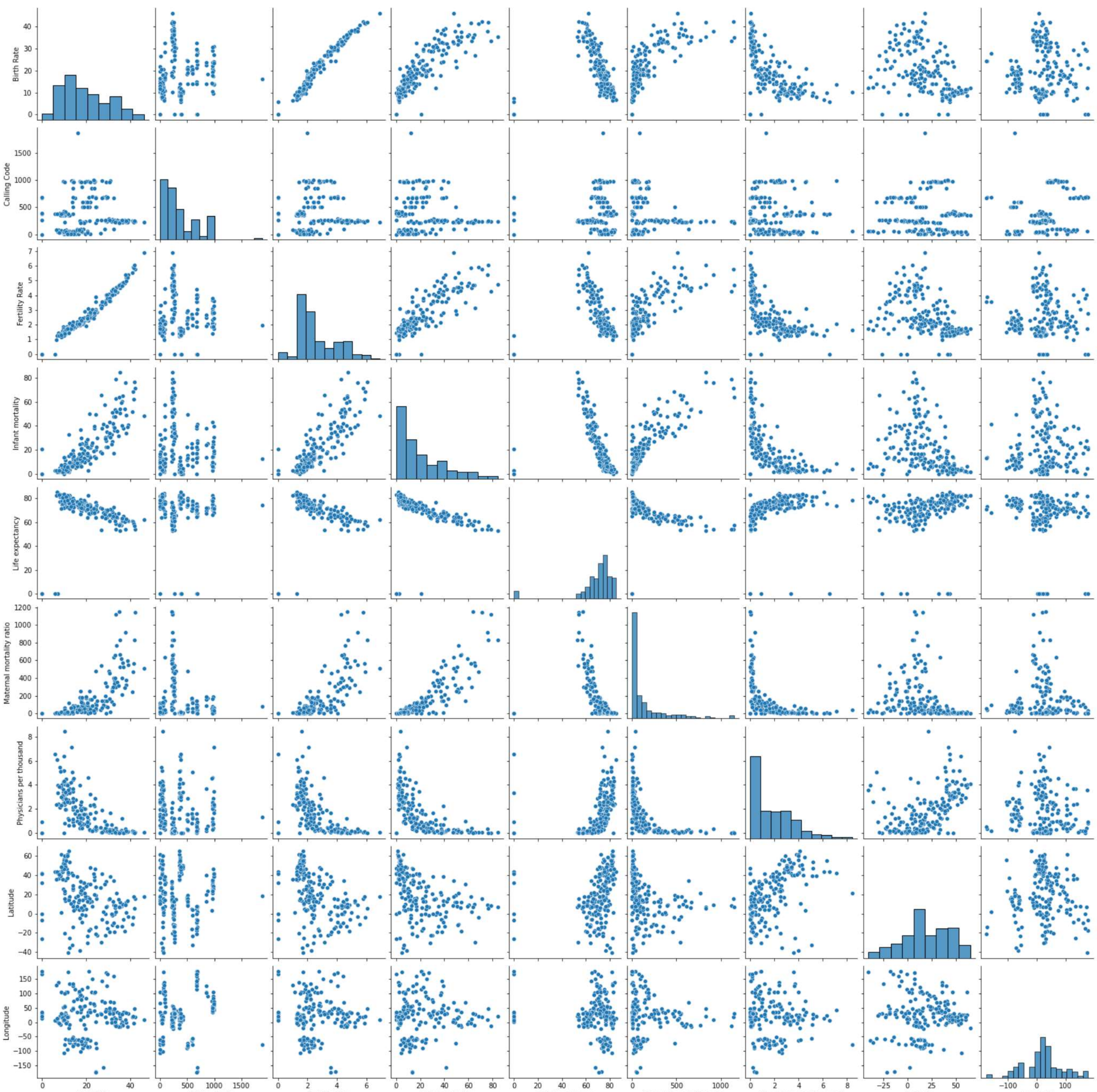
In [31]:
```
df.describe()
```

Out[31]:

	Birth Rate	Calling Code	Fertility Rate	Infant mortality	Life expectancy	Maternal mortality ratio	Physicians per thousand	Latitude	Longitude
count	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000
mean	19.592974	358.697436	2.601282	20.676410	69.314359	148.876923	1.773795	18.994442	20.128678
std	10.397534	323.434462	1.355777	19.594644	16.133643	228.717593	1.688826	23.939018	66.559711
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-40.900557	-175.198242
25%	10.675000	81.500000	1.625000	5.000000	66.150000	9.000000	0.245000	4.372880	-7.658537
50%	17.800000	255.000000	2.200000	13.700000	72.800000	43.000000	1.300000	17.189877	20.939444
75%	28.445000	506.500000	3.565000	31.550000	77.250000	175.000000	2.875000	40.106102	48.046657
max	46.080000	1876.000000	6.910000	84.500000	85.400000	1150.000000	8.420000	64.963051	178.065032

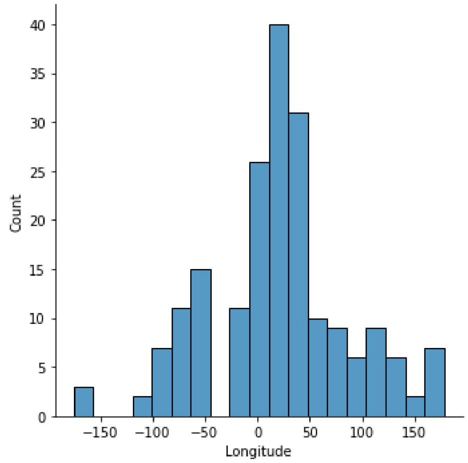
In [32]:
```
sns.pairplot(df)
```

Out[32]:
```
<seaborn.axisgrid.PairGrid at 0x2b151c30be0>
```



In [33]:
```
sns.displot(df['Longitude'])
```

Out[33]:
```
<seaborn.axisgrid.FacetGrid at 0x2b15537d970>
```



In [34]:
```
df1=df.drop(['Country'],axis=1)
df1
```

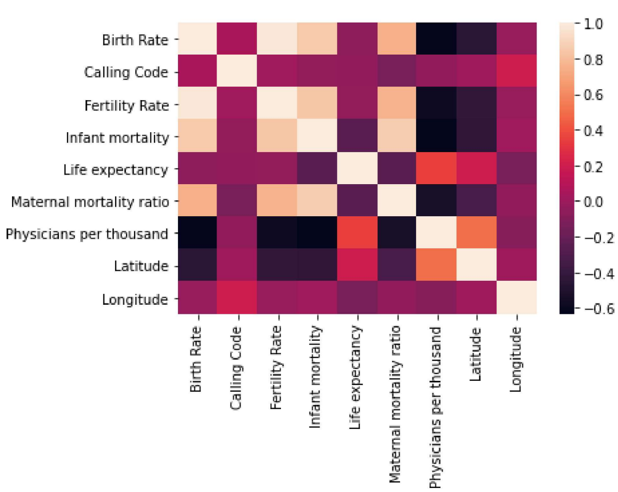
Out[34]:

	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code	Capital/Major City	Co2-Emissions	CPI	...	Out of pocket health expenditure	Physicians per thousand	Populatior
0	60	AF	58.10%	652,230	323,000	32.49	93.0	Kabul	8,672	149.9	...	78.40%	0.28	38,041,754
1	105	AL	43.10%	28,748	9,000	11.78	355.0	Tirana	4,536	119.05	...	56.90%	1.20	2,854,191
2	18	DZ	17.40%	2,381,741	317,000	24.28	213.0	Algiers	150,006	151.36	...	28.10%	1.72	43,053,054
3	164	AD	40.00%	468	0	7.20	376.0	Andorra la Vella	469	0	...	36.40%	3.33	77,142
4	26	AO	47.50%	1,246,700	117,000	40.73	244.0	Luanda	34,693	261.73	...	33.40%	0.21	31,825,295
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
190	32	VE	24.50%	912,050	343,000	17.88	58.0	Caracas	164,175	2,740.27	...	45.80%	1.92	28,515,829
191	314	VN	39.30%	331,210	522,000	16.75	84.0	Hanoi	192,668	163.52	...	43.50%	0.82	96,462,106
192	56	YE	44.60%	527,968	40,000	30.45	967.0	Sanaa	10,609	157.58	...	81.00%	0.31	29,161,922
193	25	ZM	32.10%	752,618	16,000	36.19	260.0	Lusaka	5,141	212.31	...	27.50%	1.19	17,861,030
194	38	ZW	41.90%	390,757	51,000	30.68	263.0	Harare	10,983	105.51	...	25.80%	0.21	14,645,468

195 rows × 34 columns

In [35]:
```
sns.heatmap(df1.corr())
```

Out[35]:
```
<AxesSubplot:>
```



In [36]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [37]:
```
y=df['Longitude']
x=df1.drop(['Longitude','Abbreviation','Agricultural Land( %)','Land Area(Km2)','Armed Forces size','Capital/Major City','Co2-Emi
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
print(x_train)
```

```
     Birth Rate  Calling Code   Latitude
65        29.41         233.0   7.946527
85         7.40          81.0  36.204824
67        16.47           1.0  12.116500
72        24.35         509.0  18.971187
137       17.95          51.0  -9.189967
..          ...           ...        ...
187       13.86         598.0 -32.522779
152       34.52         221.0  14.497401
81        12.50         353.0  53.412910
43        10.46         357.0  35.126413
64         9.50          49.0  51.165691

[136 rows x 3 columns]
```

In [38]:
```
model=LinearRegression()
model.fit(x_train,y_train)
model.intercept_
```

Out[38]: 2.63306406840638

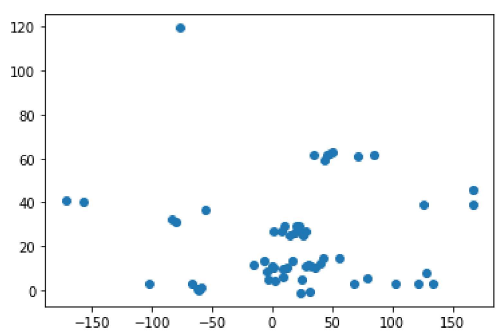
In [39]:
```
coeff=pd.DataFrame(model.coef_,x.columns,columns=["Coefficient"])
coeff
```

Out[39]:

	Coefficient
Birth Rate	-0.209231
Calling Code	0.063694
Latitude	0.039595

In [40]:
```
prediction=model.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[40]: <matplotlib.collections.PathCollection at 0x2b15762cbb0>



In [41]:
```
model.score(x_test,y_test)
```

Out[41]: -0.12023209970493265

In [42]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [43]:
```
rr = Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[43]: Ridge(alpha=10)

In [44]:
```
rr.score(x_test,y_test)
```

Out[44]: -0.12023037023015282

In [45]:
```
la = Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[45]: Lasso(alpha=10)

In [46]:
```
la.score(x_test,y_test)
```

Out[46]:
```
-0.11864899781324945
```

In [ ]: