In [2]:

```
import pandas as pd
import numpy as np
import plotly.express as px
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

Reading external dataset

In [3]:

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

In [4]:

```
df.head()
```

Out[4]:

	work_year	experience_level	employment_type	job_title	salary	salary_currency	sal
0	2023	EX	FT	Data Science Director	212000	USD	
1	2023	EX	FT	Data Science Director	190000	USD	
2	2023	MI	FT	Business Intelligence Engineer	35000	GBP	
3	2023	MI	FT	Business Intelligence Engineer	35000	GBP	
4	2023	SE	FT	Machine Learning Engineer	245700	USD	

In [5]:

```
df.shape
```

Out[5]:

```
(8805, 11)
```

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8805 entries, 0 to 8804
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   work_year           8805 non-null   int64 
 1   experience_level    8805 non-null   object
 2   employment_type     8805 non-null   object
 3   job_title           8805 non-null   object
 4   salary              8805 non-null   int64 
 5   salary_currency     8805 non-null   object
 6   salary_in_usd       8805 non-null   int64 
 7   employee_residence  8805 non-null   object
 8   remote_ratio        8805 non-null   int64 
 9   company_location    8805 non-null   object
 10  company_size        8805 non-null   object
dtypes: int64(4), object(7)
memory usage: 756.8+ KB
```

Producing various descriptive statistics of the analytic dataset

In [7]:

```
df['salary_in_usd'].describe()
```

Out[7]:

```
count      8805.000000
mean     149488.265645
std       64222.105058
min       15000.000000
25%      105000.000000
50%      142200.000000
75%      185900.000000
max      615201.000000
Name: salary_in_usd, dtype: float64
```

In [8]:

```
median_salary = df['salary_in_usd'].median()
Variance = df['salary_in_usd'].var()
salary_range = df['salary_in_usd'].max() - df['salary_in_usd'].min()
mode_salary = df['salary_in_usd'].mode()
```

In [9]:

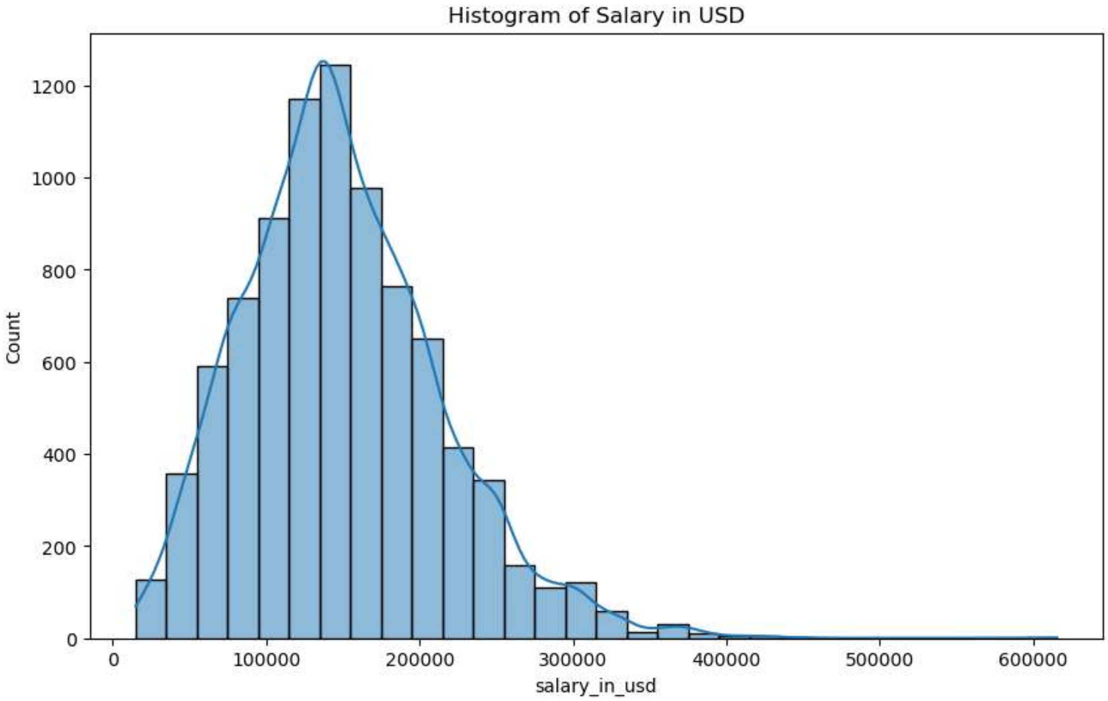
```
print("Median Salary: ", median_salary)
print("Salary Variance: ", Variance)
print("Salary Range: ", salary_range)
print("Mode Salary: ", mode_salary.tolist())
```

```
Median Salary:  142200.0
Salary Variance:  4124478778.131113
Salary Range:  600201
Mode Salary:  [150000]
```

Building various possible visualizations of dataset variables

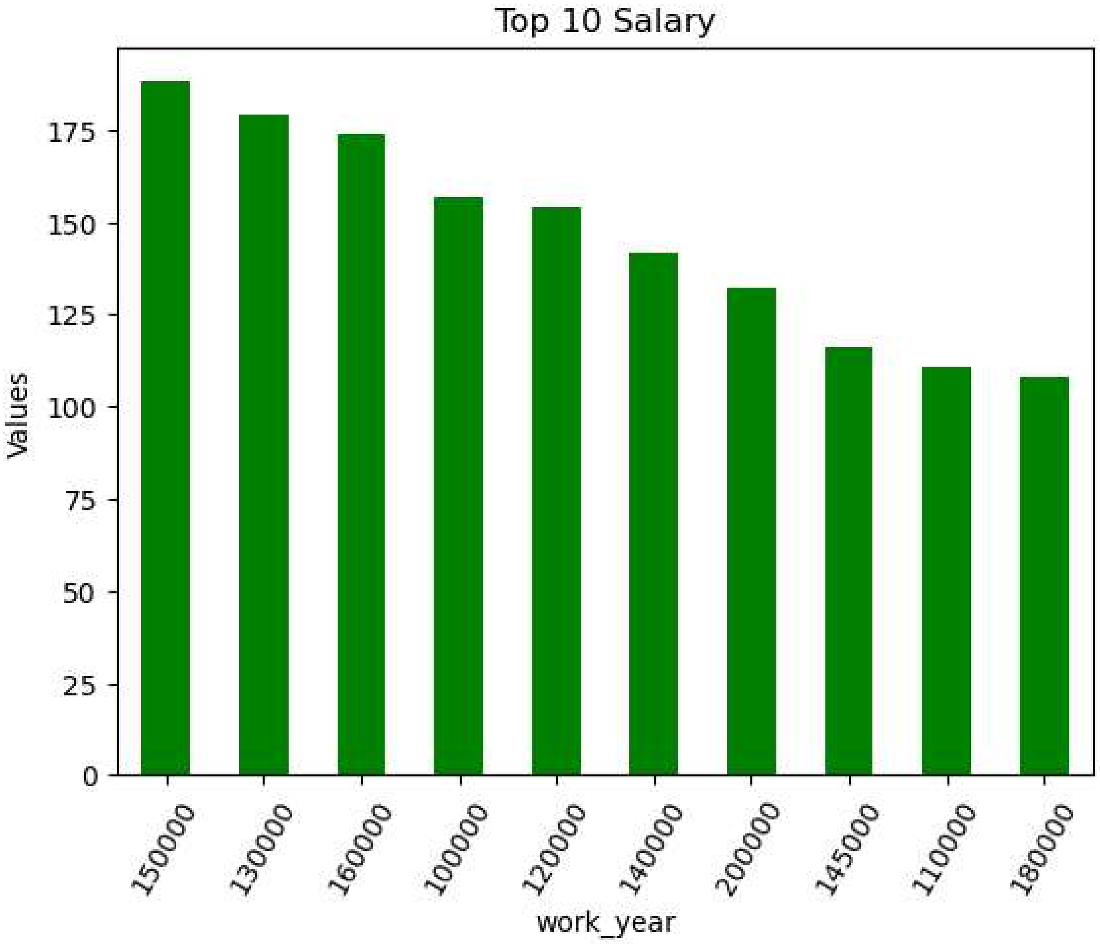
In [10]:

```
plt.figure(figsize=(10, 6))
sns.histplot(df['salary_in_usd'], bins=30, kde=True)
plt.title('Histogram of Salary in USD')
plt.show()
```



In [20]:

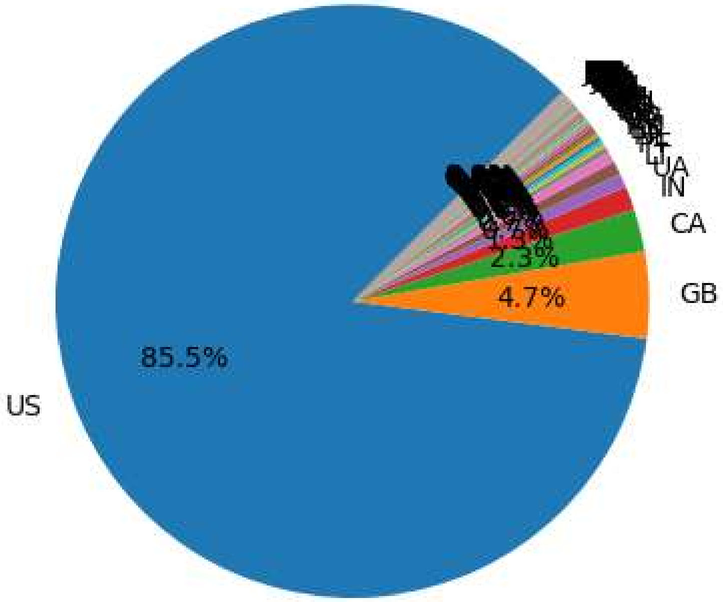
```
 Country=df["salary_in_usd"].value_counts()
Country[:10].plot(kind="bar",color="green")
plt.title("Top 10 Salary")
plt.xlabel("work_year")
plt.ylabel("Values")
plt.xticks(rotation=60)
plt.show()
```



In [18]:

```
data=df["employee_residence"].value_counts()
label=df["employee_residence"].unique()

fig,res=plt.subplots()
res.pie(data,labels=label,autopct="%1.1f%%",startangle=45)
plt.show()
```



Checking any records in the dataset with missing values and handling them appropriately

In [30]:

```
df.isnull()
```

Out[30]:

	work_year	experience_level	employment_type	job_title	salary	salary_currency	sala
0	False	False	False	False	False	False	
1	False	False	False	False	False	False	
2	False	False	False	False	False	False	
3	False	False	False	False	False	False	
4	False	False	False	False	False	False	
...	...	...	...	...	...	...	
8800	False	False	False	False	False	False	
8801	False	False	False	False	False	False	
8802	False	False	False	False	False	False	
8803	False	False	False	False	False	False	
8804	False	False	False	False	False	False	

8805 rows × 11 columns

In [31]:

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

Out[31]:

```
work_year             0
experience_level      0
employment_type       0
job_title             0
salary                0
salary_currency       0
salary_in_usd         0
employee_residence    0
remote_ratio          0
company_location      0
company_size          0
dtype: int64
```

```
In [32]: df.duplicated()
```

```
Out[32]: 0       False
         1       False
         2       False
         3        True
         4       False
                 ...
         8800    False
         8801    False
         8802    False
         8803    False
         8804    False
         Length: 8805, dtype: bool
```

```
In [33]: df.drop_duplicates(inplace=True)
```

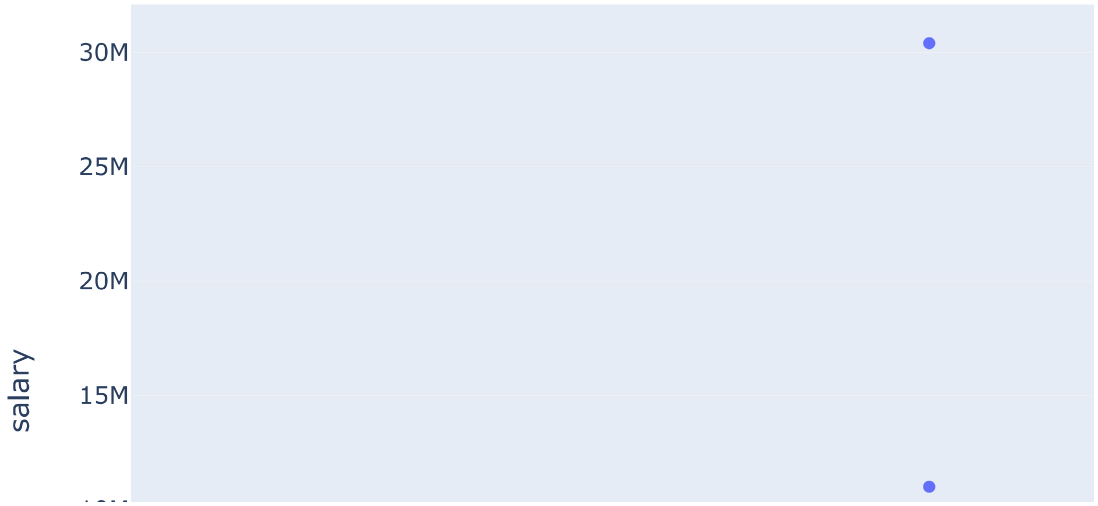
```
In [34]: df.duplicated().sum()
```

```
Out[34]: 0
```

Dealing with Outliers

In [21]:

```
fg = px.box(df,y="salary")
fg.show()
```



In [23]:

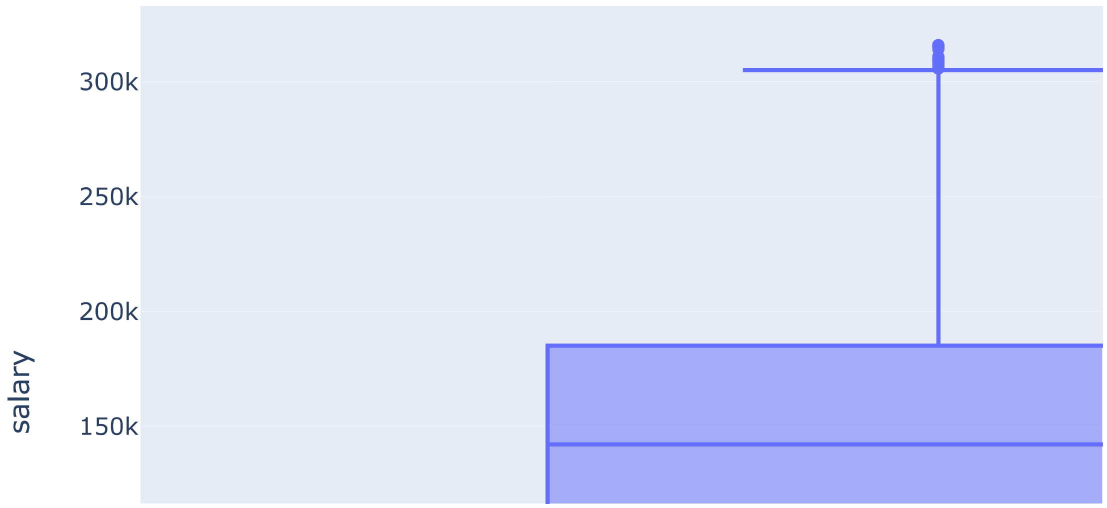
```
Q1 = df.salary.quantile(0.25)
Q3 = df.salary.quantile(0.75)
IQR = Q3-Q1
```

In [24]:

```
LowerFence = Q1-1.5*IQR
UpperFency = Q3+1.5*IQR
FilteredDF = df[(df.salary>=LowerFence) & (df.salary<=UpperFency)]
```

In [25]:

```
fg = px.box(FilteredDF,y="salary")
fg.show()
```



Investigate the unique values of categorical variables

In [72]:

```
categorical_columns = ['experience_level', 'employment_type',
                       'salary_currency', 'employee_residence', 'remote_ra
                       'company_location', 'company_size']
unique_values = {col: df[col].unique() for col in categorical_columns}
```

In [73]:

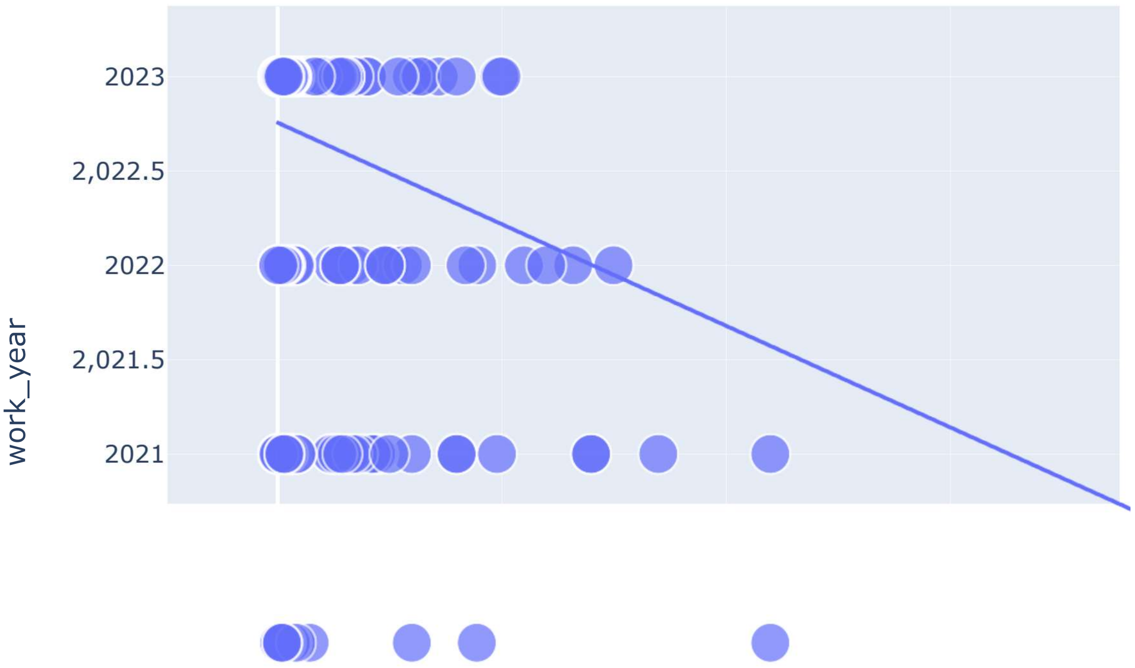
```
print(unique_values)
```

```
{'experience_level': array(['EX', 'MI', 'SE', 'EN'], dtype=object), 'emp
loyment_type': array(['FT', 'FL', 'CT', 'PT'], dtype=object), 'salary_cu
rrency': array(['USD', 'GBP', 'EUR', 'AUD', 'INR', 'CAD', 'PLN', 'BRL',
'PHP',
       'TRY', 'CHF', 'NOK', 'JPY', 'ZAR', 'HKD', 'SGD', 'ILS', 'THB',
       'HUF', 'DKK', 'MXN', 'CLP'], dtype=object), 'employee_residence':
array(['US', 'GB', 'CA', 'IN', 'UA', 'LT', 'PT', 'NL', 'ES', 'AU', 'FR',
       'UG', 'CO', 'IT', 'SI', 'RO', 'GR', 'LV', 'MU', 'DE', 'PL', 'AM',
       'HR', 'TH', 'KR', 'EE', 'TR', 'PH', 'BR', 'QA', 'RU', 'KE', 'TN',
       'GH', 'BE', 'CH', 'AD', 'EC', 'PE', 'MX', 'MD', 'NG', 'SA', 'NO',
       'AR', 'EG', 'UZ', 'GE', 'JP', 'ZA', 'HK', 'CF', 'FI', 'IE', 'IL',
       'AT', 'SG', 'SE', 'KW', 'CY', 'BA', 'PK', 'IR', 'AS', 'HU', 'CN',
       'CR', 'CL', 'PR', 'DK', 'BO', 'DO', 'ID', 'AE', 'MY', 'HN', 'CZ',
       'DZ', 'VN', 'IQ', 'BG', 'JE', 'RS', 'NZ', 'LU', 'MT'], dtype=obje
ct), 'remote_ratio': array([  0, 100,  50], dtype=int64), 'company_locat
ion': array(['US', 'GB', 'CA', 'IN', 'NL', 'LT', 'PT', 'GI', 'AU', 'FR',
'CO',
       'UA', 'SI', 'RO', 'ES', 'GR', 'LV', 'MU', 'DE', 'PL', 'RU', 'IT',
       'KR', 'EE', 'CZ', 'CH', 'BR', 'QA', 'KE', 'DK', 'GH', 'SE', 'PH',
       'TR', 'AD', 'EC', 'MX', 'IL', 'NG', 'SA', 'NO', 'AR', 'JP', 'ZA',
       'HK', 'CF', 'FI', 'IE', 'SG', 'TH', 'HR', 'AM', 'BA', 'PK', 'IR',
       'BS', 'HU', 'AT', 'PR', 'AS', 'BE', 'ID', 'EG', 'AE', 'MY', 'HN',
       'DZ', 'IQ', 'CN', 'NZ', 'CL', 'MD', 'LU', 'MT'], dtype=object),
'company_size': array(['M', 'S', 'L'], dtype=object)}
```

Linear Regression Model

In [27]:

```
fig=px.scatter(data_frame=df,x="salary",y="work_year",size="work_year",tre
fig.show()
```



In [28]:

```
features=["work_year","salary_in_usd","remote_ratio"]
x=df[features]
y=df["salary"]
```

In [29]:

```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2)
```

In [30]:

```
ml=LinearRegression()
ml.fit(xtrain,ytrain)
```

Out[30]:

```
▾ LinearRegression
LinearRegression()
```

In [31]:

```
print("Accuracy: ",ml.score(xtest,ytest))
```

```
Accuracy:  0.004417194198471863
```