Freelance Platform Project - Regression

imorting Libraries :

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from scipy.stats import skew
import xgboost as xgb
import warnings
warnings.filterwarnings('ignore')
```

importing dataset :

In [2]:
```
df = pd.read_csv("D:\Dk\Data sets\Freelance Platform Projects.csv")
```

Top 5 rows:

In [3]:
```
df.head()
```

Out[3]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Location	Freelancer Preferred From	Type	Date Posted	Description	Duration
0	Banner images for web desgin websites	Design	Entry ($)	Graphic Design	EUR	60	remote	ALL	fixed_price	4/29/2023 18:06	We are looking to improve the banner images on...	NaN
1	Make my picture a solid silhouette	Video, Photo & Image	Entry ($)	Image Editing	GBP	20	remote	ALL	fixed_price	4/29/2023 17:40	Hello \n\nI need a quick designer to make 4 pi...	NaN
2	Bookkeeper needed	Business	Entry ($)	Finance & Accounting	GBP	12	remote	ALL	fixed_price	4/29/2023 17:40	Hi - I need a bookkeeper to assist with bookke...	NaN
3	Accountant needed	Business	Entry ($)	Tax Consulting & Advising	GBP	14	remote	ALL	fixed_price	4/29/2023 17:32	Hi - I need an accountant to assist me with un...	NaN
4	Guest Post on High DA Website	Digital Marketing	Expert ($$$)	SEO	USD	10000	remote	ALL	fixed_price	4/29/2023 17:09	Hi, I am currently running a project where I w...	NaN

Showing Statistical Data :

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

Out[4]:

	Budget
count	12222.000000
mean	229.221486
std	1894.327521
min	0.000000
25%	30.000000
50%	80.000000
75%	150.000000
max	99999.000000

Number of rows and columns :

In [5]:
```
df.shape
```

Out[5]:
```
(12222, 17)
```

Datatype :

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

Out[6]:
```
Title                        object
Category Name                object
Experience                   object
Sub Category Name            object
Currency                     object
Budget                        int64
Location                     object
Freelancer Preferred From    object
Type                         object
Date Posted                  object
Description                  object
Duration                     object
Client Registration Date     object
Client City                  object
Client Country               object
Client Currency              object
Client Job Title             object
dtype: object
```

Showing Unique Values from specific columns:

In [7]:
```
df['Title'].nunique()
```

Out[7]:
```
11585
```

In [8]:
```
df['Title'].unique()
```

Out[8]:
```
array(['Banner images for web desgin websites',
       'Make my picture a solid silhouette ', 'Bookkeeper needed', ...,
       'Shopify - Filtering Work (Product Selection/Non-Selection)',
       'Create a Carbon, Water, Waste Calculating platform - Urjent',
       'COMPANY REGISTERS'], dtype=object)
```

In [9]:
```
df['Client Job Title'].nunique()
```

Out[9]:
```
1954
```

In [10]:
```
df['Location'].unique()
```

Out[10]:
```
array(['remote', 'onsite', 'remote_country'], dtype=object)
```

Null values :

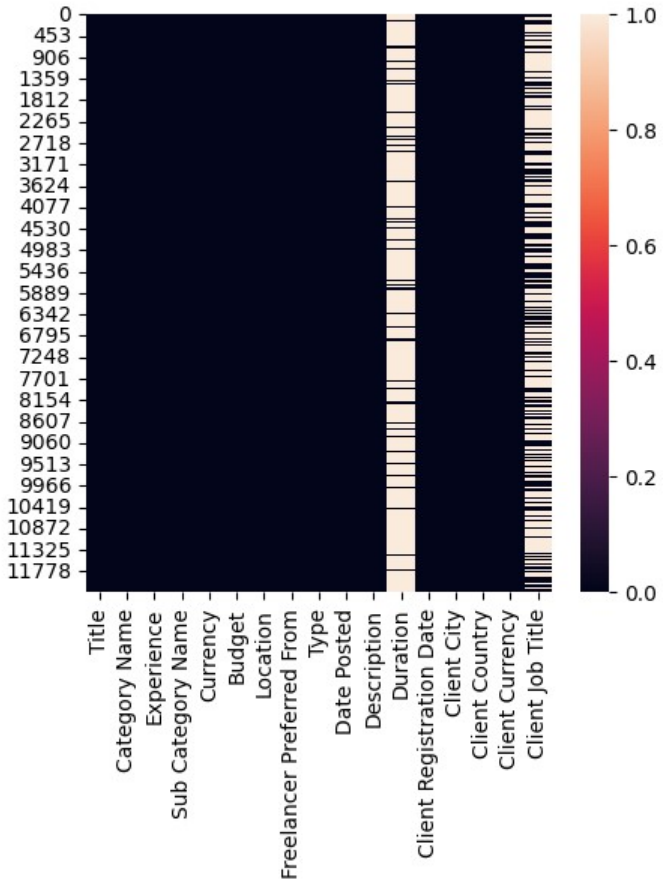
In [11]:
```
df.isna().sum()
```

Out[11]:
```
Title                            0
Category Name                    0
Experience                       0
Sub Category Name                0
Currency                         0
Budget                           0
Location                         0
Freelancer Preferred From        0
Type                             0
Date Posted                      0
Description                      0
Duration                     10620
Client Registration Date         0
Client City                      0
Client Country                   0
Client Currency                  0
Client Job Title              7634
dtype: int64
```

Visualizing Null Values by Heatmap:

In [12]:
```
plt.figure(figsize=(5,5))
sns.heatmap(df.isnull())
```

Out[12]: <Axes: >



Null values in Percentage:

In [13]:
```
null_val = df.isna().sum()
```

In [14]:
```
null_val
```

Out[14]:
```
Title                          0
Category Name                  0
Experience                     0
Sub Category Name              0
Currency                       0
Budget                         0
Location                       0
Freelancer Preferred From      0
Type                           0
Date Posted                    0
Description                    0
Duration                   10620
Client Registration Date       0
Client City                    0
Client Country                 0
Client Currency                0
Client Job Title            7634
dtype: int64
```

In [15]:
```
no_of_rows = df.shape[0]
no_of_rows
```

Out[15]: 12222

In [16]:
```
null_val_in_percentage = null_val / no_of_rows * 100
```

In [17]:
```
null_val_in_percentage
```

Out[17]:
```
Title                         0.000000
Category Name                 0.000000
Experience                    0.000000
Sub Category Name             0.000000
Currency                      0.000000
Budget                        0.000000
Location                      0.000000
Freelancer Preferred From     0.000000
Type                          0.000000
Date Posted                   0.000000
Description                   0.000000
Duration                     86.892489
Client Registration Date      0.000000
Client City                   0.000000
Client Country                0.000000
Client Currency               0.000000
Client Job Title             62.461136
dtype: float64
```

Deleting columns where null values are greater than 50

In [18]:
```
drop_col = null_val_in_percentage[null_val_in_percentage>50].keys()
drop_col
```

Out[18]:
```
Index(['Duration', 'Client Job Title'], dtype='object')
```

In [19]:
```
df = df.drop(columns=drop_col)
```

In [20]:
```
df.head()
```

Out[20]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Location	Freelancer Preferred From	Type	Date Posted	Description	Clier Registratio Dat
0	Banner images for web desgin websites	Design	Entry ($)	Graphic Design	EUR	60	remote	ALL	fixed_price	4/29/2023 18:06	We are looking to improve the banner images on...	11/3/201
1	Make my picture a solid silhouette	Video, Photo & Image	Entry ($)	Image Editing	GBP	20	remote	ALL	fixed_price	4/29/2023 17:40	Hello \n\nI need a quick designer to make 4 pi...	2/21/201
2	Bookkeeper needed	Business	Entry ($)	Finance & Accounting	GBP	12	remote	ALL	fixed_price	4/29/2023 17:40	Hi - I need a bookkeeper to assist with bookke...	4/9/202
											Hi - I need	

Now , showing there are no any null values:

In [21]:
```
df.isna().sum()
```

Out[21]:
```
Title                       0
Category Name               0
Experience                  0
Sub Category Name           0
Currency                    0
Budget                      0
Location                    0
Freelancer Preferred From   0
Type                        0
Date Posted                 0
Description                 0
Client Registration Date    0
Client City                 0
Client Country              0
Client Currency             0
dtype: int64
```

Number of rows and columns after deletion:

In [22]:
```
df.shape
```

Out[22]:
```
(12222, 15)
```

Number of Unique values in Specific column:

In [23]:
```
df['Experience'].nunique()
```

Out[23]:
```
3
```

Replacing Value of 'Experience' Column:

In [24]:
```
df['Experience'] = df['Experience'].replace({'Entry ($)':0, 'Intermediate ($$)': 1,'Expert ($$$)': 2})
```

In [25]:
```
df.head()
```

Out[25]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Location	Freelancer Preferred From	Type	Date Posted	Description	Clier Registratio Dat
0	Banner images for web desgin websites	Design	0	Graphic Design	EUR	60	remote	ALL	fixed_price	4/29/2023 18:06	We are looking to improve the banner images on...	11/3/201
1	Make my picture a solid silhouette	Video, Photo & Image	0	Image Editing	GBP	20	remote	ALL	fixed_price	4/29/2023 17:40	Hello \n\nI need a quick designer to make 4 pi...	2/21/201
2	Bookkeeper needed	Business	0	Finance & Accounting	GBP	12	remote	ALL	fixed_price	4/29/2023 17:40	Hi - I need a bookkeeper to assist with bookke...	4/9/202
3	Accountant needed	Business	0	Tax Consulting & Advising	GBP	14	remote	ALL	fixed_price	4/29/2023 17:32	Hi - I need an accountant to assist me with un...	4/9/202
4	Guest Post on High DA Website	Digital Marketing	2	SEO	USD	10000	remote	ALL	fixed_price	4/29/2023 17:09	Hi, I am currently running a project where I w...	7/1/201

Converting Client currency Values with 'USD' :

In [26]:
```
df['Currency'].unique()
```

Out[26]:
```
array(['EUR', 'GBP', 'USD'], dtype=object)
```

In [27]:
```
df['Currency'].unique()
```

Out[27]:
```
array(['EUR', 'GBP', 'USD'], dtype=object)
```

In [28]:
```
df['Client Currency'].unique()
```

Out[28]:
```
array(['EUR', 'GBP', 'USD'], dtype=object)
```

In [29]:
```
df['Client Currency'] = df['Client Currency'].replace({'EUR': 1.09565, 'GBP':1.2824, 'USD':1})
```

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

Out[30]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Location	Freelancer Preferred From	Type	Date Posted	Description	Clier Registratio Dat
0	Banner images for web desgin websites	Design	0	Graphic Design	EUR	60	remote	ALL	fixed_price	4/29/2023 18:06	We are looking to improve the banner images on...	11/3/201
1	Make my picture a solid silhouette	Video, Photo & Image	0	Image Editing	GBP	20	remote	ALL	fixed_price	4/29/2023 17:40	Hello \n\nI need a quick designer to make 4 pi...	2/21/201
2	Bookkeeper needed	Business	0	Finance & Accounting	GBP	12	remote	ALL	fixed_price	4/29/2023 17:40	Hi - I need a bookkeeper to assist with bookke...	4/9/202
3	Accountant needed	Business	0	Tax Consulting & Advising	GBP	14	remote	ALL	fixed_price	4/29/2023 17:32	Hi - I need an accountant to assist me with un...	4/9/202
4	Guest Post on High DA Website	Digital Marketing	2	SEO	USD	10000	remote	ALL	fixed_price	4/29/2023 17:09	Hi, I am currently running a project where I w...	7/1/201

In [31]:
```
df['Client Currency'].nunique()
```

Out[31]:
```
3
```

Featuring the Budget Value

In [32]:
```
def Budget_USD(row):
    if row['Currency'] == 'EUR':
        return row['Budget'] * 1.0956
    elif row['Currency'] == 'GBP':
        return row['Budget'] * 1.28
    else:
        return row['Budget']
```

In [33]:
```
df['Budget']=df.apply(Budget_USD,axis = 1)
```

In [34]:
```
df.head()
```

Out[34]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Location	Freelancer Preferred From	Type	Date Posted	Description	Cl Registra I
0	Banner images for web desgin websites	Design	0	Graphic Design	EUR	65.736	remote	ALL	fixed_price	4/29/2023 18:06	We are looking to improve the banner images on...	11/3/2
1	Make my picture a solid silhouette	Video, Photo & Image	0	Image Editing	GBP	25.600	remote	ALL	fixed_price	4/29/2023 17:40	Hello \n\nI need a quick designer to make 4 pi...	2/21/2
2	Bookkeeper needed	Business	0	Finance & Accounting	GBP	15.360	remote	ALL	fixed_price	4/29/2023 17:40	Hi - I need a bookkeeper to assist with bookke...	4/9/2
3	Accountant needed	Business	0	Tax Consulting & Advising	GBP	17.920	remote	ALL	fixed_price	4/29/2023 17:32	Hi - I need an accountant to assist me with un...	4/9/2
4	Guest Post on High DA Website	Digital Marketing	2	SEO	USD	10000.000	remote	ALL	fixed_price	4/29/2023 17:09	Hi, I am currently running a project where I w...	7/1/2

In [35]:
```
df['Budget'].dtype
```

Out[35]:
```
dtype('float64')
```

Converting Currency into USD:

In [36]:
```
df['Currency'] = df['Currency'].replace({'EUR':'USD', 'GBP':'USD'})
```

In [37]:
```
df.head()
```

Out[37]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Location	Freelancer Preferred From	Type	Date Posted	Description	Cl Registra I
0	Banner images for web desgin websites	Design	0	Graphic Design	USD	65.736	remote	ALL	fixed_price	4/29/2023 18:06	We are looking to improve the banner images on...	11/3/2
1	Make my picture a solid silhouette	Video, Photo & Image	0	Image Editing	USD	25.600	remote	ALL	fixed_price	4/29/2023 17:40	Hello \n\nI need a quick designer to make 4 pi...	2/21/2
2	Bookkeeper needed	Business	0	Finance & Accounting	USD	15.360	remote	ALL	fixed_price	4/29/2023 17:40	Hi - I need a bookkeeper to assist with bookke...	4/9/2
3	Accountant needed	Business	0	Tax Consulting & Advising	USD	17.920	remote	ALL	fixed_price	4/29/2023 17:32	Hi - I need an accountant to assist me with un...	4/9/2
4	Guest Post on High DA Website	Digital Marketing	2	SEO	USD	10000.000	remote	ALL	fixed_price	4/29/2023 17:09	Hi, I am currently running a project where I w...	7/1/2

In [38]:
```
df['Currency'] = df['Currency'].replace({'USD': 1})
```

In [39]:
```
df
```

Out[39]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Location	Freelancer Preferred From	Type	Date Posted	
0	Banner images for web desgin websites	Design	0	Graphic Design	1	65.736	remote	ALL	fixed_price	4/29/2023 18:06	im
1	Make my picture a solid silhouette	Video, Photo & Image	0	Image Editing	1	25.600	remote	ALL	fixed_price	4/29/2023 17:40	
2	Bookkeeper needed	Business	0	Finance & Accounting	1	15.360	remote	ALL	fixed_price	4/29/2023 17:40	
3	Accountant needed	Business	0	Tax Consulting & Advising	1	17.920	remote	ALL	fixed_price	4/29/2023 17:32	as
4	Guest Post on High DA Website	Digital Marketing	2	SEO	1	10000.000	remote	ALL	fixed_price	4/29/2023 17:09	
...	...	...	...	...	...	...	...	...	...	...	
12217	Published Travel Writer required for content c...	Writing & Translation	0	Content Writing	1	64.000	remote	ALL	fixed_price	1/18/2023 19:23	
12218	Shopify - Filtering Work (Product Selection/No...	Design	1	Web Design	1	83.200	remote_country	GB	fixed_price	1/18/2023 19:18	ww
12219	Simple SQL Query	Technology & Programming	0	Data Science & Analysis	1	64.000	remote	ALL	fixed_price	1/18/2023 19:18	
12220	Create a Carbon, Water, Waste Calculating plat...	Design	2	Web Design	1	39.000	remote	ALL	hourly	1/18/2023 19:18	de
12221	COMPANY REGISTERS	Business	2	Administration Assistance	1	96.000	remote	ALL	fixed_price	1/18/2023 19:18	ac

12222 rows × 15 columns

By One hot Encoding, Converting categorical Data into Numberical data :

In [40]:
```
df['Location'].unique()
```

Out[40]:
```
array(['remote', 'onsite', 'remote_country'], dtype=object)
```

In [41]:
```
df['Type'].nunique()
```

Out[41]:
```
2
```

In [42]:
```
column_to_encode=['Location','Type']
```

In [43]:
```
df = pd.get_dummies(df,columns=column_to_encode )
```

In [44]:
```
df.head()
```

Out[44]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Date Posted	Description	Client Registration Date	Client City	Clien Countr
0	Banner images for web desgin websites	Design	0	Graphic Design	1	65.736	ALL	4/29/2023 18:06	We are looking to improve the banner images on...	11/3/2010	Dublin	Irelan
1	Make my picture a solid silhouette	Video, Photo & Image	0	Image Editing	1	25.600	ALL	4/29/2023 17:40	Hello \n\nI need a quick designer to make 4 pi...	2/21/2017	London	Unite Kingdor
2	Bookkeeper needed	Business	0	Finance & Accounting	1	15.360	ALL	4/29/2023 17:40	Hi - I need a bookkeeper to assist with bookke...	4/9/2023	London	Unite Kingdor
3	Accountant needed	Business	0	Tax Consulting & Advising	1	17.920	ALL	4/29/2023 17:32	Hi - I need an accountant to assist me with un...	4/9/2023	London	Unite Kingdor
4	Guest Post on High DA Website	Digital Marketing	2	SEO	1	10000.000	ALL	4/29/2023 17:09	Hi, I am currently running a project where I w...	7/1/2016	Mumbai	Indi

In [45]:
```
df[['Location_onsite',
    'Location_remote',
    'Location_remote_country',
    'Type_fixed_price','Type_hourly']]=df[['Location_onsite',
                                           'Location_remote',
                                           'Location_remote_country',
                                           'Type_fixed_price','Type_hourly']].astype('int32')
```

In [46]:
```
df.head()
```

Out[46]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Date Posted	Description	Client Registration Date	Client City	Clien Countr
0	Banner images for web desgin websites	Design	0	Graphic Design	1	65.736	ALL	4/29/2023 18:06	We are looking to improve the banner images on...	11/3/2010	Dublin	Irelan
1	Make my picture a solid silhouette	Video, Photo & Image	0	Image Editing	1	25.600	ALL	4/29/2023 17:40	Hello \n\nI need a quick designer to make 4 pi...	2/21/2017	London	Unite Kingdor
2	Bookkeeper needed	Business	0	Finance & Accounting	1	15.360	ALL	4/29/2023 17:40	Hi - I need a bookkeeper to assist with bookke...	4/9/2023	London	Unite Kingdor
3	Accountant needed	Business	0	Tax Consulting & Advising	1	17.920	ALL	4/29/2023 17:32	Hi - I need an accountant to assist me with un...	4/9/2023	London	Unite Kingdor
4	Guest Post on High DA Website	Digital Marketing	2	SEO	1	10000.000	ALL	4/29/2023 17:09	Hi, I am currently running a project where I w...	7/1/2016	Mumbai	Indi

In [47]:
```
df.shape
```

Out[47]:
```
(12222, 18)
```

In [48]:
```
df.dtypes
```

Out[48]:
```
Title                        object
Category Name                object
Experience                    int64
Sub Category Name            object
Currency                      int64
Budget                      float64
Freelancer Preferred From    object
Date Posted                  object
Description                  object
Client Registration Date     object
Client City                  object
Client Country               object
Client Currency             float64
Location_onsite               int32
Location_remote               int32
Location_remote_country       int32
Type_fixed_price              int32
Type_hourly                   int32
dtype: object
```

By using Label-Encoding, converting categorical Data into numerical data:

In [49]:
```
col_to_encode=['Title','Category Name','Sub Category Name','Freelancer Preferred From','Description',
               'Client City', 'Client Country']
```

In [50]:
```
le = LabelEncoder()
for column in col_to_encode:
    df[column] = le.fit_transform(df[column])
```

In [51]:
```
df
```

Out[51]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Date Posted	Description	Client Registration Date	Client City	Client Country	Cu
0	969	1	0	42	1	65.736	1	4/29/2023 18:06	10434	11/3/2010	489	61	
1	6377	7	0	45	1	25.600	1	4/29/2023 17:40	1247	2/21/2017	940	129	
2	1108	0	0	37	1	15.360	1	4/29/2023 17:40	2179	4/9/2023	940	129	
3	467	0	0	90	1	17.920	1	4/29/2023 17:32	2181	4/9/2023	940	129	
4	3859	2	2	76	1	10000.000	1	4/29/2023 17:09	3024	7/1/2016	1079	58	
...	...	...	...	...	...	...	...	...	...	...	...	...	
12217	7958	8	0	21	1	64.000	1	1/18/2023 19:23	3660	6/6/2011	47	88	
12218	8803	1	1	101	1	83.200	16	1/18/2023 19:18	8718	3/23/2022	554	129	
12219	8927	6	0	25	1	64.000	1	1/18/2023 19:18	6547	3/14/2022	940	129	
12220	2057	1	2	101	1	39.000	1	1/18/2023 19:18	4154	7/21/2013	1135	58	
12221	1407	0	2	1	1	96.000	1	1/18/2023 19:18	3158	9/21/2020	632	129	

12222 rows × 18 columns

In [52]:
```
df.dtypes
```

Out[52]:
```
Title                          int32
Category Name                  int32
Experience                     int64
Sub Category Name              int32
Currency                       int64
Budget                       float64
Freelancer Preferred From      int32
Date Posted                   object
Description                    int32
Client Registration Date      object
Client City                    int32
Client Country                 int32
Client Currency              float64
Location_onsite                int32
Location_remote                int32
Location_remote_country        int32
Type_fixed_price               int32
Type_hourly                    int32
dtype: object
```

Splitting the 'Date Posted' Column into Date, Month, Year, Time

In [53]:
```
df['Date Posted']
```

Out[53]:
```
0        4/29/2023 18:06
1        4/29/2023 17:40
2        4/29/2023 17:40
3        4/29/2023 17:32
4        4/29/2023 17:09
               ...
12217    1/18/2023 19:23
12218    1/18/2023 19:18
12219    1/18/2023 19:18
12220    1/18/2023 19:18
12221    1/18/2023 19:18
Name: Date Posted, Length: 12222, dtype: object
```

In [54]:
```
df[['Posted_date','Posted_time']]=df['Date Posted'].str.split(' ',expand=True)
```

In [55]:
```
df
```

Out[55]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Date Posted	Description	Client Registration Date	Client City	Client Country	Cu
0	969	1	0	42	1	65.736	1	4/29/2023 18:06	10434	11/3/2010	489	61	
1	6377	7	0	45	1	25.600	1	4/29/2023 17:40	1247	2/21/2017	940	129	
2	1108	0	0	37	1	15.360	1	4/29/2023 17:40	2179	4/9/2023	940	129	
3	467	0	0	90	1	17.920	1	4/29/2023 17:32	2181	4/9/2023	940	129	
4	3859	2	2	76	1	10000.000	1	4/29/2023 17:09	3024	7/1/2016	1079	58	
...	...	...	...	...	...	...	...	...	...	...	...	...	
12217	7958	8	0	21	1	64.000	1	1/18/2023 19:23	3660	6/6/2011	47	88	
12218	8803	1	1	101	1	83.200	16	1/18/2023 19:18	8718	3/23/2022	554	129	
12219	8927	6	0	25	1	64.000	1	1/18/2023 19:18	6547	3/14/2022	940	129	
12220	2057	1	2	101	1	39.000	1	1/18/2023 19:18	4154	7/21/2013	1135	58	
12221	1407	0	2	1	1	96.000	1	1/18/2023 19:18	3158	9/21/2020	632	129	

12222 rows × 20 columns

In [56]:
```
df[['Posted_month','Posted_Date','Posted_year']]=df['Posted_date'].str.split('/',expand=True)
```

In [57]:
```
df[['Posted_month','Posted_Date','Posted_year']]
```

Out[57]:

	Posted_month	Posted_Date	Posted_year
0	4	29	2023
1	4	29	2023
2	4	29	2023
3	4	29	2023
4	4	29	2023
...	...	...	...
12217	1	18	2023
12218	1	18	2023
12219	1	18	2023
12220	1	18	2023
12221	1	18	2023

12222 rows × 3 columns

In [58]:
```
df.head()
```

Out[58]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Date Posted	Description	Client Registration Date	...	Location_onsite	Lo
0	969	1	0	42	1	65.736	1	4/29/2023 18:06	10434	11/3/2010	...	0	
1	6377	7	0	45	1	25.600	1	4/29/2023 17:40	1247	2/21/2017	...	0	
2	1108	0	0	37	1	15.360	1	4/29/2023 17:40	2179	4/9/2023	...	0	
3	467	0	0	90	1	17.920	1	4/29/2023 17:32	2181	4/9/2023	...	0	
4	3859	2	2	76	1	10000.000	1	4/29/2023 17:09	3024	7/1/2016	...	0	

5 rows × 23 columns

In [59]:
```
df = df.drop(columns=['Date Posted'])
```

In [60]:
```
df = df.drop(columns=['Posted_date'])
```

Showing the rows and columns after splitting

In [61]:
```
df.shape
```

Out[61]:
```
(12222, 21)
```

In [62]:
```
df
```

Out[62]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client Registration Date	Client City	...	Client Currency	Location
0	969	1	0	42	1	65.736	1	10434	11/3/2010	489	...	1.09565	
1	6377	7	0	45	1	25.600	1	1247	2/21/2017	940	...	1.28240	
2	1108	0	0	37	1	15.360	1	2179	4/9/2023	940	...	1.28240	
3	467	0	0	90	1	17.920	1	2181	4/9/2023	940	...	1.28240	
4	3859	2	2	76	1	10000.000	1	3024	7/1/2016	1079	...	1.00000	
...	...	...	...	...	...	...	...	...	...	...	...	...	
12217	7958	8	0	21	1	64.000	1	3660	6/6/2011	47	...	1.28240	
12218	8803	1	1	101	1	83.200	16	8718	3/23/2022	554	...	1.28240	
12219	8927	6	0	25	1	64.000	1	6547	3/14/2022	940	...	1.28240	
12220	2057	1	2	101	1	39.000	1	4154	7/21/2013	1135	...	1.00000	
12221	1407	0	2	1	1	96.000	1	3158	9/21/2020	632	...	1.28240	

12222 rows × 21 columns

In [63]:
```
df.dtypes
```

Out[63]:
```
Title                          int32
Category Name                  int32
Experience                     int64
Sub Category Name              int32
Currency                       int64
Budget                       float64
Freelancer Preferred From      int32
Description                    int32
Client Registration Date      object
Client City                    int32
Client Country                 int32
Client Currency              float64
Location_onsite                int32
Location_remote                int32
Location_remote_country        int32
Type_fixed_price               int32
Type_hourly                    int32
Posted_time                   object
Posted_month                  object
Posted_Date                   object
Posted_year                   object
dtype: object
```

Convtering Datatype into integer:

In [64]:
```
df[['Posted_month','Posted_Date','Posted_year']]=df[['Posted_month','Posted_Date',
                                                     'Posted_year']].astype('int32')
```

In [65]:
```
df.dtypes
```

Out[65]:
```
Title                          int32
Category Name                  int32
Experience                     int64
Sub Category Name              int32
Currency                       int64
Budget                       float64
Freelancer Preferred From      int32
Description                    int32
Client Registration Date      object
Client City                    int32
Client Country                 int32
Client Currency              float64
Location_onsite                int32
Location_remote                int32
Location_remote_country        int32
Type_fixed_price               int32
Type_hourly                    int32
Posted_time                   object
Posted_month                   int32
Posted_Date                    int32
Posted_year                    int32
dtype: object
```

In [66]:
```
df[['Client_Month','Client_Date','Client_Year']]=df['Client Registration Date'].str.split('/',
                                                                                          expand=True)
```

In [67]:
```
df[['Client_Month','Client_Date','Client_Year']]=df[['Client_Month',
                                                     'Client_Date','Client_Year']].astype('int32')
```

In [68]:
```
df.dtypes
```

Out[68]:
```
Title                          int32
Category Name                  int32
Experience                     int64
Sub Category Name              int32
Currency                       int64
Budget                       float64
Freelancer Preferred From      int32
Description                    int32
Client Registration Date      object
Client City                    int32
Client Country                 int32
Client Currency              float64
Location_onsite                int32
Location_remote                int32
Location_remote_country        int32
Type_fixed_price               int32
Type_hourly                    int32
Posted_time                   object
Posted_month                   int32
Posted_Date                    int32
Posted_year                    int32
Client_Month                   int32
Client_Date                    int32
Client_Year                    int32
dtype: object
```

In [69]:
```
df
```

Out[69]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client Registration Date	Client City	...	Location_remote_co
0	969	1	0	42	1	65.736	1	10434	11/3/2010	489	...	
1	6377	7	0	45	1	25.600	1	1247	2/21/2017	940	...	
2	1108	0	0	37	1	15.360	1	2179	4/9/2023	940	...	
3	467	0	0	90	1	17.920	1	2181	4/9/2023	940	...	
4	3859	2	2	76	1	10000.000	1	3024	7/1/2016	1079	...	
...	...	...	...	...	...	...	...	...	...	...	...	
12217	7958	8	0	21	1	64.000	1	3660	6/6/2011	47	...	
12218	8803	1	1	101	1	83.200	16	8718	3/23/2022	554	...	
12219	8927	6	0	25	1	64.000	1	6547	3/14/2022	940	...	
12220	2057	1	2	101	1	39.000	1	4154	7/21/2013	1135	...	
12221	1407	0	2	1	1	96.000	1	3158	9/21/2020	632	...	

12222 rows × 24 columns

In [70]:
```
df = df.drop(columns = 'Client Registration Date')
```

In [71]:
```
df.shape
```

Out[71]:
```
(12222, 23)
```

In [72]:
```
df.dtypes
```

Out[72]:
```
Title                          int32
Category Name                  int32
Experience                     int64
Sub Category Name              int32
Currency                       int64
Budget                       float64
Freelancer Preferred From      int32
Description                    int32
Client City                    int32
Client Country                 int32
Client Currency              float64
Location_onsite                int32
Location_remote                int32
Location_remote_country        int32
Type_fixed_price               int32
Type_hourly                    int32
Posted_time                   object
Posted_month                   int32
Posted_Date                    int32
Posted_year                    int32
Client_Month                   int32
Client_Date                    int32
Client_Year                    int32
dtype: object
```

In [73]:
```
df[['Posted_hour','Posted_Minutes']]=df['Posted_time'].str.split(':',expand=True)
```

In [74]:
```
df[['Posted_hour','Posted_Minutes']]
```

Out[74]:

	Posted_hour	Posted_Minutes
0	18	06
1	17	40
2	17	40
3	17	32
4	17	09
...	...	...
12217	19	23
12218	19	18
12219	19	18
12220	19	18
12221	19	18

12222 rows × 2 columns

In [75]:
```
df
```

Out[75]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client City	Client Country	...	Type_hourly	Posted_ti
0	969	1	0	42	1	65.736	1	10434	489	61	...	0	18
1	6377	7	0	45	1	25.600	1	1247	940	129	...	0	17
2	1108	0	0	37	1	15.360	1	2179	940	129	...	0	17
3	467	0	0	90	1	17.920	1	2181	940	129	...	0	17
4	3859	2	2	76	1	10000.000	1	3024	1079	58	...	0	17
...	...	...	...	...	...	...	...	...	...	...	...	...	
12217	7958	8	0	21	1	64.000	1	3660	47	88	...	0	19
12218	8803	1	1	101	1	83.200	16	8718	554	129	...	0	19
12219	8927	6	0	25	1	64.000	1	6547	940	129	...	0	19
12220	2057	1	2	101	1	39.000	1	4154	1135	58	...	1	19
12221	1407	0	2	1	1	96.000	1	3158	632	129	...	0	19

12222 rows × 25 columns

In [76]:
```
df = df.drop(columns='Posted_time')
```

In [77]:
```
df.shape
```

Out[77]:
```
(12222, 24)
```

In [78]:
```
df.head()
```

Out[78]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client City	Client Country	...	Type_fixed_price	Type_hou
0	969	1	0	42	1	65.736	1	10434	489	61	...	1	
1	6377	7	0	45	1	25.600	1	1247	940	129	...	1	
2	1108	0	0	37	1	15.360	1	2179	940	129	...	1	
3	467	0	0	90	1	17.920	1	2181	940	129	...	1	
4	3859	2	2	76	1	10000.000	1	3024	1079	58	...	1	

5 rows × 24 columns

In [79]:
```
df.describe()
```

Out[79]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client City	
count	12222.000000	12222.000000	12222.000000	12222.000000	12222.0	12222.000000	12222.000000	12222.000000	12222.000000	12
mean	5806.995254	3.712813	1.007282	56.839552	1.0	272.077271	2.186876	5958.065538	871.258714	
std	3340.786272	2.820344	0.936768	32.627551	0.0	2345.979319	5.015981	3446.774106	456.183611	
min	0.000000	0.000000	0.000000	0.000000	1.0	0.000000	0.000000	0.000000	0.000000	
25%	2918.250000	1.000000	0.000000	30.000000	1.0	38.400000	1.000000	2965.250000	487.250000	
50%	5842.500000	3.000000	1.000000	52.000000	1.0	96.000000	1.000000	5960.500000	940.000000	
75%	8707.750000	6.000000	2.000000	92.000000	1.0	192.000000	1.000000	8938.750000	1139.000000	
max	11584.000000	8.000000	2.000000	106.000000	1.0	127998.720000	41.000000	11924.000000	1807.000000	

8 rows × 22 columns

In [80]:
```
df.dtypes
```

Out[80]:
```
Title                          int32
Category Name                  int32
Experience                     int64
Sub Category Name              int32
Currency                       int64
Budget                       float64
Freelancer Preferred From      int32
Description                    int32
Client City                    int32
Client Country                 int32
Client Currency              float64
Location_onsite                int32
Location_remote                int32
Location_remote_country        int32
Type_fixed_price               int32
Type_hourly                    int32
Posted_month                   int32
Posted_Date                    int32
Posted_year                    int32
Client_Month                   int32
Client_Date                    int32
Client_Year                    int32
Posted_hour                   object
Posted_Minutes                object
dtype: object
```

In [81]:
```
df[['Posted_hour','Posted_Minutes']] = df[['Posted_hour','Posted_Minutes']].astype('int32')
```

In [82]:
```
df.shape
```

Out[82]:
```
(12222, 24)
```

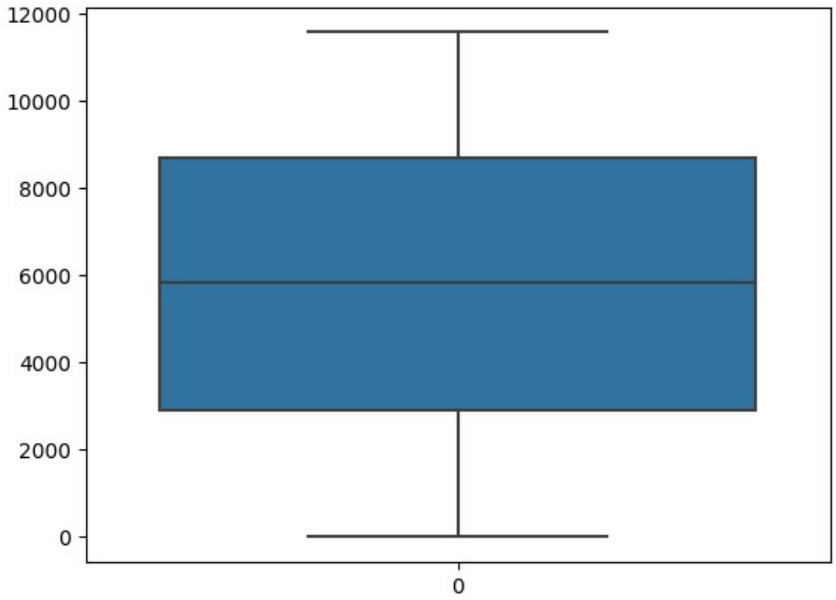
In [83]:
```
df.dtypes
```

Out[83]:
```
Title                          int32
Category Name                  int32
Experience                     int64
Sub Category Name              int32
Currency                       int64
Budget                       float64
Freelancer Preferred From      int32
Description                    int32
Client City                    int32
Client Country                 int32
Client Currency              float64
Location_onsite                int32
Location_remote                int32
Location_remote_country        int32
Type_fixed_price               int32
Type_hourly                    int32
Posted_month                   int32
Posted_Date                    int32
Posted_year                    int32
Client_Month                   int32
Client_Date                    int32
Client_Year                    int32
Posted_hour                    int32
Posted_Minutes                 int32
dtype: object
```

Checking Outliers by creating Boxplot :

In [84]:
```
sns.boxplot(df['Title'])
```

Out[84]: <Axes: >



In [85]:
```
col_num=df[['Title', 'Category Name', 'Experience']]
col_num
```

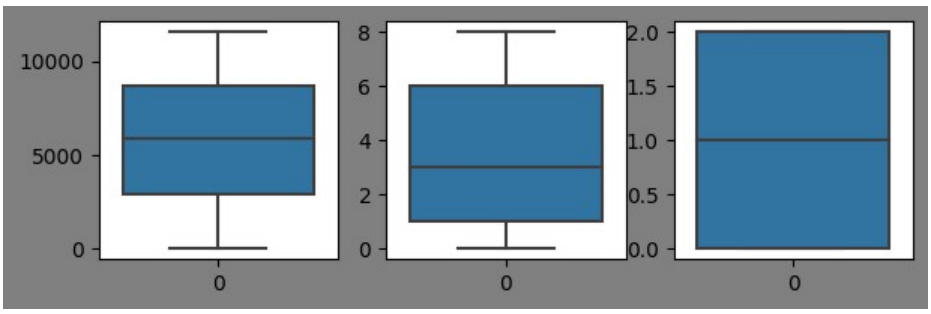
Out[85]:

	Title	Category Name	Experience
0	969	1	0
1	6377	7	0
2	1108	0	0
3	467	0	0
4	3859	2	2
...	...	...	...
12217	7958	8	0
12218	8803	1	1
12219	8927	6	0
12220	2057	1	2
12221	1407	0	2

12222 rows × 3 columns

In [86]:
```
plt.figure(figsize=(7,7),facecolor='grey', edgecolor='red')
for i,col in enumerate(col_num):
    plt.subplot(3,3,i+1)
    print(sns.boxplot(col_num[col]))
plt.figure()
plt.show()
```

```
Axes(0.125,0.653529;0.227941x0.226471)
Axes(0.398529,0.653529;0.227941x0.226471)
Axes(0.672059,0.653529;0.227941x0.226471)
```




```
<Figure size 640x480 with 0 Axes>
```

In [87]:
```
df.dtypes
```

Out[87]:
```
Title                          int32
Category Name                  int32
Experience                     int64
Sub Category Name              int32
Currency                       int64
Budget                       float64
Freelancer Preferred From      int32
Description                    int32
Client City                    int32
Client Country                 int32
Client Currency              float64
Location_onsite                int32
Location_remote                int32
Location_remote_country        int32
Type_fixed_price               int32
Type_hourly                    int32
Posted_month                   int32
Posted_Date                    int32
Posted_year                    int32
Client_Month                   int32
Client_Date                    int32
Client_Year                    int32
Posted_hour                    int32
Posted_Minutes                 int32
dtype: object
```

In [88]:
```
col_num=df[['Title','Category Name','Experience','Sub Category Name',
            'Currency','Freelancer Preferred From', 'Description', 'Client City','Client Country',
            'Client Currency', 'Location_onsite','Location_remote', 'Location_remote_country',
            'Type_fixed_price','Type_hourly', 'Posted_year', 'Posted_month', 'Posted_Date','Budget',
            'Client_Year', 'Client_Month', 'Client_Date','Posted_hour','Posted_Minutes']]
col_num
```

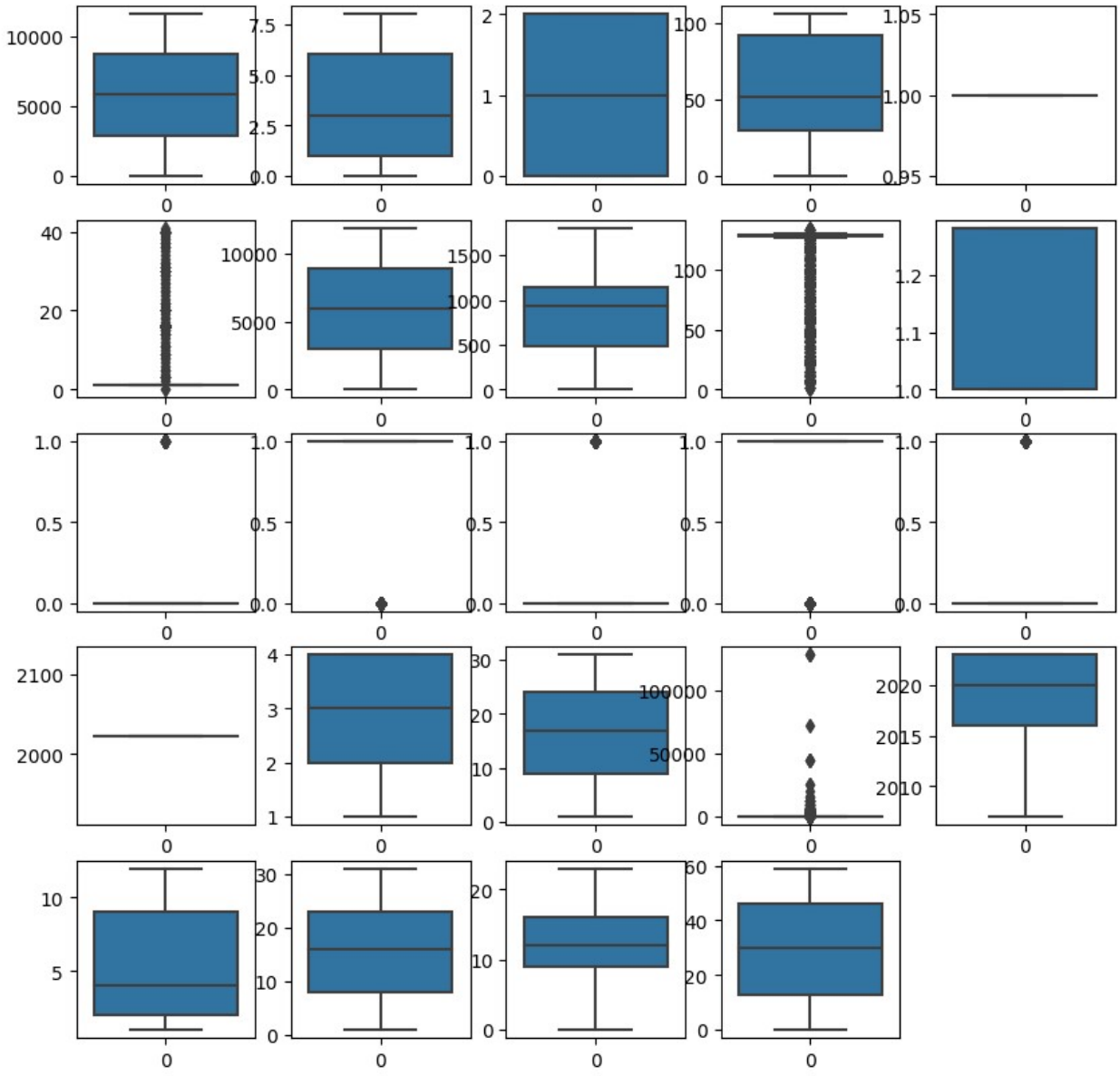
Out[88]:

	Title	Category Name	Experience	Sub Category Name	Currency	Freelancer Preferred From	Description	Client City	Client Country	Client Currency	...	Type_hourly	Posted_ye
0	969	1	0	42	1	1	10434	489	61	1.09565	...	0	20
1	6377	7	0	45	1	1	1247	940	129	1.28240	...	0	20
2	1108	0	0	37	1	1	2179	940	129	1.28240	...	0	20
3	467	0	0	90	1	1	2181	940	129	1.28240	...	0	20
4	3859	2	2	76	1	1	3024	1079	58	1.00000	...	0	20
...	...	...	...	...	...	...	...	...	...	...	...	...	
12217	7958	8	0	21	1	1	3660	47	88	1.28240	...	0	20
12218	8803	1	1	101	1	16	8718	554	129	1.28240	...	0	20
12219	8927	6	0	25	1	1	6547	940	129	1.28240	...	0	20
12220	2057	1	2	101	1	1	4154	1135	58	1.00000	...	1	20
12221	1407	0	2	1	1	1	3158	632	129	1.28240	...	0	20

12222 rows × 24 columns

In [89]:
```
plt.figure(figsize=(10,10))
for i,col in enumerate(col_num):
    plt.subplot(5,5,i+1)
    print(sns.boxplot(col_num[col]))
plt.figure()
plt.show()
```

```
Axes(0.125,0.747241;0.133621x0.132759)
Axes(0.285345,0.747241;0.133621x0.132759)
Axes(0.44569,0.747241;0.133621x0.132759)
Axes(0.606034,0.747241;0.133621x0.132759)
Axes(0.766379,0.747241;0.133621x0.132759)
Axes(0.125,0.587931;0.133621x0.132759)
Axes(0.285345,0.587931;0.133621x0.132759)
Axes(0.44569,0.587931;0.133621x0.132759)
Axes(0.606034,0.587931;0.133621x0.132759)
Axes(0.766379,0.587931;0.133621x0.132759)
Axes(0.125,0.428621;0.133621x0.132759)
Axes(0.285345,0.428621;0.133621x0.132759)
Axes(0.44569,0.428621;0.133621x0.132759)
Axes(0.606034,0.428621;0.133621x0.132759)
Axes(0.766379,0.428621;0.133621x0.132759)
Axes(0.125,0.26931;0.133621x0.132759)
Axes(0.285345,0.26931;0.133621x0.132759)
Axes(0.44569,0.26931;0.133621x0.132759)
Axes(0.606034,0.26931;0.133621x0.132759)
Axes(0.766379,0.26931;0.133621x0.132759)
Axes(0.125,0.11;0.133621x0.132759)
Axes(0.285345,0.11;0.133621x0.132759)
Axes(0.44569,0.11;0.133621x0.132759)
Axes(0.606034,0.11;0.133621x0.132759)
```



```
<Figure size 640x480 with 0 Axes>
```

Checking Ouliers Using Skewness:

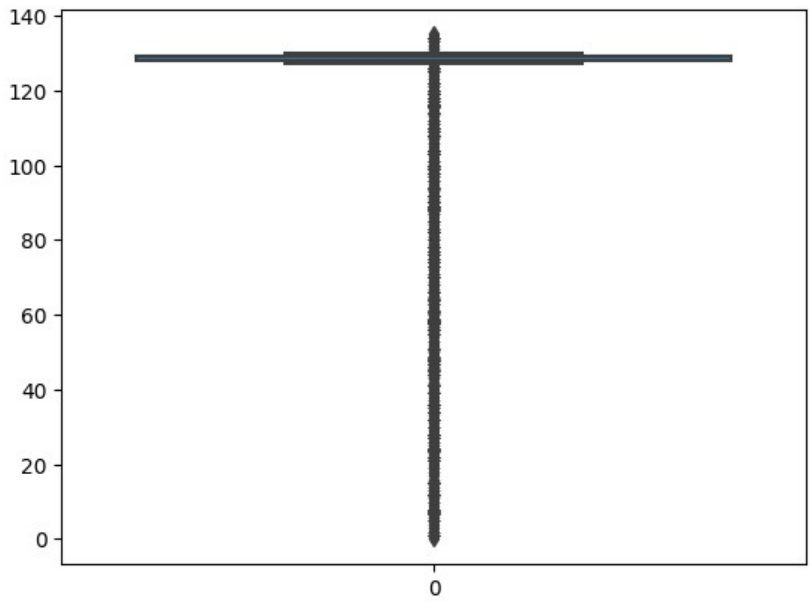
In [90]:
```
for col in col_num:
    print(col)
    print(skew(col_num[col]))

    plt.figure()
    sns.distplot(col_num[col])
```

```
Title
-0.009026401415337032
Category Name
0.13667421107983443
Experience
-0.014462463349103371
Sub Category Name
-0.018928584486655
Currency
nan
Freelancer Preferred From
5.042642835420588
Description
-0.0001446030632413126
Client City
0.037911936508722724
Client Country
-1.9572602850079996
Client Currency
```

Checking outliers in Client Country

In [91]:
```
sns.boxplot(df['Client Country'])
```

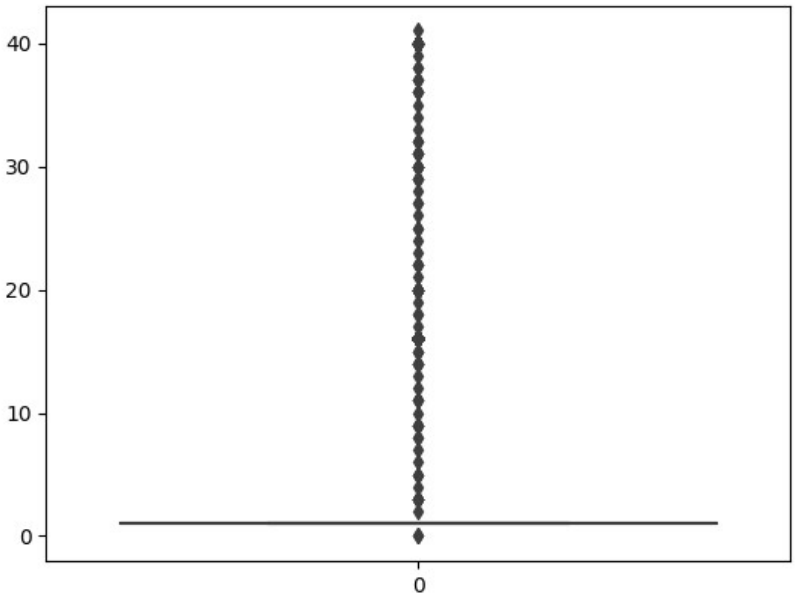
Out[91]: <Axes: >



Checking Outliers in Freelancer preferred Form

In [92]:
```
sns.boxplot(df['Freelancer Preferred From'])
```

Out[92]: <Axes: >



In [93]:
```
df['Freelancer Preferred From'].value_counts()
```

Out[93]:
```
Freelancer Preferred From
1     11431
16      578
40       90
20       27
3        10
30        9
31        6
14        6
9         6
36        4
11        4
29        4
22        4
5         3
32        3
37        3
15        3
27        2
38        2
25        2
8         2
0         2
18        2
6         1
41        1
13        1
10        1
19        1
35        1
7         1
17        1
2         1
39        1
23        1
34        1
24        1
33        1
4         1
12        1
21        1
26        1
28        1
Name: count, dtype: int64
```

Removing Outliers Using IQR method

In [94]:
```
Freelancer_Preferred_From=sorted(df['Freelancer Preferred From'])
```

In [95]:
```
Freelancer_Preferred_From
```

Out[95]:
```
[0,
 0,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
```

In [96]:
```
Q1,Q3 = np.percentile(Freelancer_Preferred_From,[25,75])
```

In [97]:
```
Q1,Q3
```

Out[97]:
```
(1.0, 1.0)
```

In [98]:
```
IQR = Q3-Q1
```

In [99]:
```
IQR
```

Out[99]:
```
0.0
```

In [100]:
```
Lowerfence = Q1-(1.5*IQR)
Upperfence = Q3+(1.5*IQR)
```

In [101]:
```
print(Lowerfence)
print(Upperfence)
```

```
1.0
1.0
```

In [102]:
```
df = df[df['Freelancer Preferred From']<=Upperfence]
df
```

Out[102]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client City	Client Country	...	Type_fixed_price	Type
0	969	1	0	42	1	65.736	1	10434	489	61	...	1	
1	6377	7	0	45	1	25.600	1	1247	940	129	...	1	
2	1108	0	0	37	1	15.360	1	2179	940	129	...	1	
3	467	0	0	90	1	17.920	1	2181	940	129	...	1	
4	3859	2	2	76	1	10000.000	1	3024	1079	58	...	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
12216	8927	6	0	26	1	38.400	1	6546	940	129	...	1	
12217	7958	8	0	21	1	64.000	1	3660	47	88	...	1	
12219	8927	6	0	25	1	64.000	1	6547	940	129	...	1	
12220	2057	1	2	101	1	39.000	1	4154	1135	58	...	0	
12221	1407	0	2	1	1	96.000	1	3158	632	129	...	1	

11433 rows × 24 columns

In [103]:
```
df = df[df['Freelancer Preferred From']>=Upperfence]
df
```

Out[103]:

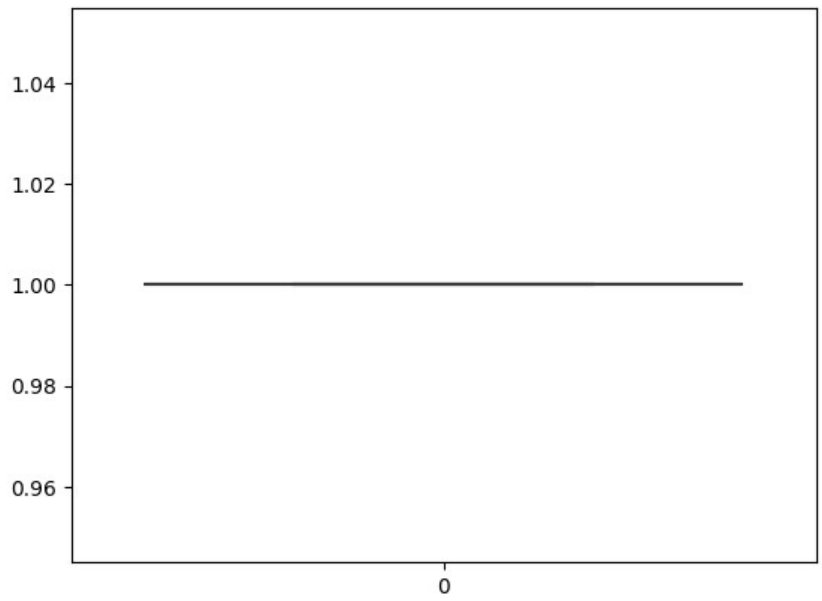
	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client City	Client Country	...	Type_fixed_price	Type
0	969	1	0	42	1	65.736	1	10434	489	61	...	1	
1	6377	7	0	45	1	25.600	1	1247	940	129	...	1	
2	1108	0	0	37	1	15.360	1	2179	940	129	...	1	
3	467	0	0	90	1	17.920	1	2181	940	129	...	1	
4	3859	2	2	76	1	10000.000	1	3024	1079	58	...	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
12216	8927	6	0	26	1	38.400	1	6546	940	129	...	1	
12217	7958	8	0	21	1	64.000	1	3660	47	88	...	1	
12219	8927	6	0	25	1	64.000	1	6547	940	129	...	1	
12220	2057	1	2	101	1	39.000	1	4154	1135	58	...	0	
12221	1407	0	2	1	1	96.000	1	3158	632	129	...	1	

11431 rows × 24 columns

Showing Boxplot after removing outliers:

In [104]:
```
sns.boxplot(df['Freelancer Preferred From'])
```

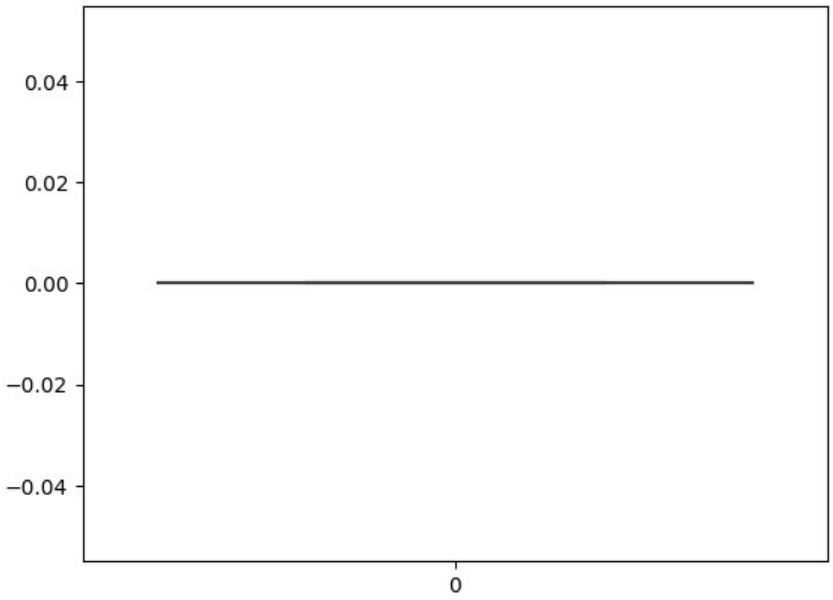
Out[104]: <Axes: >



Checking Outlier in Location_on_site

In [105]:
```
sns.boxplot(df['Location_onsite'])
```

Out[105]: <Axes: >



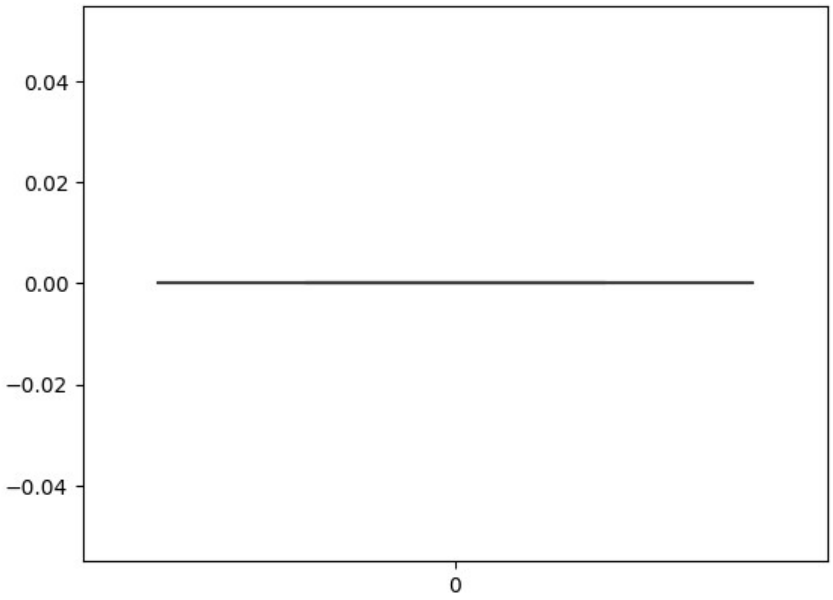
In [106]:
```
df.dtypes
```

Out[106]:
```
Title                          int32
Category Name                  int32
Experience                     int64
Sub Category Name              int32
Currency                       int64
Budget                       float64
Freelancer Preferred From      int32
Description                    int32
Client City                    int32
Client Country                 int32
Client Currency              float64
Location_onsite                int32
Location_remote                int32
Location_remote_country        int32
Type_fixed_price               int32
Type_hourly                    int32
Posted_month                   int32
Posted_Date                    int32
Posted_year                    int32
Client_Month                   int32
Client_Date                    int32
Client_Year                    int32
Posted_hour                    int32
Posted_Minutes                 int32
dtype: object
```

Checking Outliers in Location_remote_country

In [107]:
```
sns.boxplot(df['Location_remote_country'])
```

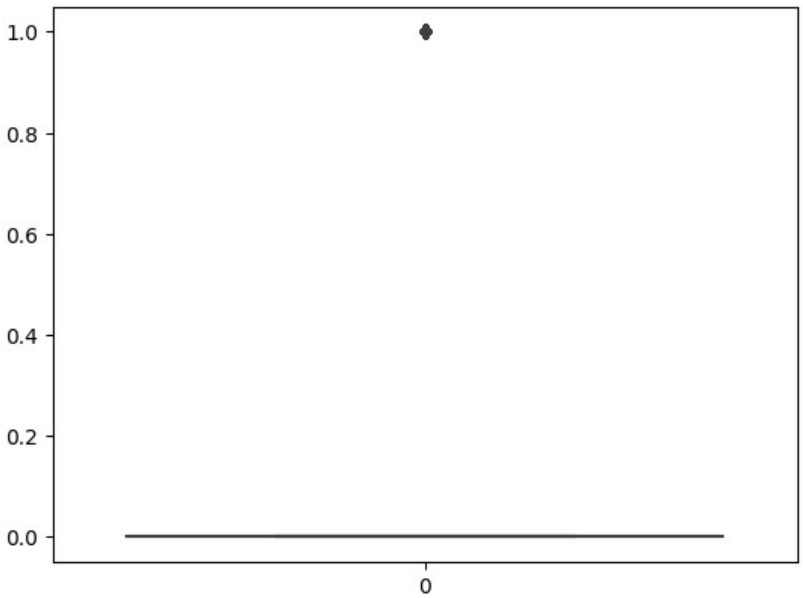
Out[107]: <Axes: >



Checking Outliers in Type_hourly

In [108]:
```
sns.boxplot(df['Type_hourly'])
```

Out[108]: <Axes: >



In [109]:
```
Type_hourly = sorted(df['Type_hourly'])
Type_hourly
```

Out[109]:
```
[0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
```

In [110]:
```
df['Type_hourly'].value_counts()
```

Out[110]:
```
Type_hourly
0    9878
1    1553
Name: count, dtype: int64
```

In [111]:
```
# IQR Method
```

In [112]:
```
Q1, Q3 = np.percentile(Type_hourly,[25,75])
Q1, Q3
```

Out[112]:
```
(0.0, 0.0)
```

In [113]:
```
IQR = Q3-Q1
IQR
```

Out[113]:
```
0.0
```

In [114]:
```
Lowerfence = Q1-(1.5*IQR)
Upperfence = Q3+(1.5*IQR)
```

In [115]:
```
print(Lowerfence)
print(Upperfence)
```

```
0.0
0.0
```

In [116]:
```
df = df[df['Type_hourly']==0]
```

In [117]:
```
df
```

Out[117]:

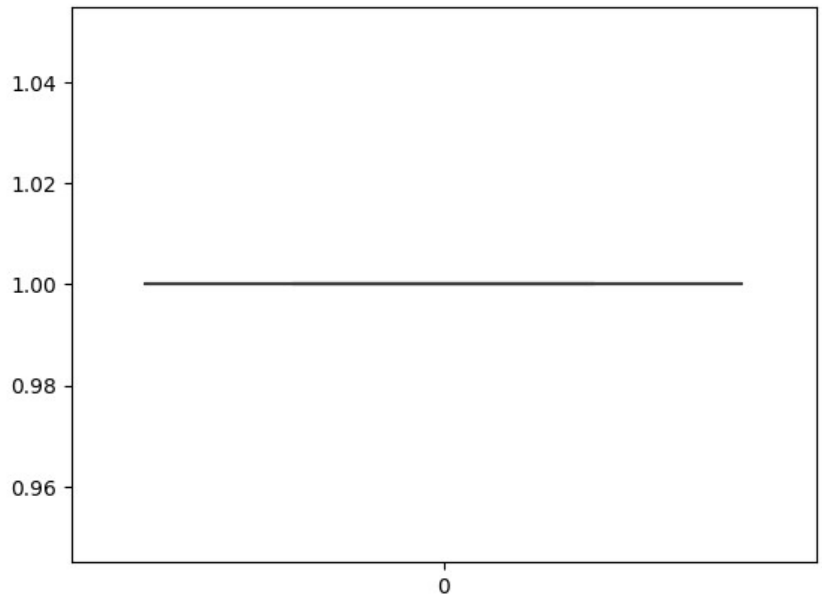
	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client City	Client Country	...	Type_fixed_price	Typ
0	969	1	0	42	1	65.736	1	10434	489	61	...	1	
1	6377	7	0	45	1	25.600	1	1247	940	129	...	1	
2	1108	0	0	37	1	15.360	1	2179	940	129	...	1	
3	467	0	0	90	1	17.920	1	2181	940	129	...	1	
4	3859	2	2	76	1	10000.000	1	3024	1079	58	...	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
12215	10800	6	0	14	1	25.600	1	4734	377	129	...	1	
12216	8927	6	0	26	1	38.400	1	6546	940	129	...	1	
12217	7958	8	0	21	1	64.000	1	3660	47	88	...	1	
12219	8927	6	0	25	1	64.000	1	6547	940	129	...	1	
12221	1407	0	2	1	1	96.000	1	3158	632	129	...	1	

9878 rows × 24 columns

In [118]:
```
# Checking outlier in Type_fixed_price
```

In [119]:
```
sns.boxplot(df['Type_fixed_price'])
```

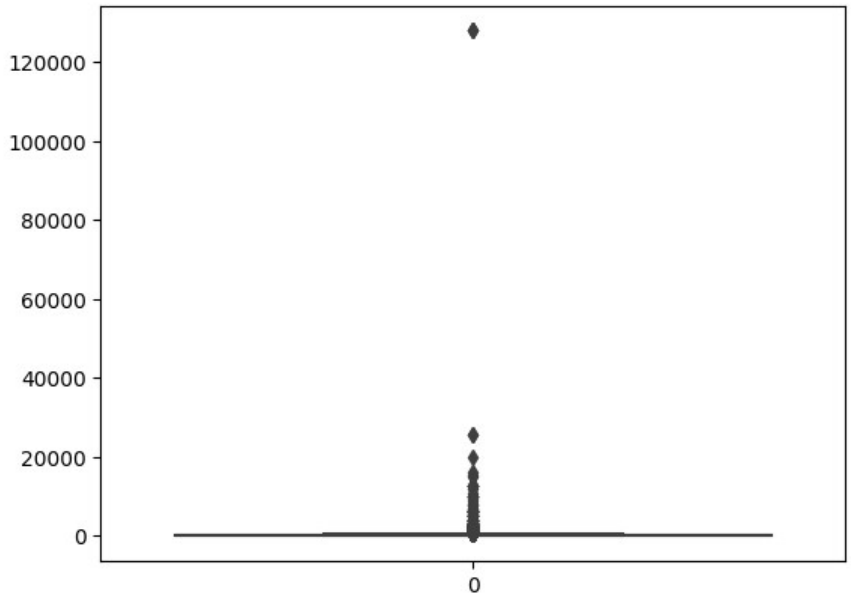
Out[119]:
```
<Axes: >
```



Checking outliers in Budget

In [120]:
```
sns.boxplot(df['Budget'])
```

Out[120]: <Axes: >



In [121]:
```
outliers=[]
def detect_outliers(data):
    threshold=3 ## 3rd standard deviation
    mean=np.mean(data)
    std=np.std(data)

    for i in data:
        z_score=(i-mean)/std
        if np.abs(z_score)> threshold:
            outliers.append(i)
    return outliers
```

In [122]:
```
detect_outliers(df['Budget'])
```

Out[122]:
```
[10000.0,
 10000.0,
 12800.0,
 12800.0,
 127998.72,
 12736.0,
 10000.0,
 127998.72,
 8000.0,
 9984.0,
 12800.0,
 12000.0,
 8832.0,
 7680.0,
 25600.0,
 7669.199999999999,
 16000.0,
 12800.0,
 127998.72,
 15488.0,
 25600.0,
 9600.0,
 20000.0,
 15000.0,
 10000.0,
 10955.999999999998,
 12800.0]
```

In [123]:
```
# Add all the desired values to this list
```

In [124]:
```
budget_values = [10000.0,
 10000.0,
 12800.0,
 12800.0,
 127998.72,
 12736.0,
 10000.0,
 127998.72,
 8000.0,
 9984.0,
 12800.0,
 12000.0,
 8832.0,
 7680.0,
 25600.0,
 7669.199999999999,
 16000.0,
 12800.0,
 127998.72,
 15488.0,
 25600.0,
 9600.0,
 20000.0,
 15000.0,
 10000.0,
 10955.999999999998,
 12800.0]
```

In [125]:
```
a = df[df['Budget'].isin(budget_values)]
a
```

Out[125]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client City	Client Country	...	Type_fixed_price	Typ
4	3859	2	2	76	1	10000.00	1	3024	1079	58	...	1	
299	5098	2	2	76	1	10000.00	1	1293	1616	24	...	1	
385	9460	1	2	18	1	12800.00	1	8679	1330	129	...	1	
742	7670	4	2	66	1	12800.00	1	7631	120	116	...	1	
837	3969	2	2	75	1	127998.72	1	6645	1281	129	...	1	
1332	7941	6	2	56	1	12736.00	1	5767	371	129	...	1	
1635	11175	2	2	20	1	10000.00	1	1336	1079	58	...	1	
3460	487	1	2	101	1	127998.72	1	1710	1513	129	...	1	
3565	3207	5	2	34	1	8000.00	1	9738	940	129	...	1	
3926	2784	6	1	102	1	9984.00	1	866	940	129	...	1	
4341	10691	6	2	69	1	12800.00	1	8960	1591	129	...	1	
4533	379	6	2	102	1	12000.00	1	4059	798	24	...	1	
4563	11118	6	2	69	1	8832.00	1	11537	741	129	...	1	
5372	6457	2	2	20	1	7680.00	1	11849	940	129	...	1	
5806	751	6	2	69	1	25600.00	1	5860	1658	24	...	1	
6180	7455	6	2	102	1	7669.20	1	11736	1328	57	...	1	
6497	2555	1	2	101	1	16000.00	1	815	690	129	...	1	
6612	11529	1	2	101	1	12800.00	1	8306	744	129	...	1	
6736	922	7	2	96	1	127998.72	1	3029	1456	129	...	1	
6971	3112	5	2	34	1	15488.00	1	517	1695	129	...	1	
7943	7128	1	2	16	1	25600.00	1	2818	613	129	...	1	
8208	2888	1	2	101	1	9600.00	1	10940	940	129	...	1	
9600	378	6	2	102	1	20000.00	1	4058	798	24	...	1	
10353	6641	7	2	96	1	15000.00	1	10534	324	55	...	1	
10419	1216	6	2	28	1	10000.00	1	9361	304	114	...	1	
10505	10979	6	2	28	1	10956.00	1	1478	59	15	...	1	
10704	3526	6	2	69	1	12800.00	1	9924	377	129	...	1	

27 rows × 24 columns

In [126]:
```
# Values to exclude
```

In [127]:
```
budget_values = [10000.0,
 10000.0,
 12800.0,
 12800.0,
 127998.72,
 12736.0,
 10000.0,
 127998.72,
 8000.0,
 9984.0,
 12800.0,
 12000.0,
 8832.0,
 7680.0,
 25600.0,
 7669.199999999999,
 16000.0,
 12800.0,
 127998.72,
 15488.0,
 25600.0,
 9600.0,
 20000.0,
 15000.0,
 10000.0,
 10955.999999999998,
 12800.0]
```

In [128]:
```
df = df[~df['Budget'].isin(budget_values)]
df
```

Out[128]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client City	Client Country	...	Type_fixed_price	Type_
0	969	1	0	42	1	65.736	1	10434	489	61	...	1	
1	6377	7	0	45	1	25.600	1	1247	940	129	...	1	
2	1108	0	0	37	1	15.360	1	2179	940	129	...	1	
3	467	0	0	90	1	17.920	1	2181	940	129	...	1	
5	1818	6	2	26	1	547.800	1	568	488	128	...	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
12215	10800	6	0	14	1	25.600	1	4734	377	129	...	1	
12216	8927	6	0	26	1	38.400	1	6546	940	129	...	1	
12217	7958	8	0	21	1	64.000	1	3660	47	88	...	1	
12219	8927	6	0	25	1	64.000	1	6547	940	129	...	1	

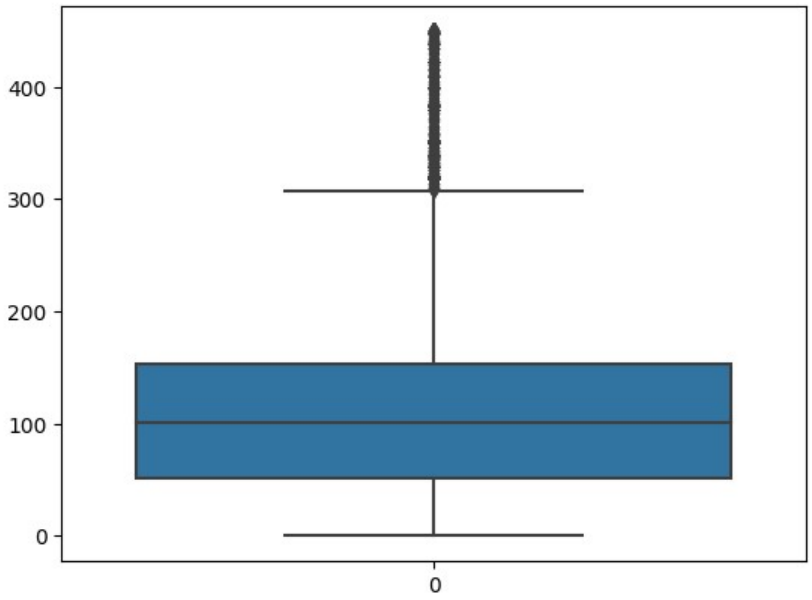
In [129]:
```
df =df[df['Budget']<=450]
```

In [130]:
```
df['Budget'].describe()
```

Out[130]:
```
count    8927.000000
mean      123.116374
std        99.486430
min         0.000000
25%        51.200000
50%       101.120000
75%       153.600000
max       450.000000
Name: Budget, dtype: float64
```

In [131]:
```
sns.boxplot(df['Budget'])
```

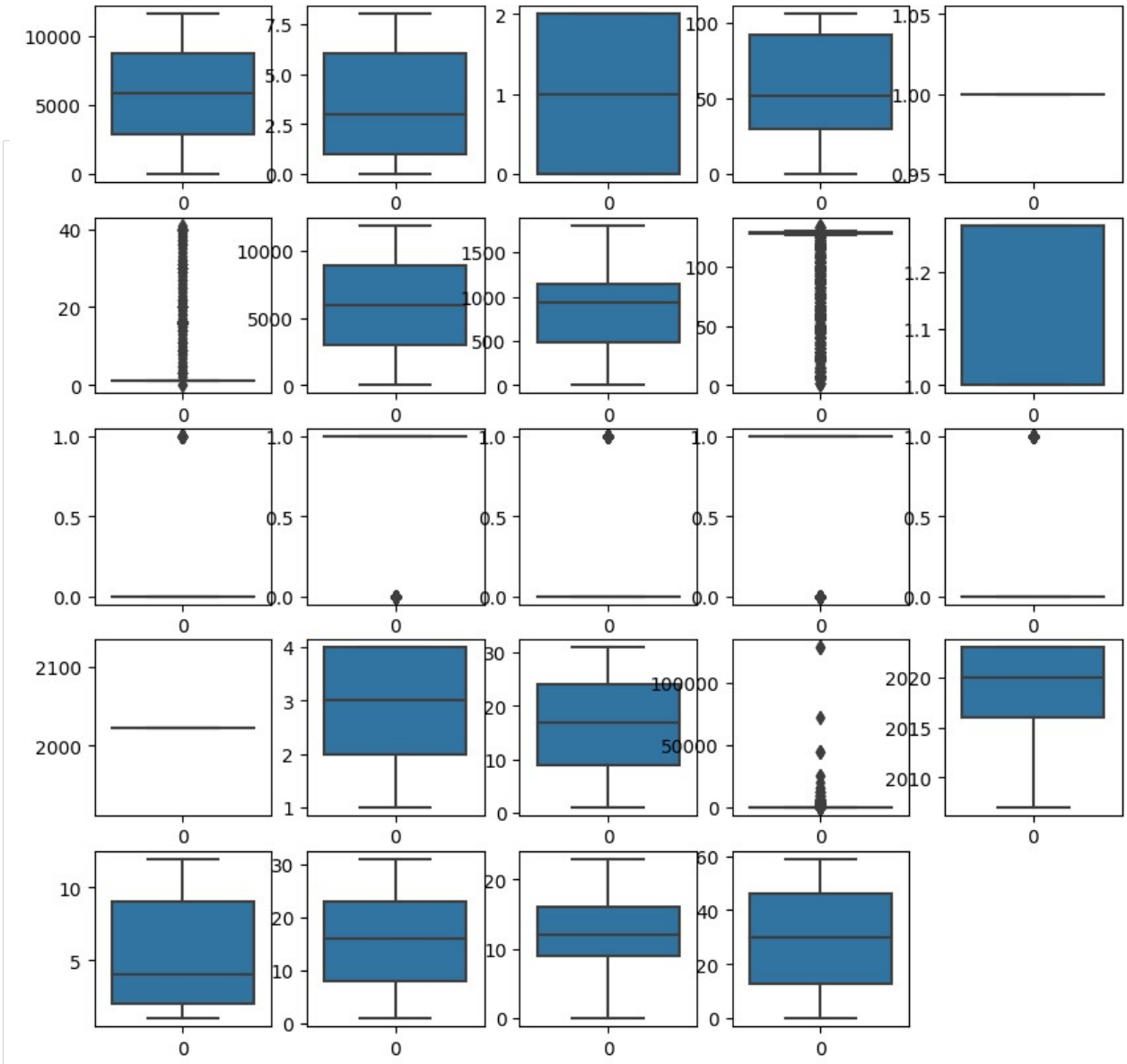
Out[131]: <Axes: >



Checking Outliers:

In [132]:
```
plt.figure(figsize=(10,10))
for i,col in enumerate(col_num):
    plt.subplot(5,5,i+1)
    print(sns.boxplot(col_num[col]))
plt.figure()
plt.show()
```

```
Axes(0.125,0.747241;0.133621x0.132759)
Axes(0.285345,0.747241;0.133621x0.132759)
Axes(0.44569,0.747241;0.133621x0.132759)
Axes(0.606034,0.747241;0.133621x0.132759)
Axes(0.766379,0.747241;0.133621x0.132759)
Axes(0.125,0.587931;0.133621x0.132759)
Axes(0.285345,0.587931;0.133621x0.132759)
Axes(0.44569,0.587931;0.133621x0.132759)
Axes(0.606034,0.587931;0.133621x0.132759)
Axes(0.766379,0.587931;0.133621x0.132759)
Axes(0.125,0.428621;0.133621x0.132759)
Axes(0.285345,0.428621;0.133621x0.132759)
Axes(0.44569,0.428621;0.133621x0.132759)
Axes(0.606034,0.428621;0.133621x0.132759)
Axes(0.766379,0.428621;0.133621x0.132759)
Axes(0.125,0.26931;0.133621x0.132759)
Axes(0.285345,0.26931;0.133621x0.132759)
Axes(0.44569,0.26931;0.133621x0.132759)
Axes(0.606034,0.26931;0.133621x0.132759)
Axes(0.766379,0.26931;0.133621x0.132759)
Axes(0.125,0.11;0.133621x0.132759)
Axes(0.285345,0.11;0.133621x0.132759)
Axes(0.44569,0.11;0.133621x0.132759)
Axes(0.606034,0.11;0.133621x0.132759)
```



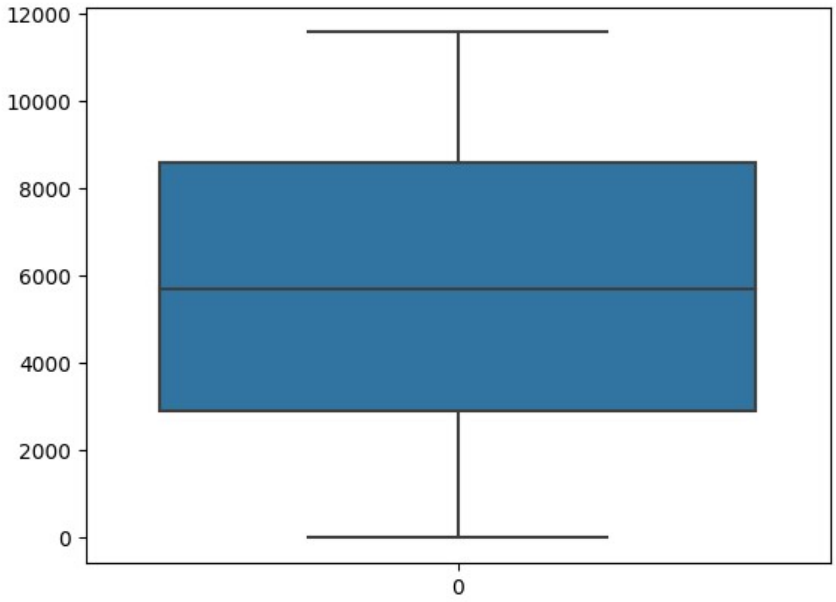
```
<Figure size 640x480 with 0 Axes>
```

In [133]:
```
df.dtypes
```

Out[133]:
```
Title                          int32
Category Name                  int32
Experience                     int64
Sub Category Name              int32
Currency                       int64
Budget                       float64
Freelancer Preferred From      int32
Description                    int32
Client City                    int32
Client Country                 int32
Client Currency              float64
Location_onsite                int32
Location_remote                int32
Location_remote_country        int32
Type_fixed_price               int32
Type_hourly                    int32
Posted_month                   int32
Posted_Date                    int32
Posted_year                    int32
Client_Month                   int32
Client_Date                    int32
Client_Year                    int32
Posted_hour                    int32
Posted_Minutes                 int32
dtype: object
```

In [134]:
```
sns.boxplot(df['Title'])
```

Out[134]:
```
<Axes: >
```

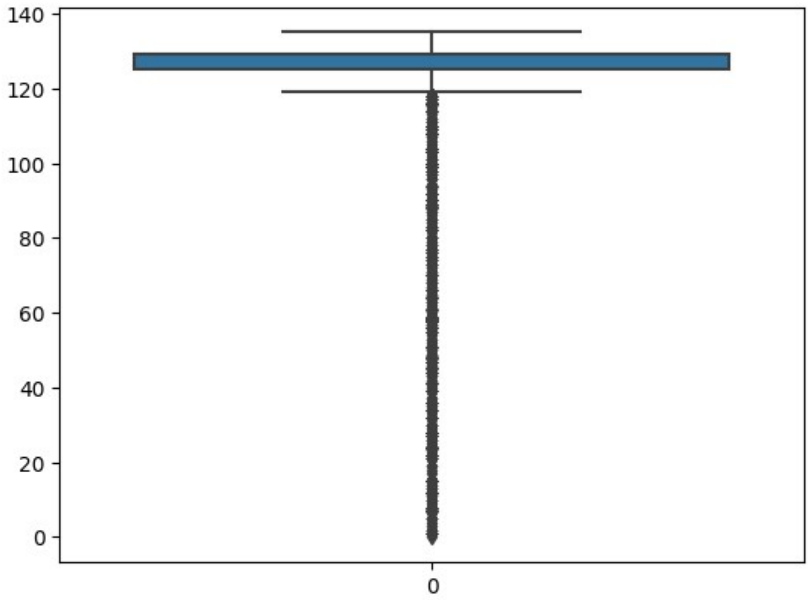


In [135]:
```
df.shape
```

Out[135]:
```
(8927, 24)
```

In [136]:
```
sns.boxplot(df['Client Country'])
```

Out[136]: <Axes: >



Removing Outliers from 'Client Country'

In [137]:
```
Client_country = sorted(df['Client Country'])
Client_country
```

Out[137]:
```
[0,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 2,
 2,
 2,
 2,
```

IQR Method

In [138]:
```
Q1,Q3 = np.percentile(Client_country,[25,75])
Q1, Q3
```

Out[138]: (125.0, 129.0)

In [139]:
```
IQR = Q3 - Q1
IQR
```

Out[139]: 4.0

In [140]:
```
Lowerfence = Q1-(1.5*IQR)
Upperfence = Q3+(1.5*IQR)
```

In [141]:
```
Lowerfence, Upperfence
```

Out[141]: (119.0, 135.0)

In [142]:
```
df = df[df['Client Country']<Upperfence]
```

In [143]:
```
df
```

Out[143]:

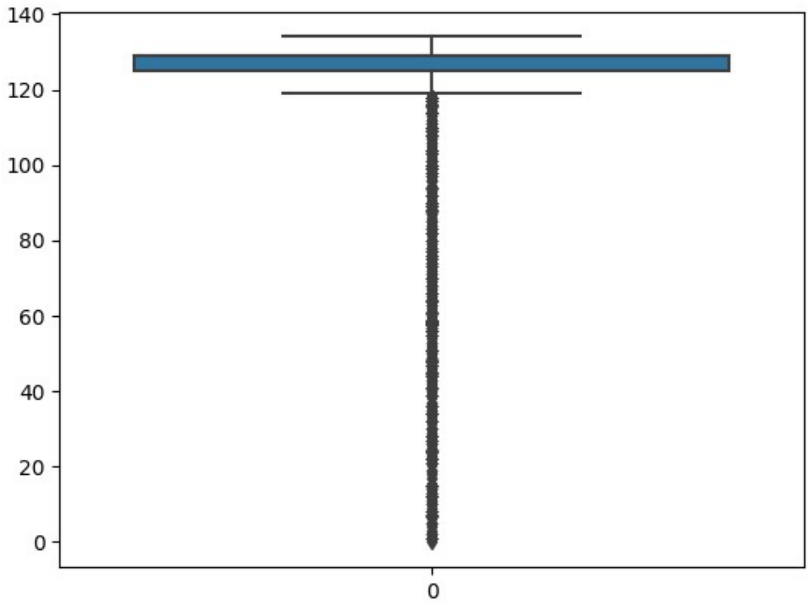
	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client City	Client Country	...	Type_fixed_price	Type_I
0	969	1	0	42	1	65.736	1	10434	489	61	...	1	
1	6377	7	0	45	1	25.600	1	1247	940	129	...	1	
2	1108	0	0	37	1	15.360	1	2179	940	129	...	1	
3	467	0	0	90	1	17.920	1	2181	940	129	...	1	
6	6384	1	0	101	1	10.000	1	6724	940	129	...	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
12215	10800	6	0	14	1	25.600	1	4734	377	129	...	1	
12216	8927	6	0	26	1	38.400	1	6546	940	129	...	1	
12217	7958	8	0	21	1	64.000	1	3660	47	88	...	1	
12219	8927	6	0	25	1	64.000	1	6547	940	129	...	1	
12221	1407	0	2	1	1	96.000	1	3158	632	129	...	1	

8926 rows × 24 columns

In [144]:
```
# After removing Outliers
```

In [145]:
```
sns.boxplot(df['Client Country'])
```

Out[145]: <Axes: >



Checking Ouliers Using Skewness:

In [146]:
```
col_num
```

Out[146]:

	Title	Category Name	Experience	Sub Category Name	Currency	Freelancer Preferred From	Description	Client City	Client Country	Client Currency	...	Type_hourly	Posted_ye
0	969	1	0	42	1	1	10434	489	61	1.09565	...	0	20
1	6377	7	0	45	1	1	1247	940	129	1.28240	...	0	20
2	1108	0	0	37	1	1	2179	940	129	1.28240	...	0	20
3	467	0	0	90	1	1	2181	940	129	1.28240	...	0	20
4	3859	2	2	76	1	1	3024	1079	58	1.00000	...	0	20
...	...	...	...	...	...	...	...	...	...	...	...	...	
12217	7958	8	0	21	1	1	3660	47	88	1.28240	...	0	20
12218	8803	1	1	101	1	16	8718	554	129	1.28240	...	0	20
12219	8927	6	0	25	1	1	6547	940	129	1.28240	...	0	20
12220	2057	1	2	101	1	1	4154	1135	58	1.00000	...	1	20
12221	1407	0	2	1	1	1	3158	632	129	1.28240	...	0	20

12222 rows × 24 columns

In [147]:
```
for col in col_num:
    print(col)
    print(skew(col_num[col]))

    plt.figure()
    sns.distplot(col_num[col])
```

```
Title
-0.009026401415337032
Category Name
0.13667421107983443
Experience
-0.014462463349103371
Sub Category Name
-0.018928584486655
Currency
nan
Freelancer Preferred From
5.042642835420588
Description
-0.0001446030632413126
Client City
0.037911936508722724
Client Country
-1.9572602850079996
Client Currency
```

In [148]:
```
df.describe()
```

Out[148]:

	Title	Category Name	Experience	Sub Category Name	Currency	Budget	Freelancer Preferred From	Description	Client City	Client Country
count	8926.000000	8926.000000	8926.000000	8926.000000	8926.0	8926.000000	8926.0	8926.000000	8926.000000	8926.000000
mean	5762.265741	3.681492	0.868698	57.143401	1.0	123.117978	1.0	5854.292404	873.936926	112.857831
std	3320.270340	2.837553	0.919756	32.275617	0.0	99.491888	0.0	3385.251437	454.237635	32.893589
min	0.000000	0.000000	0.000000	0.000000	1.0	0.000000	1.0	0.000000	0.000000	0.000000
25%	2926.250000	1.000000	0.000000	34.250000	1.0	51.200000	1.0	2977.250000	488.000000	125.000000
50%	5710.500000	3.000000	1.000000	52.000000	1.0	101.120000	1.0	5761.500000	940.000000	129.000000
75%	8605.750000	6.000000	2.000000	93.000000	1.0	153.600000	1.0	8651.750000	1136.750000	129.000000
max	11584.000000	8.000000	2.000000	106.000000	1.0	450.000000	1.0	11924.000000	1807.000000	134.000000

8 rows × 24 columns

Scaling the dataset by MinMaxSCaler:

In [149]:
```
from sklearn.preprocessing import MinMaxScaler
```

In [150]:
```
# Initialize the MinMaxScaler

scaler = MinMaxScaler()

# Specify the column(s) to be scaled

col_to_scale = df.columns

# Apply Min-Max scaling to the selected column(s)

df[col_to_scale] = scaler.fit_transform(df[col_to_scale])
```

In [151]:
```
# Print the scaled DataFrame

print(df)
```

```
          Title  Category Name  Experience  Sub Category Name  Currency
0      0.083650          0.125         0.0           0.396226       0.0  \
1      0.550501          0.875         0.0           0.424528       0.0
2      0.095649          0.000         0.0           0.349057       0.0
3      0.040314          0.000         0.0           0.849057       0.0
6      0.551105          0.125         0.0           0.952830       0.0
...         ...            ...         ...                ...       ...
12215  0.932320          0.750         0.0           0.132075       0.0
12216  0.770632          0.750         0.0           0.245283       0.0
12217  0.686982          1.000         0.0           0.198113       0.0
12219  0.770632          0.750         0.0           0.235849       0.0
12221  0.121461          0.000         1.0           0.009434       0.0

          Budget  Freelancer Preferred From  Description  Client City
0      0.146080                        0.0     0.875042     0.270614  \
1      0.056889                        0.0     0.104579     0.520199
2      0.034133                        0.0     0.182741     0.520199
3      0.039822                        0.0     0.182908     0.520199
6      0.022222                        0.0     0.563905     0.520199
...         ...                        ...          ...          ...
12215  0.056889                        0.0     0.397014     0.208633
12216  0.085333                        0.0     0.548977     0.520199
12217  0.142222                        0.0     0.306944     0.026010
12219  0.142222                        0.0     0.549061     0.520199
12221  0.213333                        0.0     0.264844     0.349751

       Client Country  ...  Type_fixed_price  Type_hourly  Posted_month
0            0.455224  ...               0.0          0.0           1.0  \
1            0.962687  ...               0.0          0.0           1.0
2            0.962687  ...               0.0          0.0           1.0
3            0.962687  ...               0.0          0.0           1.0
6            0.962687  ...               0.0          0.0           1.0
...               ...  ...               ...          ...           ...
12215        0.962687  ...               0.0          0.0           0.0
12216        0.962687  ...               0.0          0.0           0.0
12217        0.656716  ...               0.0          0.0           0.0
12219        0.962687  ...               0.0          0.0           0.0
12221        0.962687  ...               0.0          0.0           0.0

       Posted_Date  Posted_year  Client_Month  Client_Date  Client_Year
0         0.933333          0.0      0.909091     0.066667       0.1875  \
1         0.933333          0.0      0.090909     0.666667       0.6250
2         0.933333          0.0      0.272727     0.266667       1.0000
3         0.933333          0.0      0.272727     0.266667       1.0000
6         0.933333          0.0      0.272727     0.866667       1.0000
...            ...          ...           ...          ...          ...
12215     0.566667          0.0      0.181818     0.033333       0.1875
12216     0.566667          0.0      0.181818     0.433333       0.9375
12217     0.566667          0.0      0.454545     0.166667       0.2500
12219     0.566667          0.0      0.181818     0.433333       0.9375
12221     0.566667          0.0      0.727273     0.666667       0.8125

       Posted_hour  Posted_Minutes
0         0.782609        0.101695
1         0.739130        0.677966
2         0.739130        0.677966
3         0.739130        0.542373
6         0.695652        0.864407
...            ...             ...
12215     0.826087        0.644068
12216     0.826087        0.389831
12217     0.826087        0.389831
12219     0.826087        0.305085
12221     0.826087        0.305085

[8926 rows x 24 columns]
```

In [152]:
```
df.dtypes
```

Out[152]:
```
Title                        float64
Category Name                float64
Experience                   float64
Sub Category Name            float64
Currency                     float64
Budget                       float64
Freelancer Preferred From    float64
Description                  float64
Client City                  float64
Client Country               float64
Client Currency              float64
Location_onsite              float64
Location_remote              float64
Location_remote_country      float64
Type_fixed_price             float64
Type_hourly                  float64
Posted_month                 float64
Posted_Date                  float64
Posted_year                  float64
Client_Month                 float64
Client_Date                  float64
Client_Year                  float64
Posted_hour                  float64
Posted_Minutes               float64
dtype: object
```

In [153]:
```
df.isna().sum()
```

Out[153]:
```
Title                        0
Category Name                0
Experience                   0
Sub Category Name            0
Currency                     0
Budget                       0
Freelancer Preferred From    0
Description                  0
Client City                  0
Client Country               0
Client Currency              0
Location_onsite              0
Location_remote              0
Location_remote_country      0
Type_fixed_price             0
Type_hourly                  0
Posted_month                 0
Posted_Date                  0
Posted_year                  0
Client_Month                 0
Client_Date                  0
Client_Year                  0
Posted_hour                  0
Posted_Minutes               0
dtype: int64
```

In [154]:
```
df.shape
```

Out[154]:
```
(8926, 24)
```

After evaluating various regression algorithms, I have decided to finalize the XGBoost regression model as it consistently demonstrates higher accuracy and superior performance compared to others.

XG-Boost Regression Model

split the data into x and y

In [155]:
```
x = df.drop('Budget', axis=1)
y = df['Budget']
```

In [156]:
```
x.head()
```

Out[156]:

	Title	Category Name	Experience	Sub Category Name	Currency	Freelancer Preferred From	Description	Client City	Client Country	Client Currency	...	Type_fixed_price	Ty
0	0.083650	0.125	0.0	0.396226	0.0	0.0	0.875042	0.270614	0.455224	0.338704	...	0.0	
1	0.550501	0.875	0.0	0.424528	0.0	0.0	0.104579	0.520199	0.962687	1.000000	...	0.0	
2	0.095649	0.000	0.0	0.349057	0.0	0.0	0.182741	0.520199	0.962687	1.000000	...	0.0	
3	0.040314	0.000	0.0	0.849057	0.0	0.0	0.182908	0.520199	0.962687	1.000000	...	0.0	
6	0.551105	0.125	0.0	0.952830	0.0	0.0	0.563905	0.520199	0.962687	0.000000	...	0.0	

5 rows × 23 columns

In [157]:
```
y.head()
```

Out[157]:
```
0    0.146080
1    0.056889
2    0.034133
3    0.039822
6    0.022222
Name: Budget, dtype: float64
```

In [158]:
```
# Split the data into training and testing

from sklearn.model_selection import train_test_split

xtrain, xtest , ytrain, ytest= train_test_split(x,y,test_size=0.2, random_state=101)
```

In [159]:
```
xtrain.dtypes
```

Out[159]:
```
Title                        float64
Category Name                float64
Experience                   float64
Sub Category Name            float64
Currency                     float64
Freelancer Preferred From    float64
Description                  float64
Client City                  float64
Client Country               float64
Client Currency              float64
Location_onsite              float64
Location_remote              float64
Location_remote_country      float64
Type_fixed_price             float64
Type_hourly                  float64
Posted_month                 float64
Posted_Date                  float64
Posted_year                  float64
Client_Month                 float64
Client_Date                  float64
Client_Year                  float64
Posted_hour                  float64
Posted_Minutes               float64
dtype: object
```

XG-Boosting Regression:

In [160]:
```
from sklearn.metrics import mean_squared_error, r2_score

# XGBoost regressor model

xgb_model = xgb.XGBRegressor()

# Training the model
xgb_model.fit(xtrain, ytrain)

# Making predictions on the test set
trainpred = xgb_model.predict(xtrain)
testpred = xgb_model.predict(xtest)

# Evaluating the model
mse_train = mean_squared_error(ytrain, trainpred)
mse_test = mean_squared_error(ytest, testpred)
r2_train = r2_score(ytrain, trainpred)
r2_test = r2_score(ytest, testpred)

print('Mean square error for training data:', mse_train)
print('Mean square error for testing data :', mse_test)
print('R-squared for training data        :', r2_train)
print('R-squared for testing data         :', r2_test)
```

```
Mean square error for training data: 0.004095709609876648
Mean square error for testing data : 0.019768188426580877
R-squared for training data        : 0.9147987473478175
R-squared for testing data         : 0.6191819972310917
```

In [161]:
```
df.shape
```

Out[161]: (8926, 24)

After evaluating various regression algorithms, I have decided to finalize the XGBoost regression model as it consistently demonstrates higher accuracy and superior performance compared to others.

1. Mean square error for training data: 0.004095709609876648
2. Mean square error for testing data : 0.019768188426580877
3. R-squared for training data : 0.9147987473478175
4. R-squared for testing data : 0.6191819972310917