In [1]:

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

In [2]:

```python
df = pd.read_csv(r"C:\Users\Admin\Downloads\COVID clinical trials.csv" , index_col = 0)
```

Exploratory Data Analysis

In [3]:

```python
df.head()
```

Out[3]:

	NCT Number	Title	Acronym	Status	Study Results	Conditions	Interventions	Ou
Rank								
1	NCT04785898	Diagnostic Performance of the ID Now™ COVID-19...	COVID-IDNow	Active, not recruiting	No Results Available	Covid19	Diagnostic Test: ID Now™ COVID-19 Screening Test	|
2	NCT04595136	Study to Evaluate the Efficacy of COVID19-0001...	COVID-19	Not yet recruiting	No Results Available	SARS-CoV-2 Infection	Drug: Drug COVID19-0001-USR|Drug: normal saline	Cl re
3	NCT04395482	Lung CT Scan Analysis of SARS-CoV2 Induced Lun...	TAC-COVID19	Recruiting	No Results Available	covid19	Other: Lung CT scan analysis in COVID-19 patients	A of
4	NCT04416061	The Role of a Private Hospital in Hong Kong Am...	COVID-19	Active, not recruiting	No Results Available	COVID	Diagnostic Test: COVID 19 Diagnostic Test	su
5	NCT04395924	Maternal-foetal Transmission of SARS-Cov-2	TMF-COVID-19	Recruiting	No Results Available	Maternal Fetal Infection Transmission|COVID-19...	Diagnostic Test: Diagnosis of SARS-Cov2 by RT-...	pc

5 rows × 26 columns

In [4]:

```python
df.shape
```

```
Out[4]:
(5783, 26)
```

```
In [5]:
```

```
df.columns
```

```
Out[5]:
Index(['NCT Number', 'Title', 'Acronym', 'Status', 'Study Results',
       'Conditions', 'Interventions', 'Outcome Measures',
       'Sponsor/Collaborators', 'Gender', 'Age', 'Phases', 'Enrollment',
       'Funded Bys', 'Study Type', 'Study Designs', 'Other IDs', 'Start Date',
       'Primary Completion Date', 'Completion Date', 'First Posted',
       'Results First Posted', 'Last Update Posted', 'Locations',
       'Study Documents', 'URL'],
      dtype='object')
```

```
In [6]:
```

```
df.select_dtypes(include = 'object') .columns
```

```
Out[6]:
Index(['NCT Number', 'Title', 'Acronym', 'Status', 'Study Results',
       'Conditions', 'Interventions', 'Outcome Measures',
       'Sponsor/Collaborators', 'Gender', 'Age', 'Phases', 'Funded Bys',
       'Study Type', 'Study Designs', 'Other IDs', 'Start Date',
       'Primary Completion Date', 'Completion Date', 'First Posted',
       'Results First Posted', 'Last Update Posted', 'Locations',
       'Study Documents', 'URL'],
      dtype='object')
```

```
In [7]:
```

```
df.select_dtypes(exclude = 'object') .columns
```

```
Out[7]:
Index(['Enrollment'], dtype='object')
```

```
In [8]:
```

```
missing_data = df.isnull().mean() * 100
missing_data
```

```
Out[8]:
NCT Number                  0.000000
Title                       0.000000
Acronym                    57.115684
Status                      0.000000
Study Results               0.000000
Conditions                  0.000000
Interventions              15.320768
Outcome Measures            0.605222
Sponsor/Collaborators       0.000000
Gender                      0.172921
Age                         0.000000
Phases                     42.555767
Enrollment                  0.587930
Funded Bys                  0.000000
Study Type                  0.000000
Study Designs               0.605222
Other IDs                   0.017292
Start Date                  0.587930
Primary Completion Date     0.622514
Completion Date             0.622514
First Posted                0.000000
```

```
Results First Posted       99.377486
Last Update Posted          0.000000
Locations                  10.115857
Study Documents            96.852845
URL                         0.000000
dtype: float64
```

In [9]:

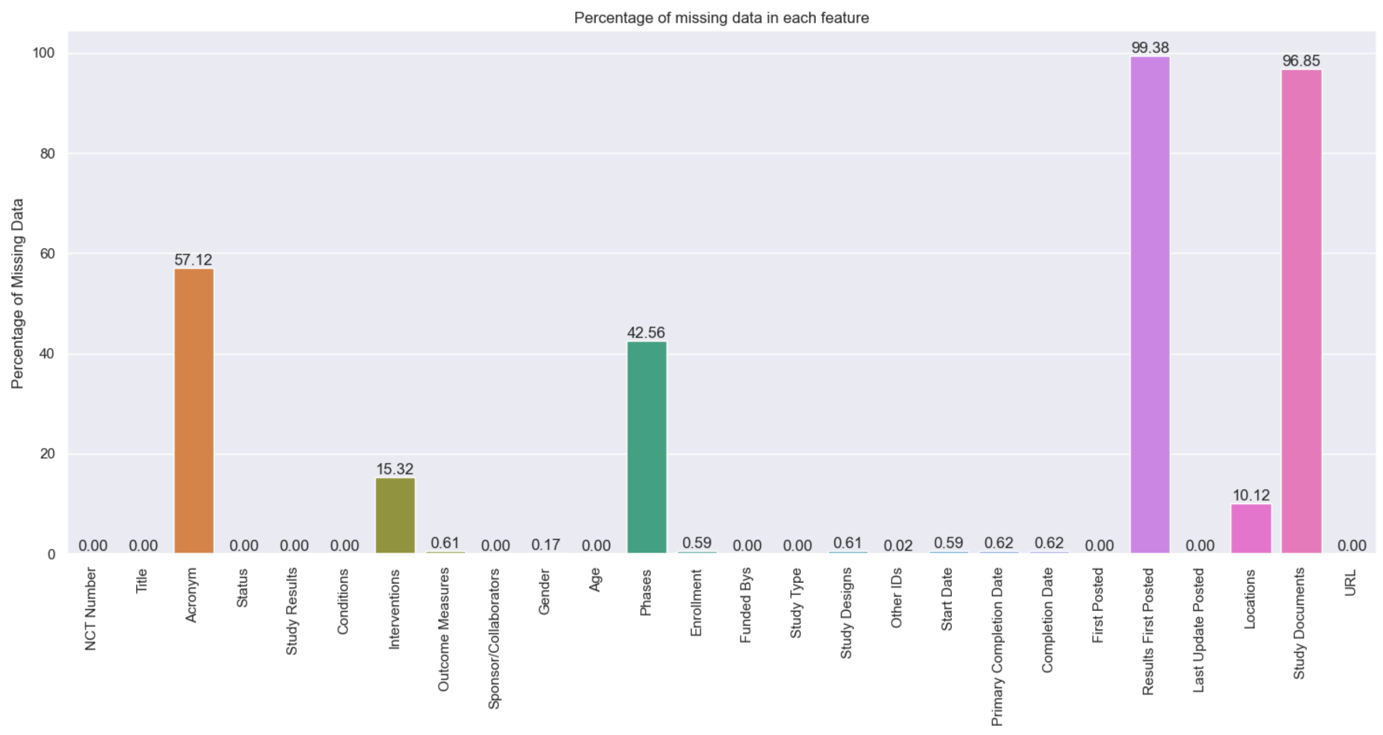
```python
import warnings
warnings.filterwarnings("ignore")
```

In [10]:

```python
import matplotlib.pyplot as plt
import seaborn as sns

def visualize_data(data, caption='', ylabel='Percentage of Missing Data'):
    sns.set(rc={'figure.figsize': (15, 8)})
    plt.xticks(rotation=90)

    # Convert to list correctly without calling .values
    x = list(data.keys())[:min(40, len(data))]
    y = list(data.values)[:min(40, len(data))]

    # Create a colored barplot where each bar has a unique color
    ax = sns.barplot(x=x, y=y, palette=sns.color_palette("husl", len(x)))
    ax.set_title(caption)
    ax.set_ylabel(ylabel)

    # Optional: add labels on bars
    for container in ax.containers:
        ax.bar_label(container, fmt='%.2f', label_type='edge')

    plt.tight_layout()
    plt.show()
```

In [11]:

```python
visualize_data(missing_data , 'Percentage of missing data in each feature')
```



In [12]:

```
df.drop(['Results First Posted', 'Study Documents'], axis=1, inplace=True)
```

In [13]:

```
df.columns
```

Out[13]:

```
Index(['NCT Number', 'Title', 'Acronym', 'Status', 'Study Results',
       'Conditions', 'Interventions', 'Outcome Measures',
       'Sponsor/Collaborators', 'Gender', 'Age', 'Phases', 'Enrollment',
       'Funded Bys', 'Study Type', 'Study Designs', 'Other IDs', 'Start Date',
       'Primary Completion Date', 'Completion Date', 'First Posted',
       'Last Update Posted', 'Locations', 'URL'],
      dtype='object')
```

In [14]:

```
print(f"Shape before dropping duplicates data {df.shape}")
df.drop_duplicates(inplace=True)
print(f"Shape after dropping duplicates data {df.shape}")
```

```
Shape before dropping duplicates data (5783, 24)
Shape after dropping duplicates data (5783, 24)
```

In [15]:

```
print(f"Shape before dropping Null rows {df.shape}")
df.dropna(axis=0, thresh=10, inplace=True)
print(f"Shape after dropping Null rows {df.shape}")
```

```
Shape before dropping Null rows (5783, 24)
Shape after dropping Null rows (5783, 24)
```

In [16]:

```
df.isnull().mean() * 100
```

Out[16]:

```
NCT Number                  0.000000
Title                       0.000000
Acronym                    57.115684
```

```
Status                      0.000000
Study Results               0.000000
Conditions                  0.000000
Interventions              15.320768
Outcome Measures            0.605222
Sponsor/Collaborators       0.000000
Gender                      0.172921
Age                         0.000000
Phases                     42.555767
Enrollment                  0.587930
Funded Bys                  0.000000
Study Type                  0.000000
Study Designs               0.605222
Other IDs                   0.017292
Start Date                  0.587930
Primary Completion Date     0.622514
Completion Date             0.622514
First Posted                0.000000
Last Update Posted          0.000000
Locations                  10.115857
URL                         0.000000
dtype: float64
```

In [17]:

```
countries = [ str(df.Locations.iloc[i]).split(',')[-1] for i in range(df.shape[0])]
df['Country'] = countries
```

In [18]:

```
df.columns
```

Out[18]:

```
Index(['NCT Number', 'Title', 'Acronym', 'Status', 'Study Results',
       'Conditions', 'Interventions', 'Outcome Measures',
       'Sponsor/Collaborators', 'Gender', 'Age', 'Phases', 'Enrollment',
       'Funded Bys', 'Study Type', 'Study Designs', 'Other IDs', 'Start Date',
       'Primary Completion Date', 'Completion Date', 'First Posted',
       'Last Update Posted', 'Locations', 'URL', 'Country'],
      dtype='object')
```

In [19]:

```
df.Country.value_counts()[:35]
```

Out[19]:

```
Country
 United States          1267
 France                  647
nan                      585
 United Kingdom          306
 Italy                   235
 Spain                   234
 Turkey                  219
 Canada                  202
 Egypt                   192
 China                   171
 Brazil                  137
 Germany                 128
 Belgium                  91
 Mexico                   88
 Switzerland              76
 Russian Federation       69
 Sweden                   57
 Denmark                  56
 Israel                   56
 India                    55
 Pakistan                 53
 Argentina                47
 Netherlands              46
 Norway                   38
 Hong Kong                36
 Colombia                 33
 Republic of              31
 Singapore                29
 Austria                  29
 Poland                   29
 Saudi Arabia             27
 Greece                   26
 Australia                26
 Islamic Republic of      23
 South Africa             22
Name: count, dtype: int64
```

In [20]:

```
print(f"Number of unique values is {df['Acronym'].nunique()}\n")
print(df['Acronym'].value_counts())
```

```
Number of unique values is 2338

Acronym
COVID-19         47
PROTECT           7
CORONA            6
SCOPE             5
RECOVER           5
                 ..
IgG4-COVID        1
Covid19-Pain      1
FACE COVID-19     1
SENTAD-COVID      1
US3R              1
Name: count, Length: 2338, dtype: int64
```

In [21]:

```
(df.Acronym.isnull().groupby(df.Country).mean().sort_values(ascending = False) * 100)[:6
```

Out[21]:

```
Country
 Bahrain                    100.000000
 Azerbaijan                 100.000000
 Bosnia and Herzegovina     100.000000
 Cape Verde                 100.000000
 Cambodia                   100.000000
 Bulgaria                   100.000000
 Belarus                    100.000000
 Cyprus                     100.000000
 Guinea-Bissau              100.000000
 Ecuador                    100.000000
 Dominican Republic         100.000000
 Iraq                       100.000000
 Rwanda                     100.000000
 South Sudan                100.000000
 North Macedonia           100.000000
 Kyrgyzstan                100.000000
 Uruguay                   100.000000
 Uzbekistan                100.000000
 Republic of                96.774194
 Taiwan                     93.750000
 Singapore                  93.103448
 Japan                      88.888889
 Kuwait                     87.500000
 China                      87.134503
 Turkey                     86.757991
 Ukraine                    85.714286
 Malaysia                   84.615385
 Egypt                      83.854167
 Hungary                    83.333333
 Hong Kong                  80.555556
 Kazakhstan                 80.000000
 Bangladesh                 80.000000
 India                      80.000000
 Saudi Arabia               77.777778
 Puerto Rico                76.470588
 Israel                     75.000000
 Zimbabwe                   75.000000
 Jordan                     72.727273
 Poland                     72.413793
 Indonesia                  71.428571
 United States              69.376480
 Romania                    69.230769
 Kenya                      66.666667
 Thailand                   66.666667
 Slovakia                   66.666667
 New Zealand                66.666667
 Lebanon                    66.666667
 Nepal                      66.666667
 Ethiopia                   66.666667
nan                         66.324786
 Russian Federation         65.217391
 Islamic Republic of        65.217391
 Chile                      64.705882
 Austria                    62.068966
 Pakistan                   60.377358
 Brazil                     59.124088
 Mexico                     57.954545
 Sweden                     57.894737
 Argentina                  57.446809
 Canada                     55.940594
Name: Acronym, dtype: float64
```

In [22]:

```
df.Acronym = df.Acronym.fillna("Missing Acronym")
```

In [23]:

```
df.isnull().mean() * 100
```

Out[23]:

```
NCT Number                 0.000000
Title                      0.000000
Acronym                    0.000000
Status                     0.000000
```

```
Study Results              0.000000
Conditions                 0.000000
Interventions             15.320768
Outcome Measures           0.605222
Sponsor/Collaborators      0.000000
Gender                     0.172921
Age                        0.000000
Phases                    42.555767
Enrollment                 0.587930
Funded Bys                 0.000000
Study Type                 0.000000
Study Designs              0.605222
Other IDs                  0.017292
Start Date                 0.587930
Primary Completion Date    0.622514
Completion Date            0.622514
First Posted               0.000000
Last Update Posted         0.000000
Locations                 10.115857
URL                        0.000000
Country                    0.000000
dtype: float64
```

In [24]:

```
categorical_features = df.select_dtypes(include='object').columns

# Select categorical features with missing values
features = categorical_features[df[categorical_features].isnull().mean() > 0]

# Fill missing values in each categorical feature
for feature in features:
    df[feature] = df[feature].fillna(f"Missing {feature}")
```

In [25]:

```
df.isnull().mean() * 100
```

Out[25]:

```
NCT Number                 0.00000
Title                      0.00000
Acronym                    0.00000
Status                     0.00000
Study Results              0.00000
Conditions                 0.00000
Interventions              0.00000
Outcome Measures           0.00000
Sponsor/Collaborators      0.00000
Gender                     0.00000
Age                        0.00000
Phases                     0.00000
Enrollment                 0.58793
Funded Bys                 0.00000
Study Type                 0.00000
Study Designs              0.00000
Other IDs                  0.00000
Start Date                 0.00000
Primary Completion Date    0.00000
Completion Date            0.00000
First Posted               0.00000
Last Update Posted         0.00000
```

```
Locations                    0.00000
URL                          0.00000
Country                      0.00000
dtype: float64
```

In [26]:

```
df.Enrollment.skew()
```

Out[26]:

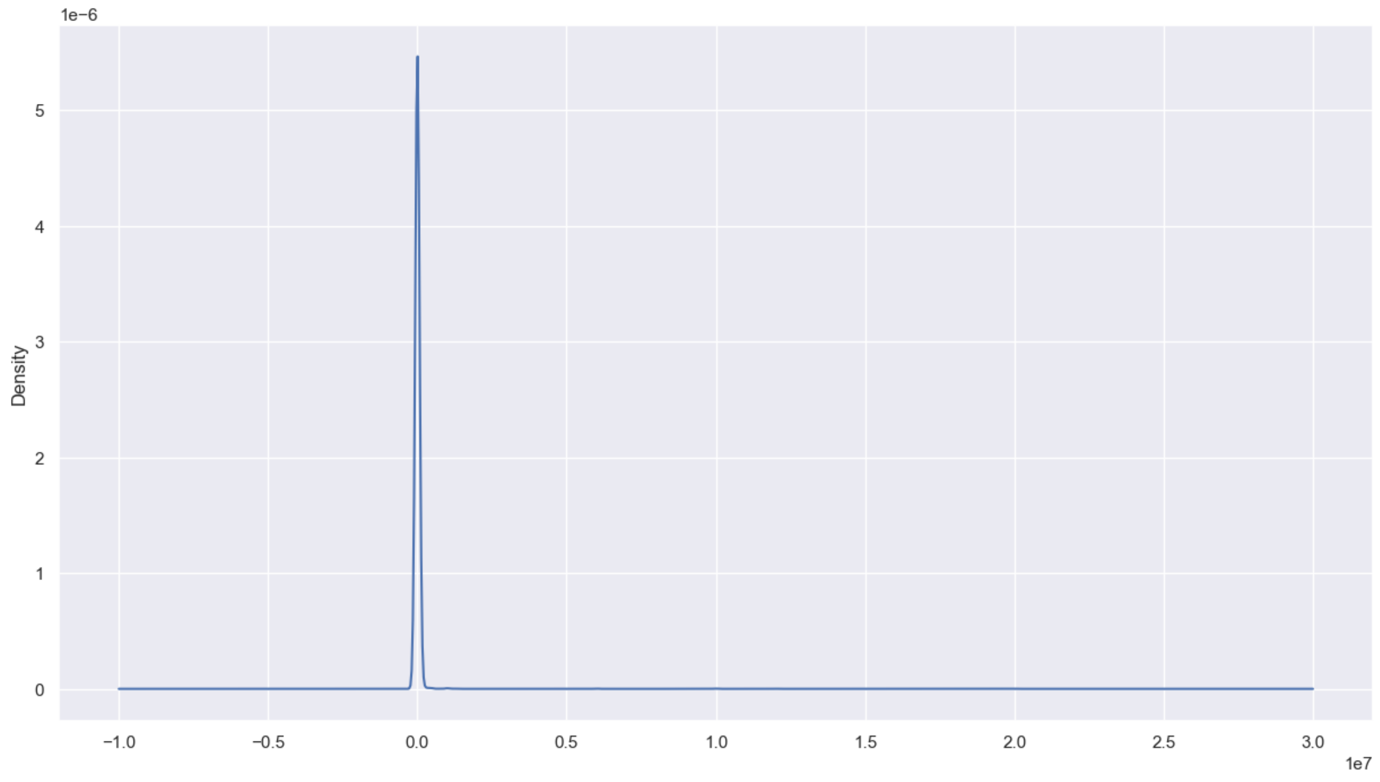
```
np.float64(34.06593382031148)
```

In [27]:

```
df.Enrollment.plot(kind = 'kde')
```

Out[27]:

```
<Axes: ylabel='Density'>
```



In [28]:

```
min_Value = df.Enrollment.min()
max_Value = df.Enrollment.max()
mean_Value = df.Enrollment.mean()
median_Value = df.Enrollment.median()
std_Value = df.Enrollment.std()

print(f"the min value is {min_Value} \n \
The max value is {max_Value} \n \
The mean is {mean_Value} \n \
The median is {median_Value} \n \
Standard Deviation is {std_Value}")
```

```
the min value is 0.0
 The max value is 20000000.0
 The mean is 18319.48860671421
 The median is 170.0
 Standard Deviation is 404543.7287841073
```

In [29]:

```
df.Enrollment = df.Enrollment.fillna(median_Value)
```

In [30]:

```
df.isnull().mean() * 100
```

Out[30]:

```
NCT Number                 0.0
Title                      0.0
Acronym                    0.0
Status                     0.0
Study Results              0.0
Conditions                 0.0
Interventions              0.0
Outcome Measures           0.0
Sponsor/Collaborators      0.0
Gender                     0.0
Age                        0.0
Phases                     0.0
Enrollment                 0.0
Funded Bys                 0.0
Study Type                 0.0
Study Designs              0.0
Other IDs                  0.0
Start Date                 0.0
Primary Completion Date    0.0
Completion Date            0.0
First Posted               0.0
Last Update Posted         0.0
Locations                  0.0
URL                        0.0
Country                    0.0
dtype: float64
```

In [31]:

```
df.head()
```

Out[31]:

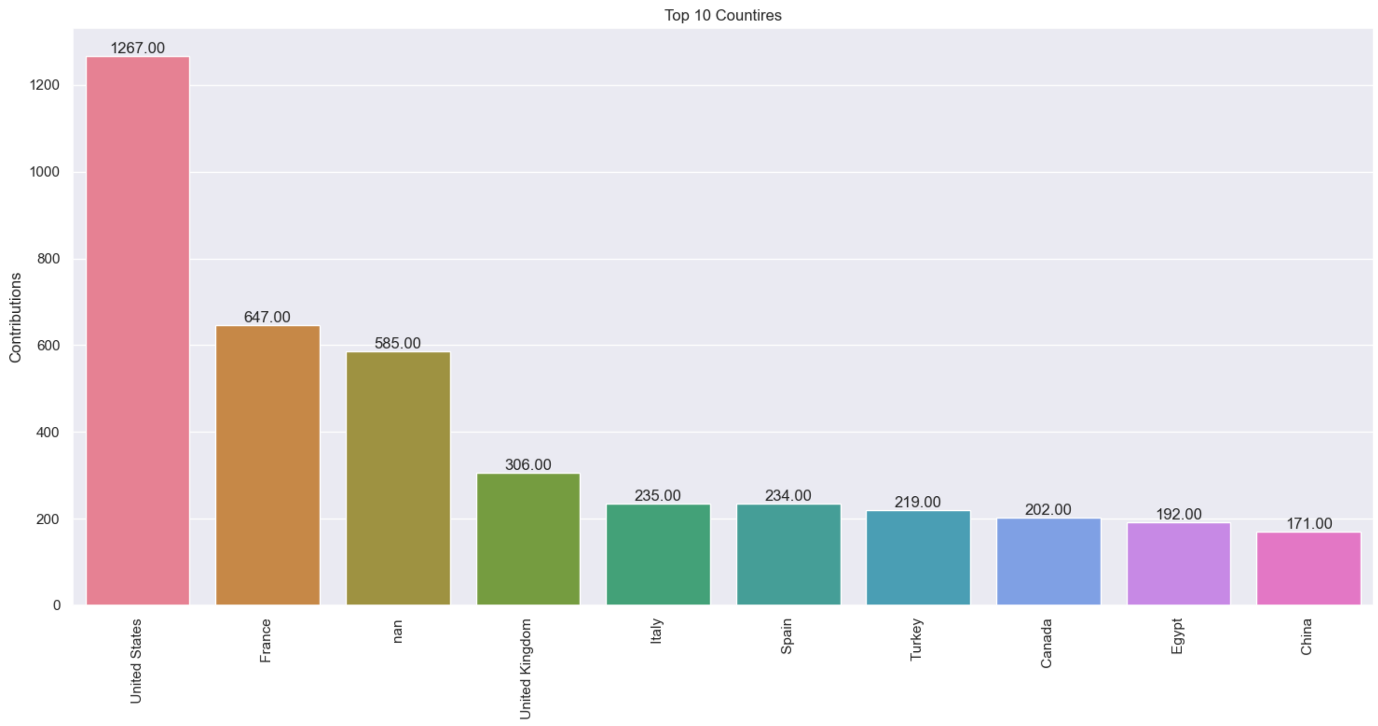
	NCT Number	Title	Acronym	Status	Study Results	Conditions	Interventions	Ou
Rank								
1	NCT04785898	Diagnostic Performance of the ID Now™ COVID-19...	COVID-IDNow	Active, not recruiting	No Results Available	Covid19	Diagnostic Test: ID Now™ COVID-19 Screening Test	
2	NCT04595136	Study to Evaluate the Efficacy of COVID19-0001...	COVID-19	Not yet recruiting	No Results Available	SARS-CoV-2 Infection	Drug: Drug COVID19-0001-USR|Drug: normal saline	Cl re
3	NCT04395482	Lung CT Scan Analysis of SARS-CoV2	TAC-COVID19	Recruiting	No Results Available	covid19	Other: Lung CT scan analysis in COVID-19 patients	A of

	NCT Number	Title	Acronym	Status	Study Results	Conditions	Interventions	Ou
Rank								
		Induced Lun...						
4	NCT04416061	The Role of a Private Hospital in Hong Kong Am...	COVID-19	Active, not recruiting	No Results Available	COVID	Diagnostic Test: COVID 19 Diagnostic Test	su
5	NCT04395924	Maternal-foetal Transmission of SARS-Cov-2	TMF-COVID-19	Recruiting	No Results Available	Maternal Fetal Infection Transmission|COVID-19...	Diagnostic Test: Diagnosis of SARS-Cov2 by RT-...	pc

5 rows × 25 columns

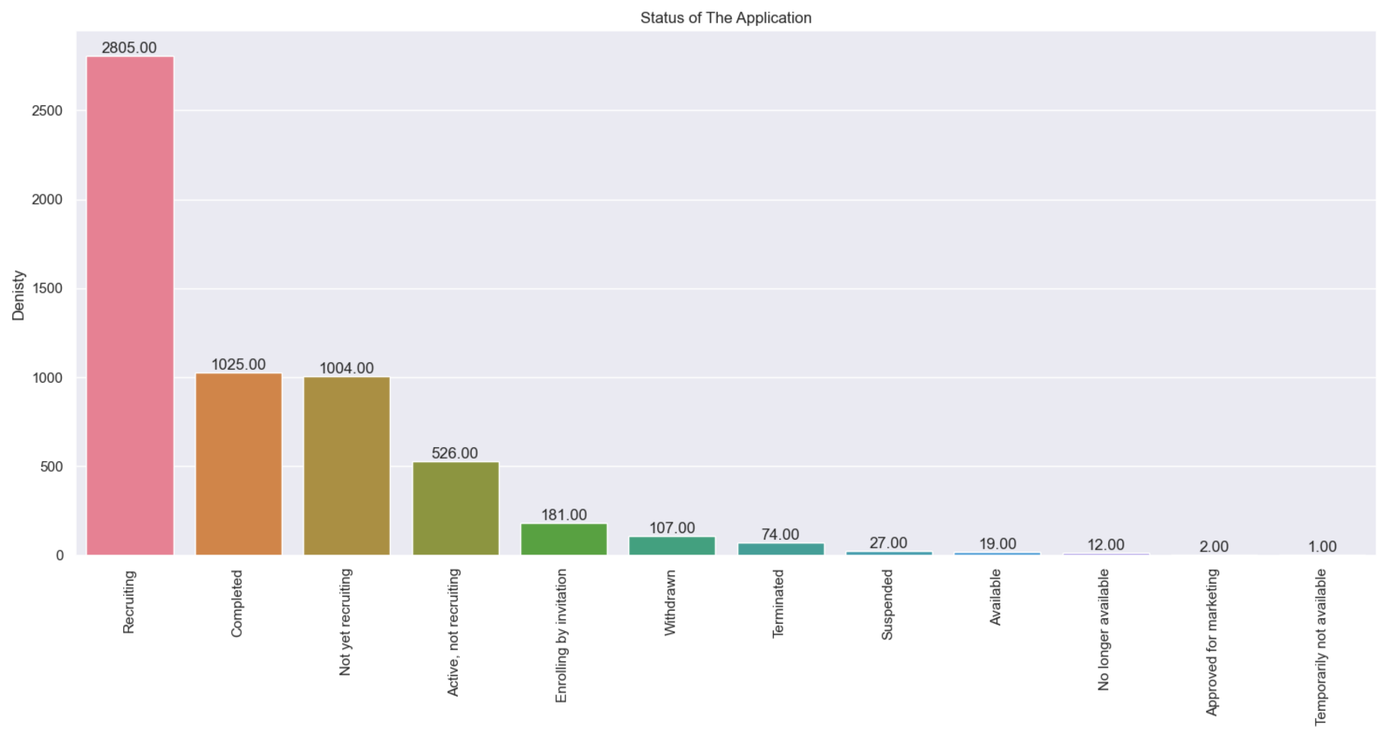
In [32]:

```python
top_10_Countires = df.Country.value_counts()[:10]
visualize_data(top_10_Countires , caption = 'Top 10 Countires'
               , ylabel = 'Contributions')
```



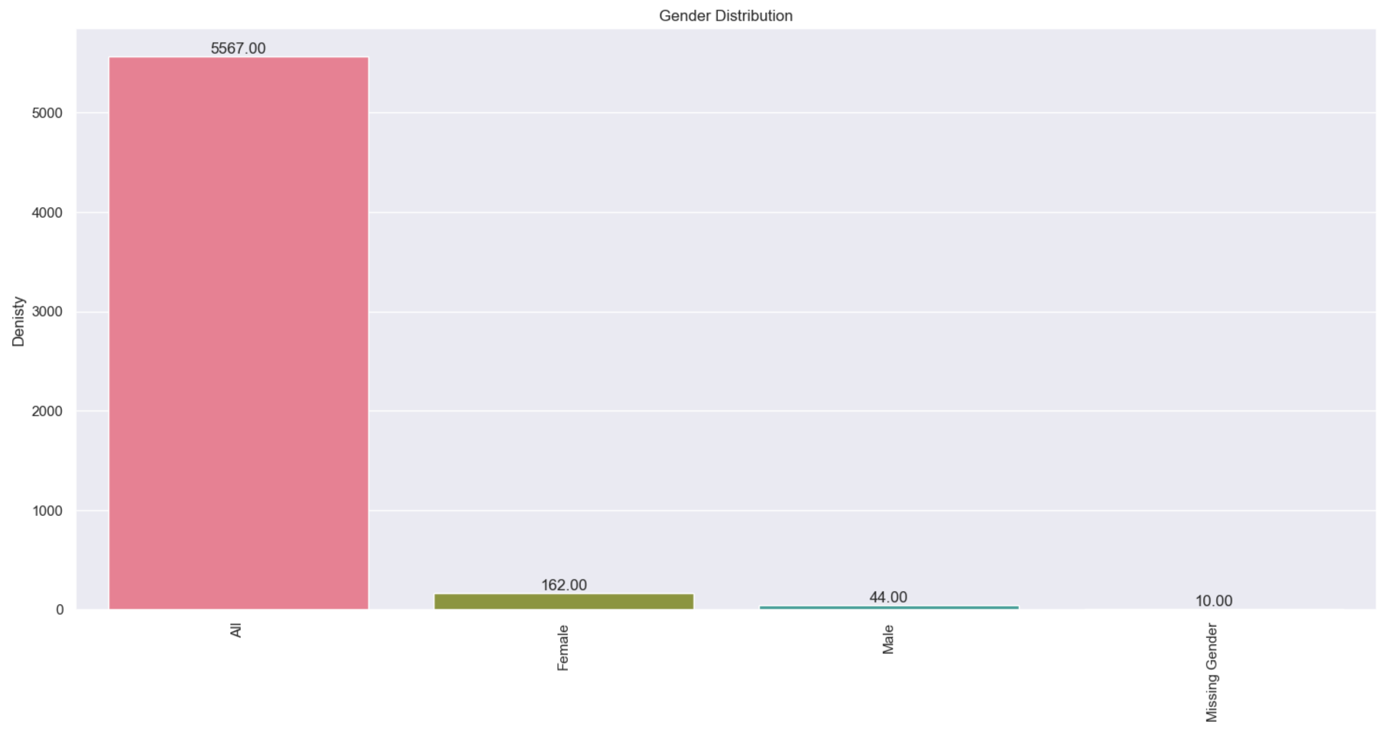
In [33]:

```python
status = df.Status.value_counts()

visualize_data(status , caption = 'Status of The Application' ,
               ylabel = 'Denisty')
```



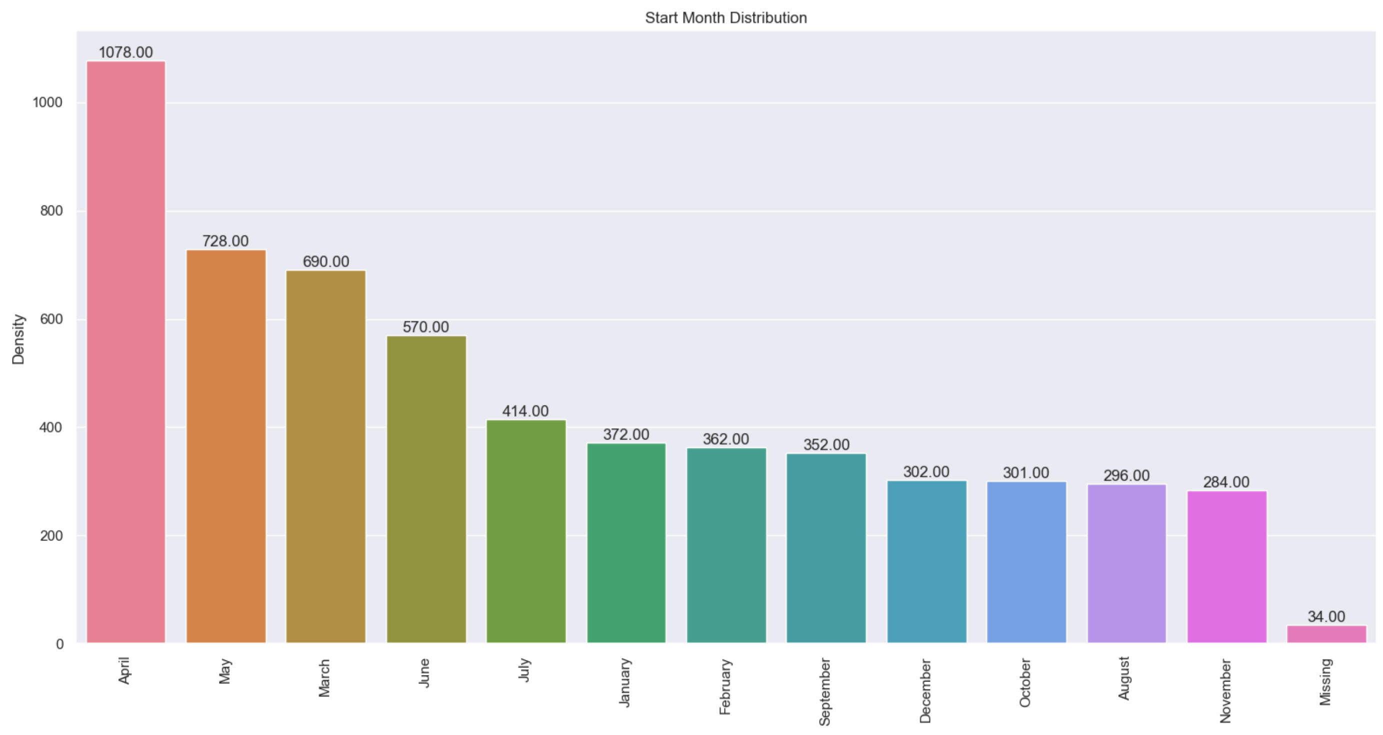
In [34]:

```
gender = df.Gender.value_counts()
visualize_data(gender , caption = 'Gender Distribution' ,
               ylabel = 'Denisty')
```



In [35]:

```
# Extract month name from 'Start Date'
start_month = pd.Series([df['Start Date'].iloc[i].split(' ')[0] for i in range(df.shape[

# Count frequency of each month
start_month_distribution = start_month.value_counts()

# Visualize
visualize_data(start_month_distribution, caption='Start Month Distribution', ylabel='Den
```



In [36]:

```
print(f"The shape of data frame is {df.shape}")
print(f"Nunique in NCT Number is {df['NCT Number']. nunique()}")
print(f"Nunique in URL is {df.URL.nunique()}")
```

```
The shape of data frame is (5783, 25)
Nunique in NCT Number is 5783
Nunique in URL is 5783
```

In [ ]: