

IMPORT LIBRARIES

In [3]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
```

IMPORT DATASET

In [4]:

```
df=pd.read_csv("WA_Fn-UseC_-HR-Employee-Attrition.csv")
```

In [5]:

df

Out[5]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical
...
1465	36	No	Travel_Frequently	884	Research & Development	23	2	Medical
1466	39	No	Travel_Rarely	613	Research & Development	6	1	Medical
1467	27	No	Travel_Rarely	155	Research & Development	4	3	Life Sciences
1468	49	No	Travel_Frequently	1023	Sales	2	3	Medical
1469	34	No	Travel_Rarely	628	Research & Development	8	3	Medical

1470 rows × 35 columns



In [6]:

```
df.head()
```

Out[6]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	Em
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1

5 rows × 35 columns



In [7]:

```
df.tail()
```

Out[7]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField
1465	36	No	Travel_Frequently	884	Research & Development	23	2	Medical
1466	39	No	Travel_Rarely	613	Research & Development	6	1	Medical
1467	27	No	Travel_Rarely	155	Research & Development	4	3	Life Sciences
1468	49	No	Travel_Frequently	1023	Sales	2	3	Medical
1469	34	No	Travel_Rarely	628	Research & Development	8	3	Medical

5 rows × 35 columns



In [8]:

```
df.shape
```

```
Out[8]:
```

```
(1470, 35)
```

```
In [9]:
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 1470 entries, 0 to 1469
```

```
Data columns (total 35 columns):
```

#	Column	Non-Null Count	Dtype
0	Age	1470 non-null	int64
1	Attrition	1470 non-null	object
2	BusinessTravel	1470 non-null	object
3	DailyRate	1470 non-null	int64
4	Department	1470 non-null	object
5	DistanceFromHome	1470 non-null	int64
6	Education	1470 non-null	int64
7	EducationField	1470 non-null	object
8	EmployeeCount	1470 non-null	int64
9	EmployeeNumber	1470 non-null	int64
10	EnvironmentSatisfaction	1470 non-null	int64
11	Gender	1470 non-null	object
12	HourlyRate	1470 non-null	int64
13	JobInvolvement	1470 non-null	int64
14	JobLevel	1470 non-null	int64
15	JobRole	1470 non-null	object
16	JobSatisfaction	1470 non-null	int64
17	MaritalStatus	1470 non-null	object
18	MonthlyIncome	1470 non-null	int64
19	MonthlyRate	1470 non-null	int64
20	NumCompaniesWorked	1470 non-null	int64
21	Over18	1470 non-null	object
22	Overtime	1470 non-null	object
23	PercentSalaryHike	1470 non-null	int64
24	PerformanceRating	1470 non-null	int64
25	RelationshipSatisfaction	1470 non-null	int64
26	StandardHours	1470 non-null	int64
27	StockOptionLevel	1470 non-null	int64
28	TotalWorkingYears	1470 non-null	int64
29	TrainingTimesLastYear	1470 non-null	int64
30	WorkLifeBalance	1470 non-null	int64
31	YearsAtCompany	1470 non-null	int64
32	YearsInCurrentRole	1470 non-null	int64
33	YearsSinceLastPromotion	1470 non-null	int64
34	YearsWithCurrManager	1470 non-null	int64

```
dtypes: int64(26), object(9)
```

```
memory usage: 402.1+ KB
```

In [10]:

```
df.describe()
```

Out[10]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	Environnr
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.000000
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306	2.721769
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335	1.093082
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000	1.000000
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000	2.000000
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000	3.000000
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000	4.000000
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000	4.000000

8 rows × 26 columns



In [11]:

```
corr=df.corr()  
corr
```

```
<ipython-input-11-7d5195e2bf4d>:1: FutureWarning: The default value of  
numeric_only in DataFrame.corr is deprecated. In a future version, it  
will default to False. Select only valid columns or specify the value of  
numeric_only to silence this warning.  
corr=df.corr()
```

Out[11]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber
Age	1.000000	0.010661	-0.001686	0.208034	NaN	-0.010145
DailyRate	0.010661	1.000000	-0.004985	-0.016806	NaN	-0.050990
DistanceFromHome	-0.001686	-0.004985	1.000000	0.021042	NaN	0.032916
Education	0.208034	-0.016806	0.021042	1.000000	NaN	0.042070
EmployeeCount	NaN	NaN	NaN	NaN	NaN	NaN
EmployeeNumber	-0.010145	-0.050990	0.032916	0.042070	NaN	1.000000
EnvironmentSatisfaction	0.010146	0.018355	-0.016075	-0.027128	NaN	0.017621
HourlyRate	0.024287	0.023381	0.031131	0.016775	NaN	0.035179
JobInvolvement	0.029820	0.046135	0.008783	0.042438	NaN	-0.006888

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber
JobLevel	0.509604	0.002966	0.005303	0.101589	NaN	-0.018519
JobSatisfaction	-0.004892	0.030571	-0.003669	-0.011296	NaN	-0.046247
MonthlyIncome	0.497855	0.007707	-0.017014	0.094961	NaN	-0.014829
MonthlyRate	0.028051	-0.032182	0.027473	-0.026084	NaN	0.012648
NumCompaniesWorked	0.299635	0.038153	-0.029251	0.126317	NaN	-0.001251
PercentSalaryHike	0.003634	0.022704	0.040235	-0.011111	NaN	-0.012944
PerformanceRating	0.001904	0.000473	0.027110	-0.024539	NaN	-0.020359
RelationshipSatisfaction	0.053535	0.007846	0.006557	-0.009118	NaN	-0.069861
StandardHours	NaN	NaN	NaN	NaN	NaN	NaN
StockOptionLevel	0.037510	0.042143	0.044872	0.018422	NaN	0.062227
TotalWorkingYears	0.680381	0.014515	0.004628	0.148280	NaN	-0.014365
TrainingTimesLastYear	-0.019621	0.002453	-0.036942	-0.025100	NaN	0.023603
WorkLifeBalance	-0.021490	-0.037848	-0.026556	0.009819	NaN	0.010309
YearsAtCompany	0.311309	-0.034055	0.009508	0.069114	NaN	-0.011240
YearsInCurrentRole	0.212901	0.009932	0.018845	0.060236	NaN	-0.008416
YearsSinceLastPromotion	0.216513	-0.033229	0.010029	0.054254	NaN	-0.009019
YearsWithCurrManager	0.202089	-0.026363	0.014406	0.069065	NaN	-0.009197

26 rows × 26 columns

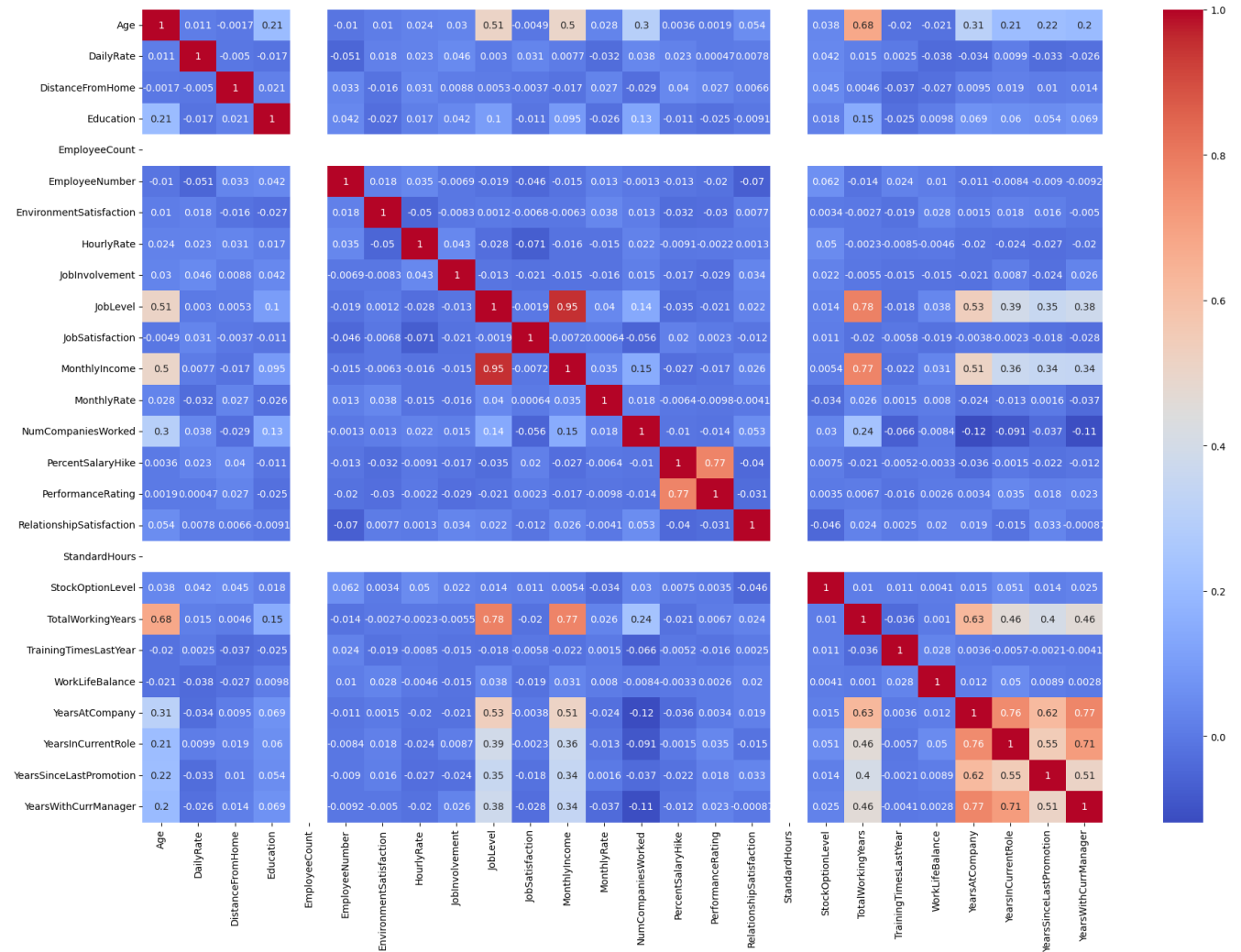


In [12]:

```
plt.subplots(figsize=(22,15))
sns.heatmap(corr,annot=True,cmap="coolwarm")
```

Out[12]:

<Axes: >



In [13]:

```
df.Attrition.value_counts()
```

Out[13]:

```
No      1233
Yes      237
Name: Attrition, dtype: int64
```

Checking for NULL Values

In [14]:

```
df.isnull().any()
```

Out[14]:

```
Age      False
Attrition False
BusinessTravel False
DailyRate False
...
Loading [MathJax]/extensions/Safe.js
...
False
```

DistanceFromHome	False
Education	False
EducationField	False
EmployeeCount	False
EmployeeNumber	False
EnvironmentSatisfaction	False
Gender	False
HourlyRate	False
JobInvolvement	False
JobLevel	False
JobRole	False
JobSatisfaction	False
MaritalStatus	False
MonthlyIncome	False
MonthlyRate	False
NumCompaniesWorked	False
Over18	False
Overtime	False
PercentSalaryHike	False
PerformanceRating	False
RelationshipSatisfaction	False
StandardHours	False
StockOptionLevel	False
TotalWorkingYears	False
TrainingTimesLastYear	False
WorkLifeBalance	False
YearsAtCompany	False
YearsInCurrentRole	False
YearsSinceLastPromotion	False
YearsWithCurrManager	False
dtype:	bool

Data Visualization

In [15]:

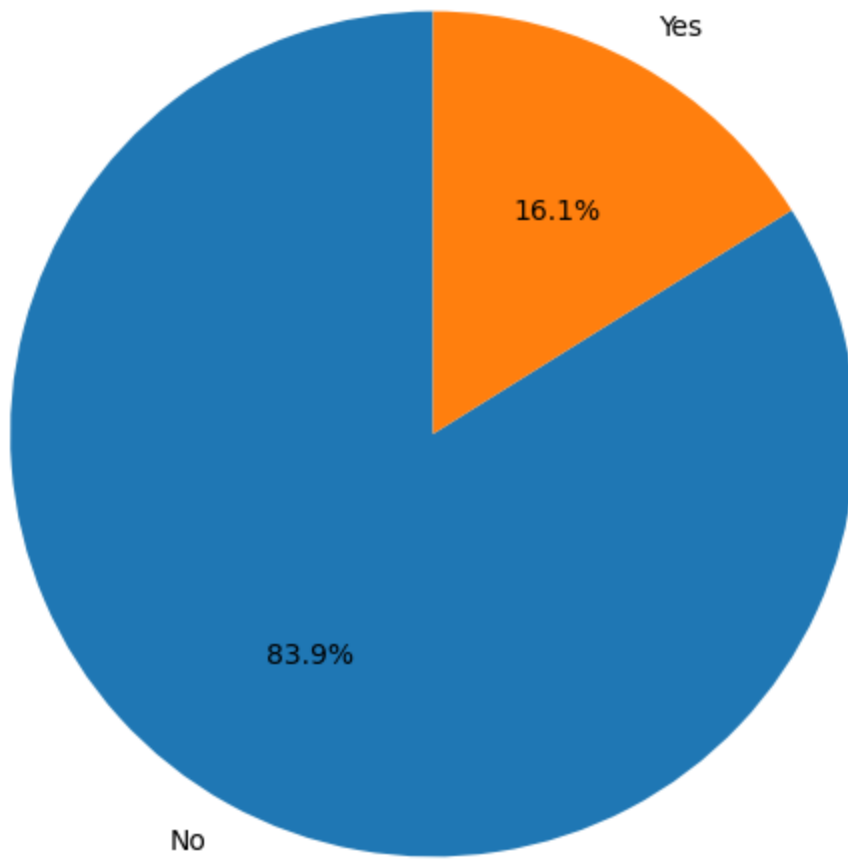
```

attrition_counts = df['Attrition'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(attrition_counts, labels=attrition_counts.index,
autopct='%1.1f%%', startangle=90)
plt.title('Attrition Distribution')
plt.axis('equal')

plt.show()

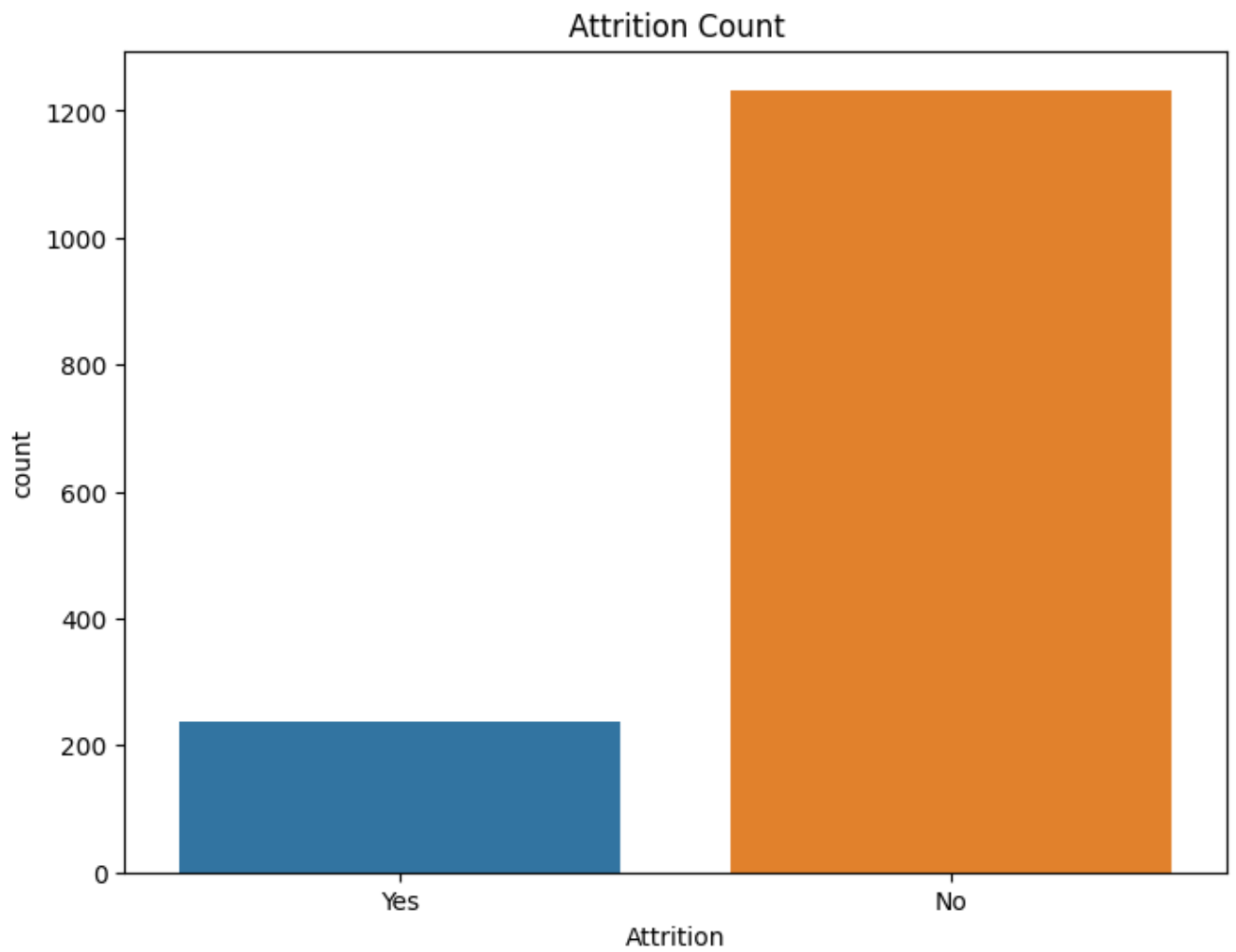
```

Attrition Distribution



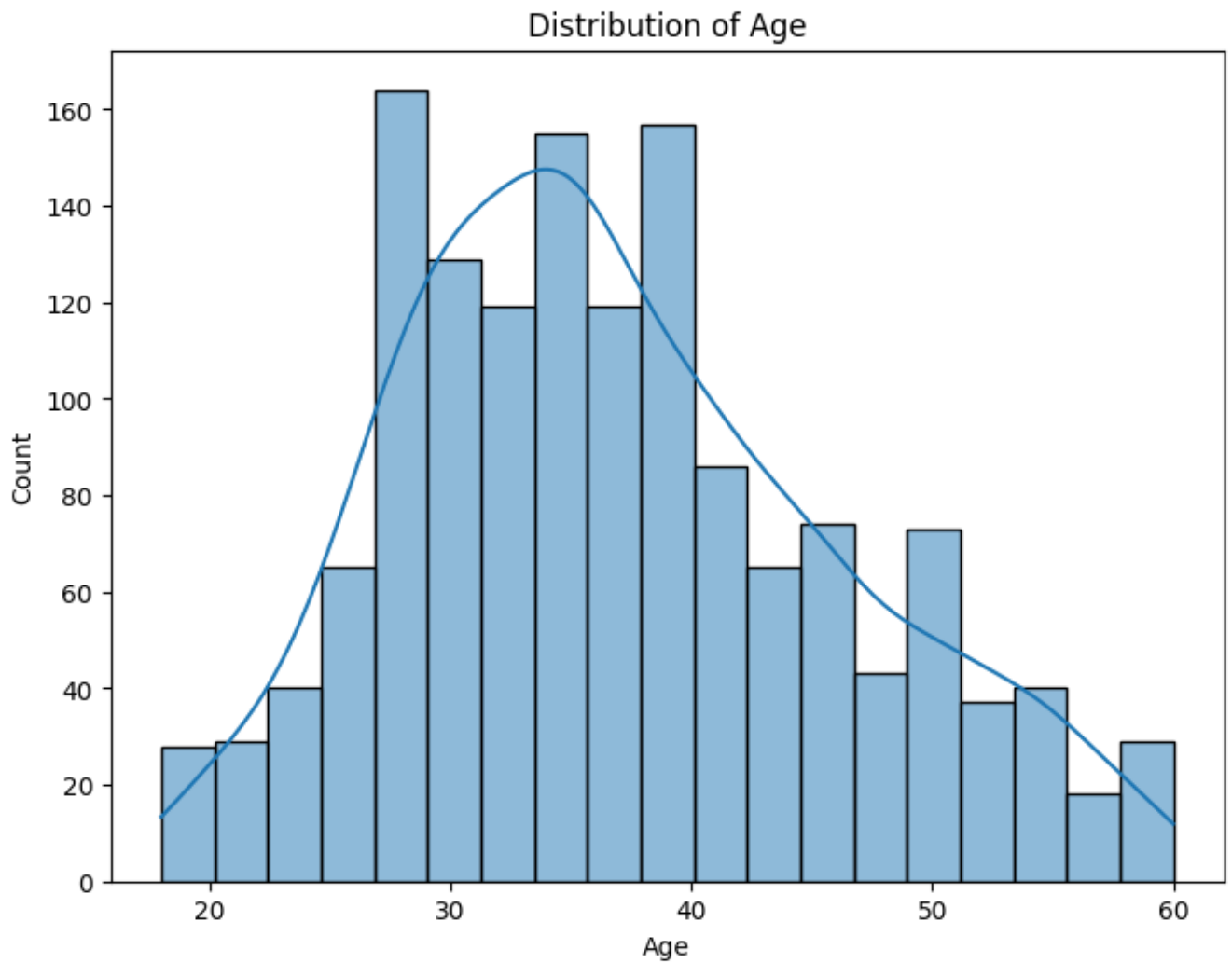
In [16]:

```
plt.figure(figsize=(8, 6))  
sns.countplot(x="Attrition", data=df)  
plt.title("Attrition Count")  
plt.show()
```

In [17]:

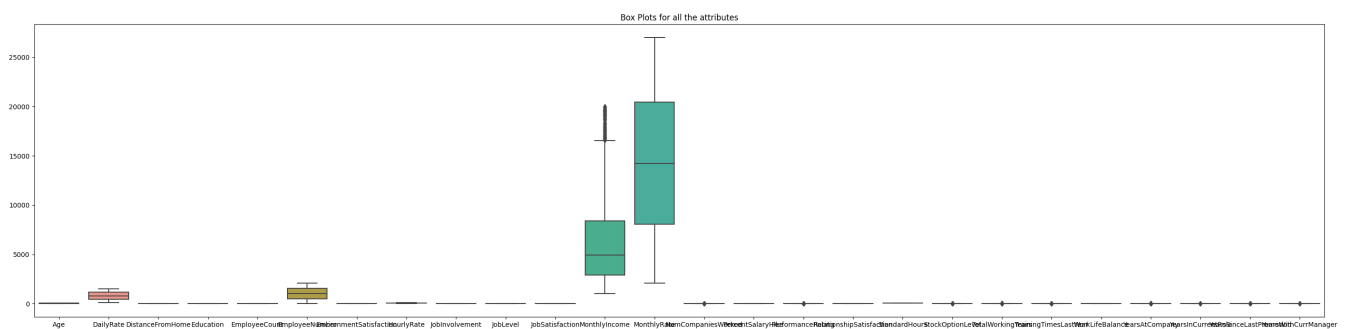
```
plt.figure(figsize=(8, 6))  
sns.histplot(data=df, x="Age", kde=True)  
plt.title("Distribution of Age")  
plt.show()
```



Outlier Detection

In [18]:

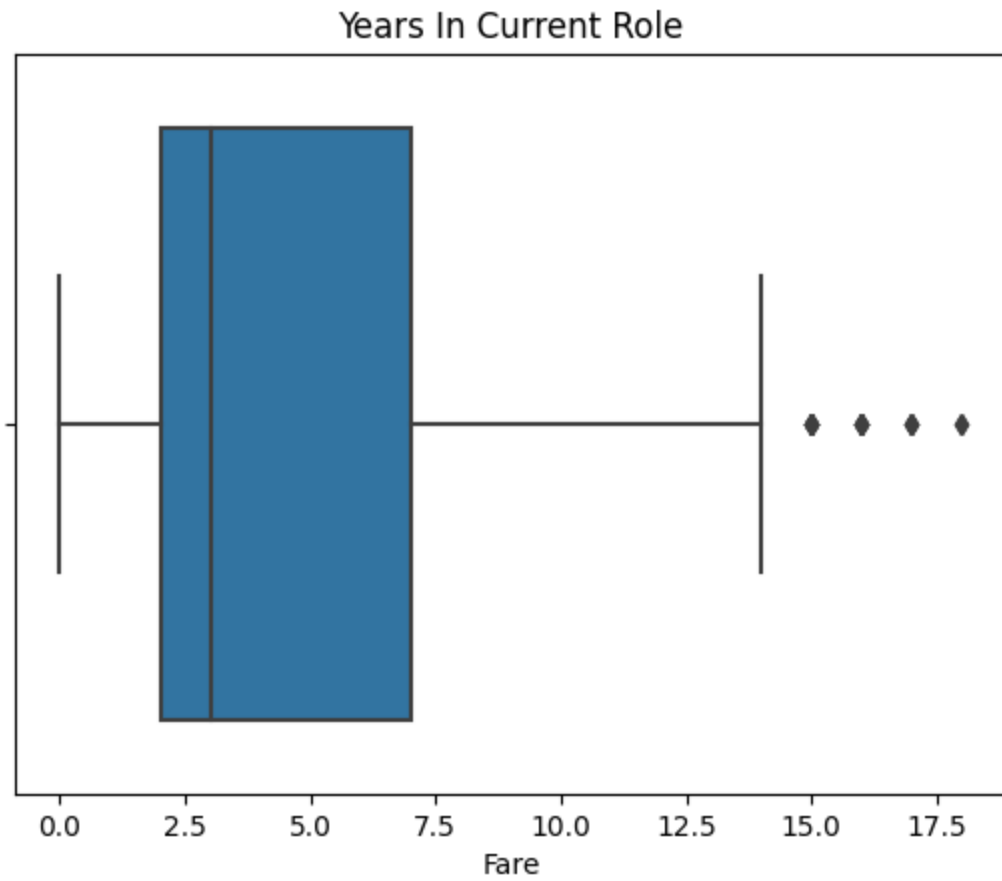
```
plt.figure(figsize=(35, 8))
sns.boxplot(data=df)
plt.title('Box Plots for all the attributes')
plt.show()
```



In [19]:

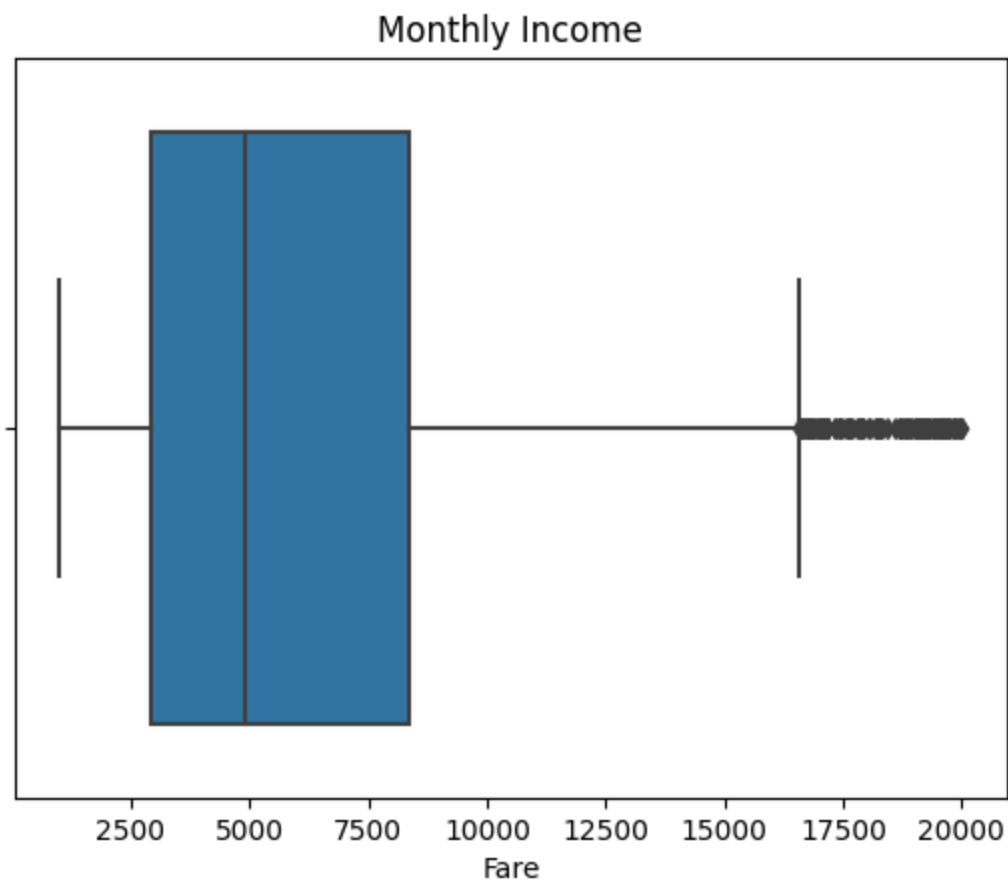
```
sns.boxplot(data=df, x='YearsInCurrentRole')
plt.title('Years In Current Role')
plt.xlabel('Fare')
```

```
plt.show()
```



In [20]:

```
sns.boxplot(data=df, x='MonthlyIncome')  
plt.title('Monthly Income')  
plt.xlabel('Fare')  
plt.show()
```



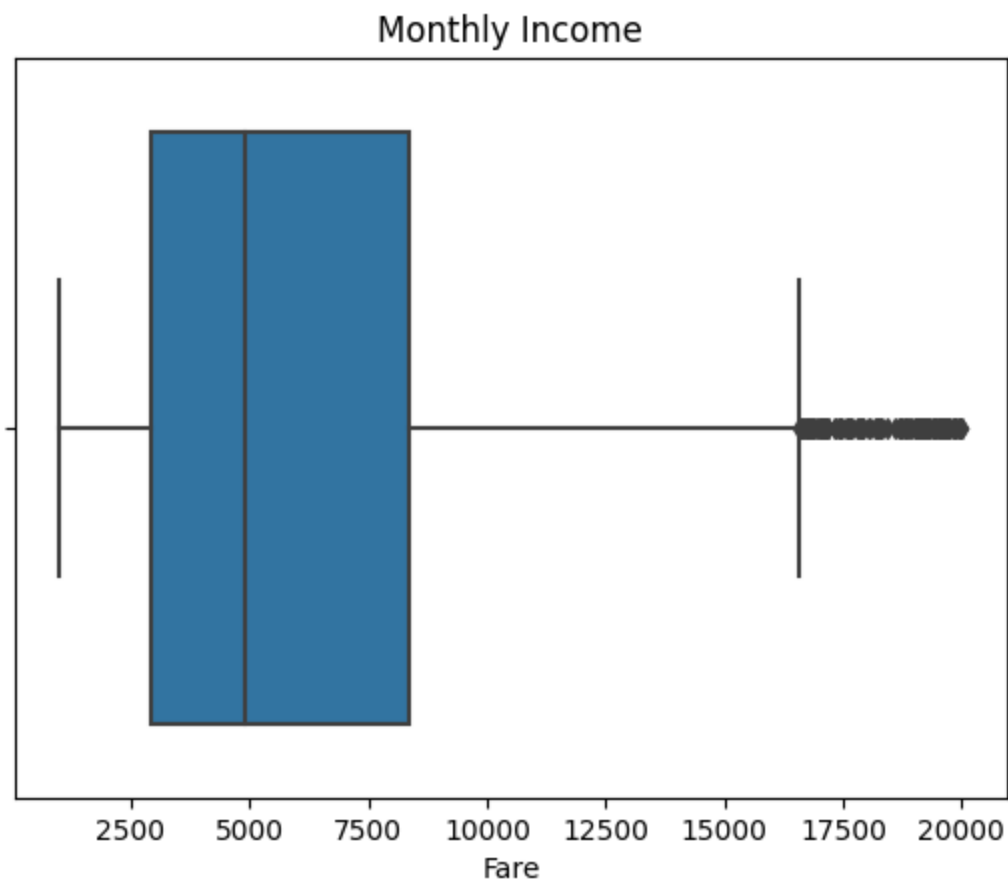
In [21]:

```
from scipy import stats

z_scores = stats.zscore(df['MonthlyIncome'])
z_score_threshold = 3
df_cleaned = df[(np.abs(z_scores) <= z_score_threshold)]
```

In [22]:

```
sns.boxplot(data=df_cleaned, x='MonthlyIncome')
plt.title('Monthly Income')
plt.xlabel('Fare')
plt.show()
```



So the outliers are in large quantity, and they are inside the threshold, so let us not remove the outliers

SPLITTING INDEPENDENT AND DEPENDENT VARIABLES

In [23]:

```
x= df.drop(columns=["Attrition"])
y = df["Attrition"]
```

In [24]:

```
x.head()
```

Out[24]:

	Age	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCou
0	41	Travel_Rarely	1102	Sales	1	2	Life Sciences	1
1	49	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1
2	37	Travel_Rarely	1373	Research & Development	2	2	Other	1
3	33	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1
4	27	Travel_Rarely	591	Research & Development	2	1	Medical	1



In [25]:

```
y.head()
```

Out[25]:

```
0    Yes
1     No
2    Yes
3     No
4     No
Name: Attrition, dtype: object
```

ENCODING

In [26]:

```
categorical_features = x.select_dtypes(include=
['object']).columns.tolist()
x_encoded = pd.get_dummies(x, columns=categorical_features,
drop_first=True)
```

In [27]:

```
x_encoded.head()
```

Out[27]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction
0	41	1102	1	2	1	1	2
1	49	279	8	1	1	2	3
2	37	1373	2	2	1	4	4
3	33	1392	3	4	1	5	4
4	27	591	2	1	1	7	1

5 rows × 47 columns



FEATURE SCALING

In [28]:

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
x_scaled = pd.DataFrame(scaler.fit_transform(x_encoded),
                        columns=x_encoded.columns)
```

In [29]:

```
x_scaled.head()
```

Out[29]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfac
0	0.446350	0.742527	-1.010909	-0.891688	0.0	-1.701283	-0.660531
1	1.322365	-1.297775	-0.147150	-1.868426	0.0	-1.699621	0.254625
2	0.008343	1.414363	-0.887515	-0.891688	0.0	-1.696298	1.169781
3	-0.429664	1.461466	-0.764121	1.061787	0.0	-1.694636	1.169781
4	-1.086676	-0.524295	-0.887515	-1.868426	0.0	-1.691313	-1.575686

5 rows × 47 columns



In [30]:

```
x=x_scaled
```

Train and test split

In [31]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,
                                                    random_state=42)
```

MODEL BUILDING

In [32]:

```
# Import the necessary libraries
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report,
```

```
confusion_matrix
from joblib import dump
```

In [33]:

```
logreg_model = LogisticRegression(random_state=42)
dt_model = DecisionTreeClassifier(random_state=42)
```

In [34]:

```
logreg_model.fit(x_train, y_train)
dt_model.fit(x_train, y_train)
```

Out[34]:

```
DecisionTreeClassifier(random_state=42)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

DecisionTreeClassifier

```
DecisionTreeClassifier(random_state=42)
```

In [35]:

```
logreg_predictions = logreg_model.predict(x_test)

dt_predictions = dt_model.predict(x_test)

logreg_accuracy = accuracy_score(y_test, logreg_predictions)
print("Logistic Regression Accuracy:", logreg_accuracy)

dt_accuracy = accuracy_score(y_test, dt_predictions)
print("Decision Tree Accuracy:", dt_accuracy)

logreg_report = classification_report(y_test, logreg_predictions)
print("Classification Report for Logistic Regression:\n", logreg_report)

dt_report = classification_report(y_test, dt_predictions)
print("Classification Report for Decision Tree Classifier:\n",
      dt_report)

logreg_conf_matrix = confusion_matrix(y_test, logreg_predictions)
print("Confusion Matrix for Logistic Regression:\n", logreg_conf_matrix)
```

```
dt_conf_matrix = confusion_matrix(y_test, dt_predictions)
```



```
print("Confusion Matrix for Decision Tree Classifier:\n",
dt_conf_matrix)
```

Logistic Regression Accuracy: 0.8809523809523809

Decision Tree Accuracy: 0.7721088435374149

Classification Report for Logistic Regression:

	precision	recall	f1-score	support
No	0.92	0.95	0.93	255
Yes	0.56	0.46	0.51	39
accuracy			0.88	294
macro avg	0.74	0.70	0.72	294
weighted avg	0.87	0.88	0.88	294

Classification Report for Decision Tree Classifier:

	precision	recall	f1-score	support
No	0.87	0.86	0.87	255
Yes	0.17	0.18	0.17	39
accuracy			0.77	294
macro avg	0.52	0.52	0.52	294
weighted avg	0.78	0.77	0.78	294

Confusion Matrix for Logistic Regression:

```
[[241  14]
```

```
[ 21  18]]
```

Confusion Matrix for Decision Tree Classifier:

```
[[220  35]
```

```
[ 32   7]]
```