Students adaptability in online education

Importing all the modules

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import plotly.express as px
import statistics as s
```

```
df=pd.read_csv('/content/students_adaptability_level_online_education.csv')
```

```
df
```

	Gender	Age	Education Level	Institution Type	IT Student	Location	Load-shedding	Financial Condition	Internet Type	Network Type	Class Duration	Se
0	Boy	21-25	University	Non Government	No	Yes	Low	Mid	Wifi	4G	3-6	
1	Girl	21-25	University	Non Government	No	Yes	High	Mid	Mobile Data	4G	1-3	
2	Girl	16-20	College	Government	No	Yes	Low	Mid	Wifi	4G	1-3	
3	Girl	11-15	School	Non Government	No	Yes	Low	Mid	Mobile Data	4G	1-3	
4	Girl	16-20	School	Non Government	No	Yes	Low	Poor	Mobile Data	3G	0	
...	...	...	...	...	...	...	...	...	...	...	...	...
1200	Girl	16-20	College	Non Government	No	Yes	Low	Mid	Wifi	4G	1-3	
1201	Girl	16-20	College	Non Government	No	No	High	Mid	Wifi	4G	3-6	
1202	Boy	11-15	School	Non Government	No	Yes	Low	Mid	Mobile Data	3G	1-3	
1203	Girl	16-20	College	Non Government	No	No	Low	Mid	Wifi	4G	1-3	
1204	Girl	11-15	School	Non Government	No	Yes	Low	Poor	Mobile Data	3G	1-3	

1205 rows × 14 columns

```
df.shape
```

```
(1205, 14)
```

```
df.loc[100]
```

```
Gender                          Boy
Age                           11-15
Education Level              School
Institution Type       Non Government
IT Student                         No
Location                          Yes
Load-shedding                     Low
Financial Condition               Mid
Internet Type                    Wifi
Network Type                       4G
Class Duration                      0
Self Lms                           No
```

```
Device                          Mobile
Adaptivity Level                   Low
Name: 100, dtype: object
```

```
df.size
```

```
16870
```

```
df.dtypes
```

```
Gender                 object
Age                    object
Education Level        object
Institution Type       object
IT Student             object
Location               object
Load-shedding          object
Financial Condition    object
Internet Type          object
Network Type           object
Class Duration         object
Self Lms               object
Device                 object
Adaptivity Level       object
dtype: object
```

```
df.sample()
```

	Gender	Age	Education Level	Institution Type	IT Student	Location	Load-shedding	Financial Condition	Internet Type	Network Type	Class Duration	Se... L...
533	Boy	11-15	School	Non Government	No	Yes	Low	Poor	Mobile Data	3G	1-3	M

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1205 entries, 0 to 1204
Data columns (total 14 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Gender               1205 non-null   object
 1   Age                  1205 non-null   object
 2   Education Level      1205 non-null   object
 3   Institution Type     1205 non-null   object
 4   IT Student           1205 non-null   object
 5   Location             1205 non-null   object
 6   Load-shedding        1205 non-null   object
 7   Financial Condition  1205 non-null   object
 8   Internet Type        1205 non-null   object
 9   Network Type         1205 non-null   object
 10  Class Duration       1205 non-null   object
 11  Self Lms             1205 non-null   object
 12  Device               1205 non-null   object
 13  Adaptivity Level     1205 non-null   object
dtypes: object(14)
memory usage: 131.9+ KB
```

```
df.nunique()
```

```
Gender                 2
Age                    6
Education Level        3
Institution Type       2
IT Student             2
Location               2
Load-shedding          2
Financial Condition    3
Internet Type          2
Network Type           3
Class Duration         3
Self Lms               2
```

```
Device                3
Adaptivity Level      3
dtype: int64
```

```
df.isna()
```

	Gender	Age	Education Level	Institution Type	IT Student	Location	Load-shedding	Financial Condition	Internet Type	Network Type	Class Duration	S
0	False	False	False	False	False	False	False	False	False	False	False	I
1	False	False	False	False	False	False	False	False	False	False	False	I
2	False	False	False	False	False	False	False	False	False	False	False	I
3	False	False	False	False	False	False	False	False	False	False	False	I
4	False	False	False	False	False	False	False	False	False	False	False	I
...	...	...	...	...	...	...	...	...	...	...	...	
1200	False	False	False	False	False	False	False	False	False	False	False	I
1201	False	False	False	False	False	False	False	False	False	False	False	I
1202	False	False	False	False	False	False	False	False	False	False	False	I
1203	False	False	False	False	False	False	False	False	False	False	False	I
1204	False	False	False	False	False	False	False	False	False	False	False	I

1205 rows × 14 columns

```
df.isna().sum()
```

```
Gender                0
Age                   0
Education Level       0
Institution Type      0
IT Student            0
Location              0
Load-shedding         0
Financial Condition   0
Internet Type         0
Network Type          0
Class Duration        0
Self Lms              0
Device                0
Adaptivity Level      0
dtype: int64
```

```
df.describe().transpose()
```

	count	unique	top	freq
Gender	1205	2	Boy	663
Age	1205	6	21-25	374
Education Level	1205	3	School	530
Institution Type	1205	2	Non Government	823
IT Student	1205	2	No	901
Location	1205	2	Yes	935
Load-shedding	1205	2	Low	1004
Financial Condition	1205	3	Mid	878
Internet Type	1205	2	Mobile Data	695
Network Type	1205	3	4G	775
Class Duration	1205	3	1-3	840
Self Lms	1205	2	No	995
Device	1205	3	Mobile	1013
Adaptivity Level	1205	3	Moderate	625

```
df['Adaptivity Level'].describe().transpose()
```

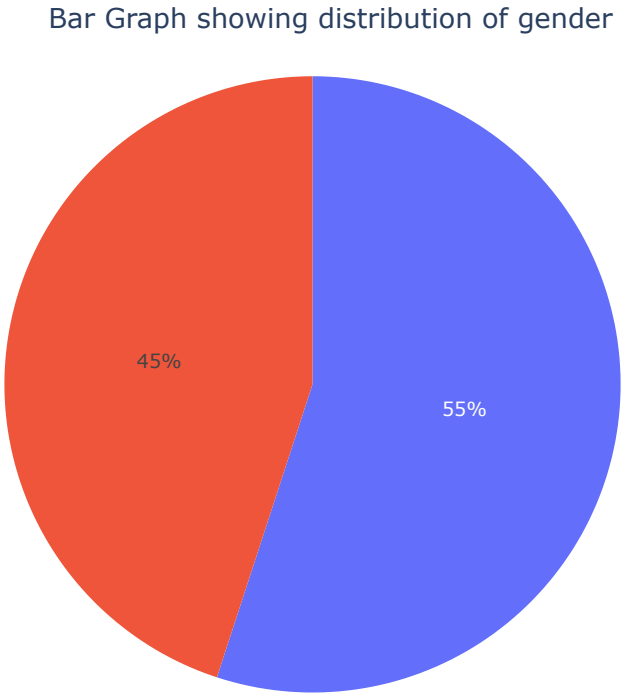
```
count         1205
unique           3
top       Moderate
freq           625
Name: Adaptivity Level, dtype: object
```

```
for i in df.columns:
  print(f"{i},':',{df[i].unique().tolist()}")
```

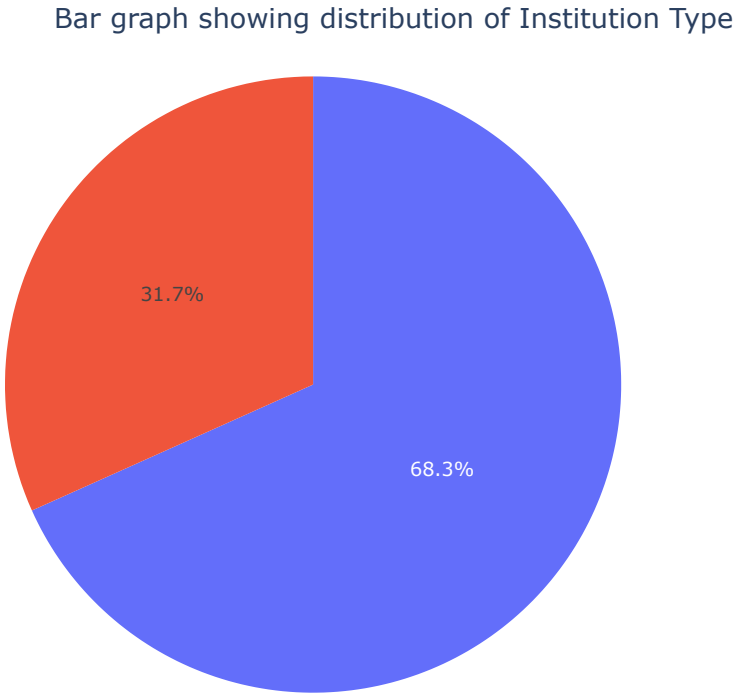
```
Gender,':',['Boy', 'Girl']
Age,':',['21-25', '16-20', '11-15', '26-30', '6-10', '1-5']
Education Level,':',['University', 'College', 'School']
Institution Type,':',['Non Government', 'Government']
IT Student,':',['No', 'Yes']
Location,':',['Yes', 'No']
Load-shedding,':',['Low', 'High']
Financial Condition,':',['Mid', 'Poor', 'Rich']
Internet Type,':',['Wifi', 'Mobile Data']
Network Type,':',['4G', '3G', '2G']
Class Duration,':',['3-6', '1-3', '0']
Self Lms,':',['No', 'Yes']
Device,':',['Tab', 'Mobile', 'Computer']
Adaptivity Level,':',['Moderate', 'Low', 'High']
```

Univarient Analysis

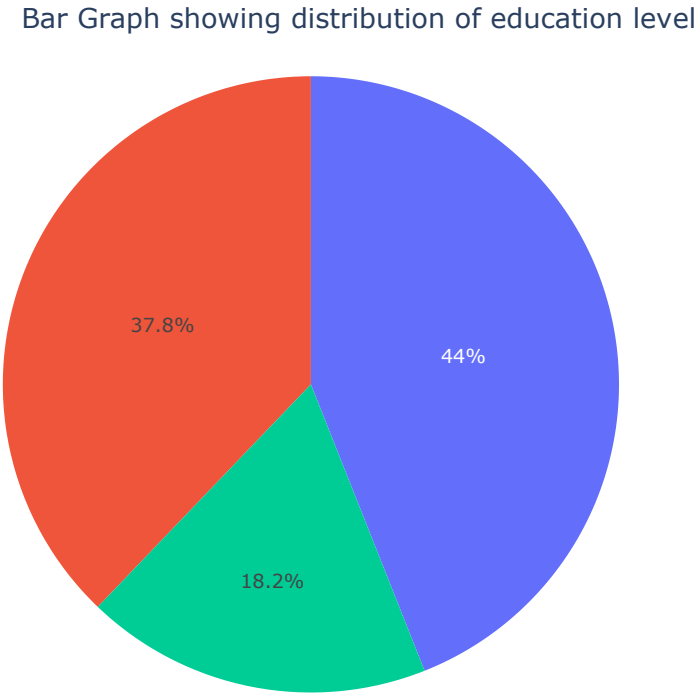
```
plot=px.pie(df,'Gender')
plot.update_layout(title='Bar Graph showing distribution of gender' ,title_x=0.5)
```



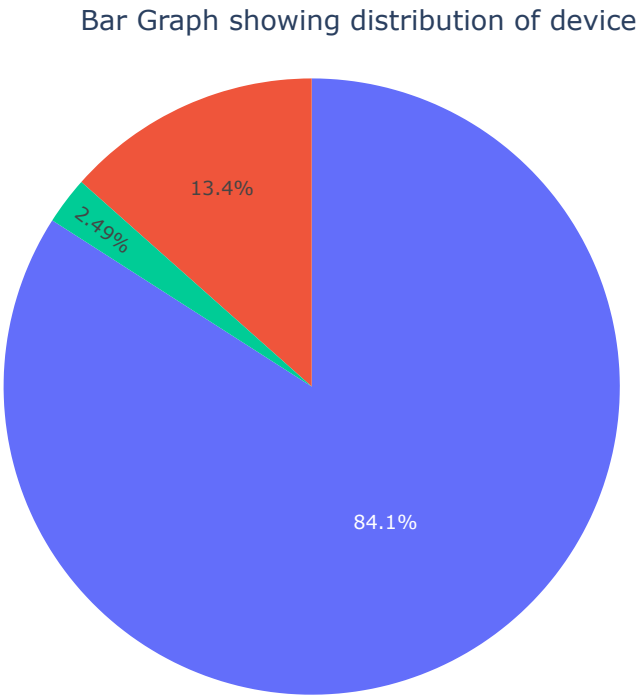

```
plot=px.pie(df,'Institution Type')
plot.update_layout(title='Bar graph showing distribution of Institution Type' ,title_x=0.5)
```




```
plot=px.pie(df,'Education Level')
plot.update_layout(title='Bar Graph showing distribution of education level' ,title_x=0.5)
```

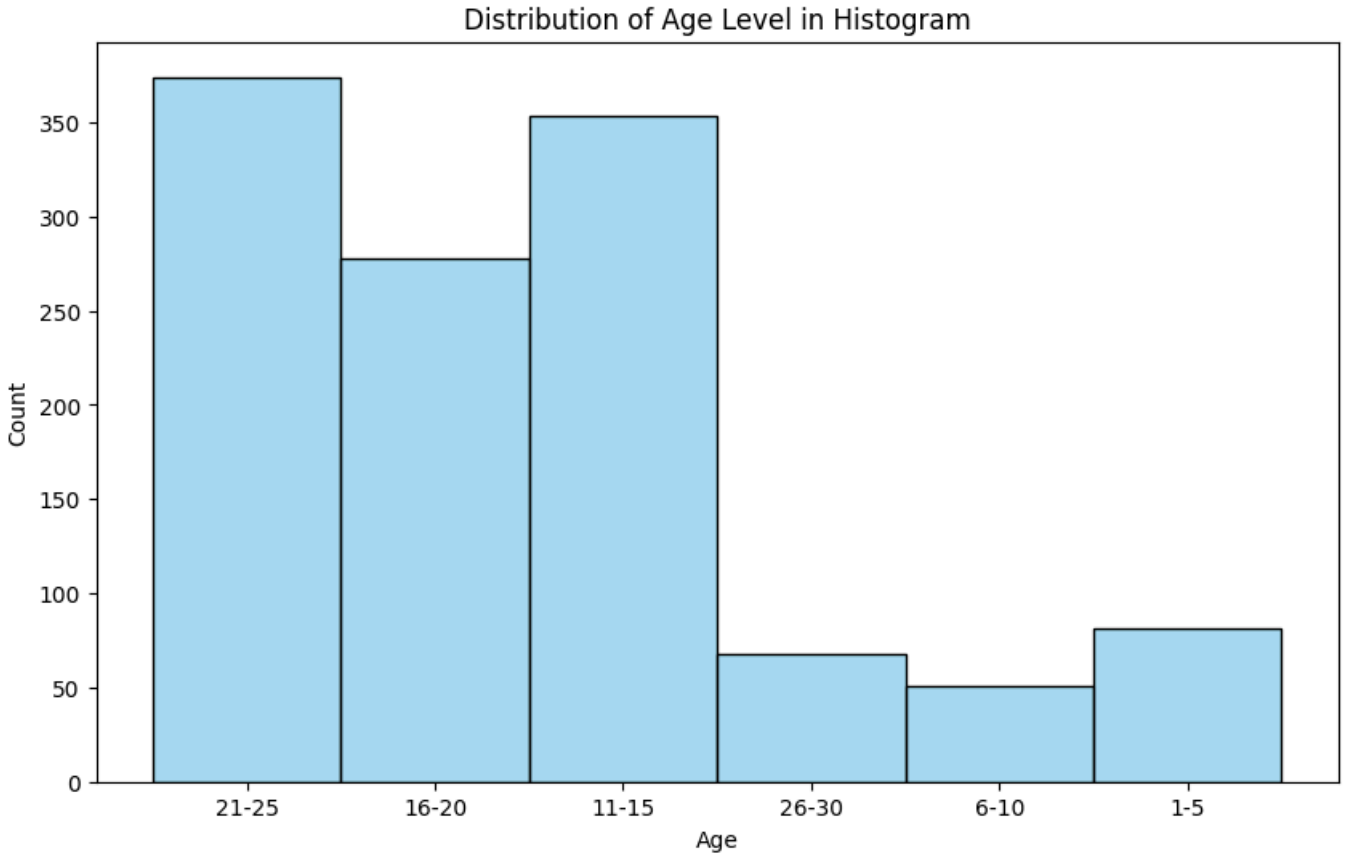



```
plot=px.pie(df,'Device')
plot.update_layout(title='Bar Graph showing distribution of device' ,title_x=0.5)
```

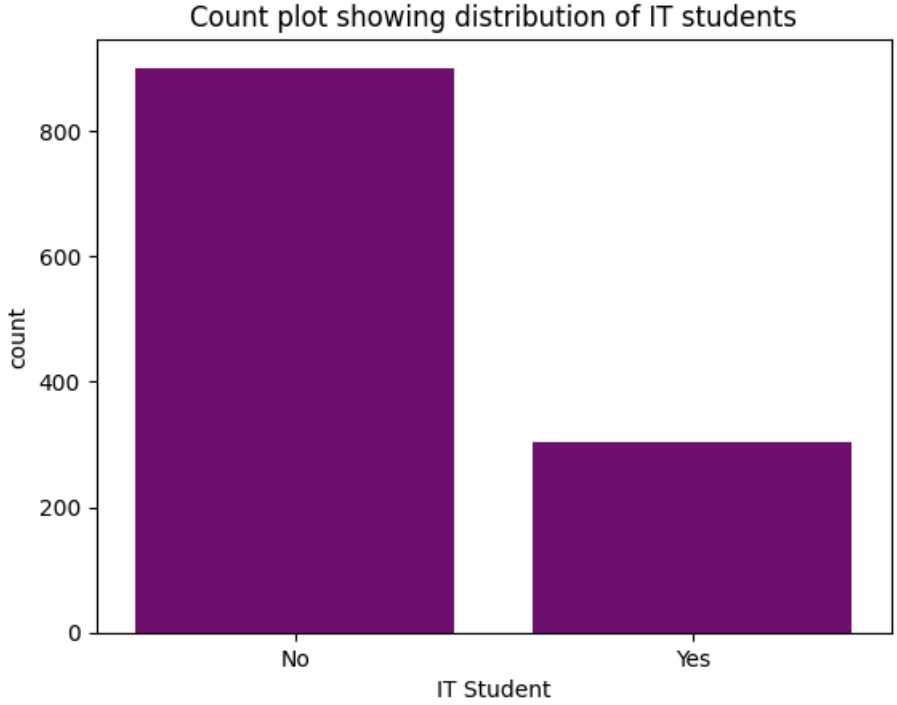



```
plt.figure(figsize=(10, 6))
sns.histplot(df['Age'],color='skyblue')
plt.title('Distribution of Age Level in Histogram')
```

```
Text(0.5, 1.0, 'Distribution of Age Level in Histogram')
```

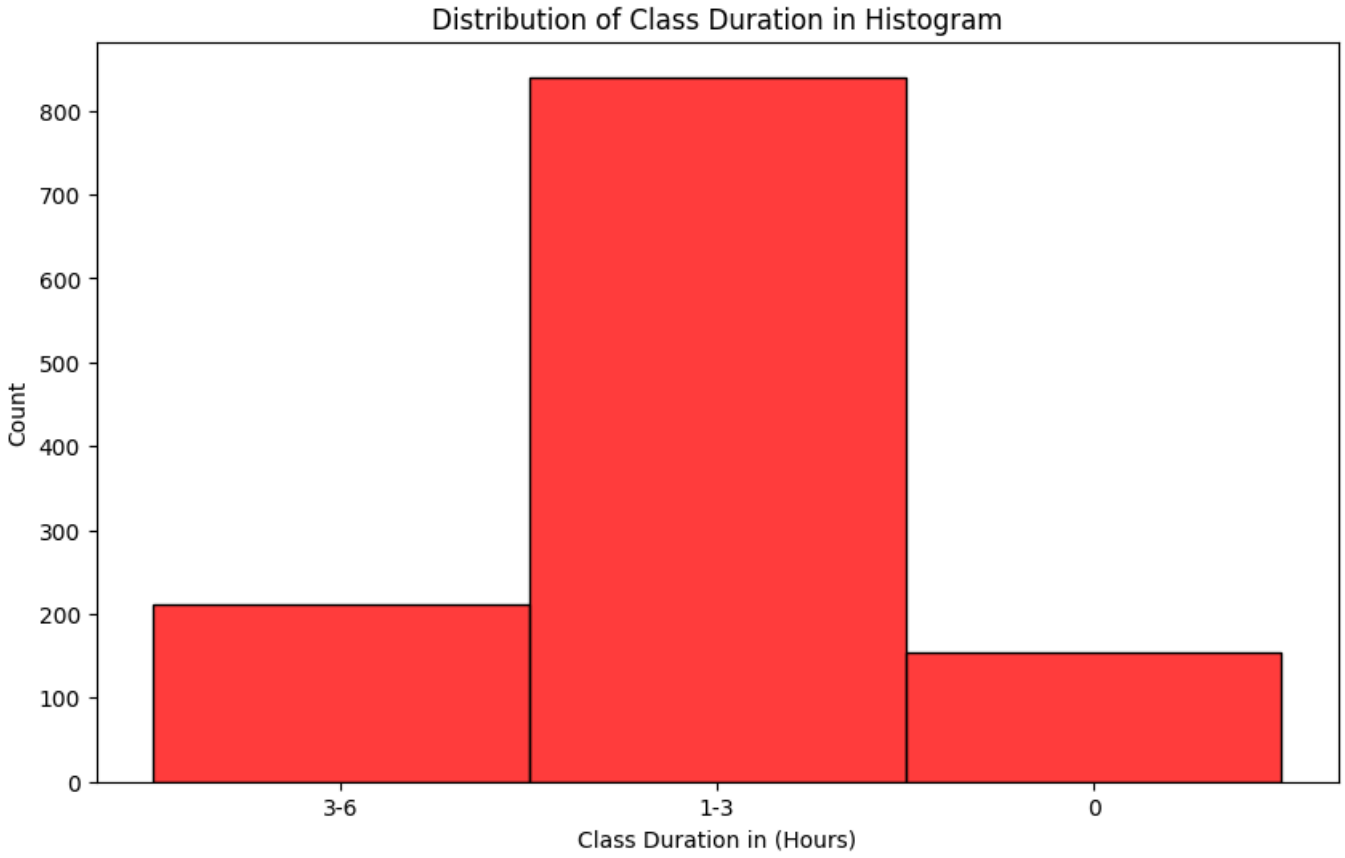



```
sns.countplot( x= 'IT Student' ,data=df, linewidth=2,color='purple')
plt.title("Count plot showing distribution of IT students")
plt.show()
```

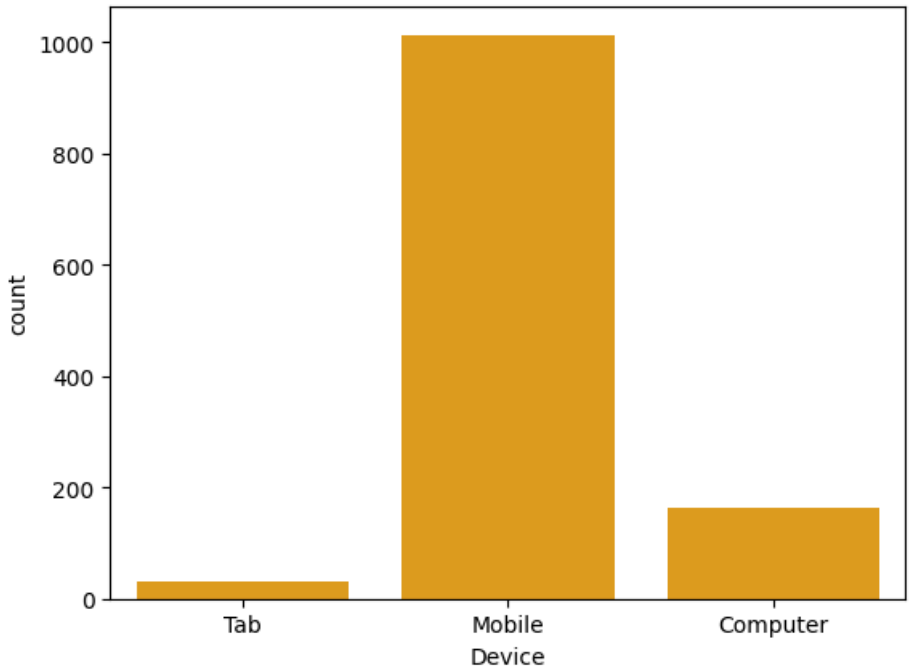



```
plt.figure(figsize=(10, 6))
sns.histplot(df['Class Duration'],color='red')
plt.xlabel('Class Duration in (Hours)')
plt.title('Distribution of Class Duration in Histogram')
```

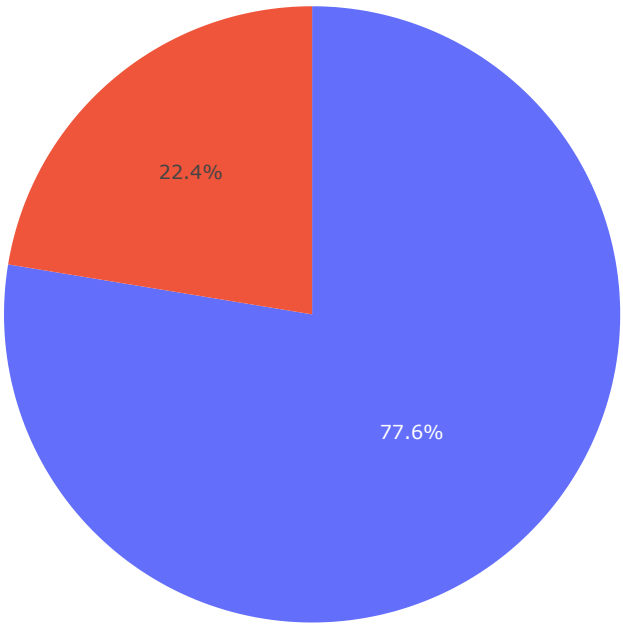
```
Text(0.5, 1.0, 'Distribution of Class Duration in Histogram')
```



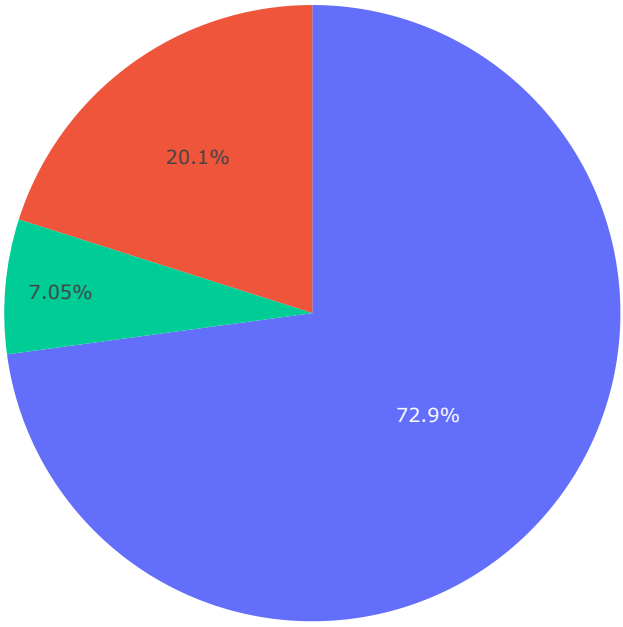

```
sns.countplot( x= 'Device' ,data=df, linewidth=2,color='orange')
plt.show()
```



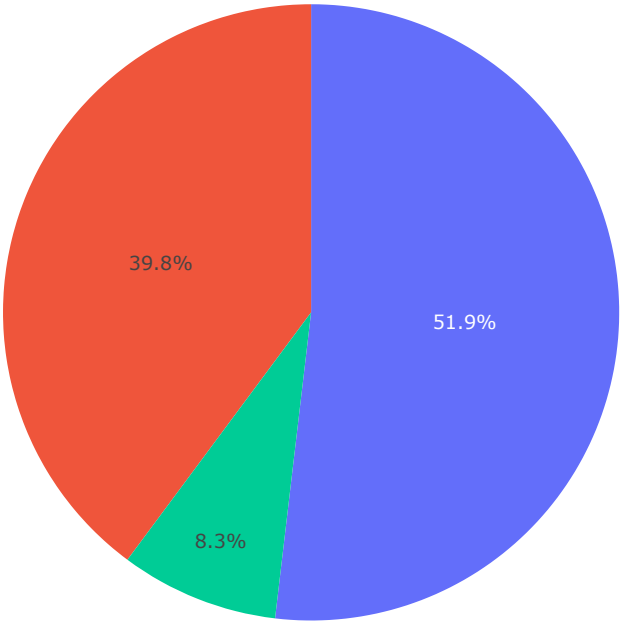

```
plot=px.pie(df,'Location')
plot.update_layout(title='Bar Graph Showing distribution of Location' ,title_x=0.5)
```




```
px.pie(df,'Financial Condition')
```

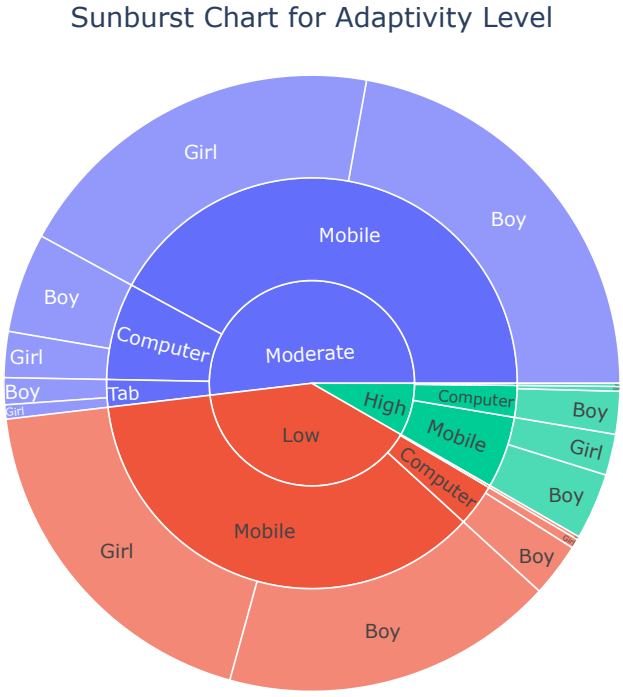



```
px.pie(df,'Adaptivity Level')
```



Multivarient Analysis

```
plot=px. sunburst(
    df,
    path=['Adaptivity Level', 'Device', 'Gender'],
    color_discrete_map={'Mid': 'blue', 'Other': 'red'},
)
plot.update_layout(title='Sunburst Chart for Adaptivity Level' ,title_x=0.5)
```



Double-click (or enter) to edit

```
df.columns.tolist()
```

```
['Gender',
 'Age',
 'Education Level',
 'Institution Type',
 'IT Student',
 'Location',
 'Load-shedding',
 'Financial Condition',
 'Internet Type',
 'Network Type',
 'Class Duration',
 'Self Lms',
 'Device',
 'Adaptivity Level']
```

Encoding

Since data are composed of categorical values,we use LabelEncoder() to encode into numeric values

```
from sklearn.preprocessing import LabelEncoder


data=df

label_encoders = {}
categorical_columns = data.columns

for column in categorical_columns:
    label_encoders[column] = LabelEncoder()
    data[column] = label_encoders[column].fit_transform(data[column])



df.head()
```

	Gender	Age	Education Level	Institution Type	IT Student	Location	Load-shedding	Financial Condition	Internet Type	Network Type	Class Duration	Self Lms
0	0	3	2	1	0	1	1	0	1	2	2	0
1	1	3	2	1	0	1	0	0	0	2	1	1
2	1	2	0	0	0	1	1	0	1	2	1	0
3	1	1	1	1	0	1	1	0	0	2	1	0
4	1	2	1	1	0	1	1	1	0	1	0	0

```
from sklearn.model_selection import train_test_split


X=df.drop('Adaptivity Level', axis=1)
y=df['Adaptivity Level']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

print(f"X train {X_train.shape}")
print(f"X test {X_test.shape}")
print(f"Y train {y_train.shape}")
print(f"Y test {y_test.shape}")
```

```
X train (964, 13)
X test (241, 13)
Y train (964,)
Y test (241,)
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score,r2_score
from sklearn.metrics import confusion_matrix,accuracy_score, classification_report,ConfusionMatrixDisplay


log_reg = LogisticRegression(max_iter=1000)

# Train the model
log_reg.fit(X_train, y_train)

# Predict on the test set
y_pred = log_reg.predict(X_test)

# Calculate accuracy
accuracy_log_reg = accuracy_score(y_test, y_pred)

#check accuracy of training and testing
print("Train Accuracy :", accuracy_score(y_train, log_reg.predict(X_train)))
print("Train Confusion Matrix:")
print(confusion_matrix(y_train, log_reg.predict(X_train)))
#plot confusion matrix for training set
plt.figure(figsize=(3, 3))
cm_train = confusion_matrix(y_train, log_reg.predict(X_train))
sns.heatmap(cm_train, annot=True, fmt='d', cmap='Blues', xticklabels=['Predicted 0', 'Predicted 1'], yticklabels=
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix - Training Set')
plt.show()

print("-"*45)

print("Test Accuracy :", accuracy_score(y_test, log_reg.predict(X_test)))
print("Test Confusion Matrix:")
print(confusion_matrix(y_test,log_reg.predict(X_test)))
# Plot the Confusion Matrix for Test Set
plt.figure(figsize=(3, 3))
cm_test = confusion_matrix(y_test, log_reg.predict(X_test))
sns.heatmap(cm_test, annot=True, fmt='d', cmap='Blues', xticklabels=['Predicted 0', 'Predicted 1'], yticklabels=[
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix - Test Set')
plt.show()
```

```
Train Accuracy : 0.6929460580912863
Train Confusion Matrix:
[[ 35  17  25]
```