In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
loan_data = pd.read_csv("loan_train.csv")
```

In [3]:

```
loan_data.head()
```

Out[3]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplic
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	

In [4]:

```
loan_data.tail()
```

Out[4]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapp
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	

In [5]:

```
loan_data.shape
```

Out[5]:

```
(614, 13)
```

In [6]:

```
loan_data.columns
```

Out[6]:

```
Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
       'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmoun
t',
       'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Statu
s'],
      dtype='object')
```

In [7]:

```
loan_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            614 non-null    object
 1   Gender             601 non-null    object
 2   Married            611 non-null    object
 3   Dependents         599 non-null    object
 4   Education          614 non-null    object
 5   Self_Employed      582 non-null    object
 6   ApplicantIncome    614 non-null    int64
 7   CoapplicantIncome  614 non-null    float64
 8   LoanAmount         592 non-null    float64
 9   Loan_Amount_Term   600 non-null    float64
 10  Credit_History     564 non-null    float64
 11  Property_Area      614 non-null    object
 12  Loan_Status        614 non-null    object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.5+ KB
```

In [8]:

```
loan_data.describe()
```

Out[8]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	614.000000	614.000000	592.000000	600.00000	564.000000
mean	5403.459283	1621.245798	146.412162	342.00000	0.842199
std	6109.041673	2926.248369	85.587325	65.12041	0.364878
min	150.000000	0.000000	9.000000	12.00000	0.000000
25%	2877.500000	0.000000	100.000000	360.00000	1.000000
50%	3812.500000	1188.500000	128.000000	360.00000	1.000000
75%	5795.000000	2297.250000	168.000000	360.00000	1.000000
max	81000.000000	41667.000000	700.000000	480.00000	1.000000

In [9]:

```
loan_data.isnull().sum()
```

Out[9]:

```
Loan_ID               0
Gender               13
Married               3
Dependents           15
Education             0
Self_Employed        32
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount           22
Loan_Amount_Term     14
Credit_History       50
Property_Area         0
Loan_Status           0
dtype: int64
```

In [10]:

```
loan_data['Gender']= loan_data['Gender'].map({'Male':0, 'Female':1})
loan_data['Married']= loan_data['Married'].map({'No':0, 'Yes':1})
loan_data['Loan_Status']= loan_data['Loan_Status'].map({'N':0, 'Y':1})
```

In [11]:

```
loan_data
```

Out[11]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapp
0	LP001002	0.0	0.0	0	Graduate	No	5849	
1	LP001003	0.0	1.0	1	Graduate	No	4583	
2	LP001005	0.0	1.0	0	Graduate	Yes	3000	
3	LP001006	0.0	1.0	0	Not Graduate	No	2583	
4	LP001008	0.0	0.0	0	Graduate	No	6000	
...	...	...	...	...	...	...	...	
609	LP002978	1.0	0.0	0	Graduate	No	2900	
610	LP002979	0.0	1.0	3+	Graduate	No	4106	
611	LP002983	0.0	1.0	1	Graduate	No	8072	
612	LP002984	0.0	1.0	2	Graduate	No	7583	
613	LP002990	1.0	0.0	0	Graduate	Yes	4583	

614 rows × 13 columns

In [12]:

```
loan_data['Gender'] = loan_data['Gender'].fillna(loan_data['Gender'].mode()[0])
```

In [13]:

```
loan_data['Married'] = loan_data['Married'].fillna(loan_data['Married'].mode()[0])
```

In [14]:

```
loan_data['Dependents'] = loan_data['Dependents'].fillna(loan_data['Dependents'].mode()
```

In [15]:

```
loan_data['Self_Employed'].fillna('No',inplace=True)
```

In [17]:

```
loan_data['Credit_History'] = loan_data['Credit_History'].fillna(loan_data['Credit_Histo
```

In [18]:

```
loan_data['LoanAmount'] = loan_data['LoanAmount'].fillna(loan_data['LoanAmount'].median(
```

In [19]:

```
loan_data['Loan_Amount_Term'] = loan_data['Loan_Amount_Term'].fillna(loan_data['Loan_Amo
```

In [20]:

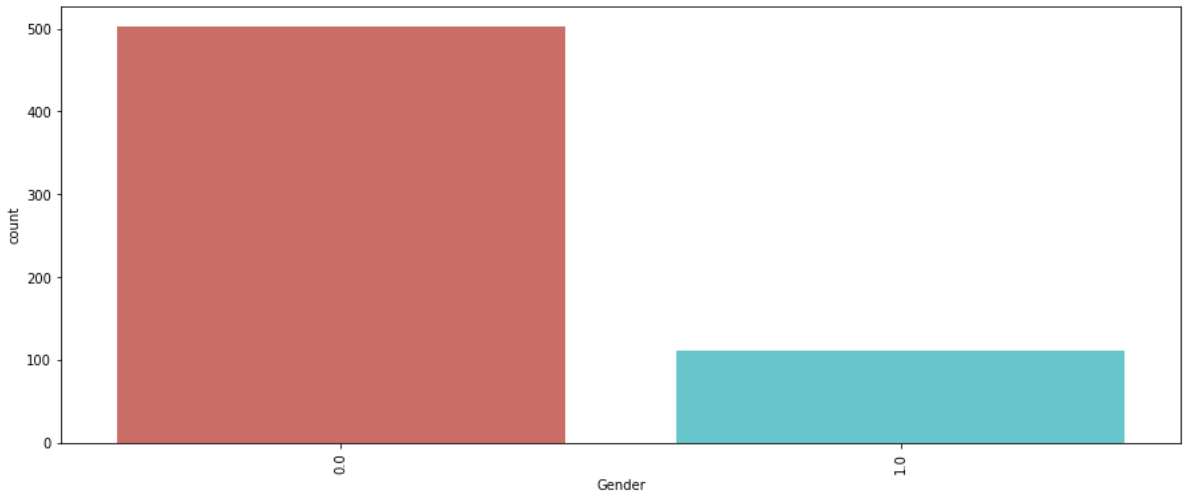
```
loan_data['Gender'].value_counts()
```

Out[20]:

```
0.0    502
1.0    112
Name: Gender, dtype: int64
```

In [21]:

```
plt.figure(figsize=(15,6))
sns.countplot('Gender', data = loan_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [22]:

```
loan_data['LoanAmount'].value_counts()
```

Out[22]:

```
128.0    33
120.0    20
110.0    17
100.0    15
160.0    12
         ..
240.0     1
214.0     1
59.0      1
166.0     1
253.0     1
Name: LoanAmount, Length: 203, dtype: int64
```

In [23]:

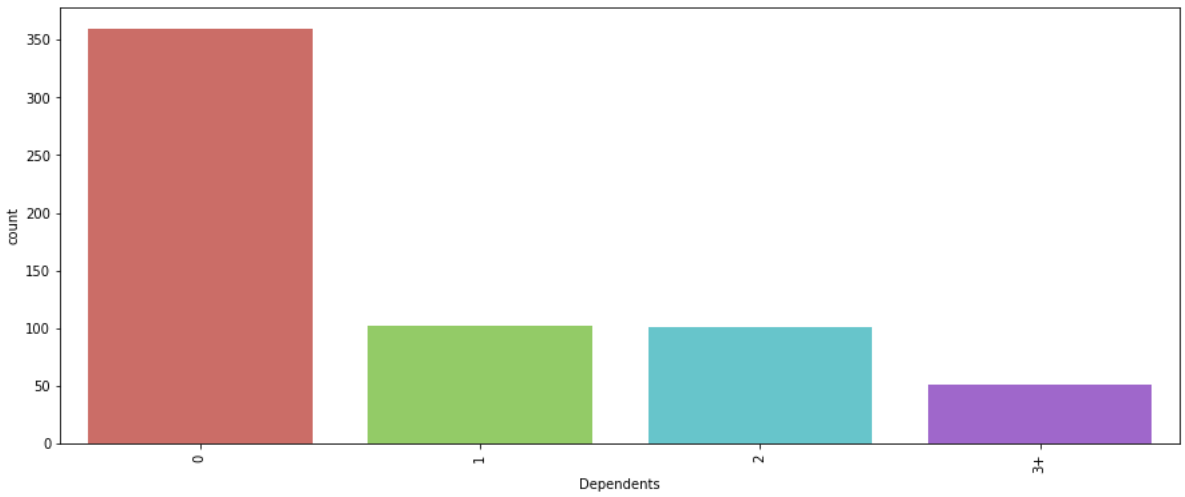
```
loan_data['Dependents'].value_counts()
```

Out[23]:

```
0     360
1     102
2     101
3+     51
Name: Dependents, dtype: int64
```

In [26]:

```
plt.figure(figsize=(15,6))
sns.countplot('Dependents', data = loan_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [24]:

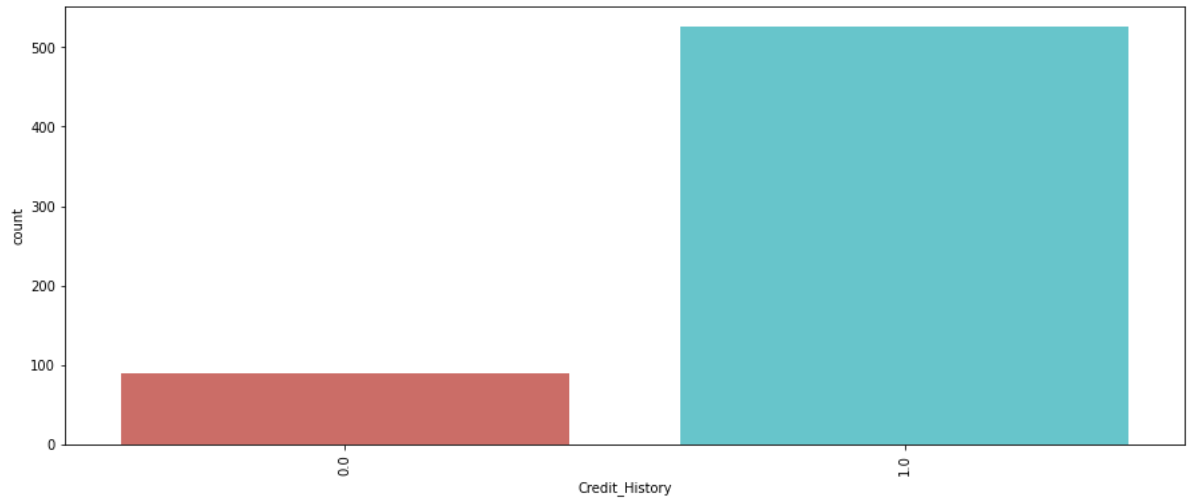
```
loan_data['Credit_History'].value_counts()
```

Out[24]:

```
1.0    525
0.0     89
Name: Credit_History, dtype: int64
```

In [28]:

```
plt.figure(figsize=(15,6))
sns.countplot('Credit_History', data = loan_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [25]:

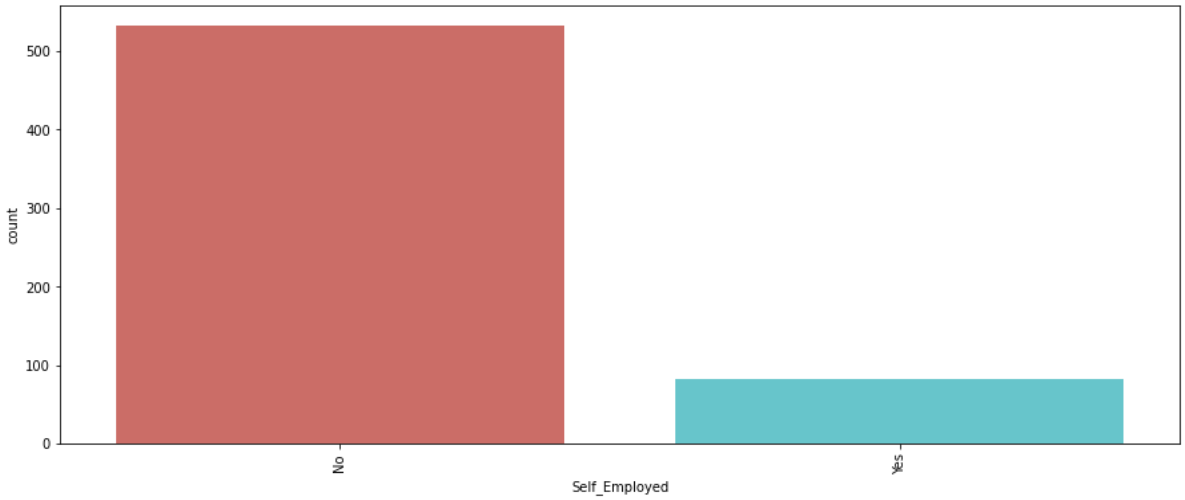
```
loan_data['Self_Employed'].value_counts()
```

Out[25]:

```
No     532
Yes     82
Name: Self_Employed, dtype: int64
```

In [26]:

```
plt.figure(figsize=(15,6))
sns.countplot('Self_Employed', data = loan_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [27]:

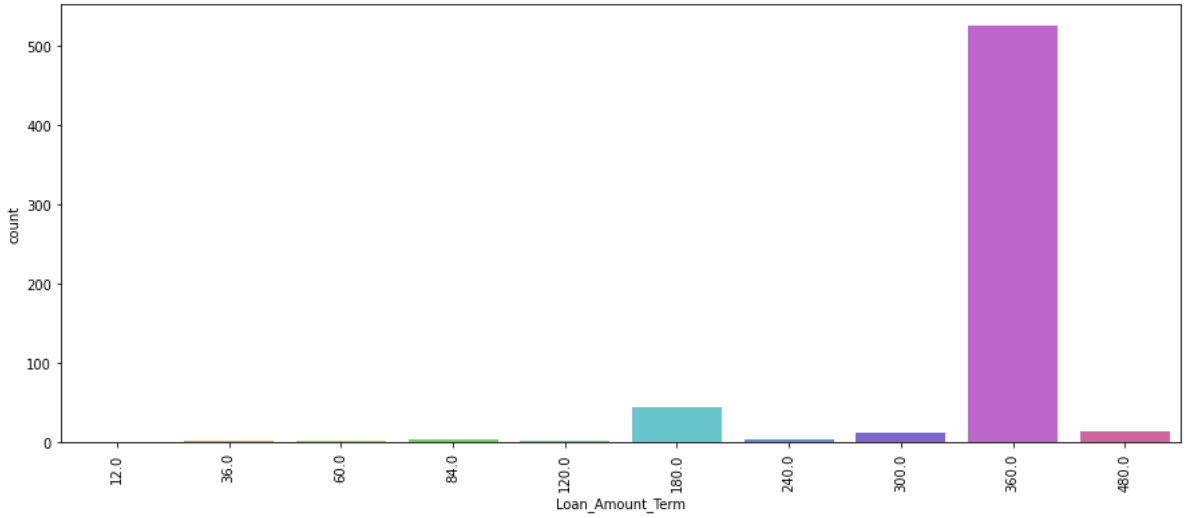
```
loan_data['Loan_Amount_Term'].value_counts()
```

Out[27]:

```
360.0    526
180.0     44
480.0     15
300.0     13
240.0      4
84.0       4
120.0      3
60.0       2
36.0       2
12.0       1
Name: Loan_Amount_Term, dtype: int64
```

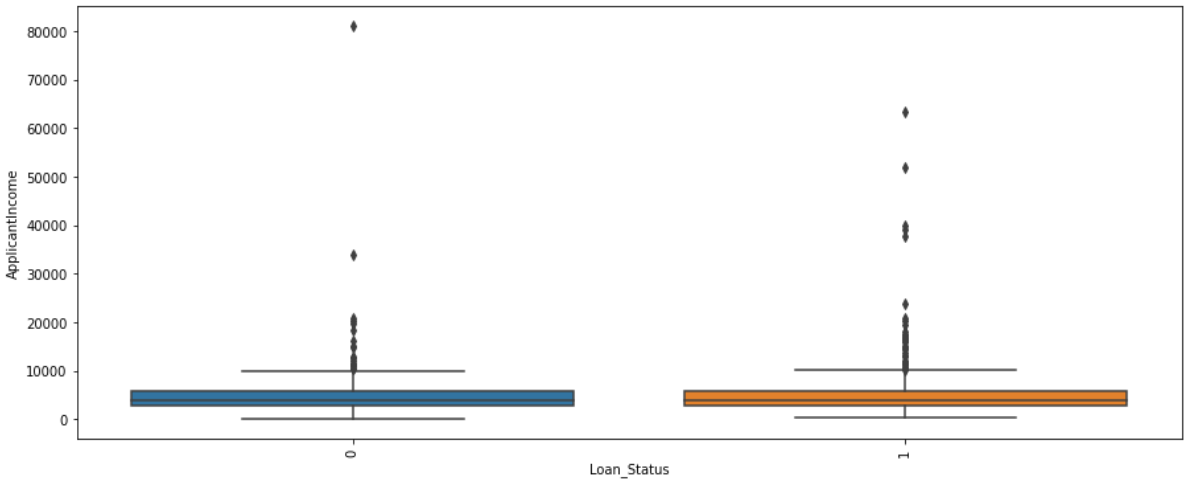
In [28]:

```
plt.figure(figsize=(15,6))
sns.countplot('Loan_Amount_Term', data = loan_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



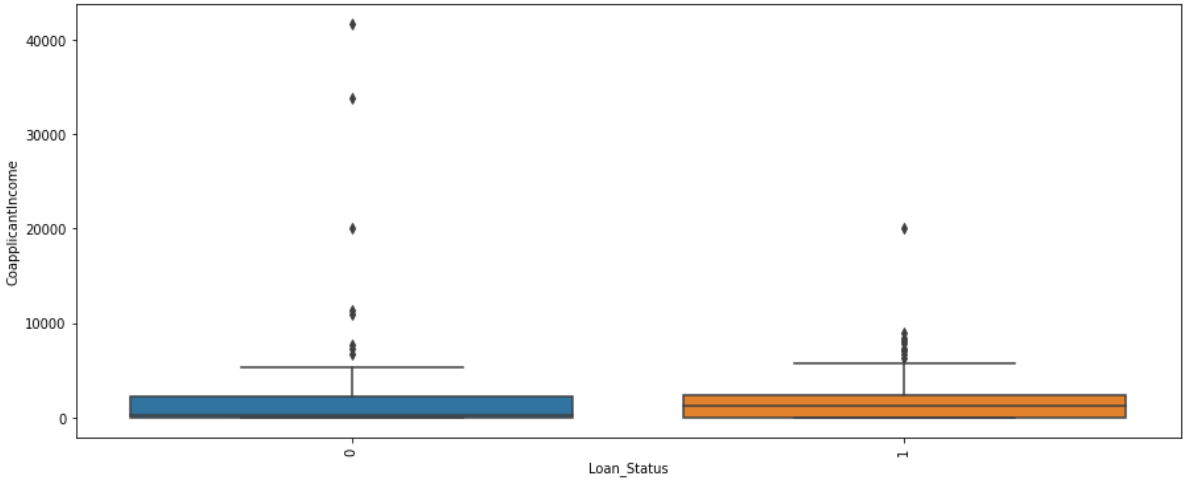
In [29]:

```
plt.figure(figsize=(15,6))
sns.boxplot(x = 'Loan_Status', y = 'ApplicantIncome', data = loan_data)
plt.xticks(rotation = 90)
plt.show()
```



In [49]:

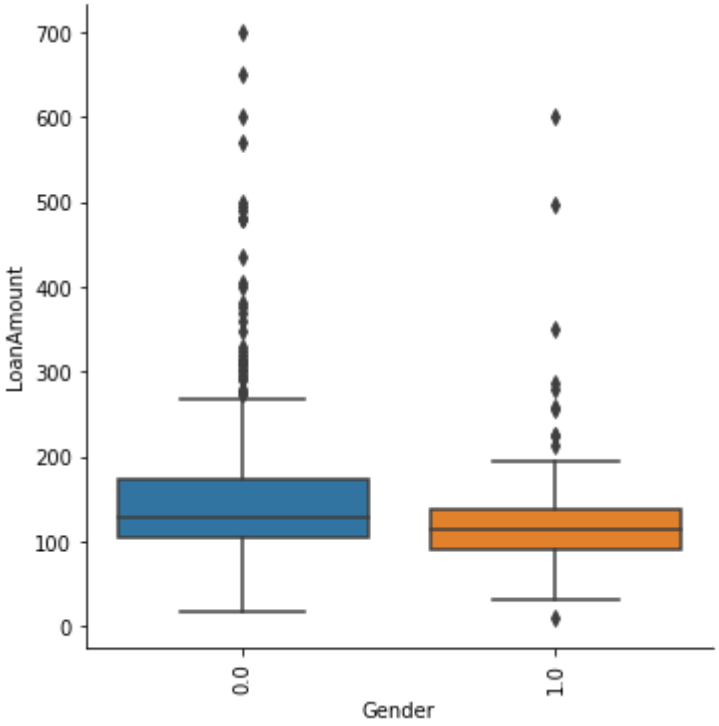
```
plt.figure(figsize=(15,6))
sns.boxplot(x = 'Loan_Status', y = 'CoapplicantIncome', data = loan_data)
plt.xticks(rotation = 90)
plt.show()
```



In [30]:

```
plt.figure(figsize=(15,6))
sns.catplot(x = 'Gender', y = 'LoanAmount', data = loan_data, kind = 'box')
plt.xticks(rotation = 90)
plt.show()
```

```
<Figure size 1080x432 with 0 Axes>
```



In [31]:

```
X = loan_data[['Gender', 'Married', 'ApplicantIncome', 'LoanAmount',
               'Credit_History']]
y = loan_data.Loan_Status
```

In [32]:

```
X.shape, y.shape
```

Out[32]:

```
((614, 5), (614,))
```

In [33]:

```
from sklearn.model_selection import train_test_split
x_train, x_cv, y_train, y_cv = train_test_split(X,y, test_size = 0.2,
                                                random_state = 10)
```

In [34]:

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(max_depth=4, random_state = 10)
model.fit(x_train, y_train)
```

Out[34]:

```
RandomForestClassifier(max_depth=4, random_state=10)
```

In [35]:

```
from sklearn.metrics import accuracy_score
pred_cv = model.predict(x_cv)
accuracy_score(y_cv,pred_cv)
```

Out[35]:

```
0.7967479674796748
```

In [36]:

```
pred_train = model.predict(x_train)
accuracy_score(y_train,pred_train)
```

Out[36]:

```
0.8167006109979633
```