DAV Experiment 1

1. Matplotlib

In [22]:

```
import pandas as pd
import matplotlib.pyplot as plt
```

In [27]:

```
# reading the csv data set
df = pd.read_csv("data_python.csv")
```

In [28]:

```
df.head()
```

Out[28]:

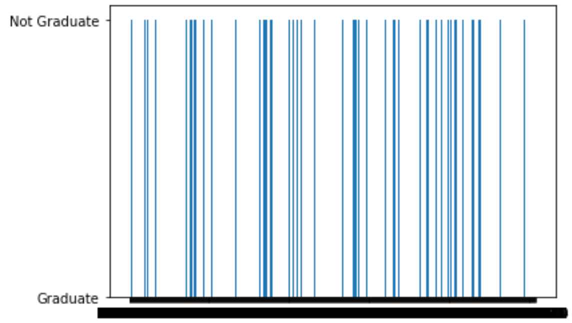
	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_His
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	

In [35]:

```
# Plotting Scatter plot of loan_ID vs Education
plt.bar(df['Loan_ID'], df['Education'])
```

Out[35]:

```
<BarContainer object of 614 artists>
```

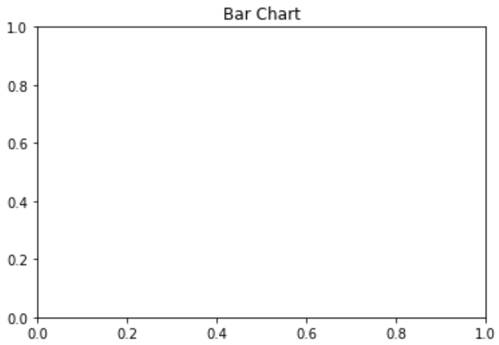


In [33]:

```
# Giving our plot a title
plt.title("Bar Chart")
```

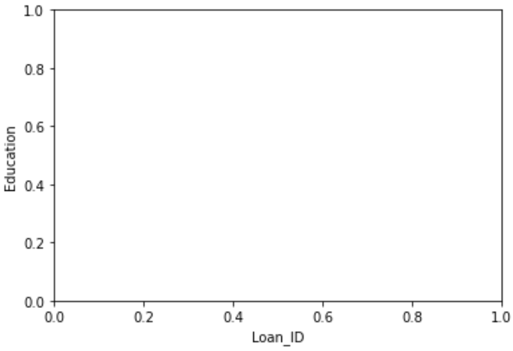
Out[33]:

```
Text(0.5, 1.0, 'Bar Chart')
```



In [36]:

```
# Giving x and y labels names
plt.xlabel('Loan_ID')
plt.ylabel('Education')
plt.show()
```



2. Seaborn

In [37]:

```
import seaborn as sns
```

In [38]:

```
# reading the csv data set using pandas
df = pd.read_csv("data_python.csv")
```

In [39]:

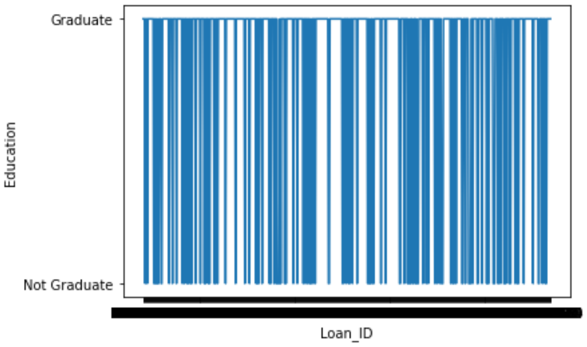
```
df.head()
```

Out[39]:

larried	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_
No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	
Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	
Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	
Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	
No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	

In [40]:

```
sns.lineplot(x='Loan_ID', y='Education', data=df)
plt.show()
```



3. Bokeh

In [41]:

```
import bokeh.plotting
```

In [42]:

```
# creating an object for the figure
chart = bokeh.plotting.figure(title="Bokeh Bar Chart")
```

In [43]:

```
# reading the csv dataset through pandas
df = pd.read_csv("data_python.csv")
```

In [44]:

```
df.head()
```

Out[44]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_His
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	

In [45]:

```
# Counting unique columns of thwe tips file
count = df['Education'].value_counts()
```

In [46]:

```
# plotting the graph
chart.line(count, df['Education'])
```

```
BokehUserWarning: ColumnDataSource's columns must be of the same length. Current lengths: ('x', 2), ('y', 614)
```

Out[46]:

```
GlyphRenderer(
id = '1211', …)
coordinates = None,
data_source = ColumnDataSource(id='1207', ...),
glyph = Line(id='1208', ...),
group = None,
hover_glyph = None,
js_event_callbacks = {},
js_property_callbacks = {},
level = 'glyph',
muted = False,
muted_glyph = Line(id='1210', ...),
name = None,
nonselection_glyph = Line(id='1209', ...),
selection_glyph = 'auto',
subscribed_events = [],
syncable = True,
tags = [],
view = CDSView(id='1212', ...),
visible = True,
x_range_name = 'default',
y_range_name = 'default')
```

In [47]:

```
# showing the figure
bokeh.plotting.show(chart)
```

4. Plotly

In [56]:

```
# importing the required modules
import plotly.express
```

In [57]:

```
# reading the csv dataset through pandas
df = pd.read_csv("data_python.csv")
```

In [58]:

```
df.head()
```

Out[58]:

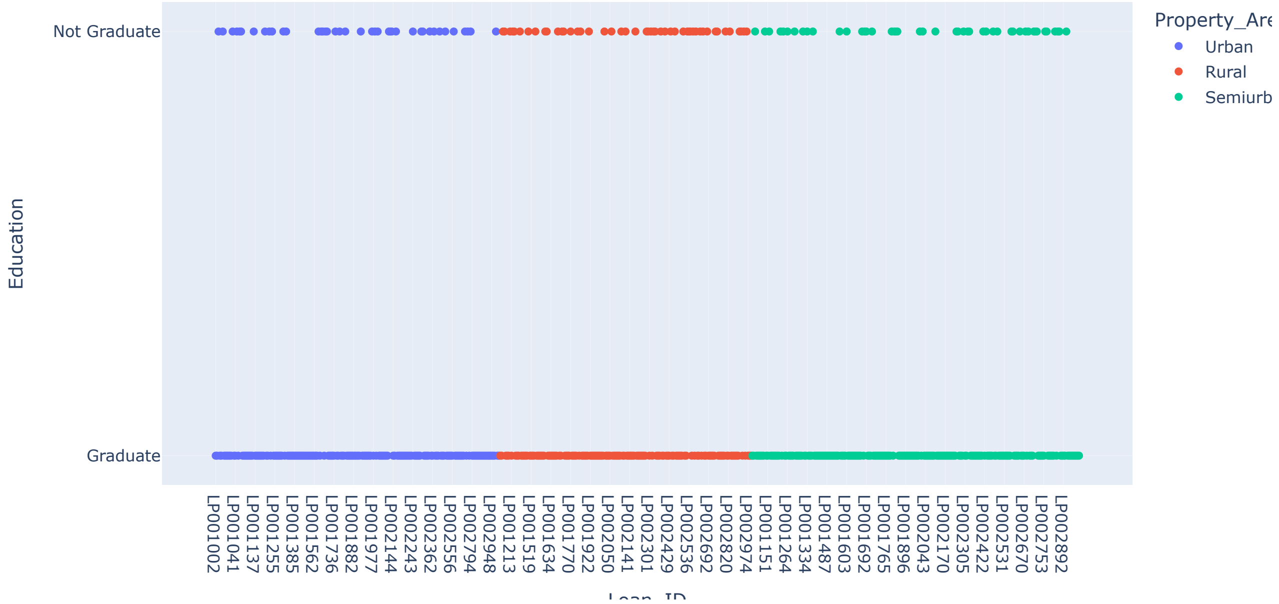
	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_His
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	

In [61]:

```
# plotting our scatter plot
graph = plotly.express.scatter(df, x="Loan_ID", y="Education", color='Property_Area')
```

In [62]:

```
# displaying the plot created
graph.show()
```



Plots to be used for visualization

- Histogram

In [63]:

```
df.head()
```

Out[63]:

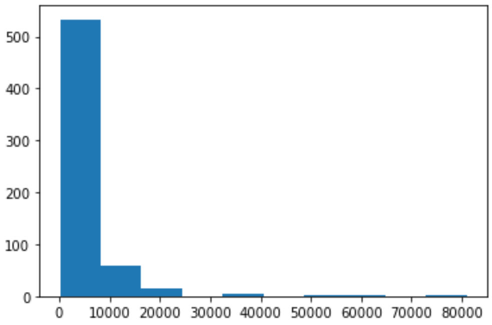
	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_His
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	

In [64]:

```
%matplotlib inline
```

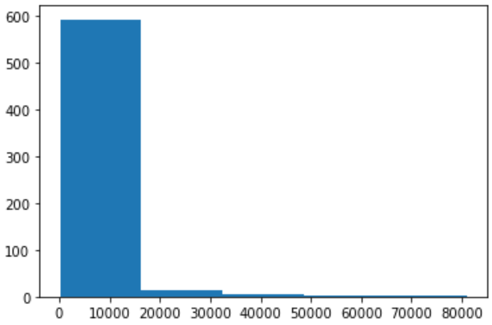
In [69]:

```
plt.hist(x='ApplicantIncome', data=df)
plt.show()
```



In [70]:

```
plt.hist(x='ApplicantIncome', data=df, bins=5)
plt.show()
```



- Scatter plot

In [71]:

```
df.head()
```

Out[71]:

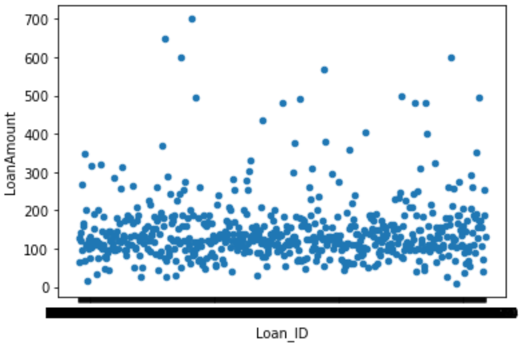
	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_His
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	

In [74]:

```
df.plot.scatter('Loan_ID','LoanAmount')
```

Out[74]:

```
<AxesSubplot:xlabel='Loan_ID', ylabel='LoanAmount'>
```



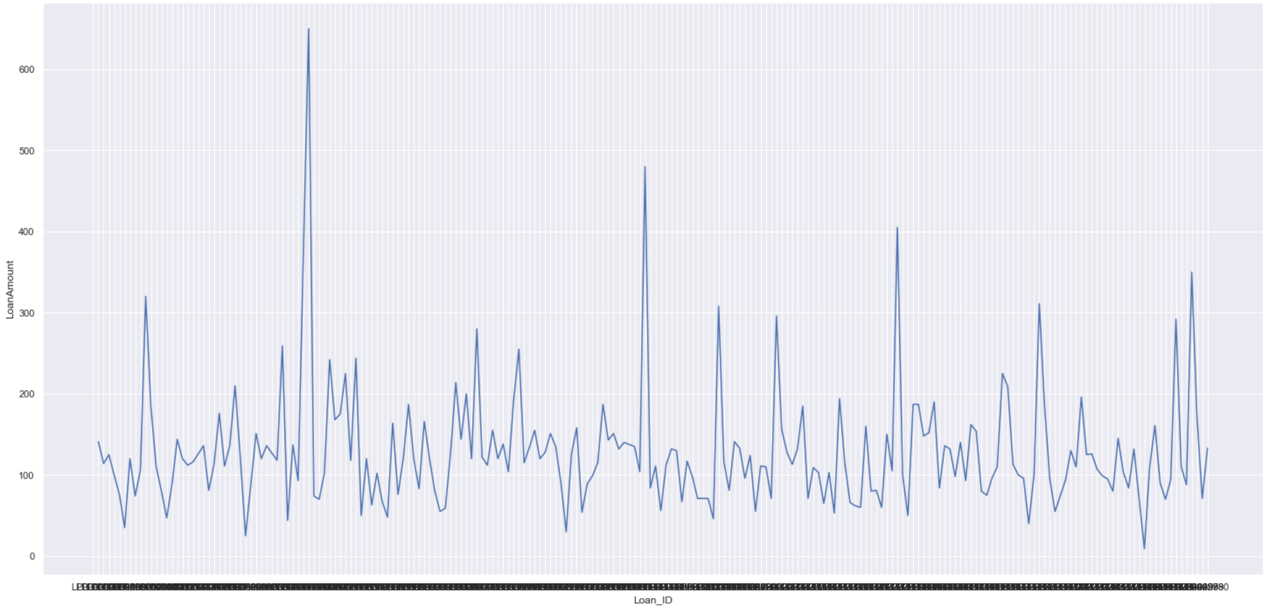
- Line plot

In [114]:

```
keep_up = df.query("Married == 'No'")
sns.lineplot(data=keep_up, x="Loan_ID", y="LoanAmount")
```

Out[114]:

```
<AxesSubplot:xlabel='Loan_ID', ylabel='LoanAmount'>
```



- Swarm plot (using seaborn)

In [78]:

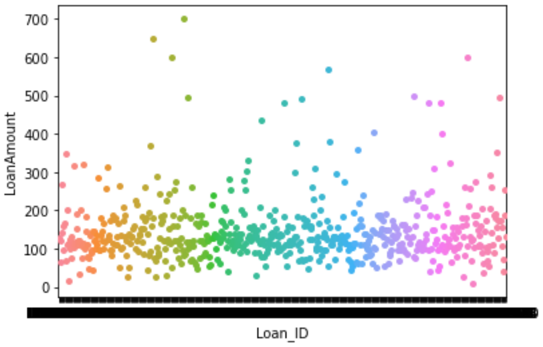
```
# ignoring warnings
import warnings
warnings.filterwarnings("ignore")
```

In [77]:

```
sns.swarmplot(data=df, x="Loan_ID", y="LoanAmount")
```

Out[77]:

```
<AxesSubplot:xlabel='Loan_ID', ylabel='LoanAmount'>
```



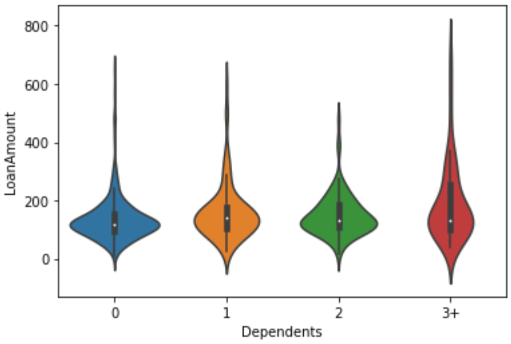
- Violin plot

In [81]:

```
sns.violinplot(x="Dependents", y="LoanAmount", data=df , size=10)
```

Out[81]:

```
<AxesSubplot:xlabel='Dependents', ylabel='LoanAmount'>
```



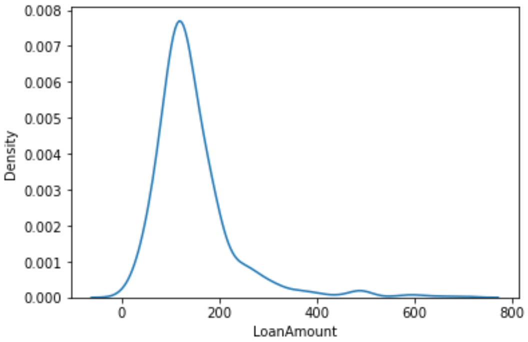
- KDE plot

In [84]:

```
sns.kdeplot(data=df, x="LoanAmount")
```

Out[84]:

```
<AxesSubplot:xlabel='LoanAmount', ylabel='Density'>
```



- Pie chart

In [100]:

```
df.head()
```

Out[100]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_His
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	

In [101]:

```
x= df[['Loan_ID']]
```

In [102]:

```
plt.figure(figsize=(8,5))
```

Out[102]:

```
<Figure size 576x360 with 0 Axes>
```

```
<Figure size 576x360 with 0 Axes>
```

In [103]:

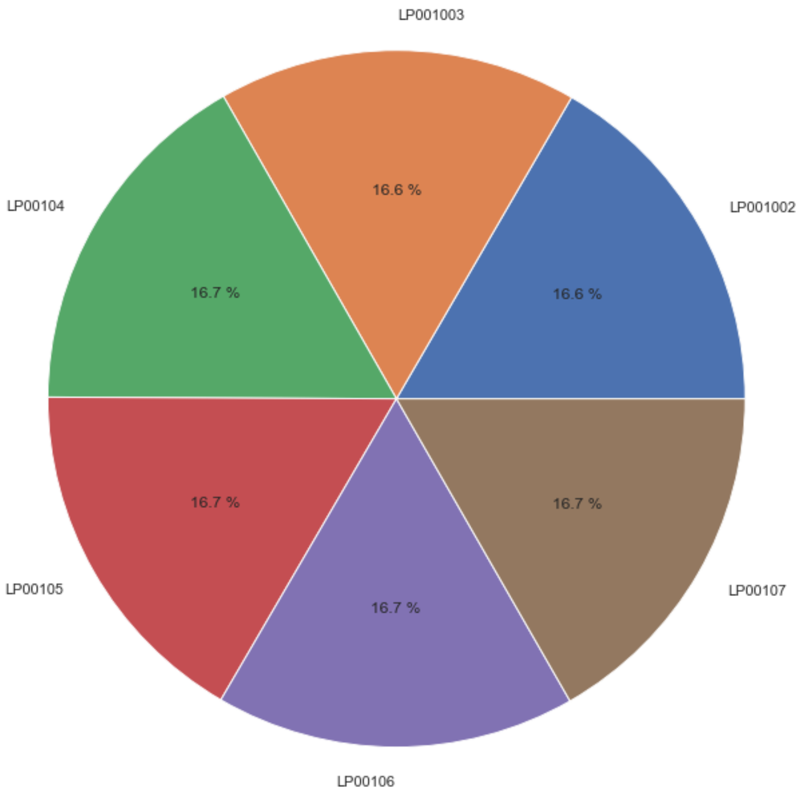
```
labels = ['LP001002', 'LP001003', 'LP00104', 'LP00105', 'LP00106', 'LP00107']
```

In [108]:

```
plt.pie([1002, 1003, 1004, 1005, 1006, 1007], labels = labels, autopct='%.1f %%')
```

Out[108]:

```
([<matplotlib.patches.Wedge at 0x262028b81f0>,
  <matplotlib.patches.Wedge at 0x262028b8970>,
  <matplotlib.patches.Wedge at 0x26202a930d0>,
  <matplotlib.patches.Wedge at 0x26202a937f0>,
  <matplotlib.patches.Wedge at 0x26202a93f10>,
  <matplotlib.patches.Wedge at 0x26202aa2670>],
 [Text(0.953343855738219, 0.5487581368200257, 'LP001002'),
  Text(0.0037269229682921315, 1.0999936863660575, 'LP001003'),
  Text(-0.9501817603644234, 0.554215321216191, 'LP00104'),
  Text(-0.9550554360384921, -0.5457738671769893, 'LP00105'),
  Text(-0.0037269229682925105, -1.0999936863660575, 'LP00106'),
  Text(0.9519104234904848, -0.5512409143470446, 'LP00107')],
 [Text(0.520005739493574, 0.29932262008365035, '16.6 %'),
  Text(0.0020328670736138896, 0.5999965561996676, '16.6 %'),
  Text(-0.5182809601987763, 0.3022992661179223, '16.7 %'),
  Text(-0.5209393287482683, -0.29769483664199414, '16.7 %'),
  Text(-0.0020328670736140964, -0.5999965561996676, '16.7 %'),
  Text(0.5192238673584463, -0.3006768623711152, '16.7 %')])
```



- Heat maps

In [85]:

```
df.corr()
```

Out[85]:

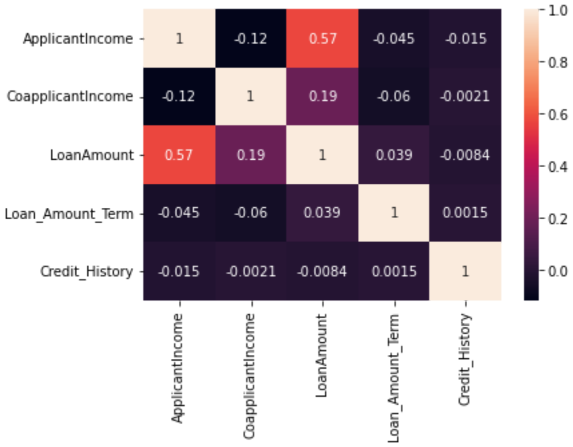
	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
ApplicantIncome	1.000000	-0.116605	0.570909	-0.045306	-0.014715
CoapplicantIncome	-0.116605	1.000000	0.188619	-0.059878	-0.002056
LoanAmount	0.570909	0.188619	1.000000	0.039447	-0.008433
Loan_Amount_Term	-0.045306	-0.059878	0.039447	1.000000	0.001470
Credit_History	-0.014715	-0.002056	-0.008433	0.001470	1.000000

In [86]:

```
sns.heatmap(df.corr(), annot=True)
```

Out[86]:

```
<AxesSubplot:>
```



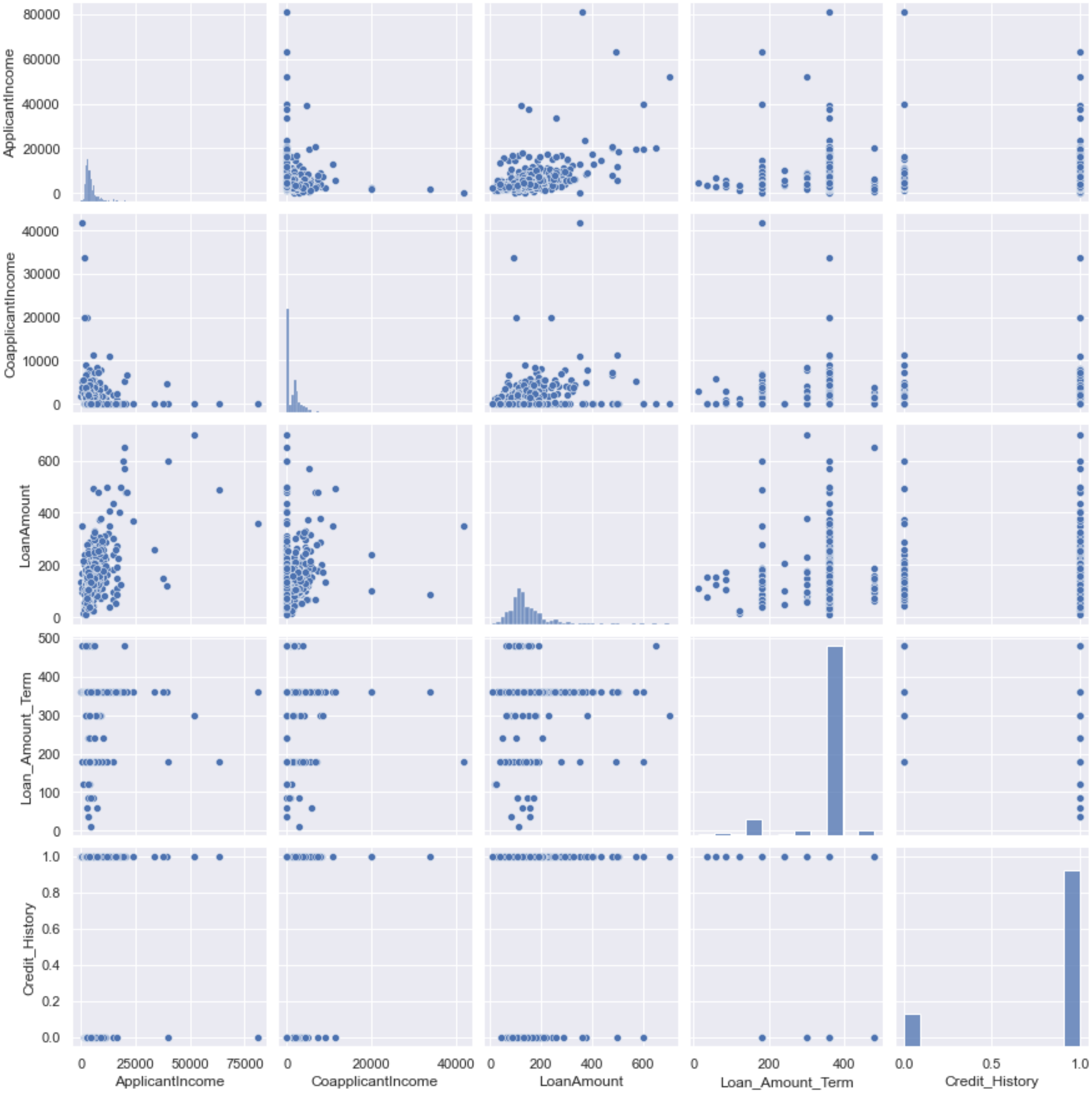
- Pair plot

In [87]:

```
sns.set(rc={'figure.figsize':(25,12)})
sns.pairplot(df)
```

Out[87]:

```
<seaborn.axisgrid.PairGrid at 0x2627f8101c0>
```



```
* Box plot
```

In [88]:

```
df.head()
```

Out[88]:

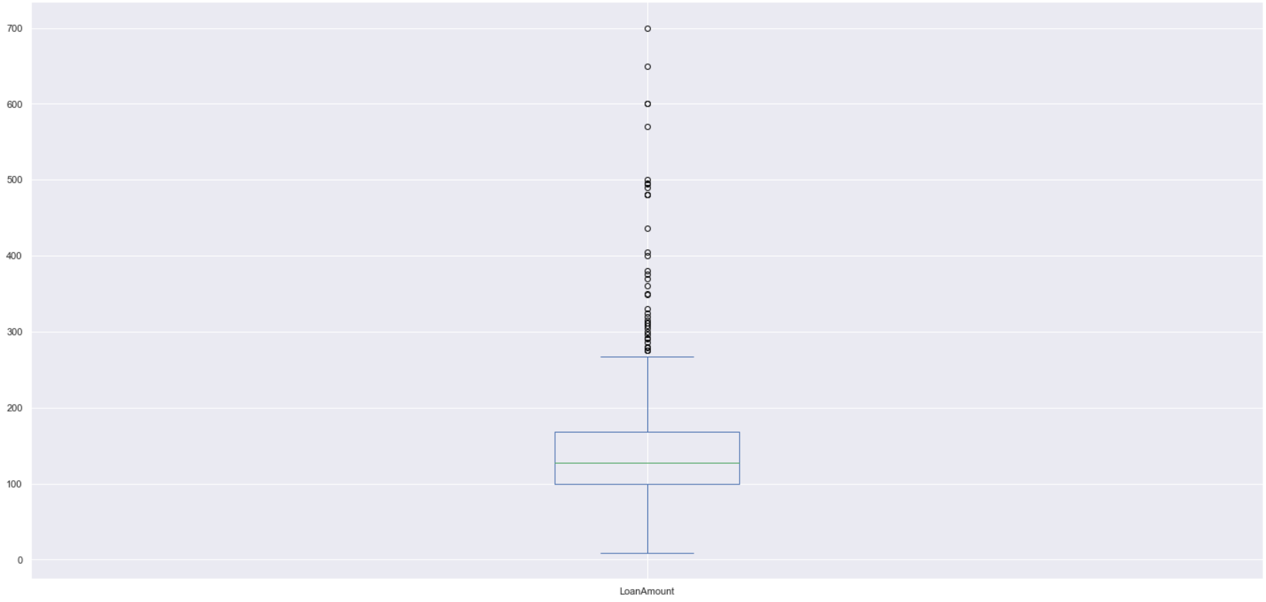
	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_His
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	

In [91]:

```
df['LoanAmount'].plot.box()
```

Out[91]:

```
<AxesSubplot:>
```



- Count plot

In [109]:

```
df.head()
```

Out[109]:

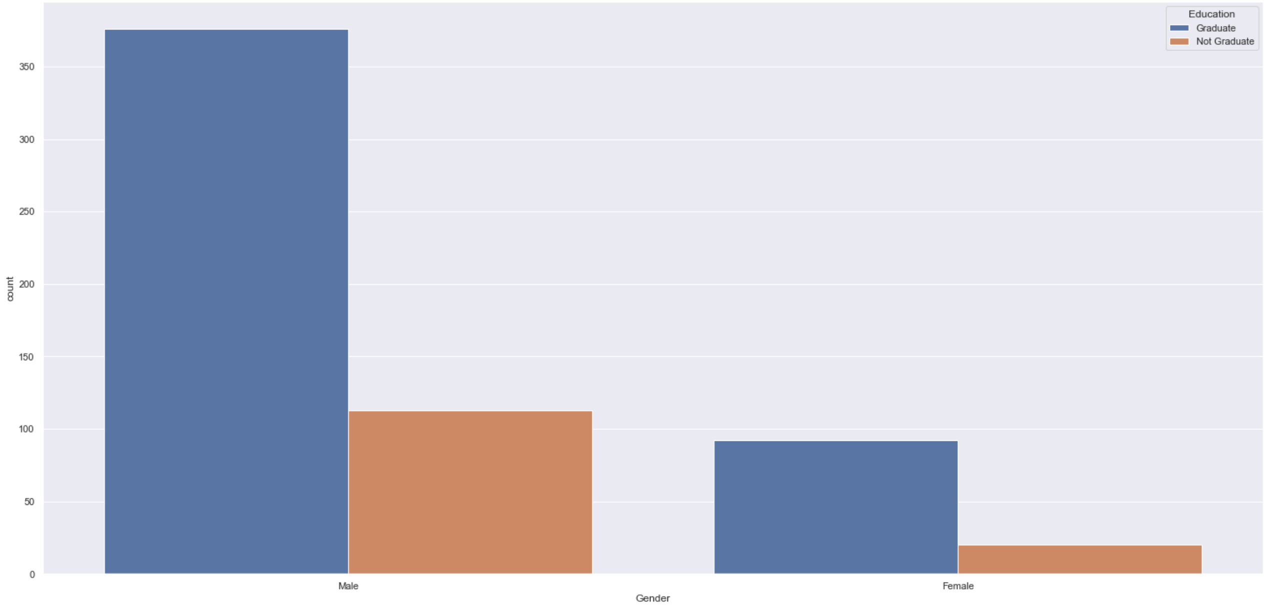
	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_His
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	

In [116]:

```
sns.countplot(x='Gender', data=df, hue='Education')
```

Out[116]:

```
<AxesSubplot:xlabel='Gender', ylabel='count'>
```



In [ ]:

```
```