In [1]:

```
#NAME : DEV CHAUHAN
#SECTION : DS
#SEMESTER : 5TH
#ROLL NO. : 2013648
#SUBJECT : BIG DATA STORAGE AND PROCESSING
#POJECT : VISUALISATON ON CREDIT CARD FRAUD DETECTON
```

In [2]:

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

In [3]:

```
data = pd.read_csv("creditcard.csv")
```

In [4]:

```
data.head(10)
```

Out[4]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533
5	2.0	-0.425966	0.960523	1.141109	-0.168252	0.420987	-0.029728	0.476201	0.260314
6	4.0	1.229658	0.141004	0.045371	1.202613	0.191881	0.272708	-0.005159	0.081213
7	7.0	-0.644269	1.417964	1.074380	-0.492199	0.948934	0.428118	1.120631	-3.807864
8	7.0	-0.894286	0.286157	-0.113192	-0.271526	2.669599	3.721818	0.370145	0.851084
9	9.0	-0.338262	1.119593	1.044367	-0.222187	0.499361	-0.246761	0.651583	0.069539

10 rows × 31 columns

In [5]:

```
data.shape
```

Out[5]:

```
(284807, 31)
```

In [6]:

```
data.describe()
```

Out[6]:

	Time	V1	V2	V3	V4	V
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+0
mean	94813.859575	3.919560e-15	5.688174e-16	-8.769071e-15	2.782312e-15	-1.552563e-1
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.380247e+0
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.137433e+0
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.915971e-0
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.433583e-0
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.119264e-0
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.480167e+0

8 rows × 31 columns

In [7]:

```
fraud =data[data['Class'] == 1]
print('Fraud Cases: {}'.format(len(data[data['Class'] == 1])))
```

```
Fraud Cases: 492
```

In [8]:

```
valid = data[data['Class'] == 0]
print('Valid Transactions: {}'.format(len(data[data['Class'] == 0])))
```

```
Valid Transactions: 284315
```

In [9]:

```
print("Amount details of the fraudulent transaction")
fraud.Amount.describe()
```

```
Amount details of the fraudulent transaction
```

Out[9]:

```
count     492.000000
mean      122.211321
std       256.683288
min         0.000000
25%         1.000000
50%         9.250000
75%       105.890000
max      2125.870000
Name: Amount, dtype: float64
```

In [10]:

```
print("details of valid transaction")
valid.Amount.describe()
```

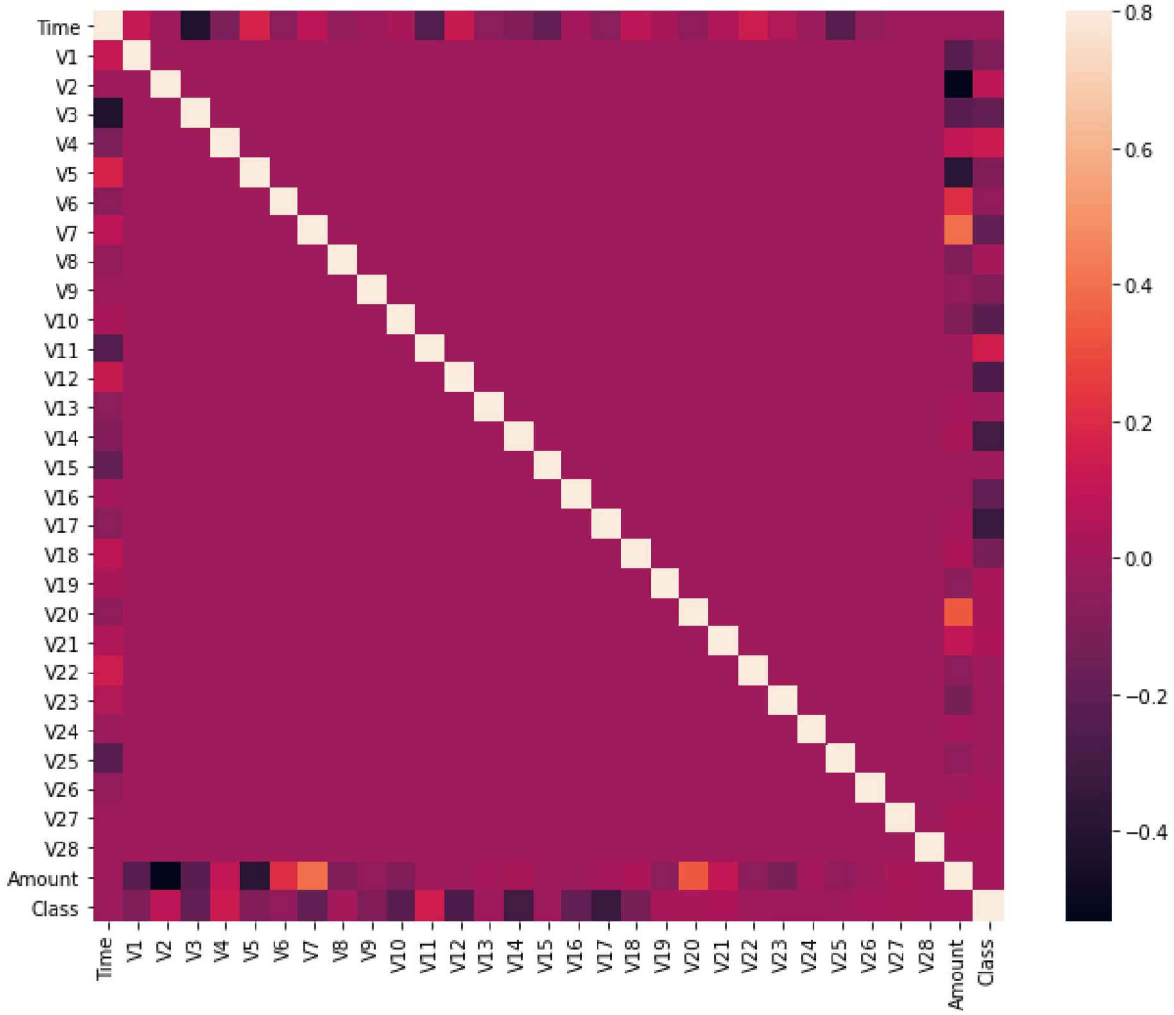
```
details of valid transaction
```

Out[10]:

```
count    284315.000000
mean         88.291022
std         250.105092
min           0.000000
25%           5.650000
50%          22.000000
75%          77.050000
max       25691.160000
Name: Amount, dtype: float64
```

In [11]:

```
# Correlation matrix
corrmat = data.corr()
fig = plt.figure(figsize = (12, 9))
sns.heatmap(corrmat, vmax = .8, square = True)
plt.show()
```



In [12]:

```
x = data.drop(['Class'], axis = 1)
y = data["Class"]
from sklearn.model_selection import train_test_split
xTrain, xTest, yTrain, yTest = train_test_split(x,y, test_size = 0.2, random_state = 42)
from sklearn.ensemble import RandomForestClassifier
m = RandomForestClassifier()
m.fit(xTrain, yTrain)
pri = m.predict(xTest)
```

In [13]:

```
from sklearn.metrics import classification_report, accuracy_score
acc = (accuracy_score(yTest,pri)*100)
print("The accuracy= {} ".format(acc))
```

```
The accuracy= 99.95786664794073
```