In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import  matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
from sklearn.linear_model import LogisticRegression

sns.set_style('darkgrid')
```

In [3]:
```
df=pd.read_csv('payment_fraud.csv')
```

In [4]:
```
df.head()
```

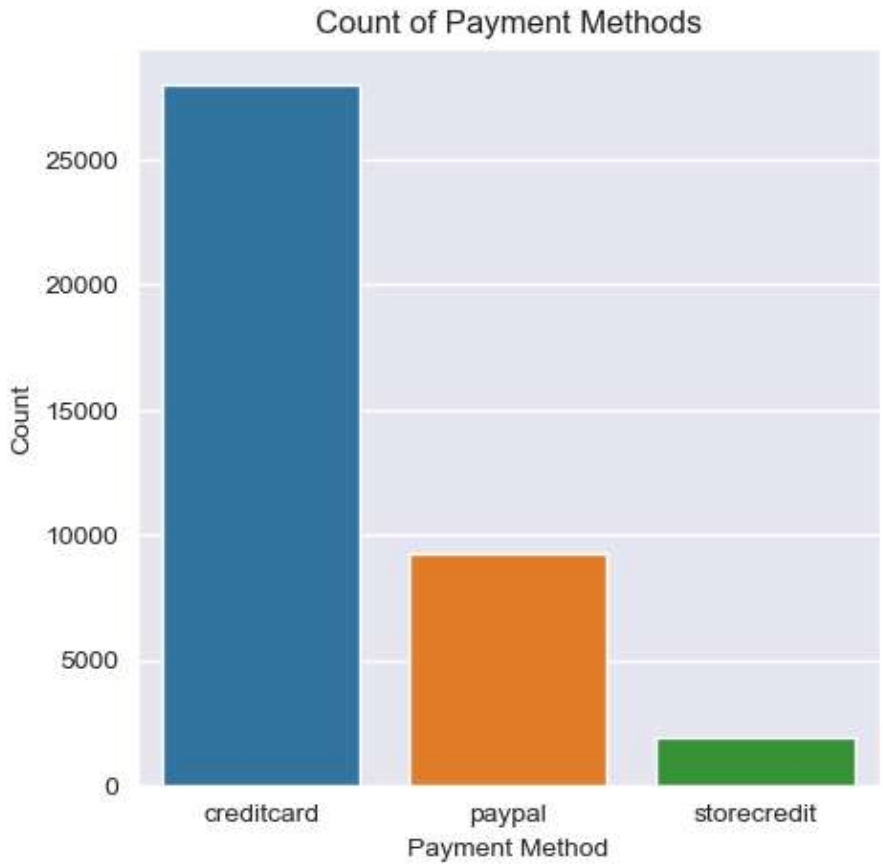
Out[4]:

	accountAgeDays	numItems	localTime	paymentMethod	paymentMethodAgeDays	label
0	29	1	4.745402	paypal	28.204861	0
1	725	1	4.742303	storecredit	0.000000	0
2	845	1	4.921318	creditcard	0.000000	0
3	503	1	4.886641	creditcard	0.000000	0
4	2000	1	5.040929	creditcard	0.000000	0

In [6]:
```
df.isnull().sum() ## checking the null values
```

Out[6]:
```
accountAgeDays          0
numItems                0
localTime               0
paymentMethod           0
paymentMethodAgeDays    0
label                   0
dtype: int64
```

In [11]:
```
paymthd = df.paymentMethod.value_counts()
plt.figure(figsize=(5, 5))
sns.barplot(x=paymthd.index, y=paymthd.values)
plt.ylabel('Count')
plt.xlabel('Payment Method')
plt.title('Count of Payment Methods')
plt.show()
```



In [12]:
```
df.label.value_counts() ##count the number of 0's and 1's
```

Out[12]:
```
label
0    38661
1      560
Name: count, dtype: int64
```

In [14]:
```
## converting paymentMethod column into label encoding
paymthd_label = {v:k for k, v in enumerate(df.paymentMethod.unique())}

df.paymentMethod =df.paymentMethod.map(paymthd_label)
```

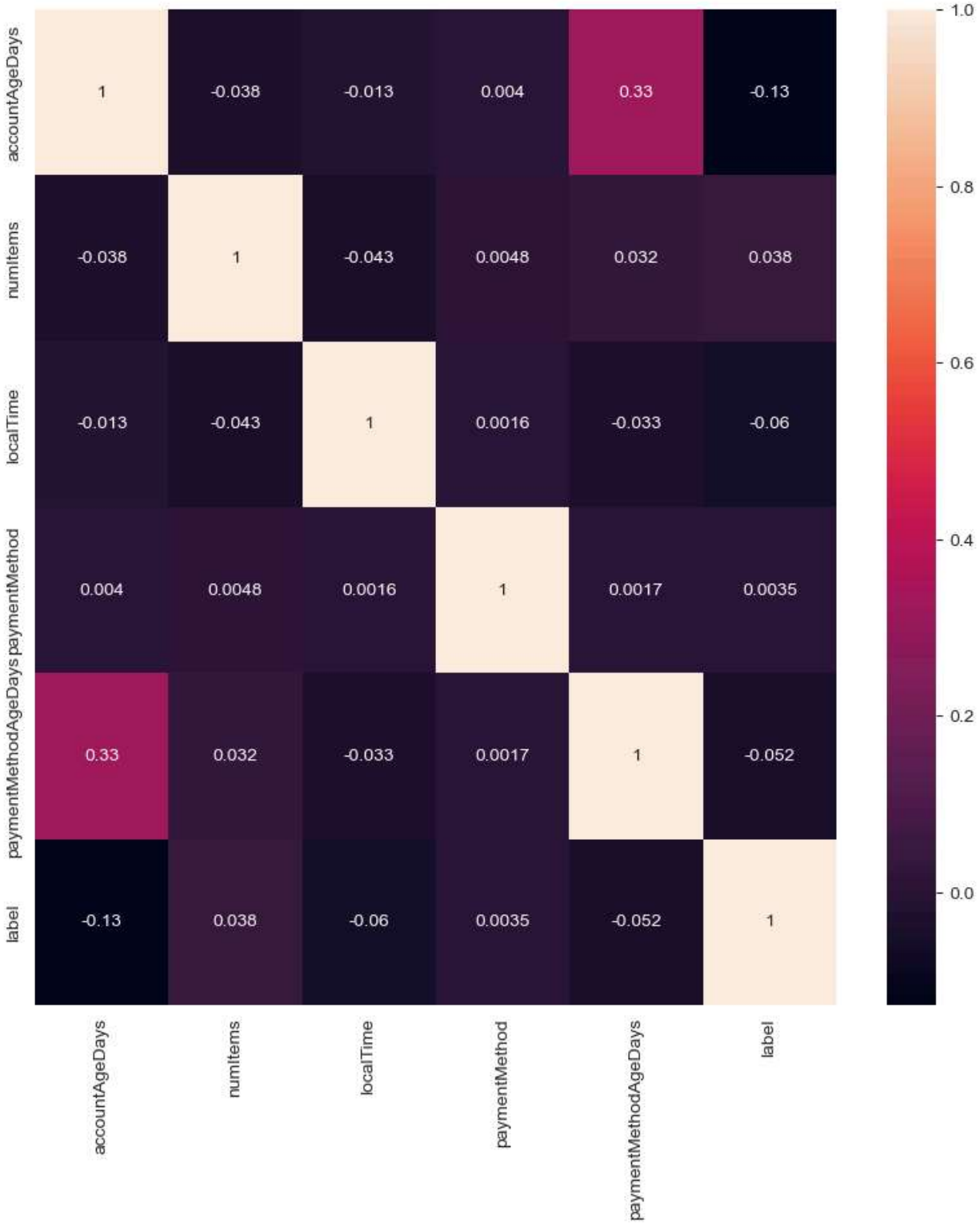
In [15]:
```
df.head()
```

Out[15]:

	accountAgeDays	numItems	localTime	paymentMethod	paymentMethodAgeDays	label
0	29	1	4.745402	0	28.204861	0
1	725	1	4.742303	1	0.000000	0
2	845	1	4.921318	2	0.000000	0
3	503	1	4.886641	2	0.000000	0
4	2000	1	5.040929	2	0.000000	0

In [17]:
```
## corr(): it gives the correlation between the featuers
plt.figure(figsize=(10,10))
sns.heatmap(df.corr(), annot=True)
```

Out[17]: <Axes: >



In [18]:
```
df.describe()
```

Out[18]:

	accountAgeDays	numItems	localTime	paymentMethod	paymentMethodAgeDays	la
count	39221.000000	39221.000000	39221.000000	39221.000000	39221.000000	39221.000(
mean	857.563984	1.084751	4.748232	1.476811	122.641326	0.014
std	804.788212	0.566899	0.389360	0.850805	283.569177	0.118
min	1.000000	1.000000	0.421214	0.000000	0.000000	0.000(
25%	72.000000	1.000000	4.742303	1.000000	0.000000	0.000(
50%	603.000000	1.000000	4.886641	2.000000	0.012500	0.000(
75%	1804.000000	1.000000	4.962055	2.000000	87.510417	0.000(
max	2000.000000	29.000000	5.040929	2.000000	1999.580556	1.000(

In [19]:
```
## independent and dependent features
X = df.iloc[:, :-1].values
y = df.iloc[:, -1].values
```

In [23]:
```
## scaling
sc = StandardScaler()
X = sc.fit_transform(X)
```

In [24]:
```
## train test split
X_train, X_test, y_train,y_test = train_test_split(X,y, test_size=0.25,random_state=4
```

In [26]:
```
print("X_train shape: ", X_train.shape)
print("X_test shape: ", X_test.shape)
print("y_train shape: ", y_train.shape)
print("y_test shape: ", y_test.shape)
```

```
X_train shape:  (29415, 5)
X_test shape:  (9806, 5)
y_train shape:  (29415,)
y_test shape:  (9806,)
```

In [27]:
```
## LogisticRegression Model
lg = LogisticRegression()

## training
lg.fit(X_train, y_train)
```

Out[27]:

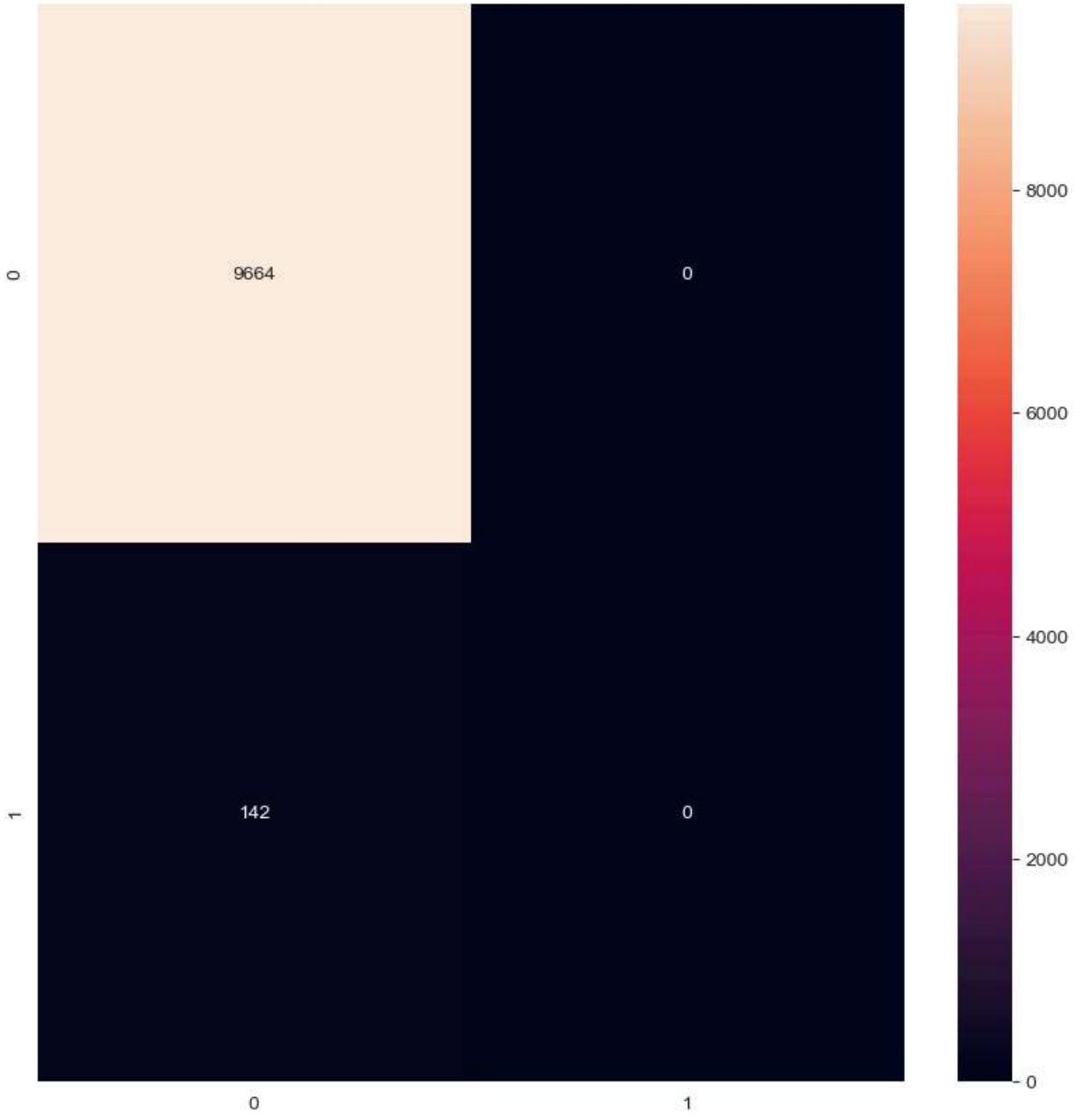
▾ LogisticRegression (https://scikit-learn.org/1.4/modules/generated/sklearn.linear_model.LogisticRegression.

LogisticRegression()

In [28]:
```
## prediction
pred = lg.predict(X_test)
```

In [30]:
```
print("--------------------------------------------------Accuracy------------------
print(accuracy_score(y_test, pred))
print()

print("--------------------------------------------------Classification Report------
print(classification_report(y_test, pred))
print()

print("--------------------------------------------------Confustion Metrics-----------
plt.figure(figsize=(10, 10));
sns.heatmap(confusion_matrix(y_test, pred), annot=True, fmt='g');
```

```
----------------------------------------------------Accuracy------------------------
------------------------------
0.9855190699571691

---------------------------------------------------Classification Report------------
---------------------------------
              precision    recall  f1-score   support

           0       0.99      1.00      0.99      9664
           1       0.00      0.00      0.00       142

    accuracy                           0.99      9806
   macro avg       0.49      0.50      0.50      9806
weighted avg       0.97      0.99      0.98      9806


-------------------------------------------------Confustion Metrics-----------------
-----------------------------------

C:\Users\Lenovo\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1509:
UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\Lenovo\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1509:
UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\Lenovo\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1509:
UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with
no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```



In [ ]: