In [51]:
```
import numpy as np  # Importing NumPy library
import pandas as pd  # Importing Pandas library
import matplotlib.pyplot as plt  # Importing Matplotlib library's "pyplot" modu
import seaborn as sns  # Importing Seaborn library


import os
```

In [52]:
```
data = pd.read_csv("Classification.csv")
data['fetal_health'] = data['fetal_health'].replace(1.0,0)
data['fetal_health'] = data['fetal_health'].replace(2.0,0)
data['fetal_health'] = data['fetal_health'].replace(3.0,1)
data
```

Out[52]:

	baseline value	accelerations	fetal_movement	uterine_contractions	light_decelerations	severe
0	120	0.000	0.000	0.000	0.000	
1	132	0.006	0.000	0.006	0.003	
2	133	0.003	0.000	0.008	0.003	
3	134	0.003	0.000	0.008	0.003	
4	132	0.007	0.000	0.008	0.000	
...	...	...	...	...	...	
2121	140	0.000	0.000	0.007	0.000	
2122	140	0.001	0.000	0.007	0.000	
2123	140	0.001	0.000	0.007	0.000	
2124	140	0.001	0.000	0.006	0.000	
2125	142	0.002	0.002	0.008	0.000	

2126 rows × 22 columns

In [53]:
```
X = data.drop(["fetal_health"], axis=1)
y = data['fetal_health']
```

In [54]:
```
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, random_st

MLP = MLPClassifier(random_state=1, max_iter=300)

MLP.fit(X_train, y_train)
y_pred=MLP.predict(X_test)
```

In [55]:
```
#Training Score
MLP.score(X_train, y_train)*100
```

Out[55]:
```
96.42409033877038
```

In [56]:
```
#Testing Score
MLP.score(X_test, y_test)*100
```

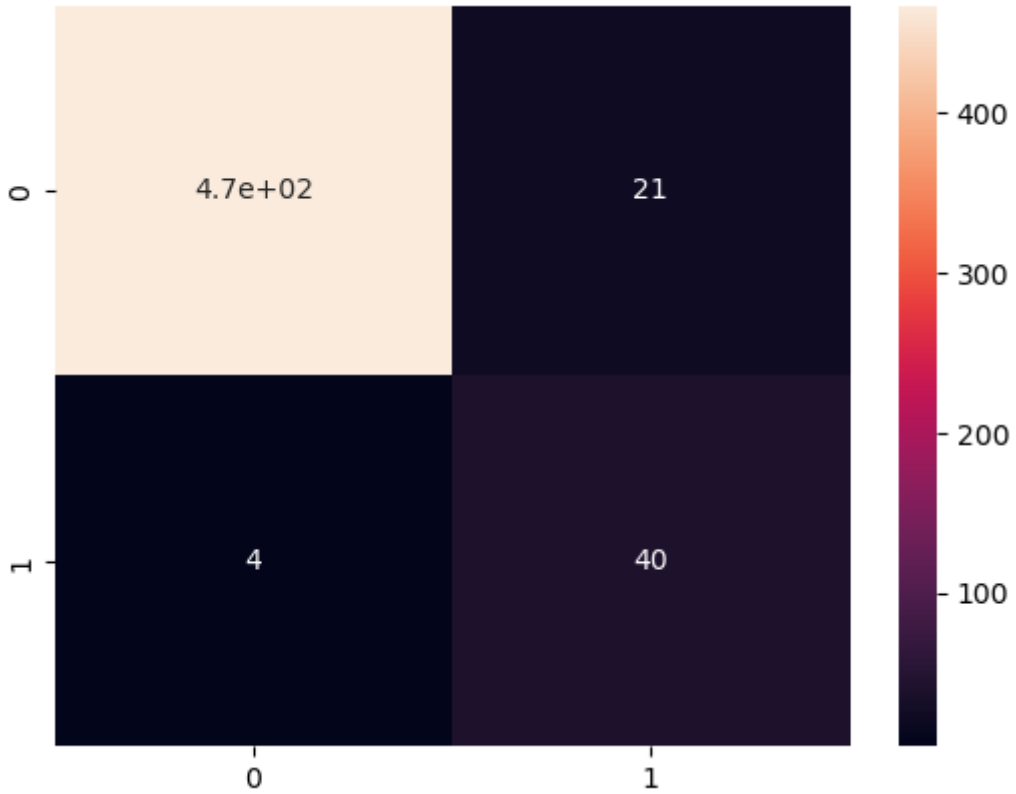
Out[56]:
```
95.30075187969925
```

In [57]:
```
from sklearn.metrics import classification_report
#evaluate model performance
print(classification_report(y_test, y_pred))
```

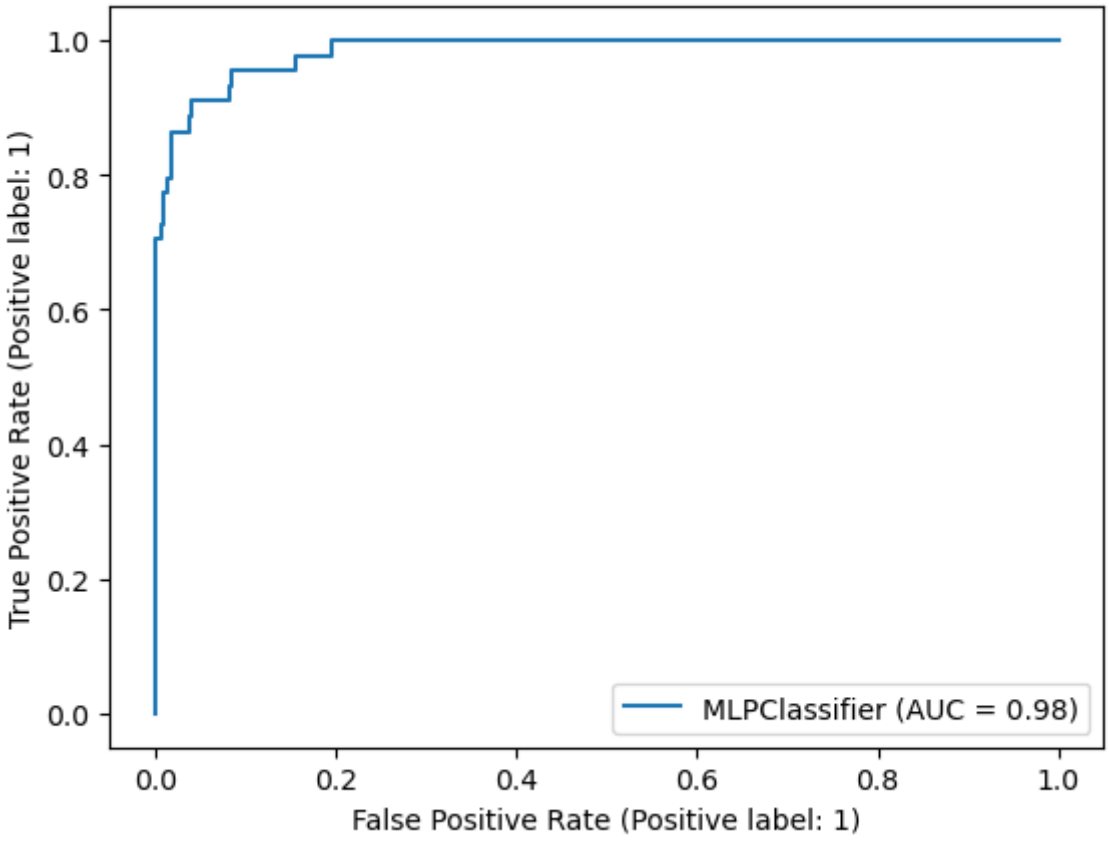
```
              precision    recall  f1-score   support

           0       0.99      0.96      0.97       488
           1       0.66      0.91      0.76        44

    accuracy                           0.95       532
   macro avg       0.82      0.93      0.87       532
weighted avg       0.96      0.95      0.96       532
```

In [58]:
```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score

cm = confusion_matrix(y_test, y_pred)

accuracy_score(y_test, y_pred)

sns.heatmap(cm, annot=True)
```

Out[58]:
```
<AxesSubplot:>
```



In [59]:
```
from sklearn.metrics import RocCurveDisplay
metrics.RocCurveDisplay.from_estimator(MLP, X_test, y_test)
plt.show()
```


True Positive Rate (Positive label: 1)
False Positive Rate (Positive label: 1)
MLPClassifier (AUC = 0.98)
0.0
0.2
0.4
0.6
0.8
1.0

In [ ]: