1. Use Autoencoder to implement anomaly detection. Build the model by using:
   a. Import required libraries
   b. Upload / access the dataset
   c. Encoder converts it into latent representation
   d. Decoder networks convert it back to the original input
   e. Compile the models with Optimizer, Loss, and Evaluation Metrics

a. Import required libraries

In [68]:
```
import numpy as np
import pandas as pd
import tensorflow as tf
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix, classification_report
from tensorflow.keras import layers, models
import matplotlib.pyplot as plt
```

b. Upload / access the dataset

In [69]:
```
# Load the ECG dataset
ecg_dataset = pd.read_csv("Datasets/ecg-csv/ecg.csv")
```

In [70]:
```
# Preprocess the data
scaler = StandardScaler()
X = scaler.fit_transform(ecg_dataset.values)
y = X  # Autoencoder input and output are the same

X_train, X_test, _, _ = train_test_split(X, X, test_size=0.2, random_state
```

In [71]:
```
# Build and train the Autoencoder model
input_dim = X_train.shape[1]
```

c. Encoder converts it into latent representation

In [72]:
```
encoder = models.Sequential([
    layers.Input(shape=(input_dim,)),
    layers.Dense(32, activation='relu'),
    layers.Dense(16, activation='relu'),
    layers.Dense(8, activation='relu')
])
```

d. Decoder networks convert it back to the original input

In [73]:
```
decoder = models.Sequential([
    layers.Input(shape=(8,)),
    layers.Dense(16, activation='relu'),
    layers.Dense(32, activation='relu'),
```

```
    layers.Dense(input_dim, activation='linear')  # Use linear activation 
])
```

e. Compile the models with Optimizer, Loss, and Evaluation Metrics

In [74]:
```
autoencoder = models.Sequential([
    encoder,
    decoder
])
autoencoder.compile(optimizer='adam', loss='mean_squared_error')
autoencoder.fit(X_train, X_train, epochs=100, batch_size=32, shuffle=True)
```

```
Epoch 1/100
125/125 [==============================] - 1s 2ms/step - loss: 0.6891
Epoch 2/100
125/125 [==============================] - 0s 1ms/step - loss: 0.3301
Epoch 3/100
125/125 [==============================] - 0s 1ms/step - loss: 0.2722
Epoch 4/100
125/125 [==============================] - 0s 2ms/step - loss: 0.2358
Epoch 5/100
125/125 [==============================] - 0s 1ms/step - loss: 0.2116
Epoch 6/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1937
Epoch 7/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1807
Epoch 8/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1683
Epoch 9/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1581
Epoch 10/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1510
Epoch 11/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1454
Epoch 12/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1408
Epoch 13/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1358
Epoch 14/100
125/125 [==============================] - 0s 2ms/step - loss: 0.1332
Epoch 15/100
125/125 [==============================] - 0s 2ms/step - loss: 0.1302
Epoch 16/100
125/125 [==============================] - 0s 2ms/step - loss: 0.1284
Epoch 17/100
125/125 [==============================] - 0s 2ms/step - loss: 0.1252
Epoch 18/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1238
Epoch 19/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1221
Epoch 20/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1206
Epoch 21/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1193
Epoch 22/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1176
Epoch 23/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1164
Epoch 24/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1152
Epoch 25/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1141
Epoch 26/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1124
Epoch 27/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1119
Epoch 28/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1097
Epoch 29/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1084
Epoch 30/100
```

```
125/125 [==============================] - 0s 1ms/step - loss: 0.1070
Epoch 31/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1066
Epoch 32/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1058
Epoch 33/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1048
Epoch 34/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1042
Epoch 35/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1031
Epoch 36/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1023
Epoch 37/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1023
Epoch 38/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1012
Epoch 39/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1005
Epoch 40/100
125/125 [==============================] - 0s 1ms/step - loss: 0.1001
Epoch 41/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0996
Epoch 42/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0995
Epoch 43/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0988
Epoch 44/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0982
Epoch 45/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0976
Epoch 46/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0975
Epoch 47/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0969
Epoch 48/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0960
Epoch 49/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0963
Epoch 50/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0961
Epoch 51/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0958
Epoch 52/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0959
Epoch 53/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0944
Epoch 54/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0949
Epoch 55/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0938
Epoch 56/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0934
Epoch 57/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0932
Epoch 58/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0931
Epoch 59/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0928
```

```
Epoch 60/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0927
Epoch 61/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0924
Epoch 62/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0922
Epoch 63/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0913
Epoch 64/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0911
Epoch 65/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0911
Epoch 66/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0910
Epoch 67/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0905
Epoch 68/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0903
Epoch 69/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0904
Epoch 70/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0912
Epoch 71/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0903
Epoch 72/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0904
Epoch 73/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0895
Epoch 74/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0893
Epoch 75/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0893
Epoch 76/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0887
Epoch 77/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0888
Epoch 78/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0885
Epoch 79/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0883
Epoch 80/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0885
Epoch 81/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0884
Epoch 82/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0878
Epoch 83/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0877
Epoch 84/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0878
Epoch 85/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0870
Epoch 86/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0874
Epoch 87/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0872
Epoch 88/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0874
Epoch 89/100
```

```
125/125 [==============================] - 0s 1ms/step - loss: 0.0870
Epoch 90/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0866
Epoch 91/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0867
Epoch 92/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0867
Epoch 93/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0861
Epoch 94/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0865
Epoch 95/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0859
Epoch 96/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0859
Epoch 97/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0862
Epoch 98/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0858
Epoch 99/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0855
Epoch 100/100
125/125 [==============================] - 0s 1ms/step - loss: 0.0854
```

Out[74]:

```
<keras.src.callbacks.History at 0x7f9b14231ed0>
```

In [75]:

```python
# Detect anomalies
y_pred = autoencoder.predict(X_test)
mse = np.mean(np.power(X_test - y_pred, 2), axis=1)
```

```
32/32 [==============================] - 0s 907us/step
```

In [76]:

```python
# Define a threshold for anomaly detection
threshold = np.percentile(mse, 95)  # Adjust the percentile as needed
```

In [77]:

```python
# Predict anomalies
anomalies = mse > threshold
```

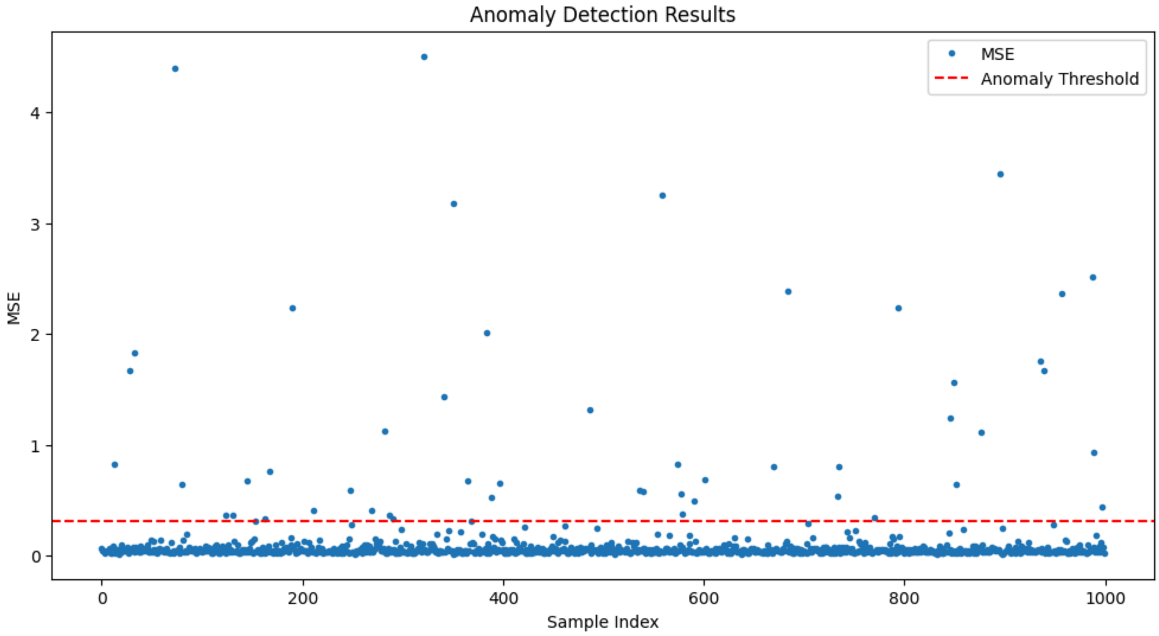
In [78]:

```python
# Calculate the number of anomalies
num_anomalies = np.sum(anomalies)
print(f"Number of Anomalies: {num_anomalies}")
```

```
Number of Anomalies: 50
```

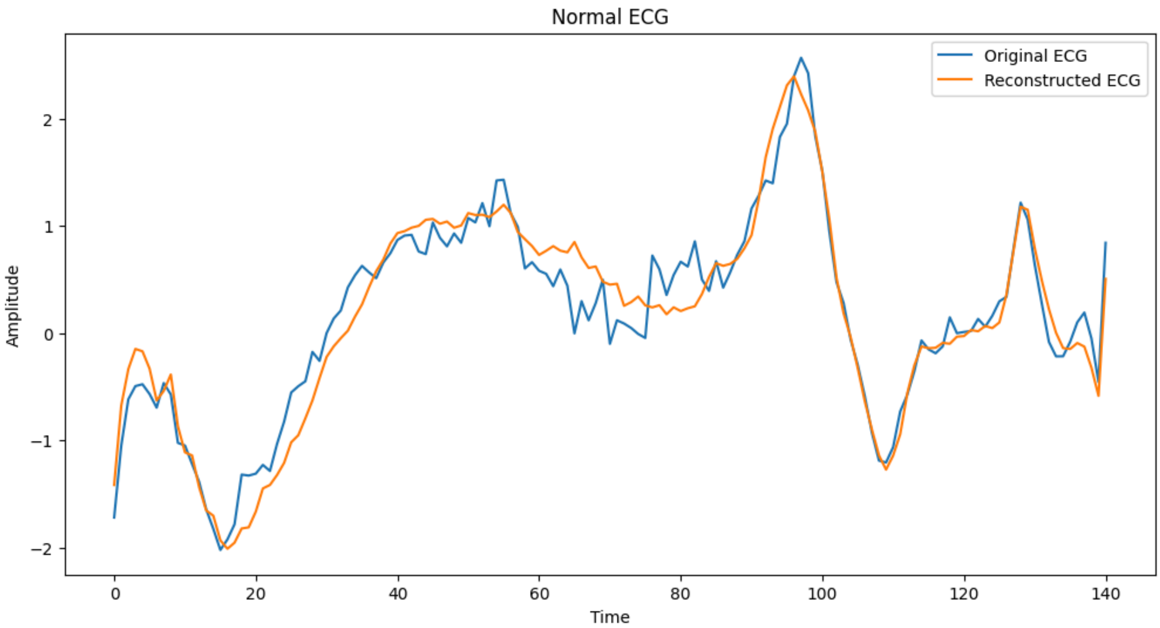
In [79]:

```python
# Plot the anomalies
plt.figure(figsize=(12, 6))
plt.plot(mse, marker='o', linestyle='', markersize=3, label='MSE')
plt.axhline(threshold, color='r', linestyle='--', label='Anomaly Threshold
plt.xlabel('Sample Index')
plt.ylabel('MSE')
plt.title('Anomaly Detection Results')
plt.legend()
plt.show()
```



Visualize Normal

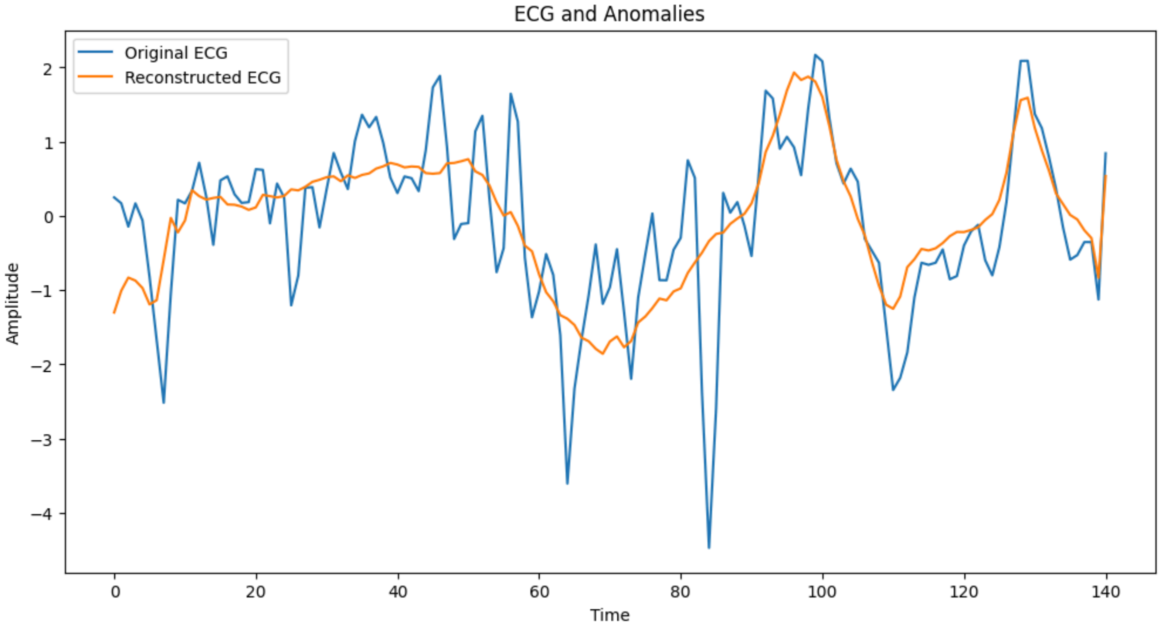
In [80]:
```
plt.figure(figsize=(12, 6))
plt.plot(X_test[0], label='Original ECG')
plt.plot(y_pred[0], label='Reconstructed ECG')
plt.xlabel('Time')
plt.ylabel('Amplitude')
plt.legend()
plt.title('Normal ECG')
plt.show()
```



Visualize Anomaly

In [81]:
```
# listing the index of anomalies in X_test
anomalies_index = []
for index, anomaly in enumerate(anomalies):
    if anomaly == True :
        anomalies_index.append(index)
```

In [82]:
```
n = 4
anomaly_index = anomalies_index[n]
plt.figure(figsize=(12, 6))
plt.plot(X_test[anomaly_index], label='Original ECG')
plt.plot(y_pred[anomaly_index], label='Reconstructed ECG')
plt.xlabel('Time')
plt.ylabel('Amplitude')
plt.legend()
plt.title('ECG and Anomalies')
plt.show()
```



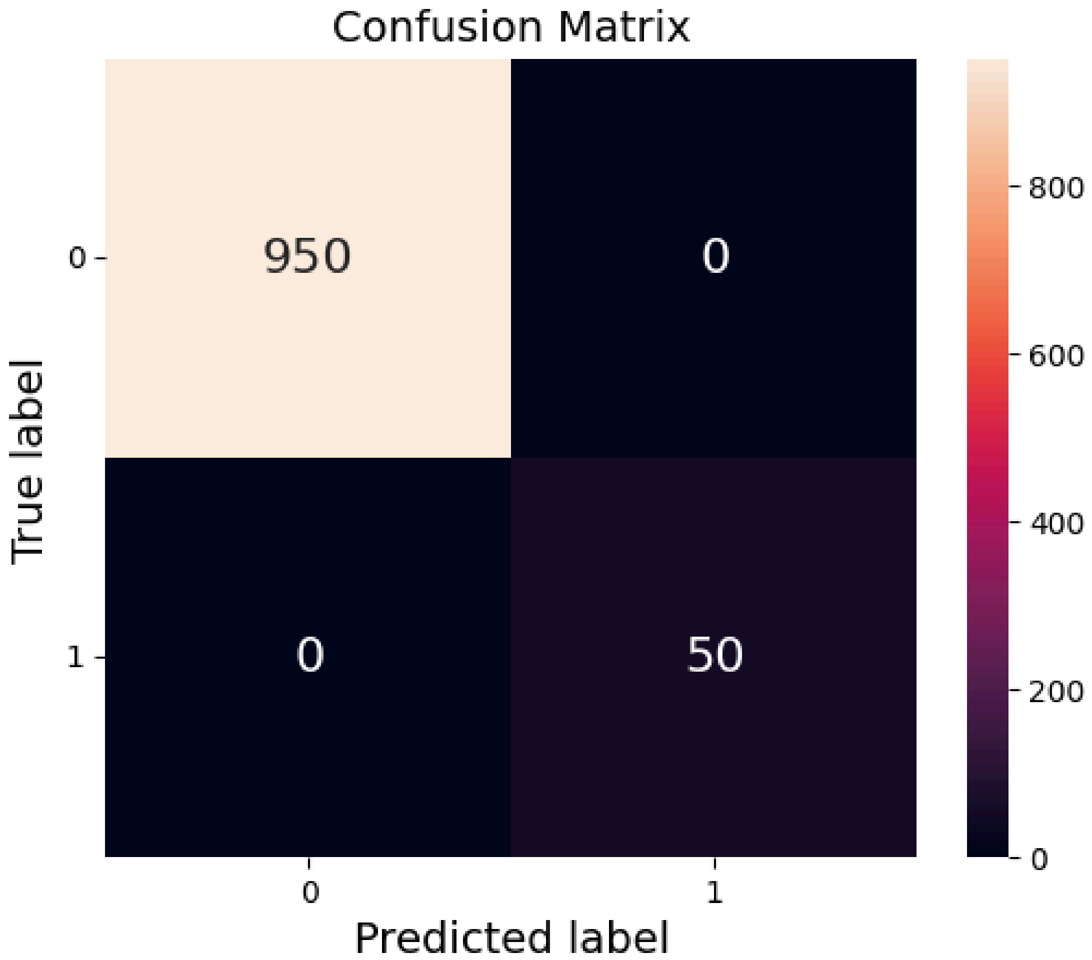
In [83]:
```
# Evaluate the model
y_true = np.zeros(len(X_test))
print("Confusion Matrix:")
print(confusion_matrix(anomalies, anomalies))

print("\nClassification Report:")
print(classification_report(anomalies, anomalies))
```

```
Confusion Matrix:
[[950   0]
 [  0  50]]

Classification Report:
              precision    recall  f1-score   support

       False       1.00      1.00      1.00       950
        True       1.00      1.00      1.00        50

    accuracy                           1.00      1000
   macro avg       1.00      1.00      1.00      1000
weighted avg       1.00      1.00      1.00      1000
```

In [84]:
```
import seaborn as sns
```

In [85]:
```
plt.figure(figsize = (6, 4.75))
sns.heatmap(confusion_matrix(anomalies, anomalies), annot = True, annot_kw
plt.xticks([0.5, 1.5],  rotation = 'horizontal')
```

```
plt.yticks([0.5, 1.5],  rotation = 'horizontal')
plt.xlabel("Predicted label", fontsize = 14)
plt.ylabel("True label", fontsize = 14)
plt.title("Confusion Matrix", fontsize = 14)
plt.grid(False)
plt.show()
```



In [ ]: