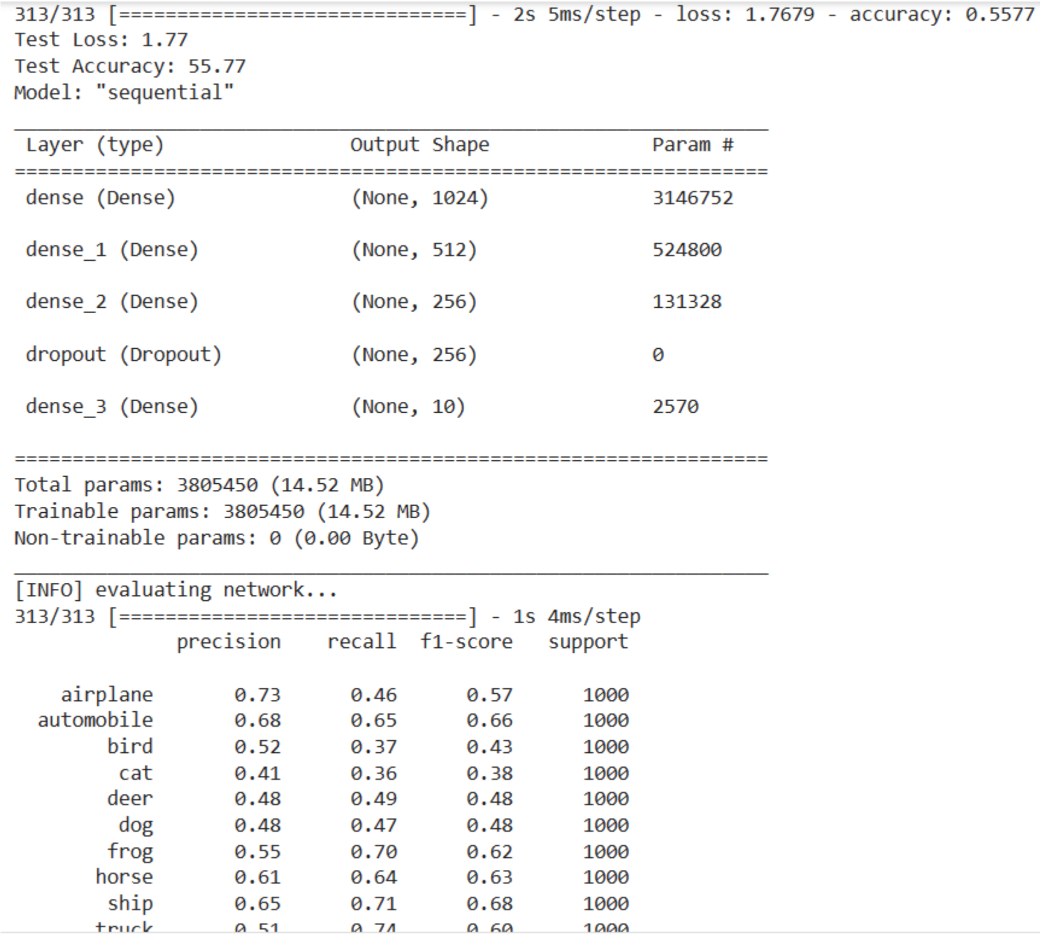
PROGRAM

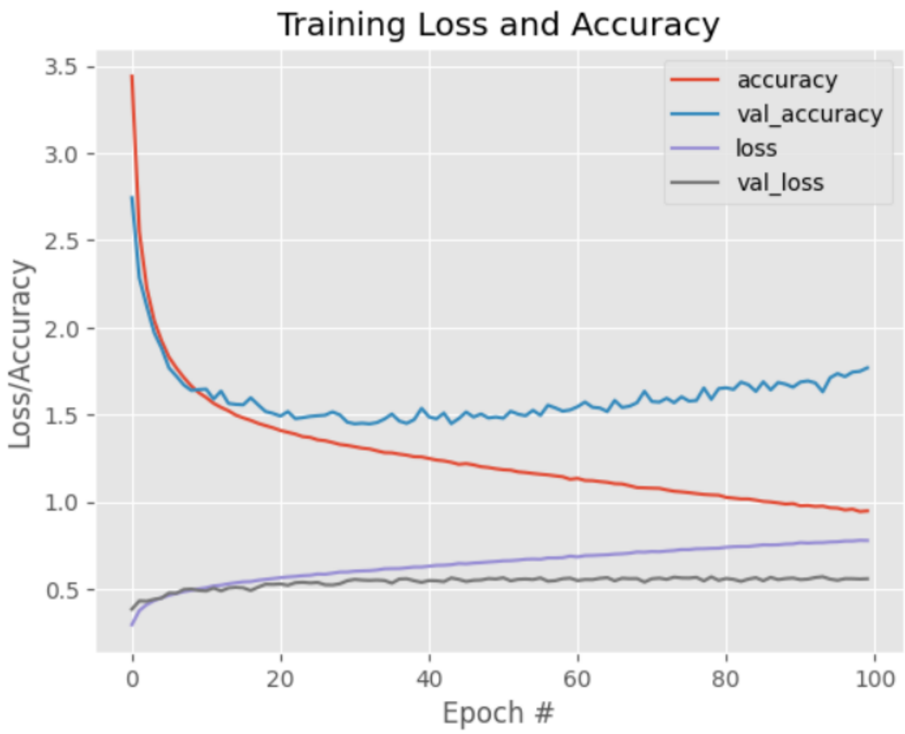
```
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras import regularizers
from tensorflow.keras.layers import Dense
from sklearn.preprocessing import LabelBinarizer
from sklearn.metrics import classification_report
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.datasets import cifar10
import matplotlib.pyplot as plt
import numpy as np

print("[INFO] loading CIFAR-10 data...")
((trainX, trainY), (testX, testY)) = cifar10.load_data()
trainX = trainX.astype("float") / 255.0
testX = testX.astype("float") / 255.0
trainX = trainX.reshape((trainX.shape[0], 3072))
testX = testX.reshape((testX.shape[0], 3072))

lb = LabelBinarizer()
trainY = lb.fit_transform(trainY)
testY = lb.transform(testY)
labelNames = ["airplane", "automobile", "bird", "cat", "deer",
 "dog", "frog", "horse", "ship", "truck"]

model = Sequential()
model.add(Dense(1024, input_shape=(3072,),
activation="relu",kernel_regularizer=regularizers.l2(0.001)))
model.add(Dense(512,
activation="relu",kernel_regularizer=regularizers.l2(0.001)))
model.add(Dense(256,
activation="relu",kernel_regularizer=regularizers.l2(0.001)))
model.add(layers.Dropout(0.5))
model.add(Dense(10, activation="softmax"))

print("[INFO] training network...")
optimizer = tf.keras.optimizers.Adam(learning_rate=0.0001)
model.compile(optimizer=optimizer, loss='categorical_crossentropy',
metrics=['accuracy'])
H = model.fit(trainX, trainY, validation_data=(testX, testY),
 epochs=100, batch_size=32)

test_loss, test_acc = model.evaluate(testX, testY)
print("Test Loss: %.2f" % test_loss)
print("Test Accuracy: %.2f" % (test_acc * 100))
model.summary()
```

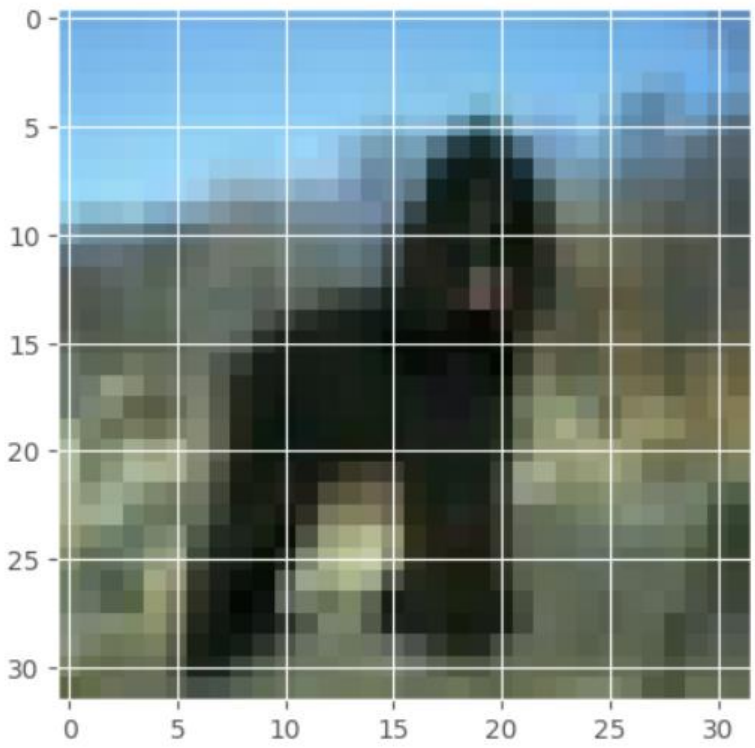
```
print("[INFO] evaluating network...")
predictions = model.predict(testX, batch_size=32)
print(classification_report(testY.argmax(axis=1),
predictions.argmax(axis=1), target_names=labelNames))

plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, 100), H.history["loss"], label="train_loss")
plt.plot(np.arange(0, 100), H.history["val_loss"], label="val_loss")
plt.plot(np.arange(0, 100), H.history["accuracy"], label="train_acc")
plt.plot(np.arange(0, 100), H.history["val_accuracy"], label="val_acc")
plt.title("Training Loss and Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(['accuracy','val_accuracy','loss','val_loss'])
plt.show()

import matplotlib.pyplot as plt
import random
n = random.randint(0, 9999)
image = testX[n].reshape(32, 32, 3)
plt.imshow(image)
plt.show()
predictions = model.predict(testX)
predicted_label = np.argmax(predictions[n])
print("Predicted Label:", predicted_label)
import matplotlib.pyplot as plt
plt.plot(H.history['accuracy'], label='Training Accuracy')
plt.plot(H.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

OUTPUT

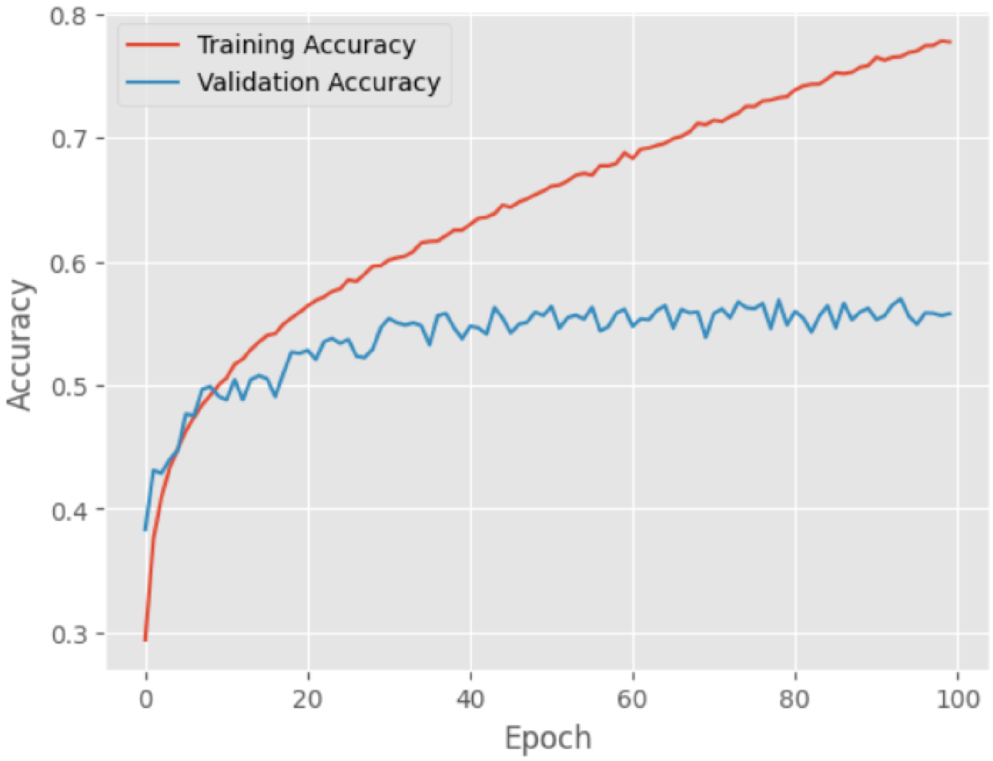


```
313/313 [==============================] - 2s 5ms/step - loss: 1.7679 - accuracy: 0.5577
Test Loss: 1.77
Test Accuracy: 55.77
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 1024)              3146752

 dense_1 (Dense)             (None, 512)               524800

 dense_2 (Dense)             (None, 256)               131328

 dropout (Dropout)           (None, 256)               0

 dense_3 (Dense)             (None, 10)                2570

=================================================================
Total params: 3805450 (14.52 MB)
Trainable params: 3805450 (14.52 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
[INFO] evaluating network...
313/313 [==============================] - 1s 4ms/step
              precision    recall  f1-score   support

    airplane       0.73      0.46      0.57      1000
  automobile       0.68      0.65      0.66      1000
        bird       0.52      0.37      0.43      1000
         cat       0.41      0.36      0.38      1000
        deer       0.48      0.49      0.48      1000
         dog       0.48      0.47      0.48      1000
        frog       0.55      0.70      0.62      1000
       horse       0.61      0.64      0.63      1000
        ship       0.65      0.71      0.68      1000
       truck       0.51      0.74      0.60      1000
```





```
313/313 [==============================] - 2s 5ms/step
Predicted Label: 7
```


Training Accuracy
Validation Accuracy
Accuracy
Epoch