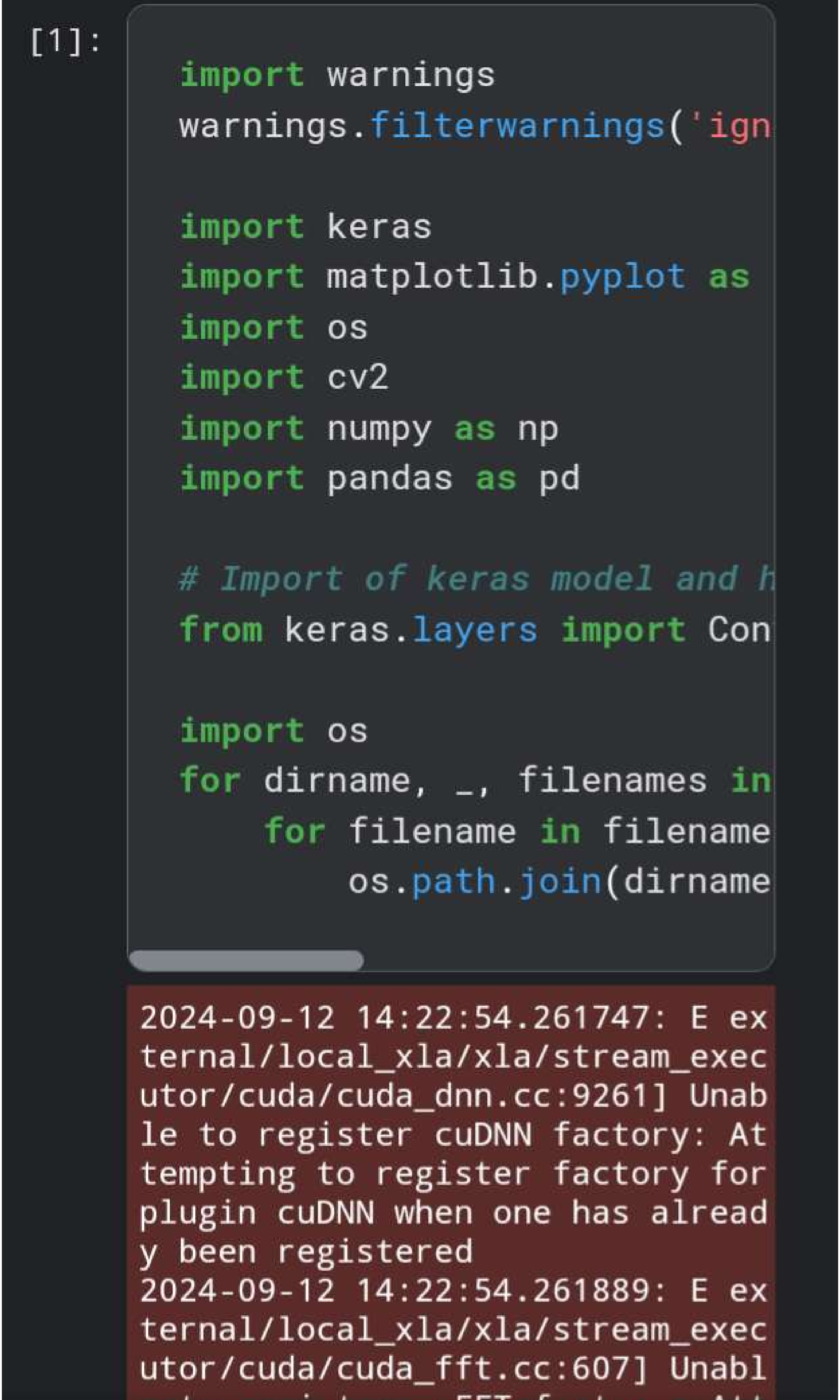
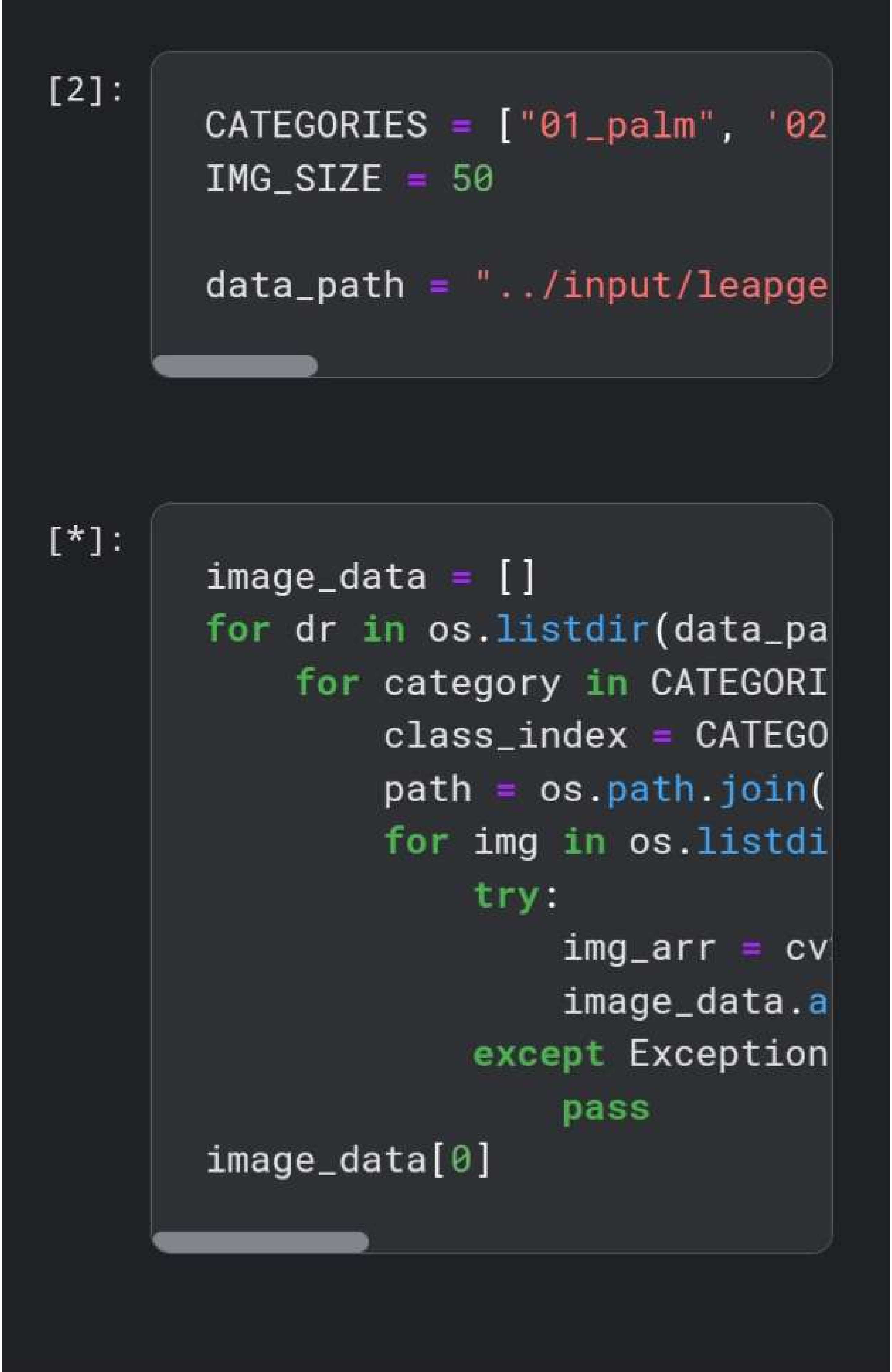


[1]:

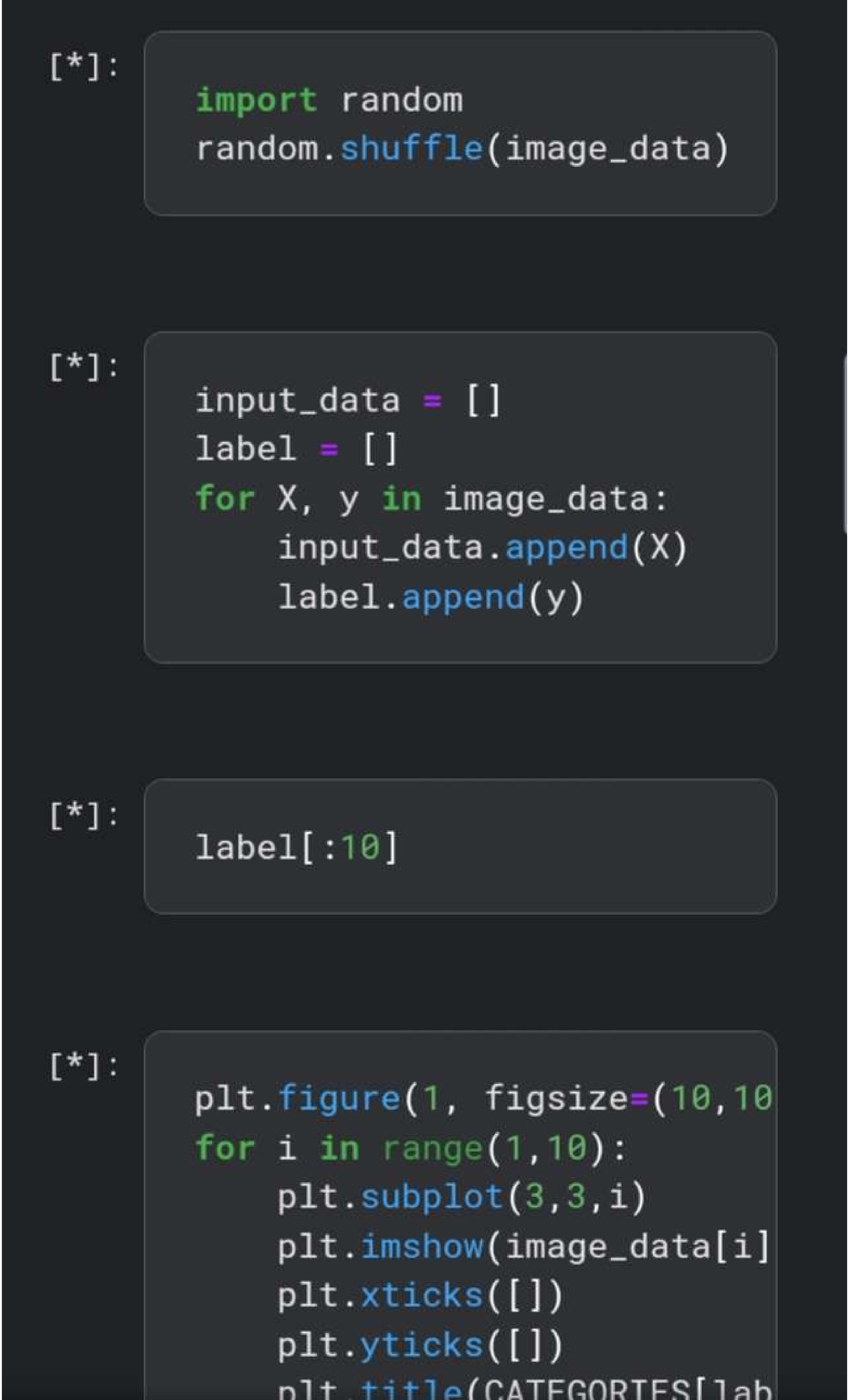
```
import warnings
warnings.filterwarnings('ign

import keras
import matplotlib.pyplot as
import os
import cv2
import numpy as np
import pandas as pd

# Import of keras model and h
from keras.layers import Con

import os
for dirname, _, filenames in
    for filename in filename
        os.path.join(dirname
```

```
2024-09-12 14:22:54.261747: E ex
ternal/local_xla/xla/stream_exec
utor/cuda/cuda_dnn.cc:9261] Unab
le to register cuDNN factory: At
tempting to register factory for
plugin cuDNN when one has alread
y been registered
2024-09-12 14:22:54.261889: E ex
ternal/local_xla/xla/stream_exec
utor/cuda/cuda_fft.cc:607] Unabl
e to register cuFFT factory: At
```



```
[2]:
CATEGORIES = ["01_palm", '02
IMG_SIZE = 50

data_path = "../input/leapge
```

```
[*]:
image_data = []
for dr in os.listdir(data_pa
    for category in CATEGORI
        class_index = CATEGO
        path = os.path.join(
        for img in os.listdi
            try:
                img_arr = cv
                image_data.a
            except Exception
                pass
image_data[0]
```



[*]:

```
import random
random.shuffle(image_data)
```

[*]:

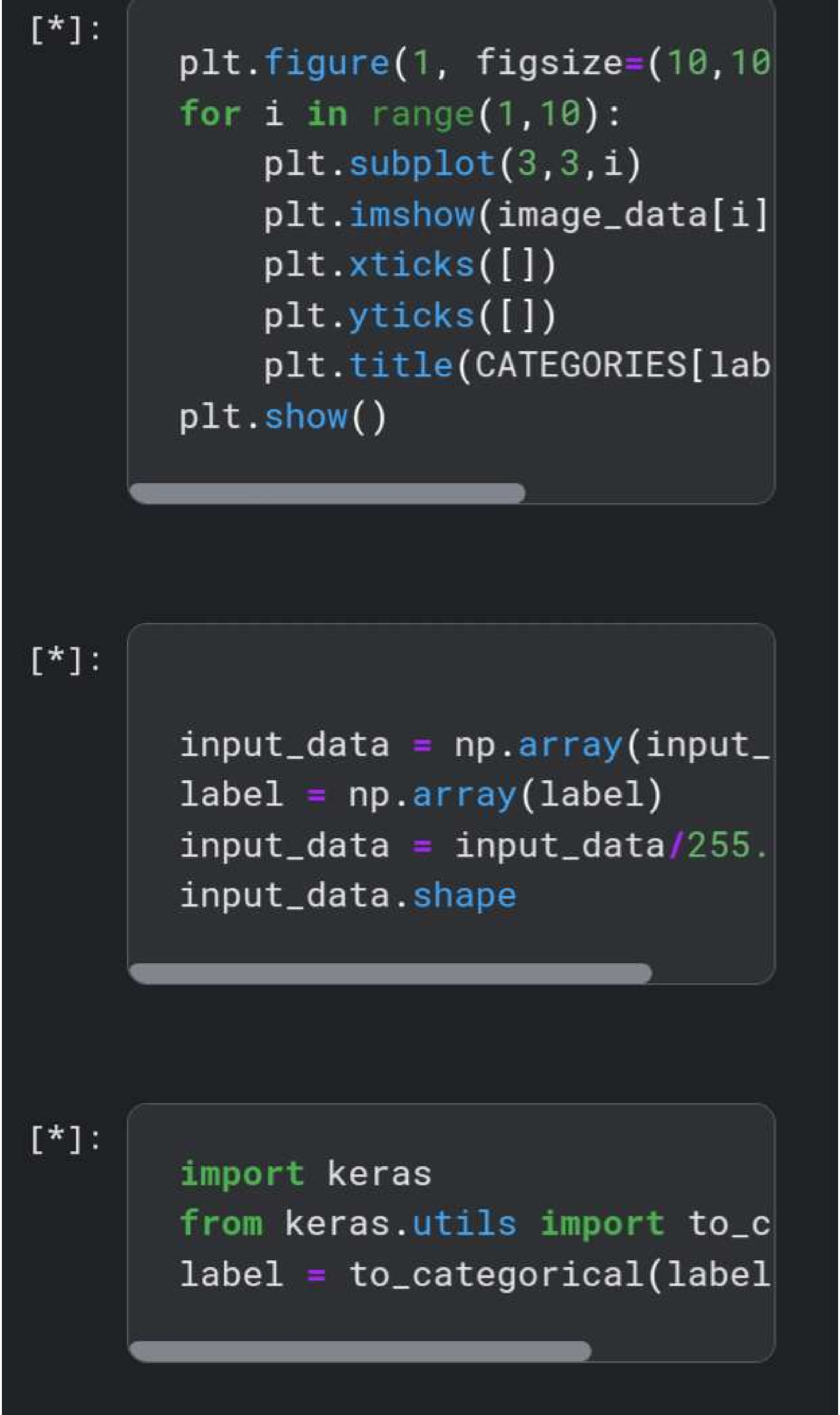
```
input_data = []
label = []
for X, y in image_data:
    input_data.append(X)
    label.append(y)
```

[*]:

```
label[:10]
```

[*]:

```
plt.figure(1, figsize=(10,10
for i in range(1,10):
    plt.subplot(3,3,i)
    plt.imshow(image_data[i]
    plt.xticks([])
    plt.yticks([])
    plt.title(CATEGORIES[lab
```



[*]:

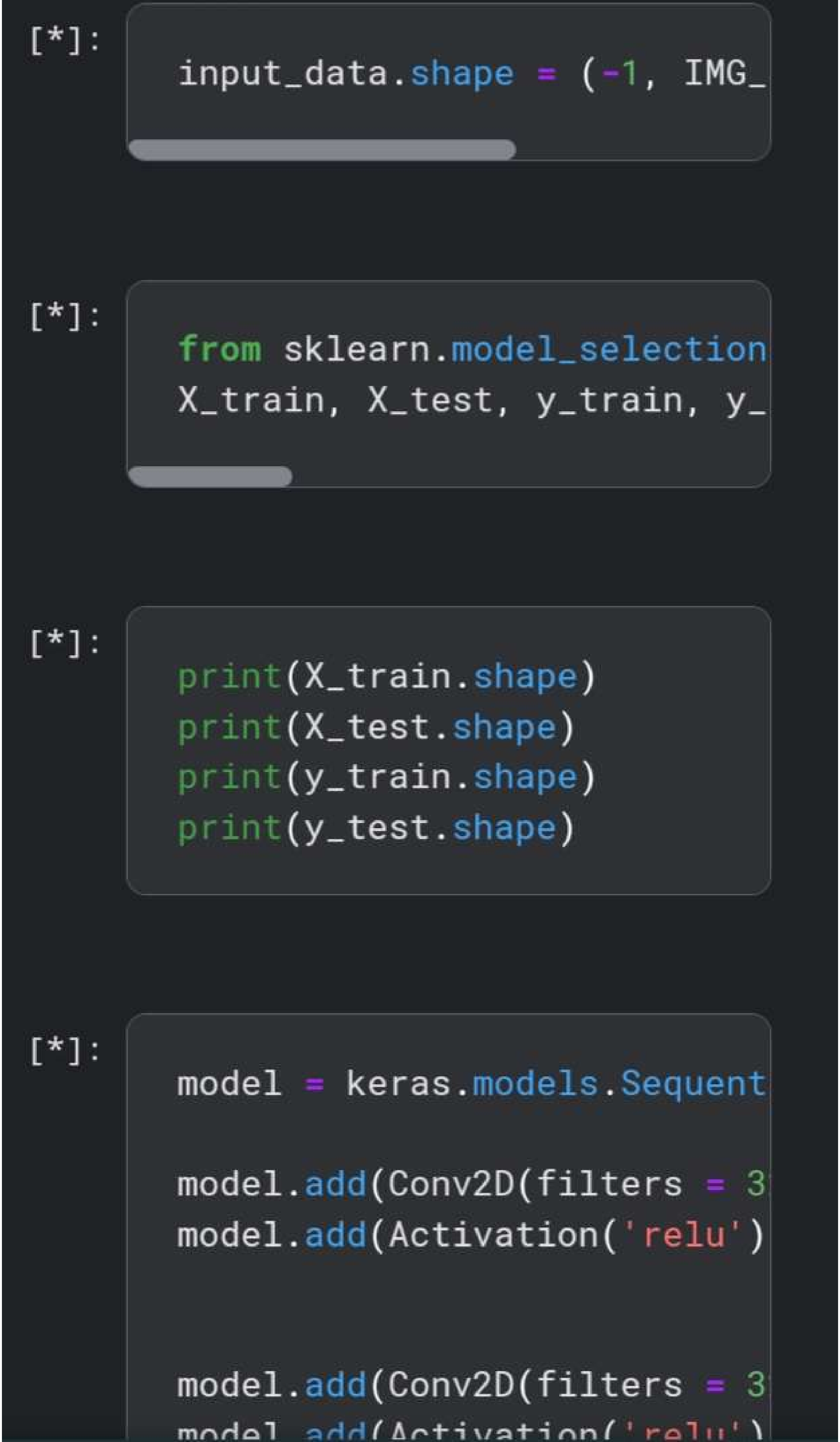
```
plt.figure(1, figsize=(10,10
for i in range(1,10):
    plt.subplot(3,3,i)
    plt.imshow(image_data[i]
    plt.xticks([])
    plt.yticks([])
    plt.title(CATEGORIES[lab
plt.show()
```

[*]:

```
input_data = np.array(input_
label = np.array(label)
input_data = input_data/255.
input_data.shape
```

[*]:

```
import keras
from keras.utils import to_c
label = to_categorical(label
```



[*]:

```
input_data.shape = (-1, IMG_
```

[*]:

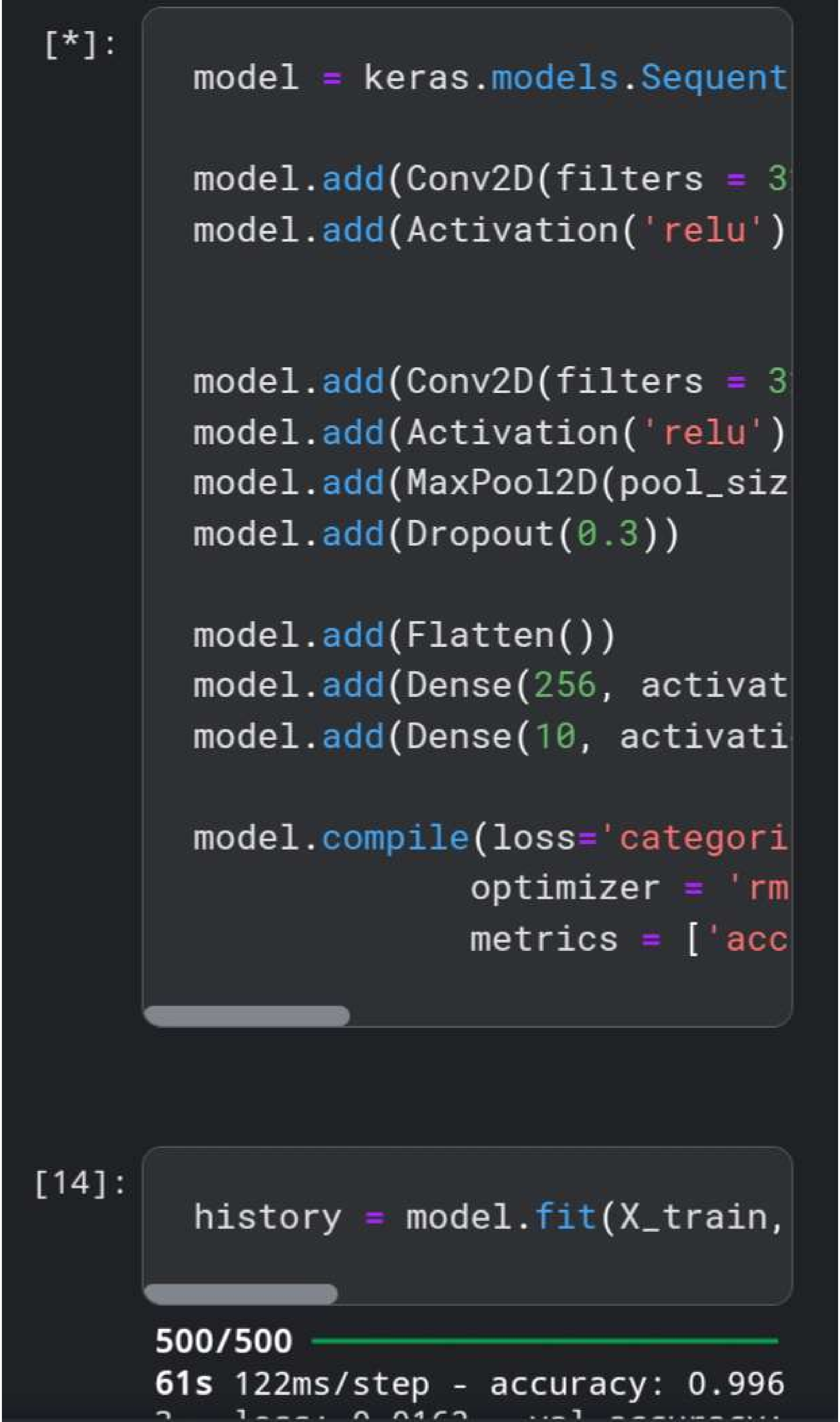
```
from sklearn.model_selection
X_train, X_test, y_train, y_
```

[*]:

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

[*]:

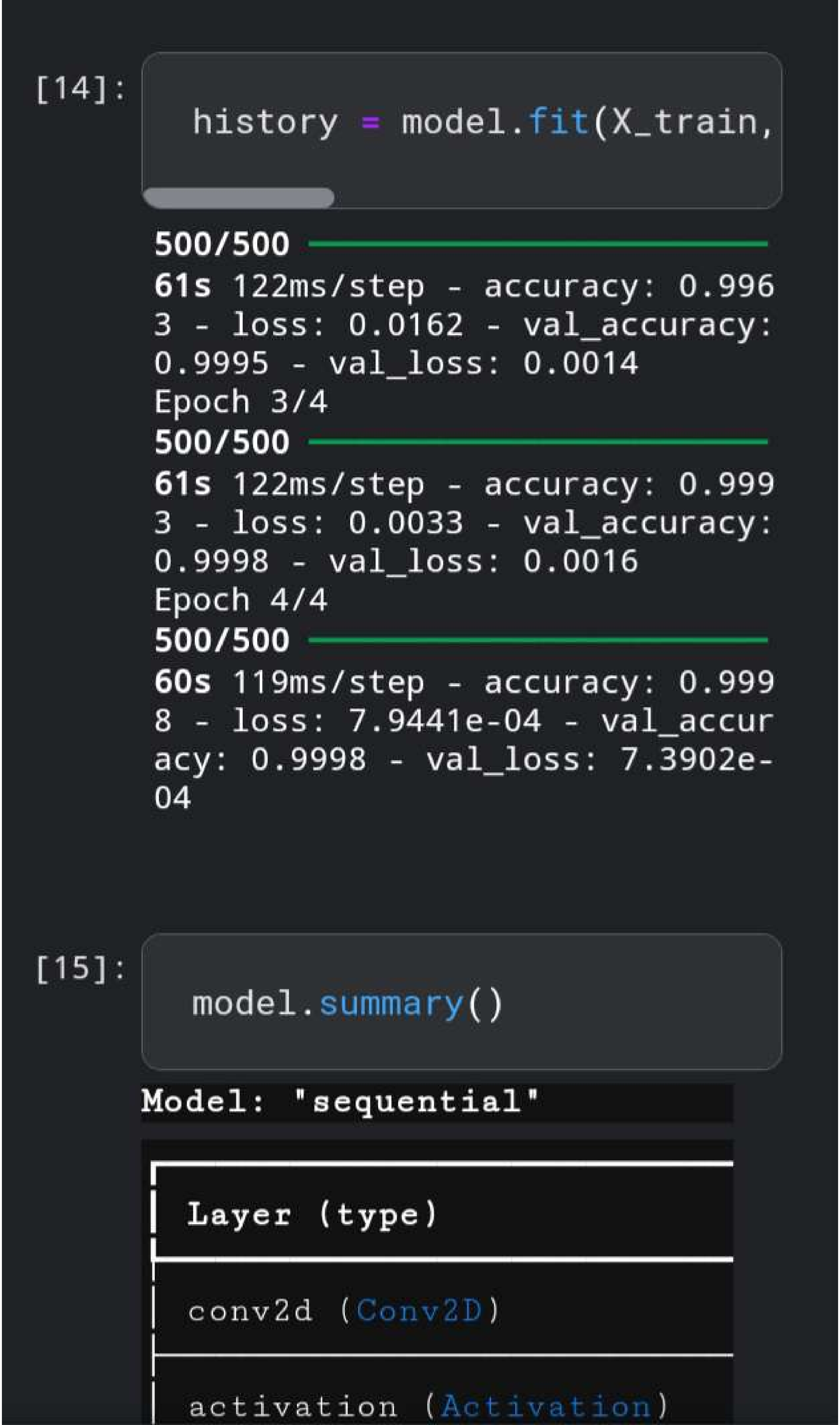
```
model = keras.models.Sequent

model.add(Conv2D(filters = 3
model.add(Activation('relu')


model.add(Conv2D(filters = 3
model.add(Activation('relu')
```



```
[*]:
model = keras.models.Sequent

model.add(Conv2D(filters = 3
model.add(Activation('relu')


model.add(Conv2D(filters = 3
model.add(Activation('relu')
model.add(MaxPool2D(pool_siz
model.add(Dropout(0.3))

model.add(Flatten())
model.add(Dense(256, activat
model.add(Dense(10, activati

model.compile(loss='categori
              optimizer = 'rm
              metrics = ['acc
```

```
[14]:
history = model.fit(X_train,
```

```
500/500 ━━━━━━━━━━━━━━━━━━━━
61s 122ms/step - accuracy: 0.996
2 - loss: 0.0162 - val_accuracy:
```



[14]:

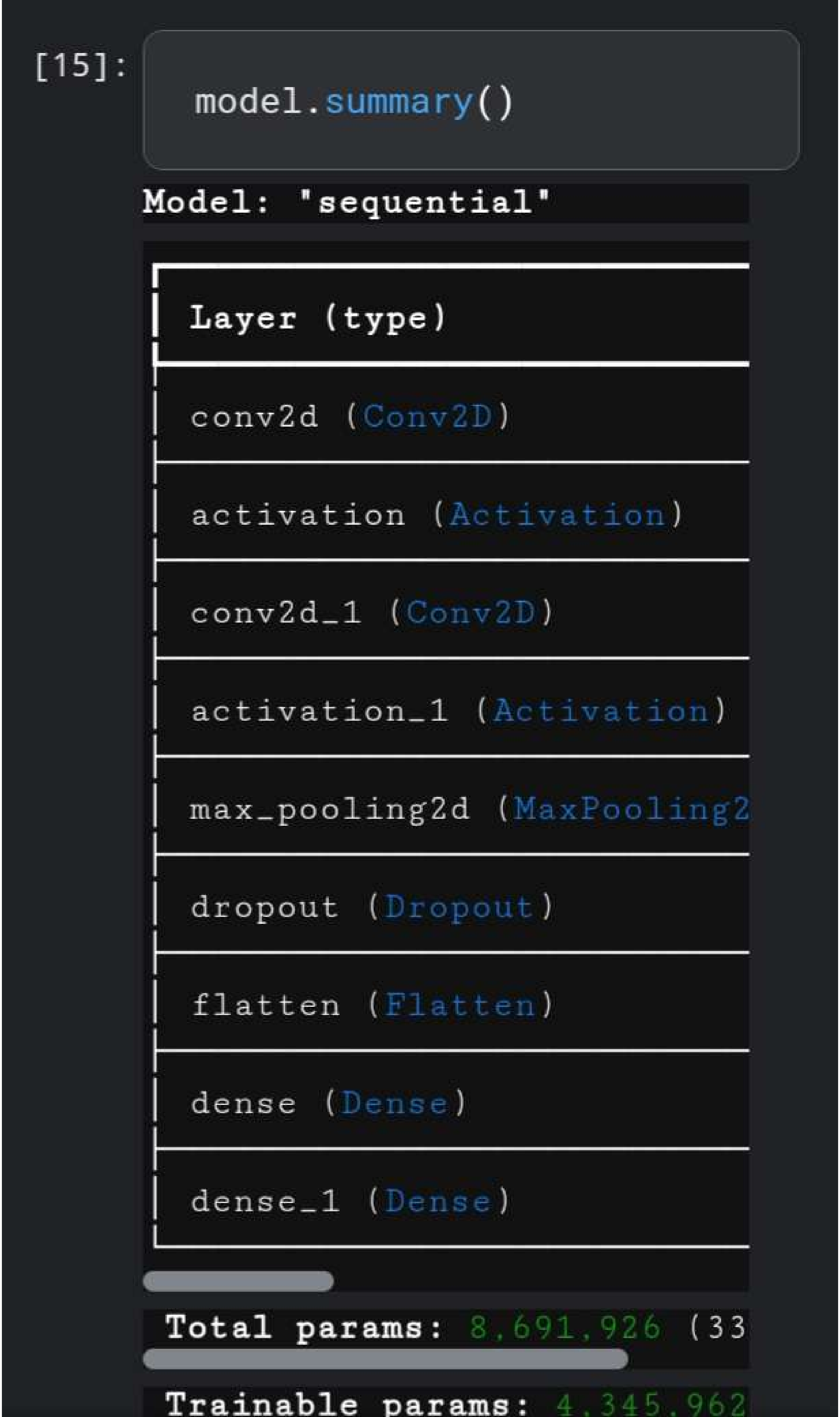
```
history = model.fit(X_train,
```

```
500/500
61s 122ms/step - accuracy: 0.996
3 - loss: 0.0162 - val_accuracy:
0.9995 - val_loss: 0.0014
Epoch 3/4
500/500
61s 122ms/step - accuracy: 0.999
3 - loss: 0.0033 - val_accuracy:
0.9998 - val_loss: 0.0016
Epoch 4/4
500/500
60s 119ms/step - accuracy: 0.999
8 - loss: 7.9441e-04 - val_accur
acy: 0.9998 - val_loss: 7.3902e-
04
```

[15]:

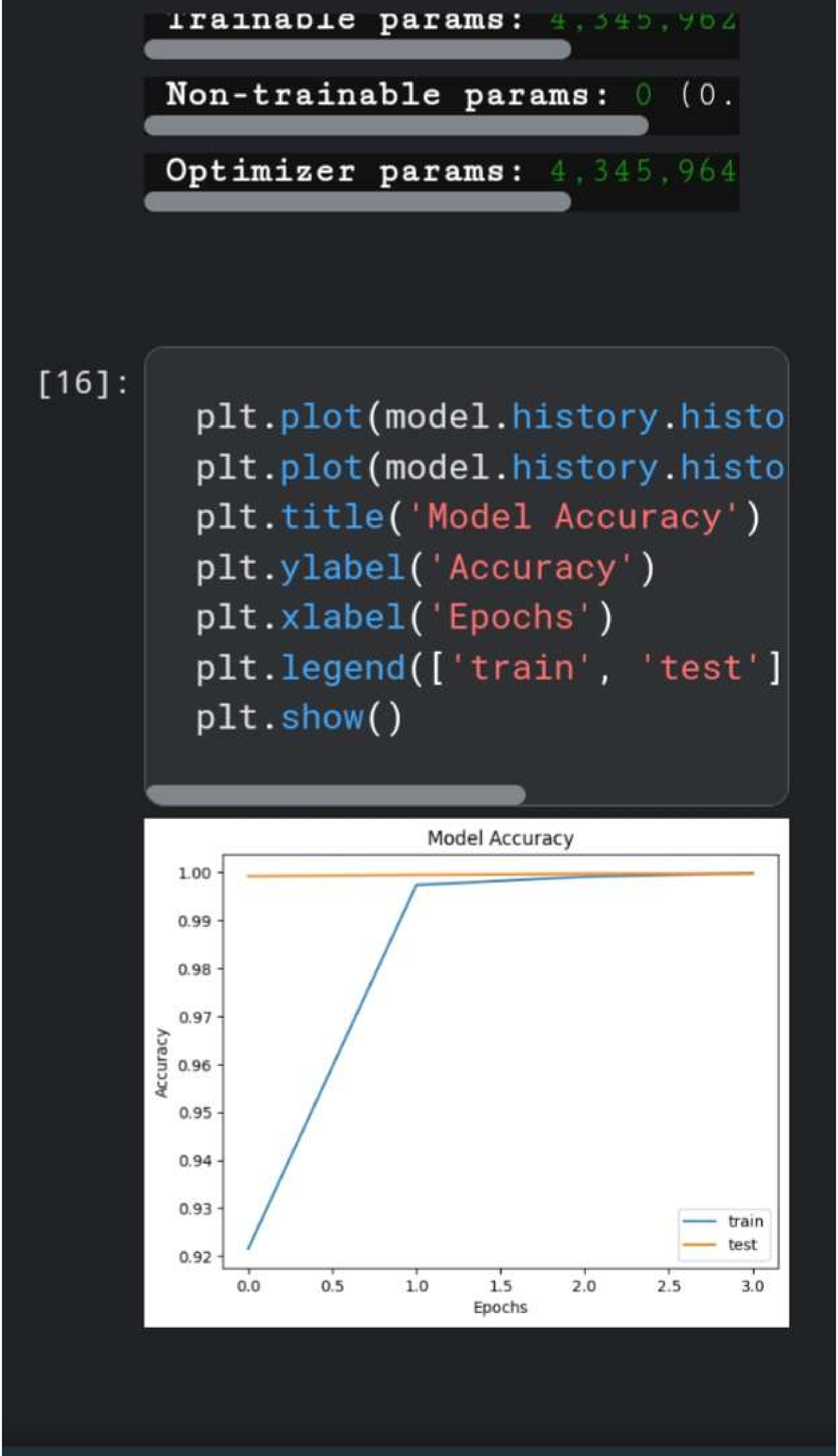
```
model.summary()
```

```
Model: "sequential"

 Layer (type)

 conv2d (Conv2D)

 activation (Activation)
```



```
[15]: model.summary()
```

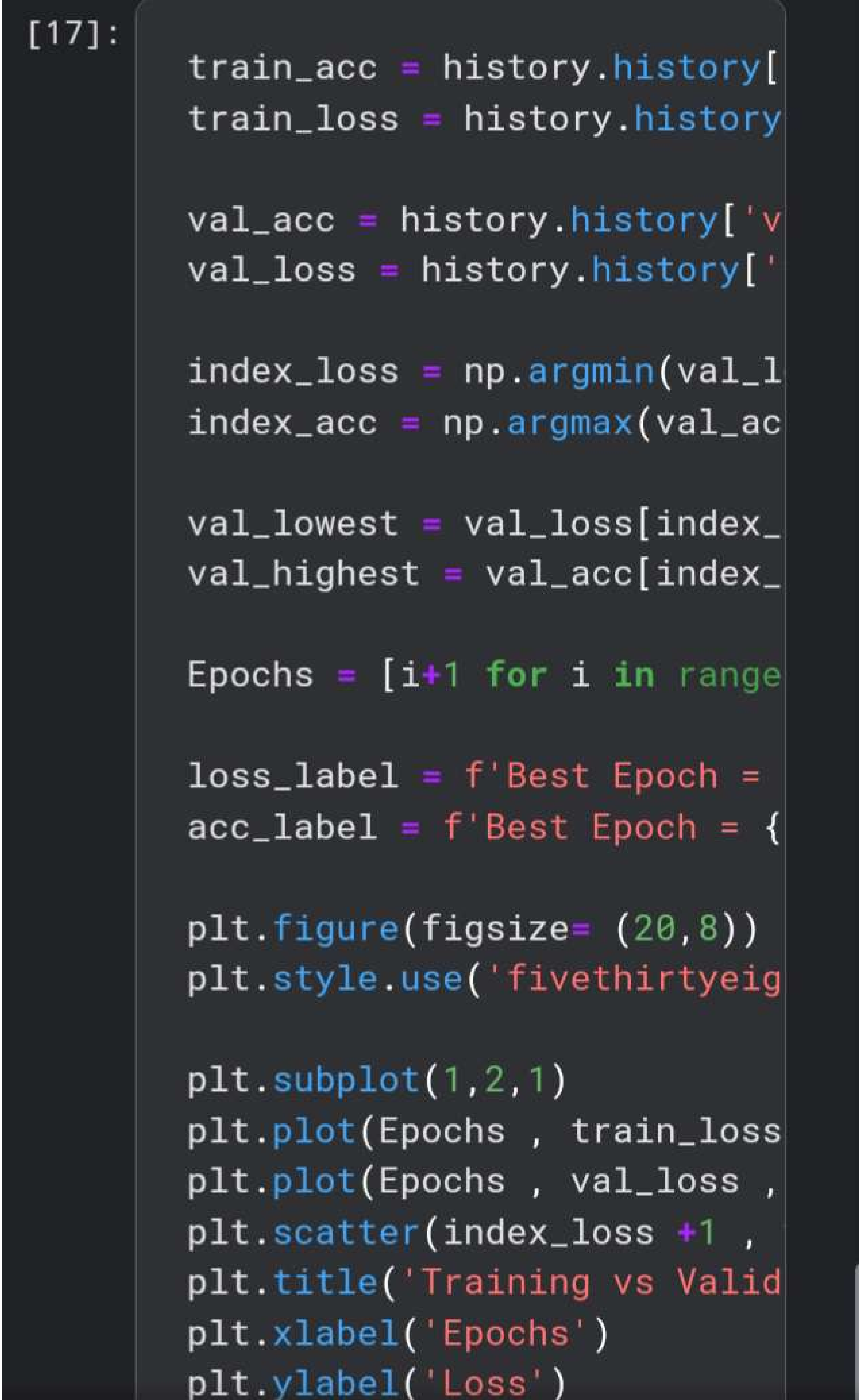
```
Model: "sequential"

Layer (type)

conv2d (Conv2D)

activation (Activation)

conv2d_1 (Conv2D)

activation_1 (Activation)

max_pooling2d (MaxPooling2

dropout (Dropout)

flatten (Flatten)

dense (Dense)

dense_1 (Dense)

 Total params: 8,691,926 (33

 Trainable params: 4,345,962
```



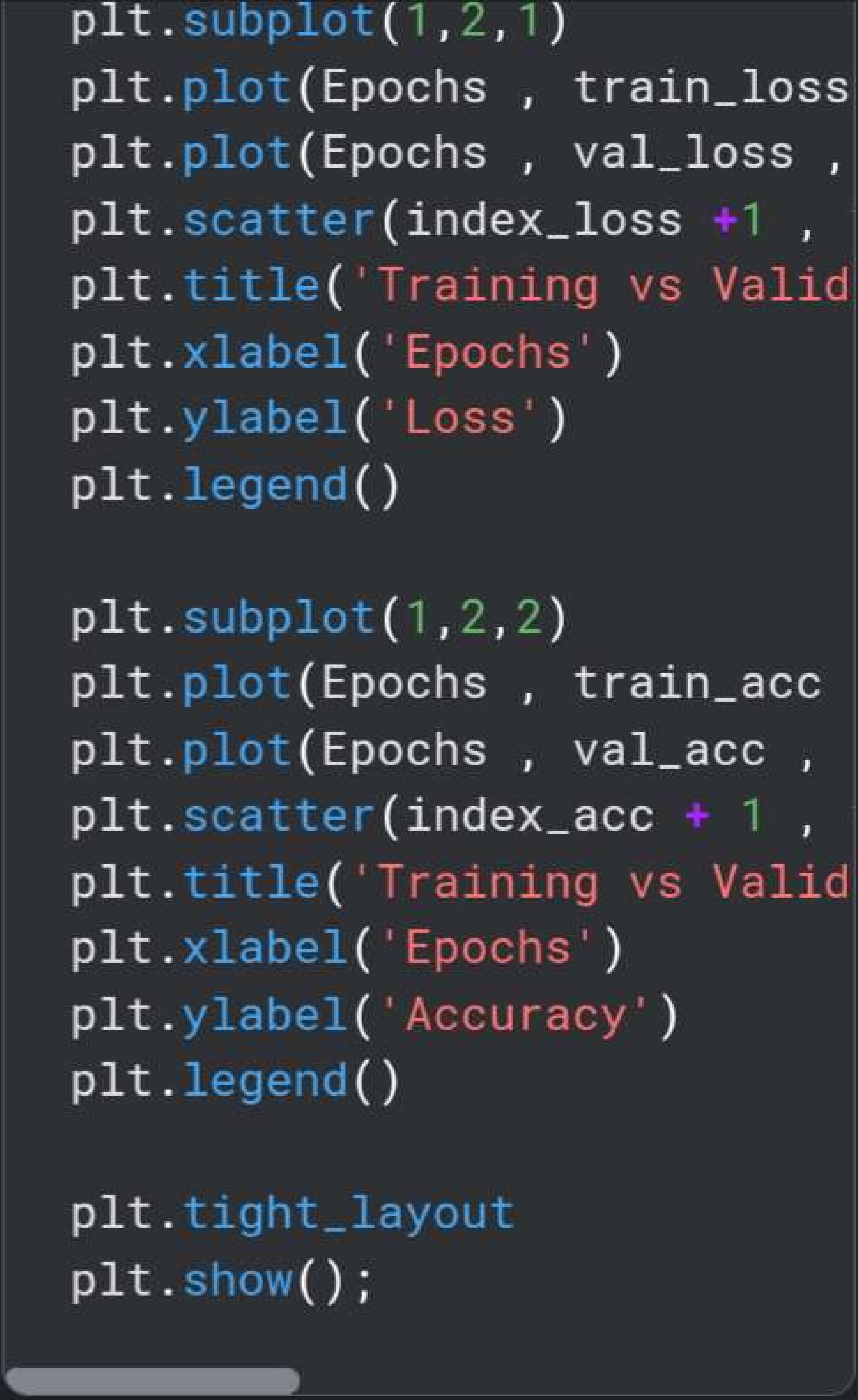
```
Trainable params: 4,345,962
Non-trainable params: 0 (0.
Optimizer params: 4,345,964
```

[16]:

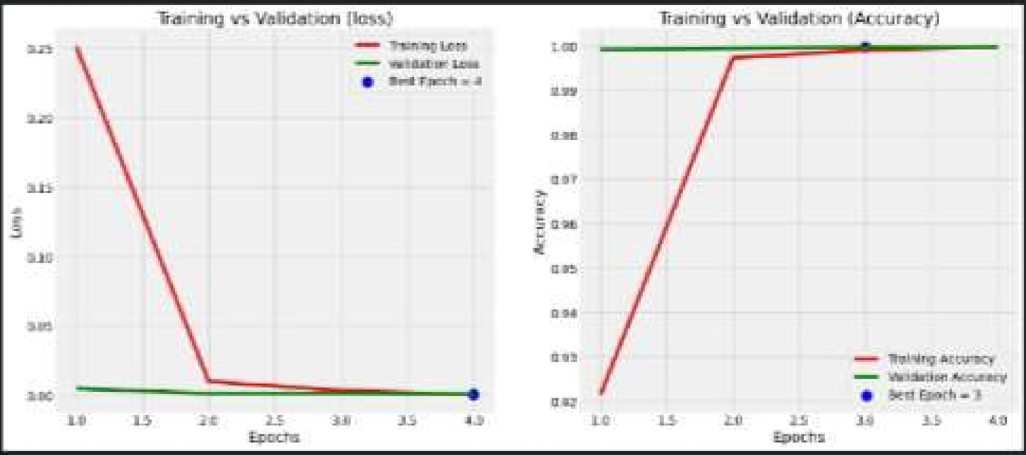
```
plt.plot(model.history.histo
plt.plot(model.history.histo
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epochs')
plt.legend(['train', 'test']
plt.show()
```

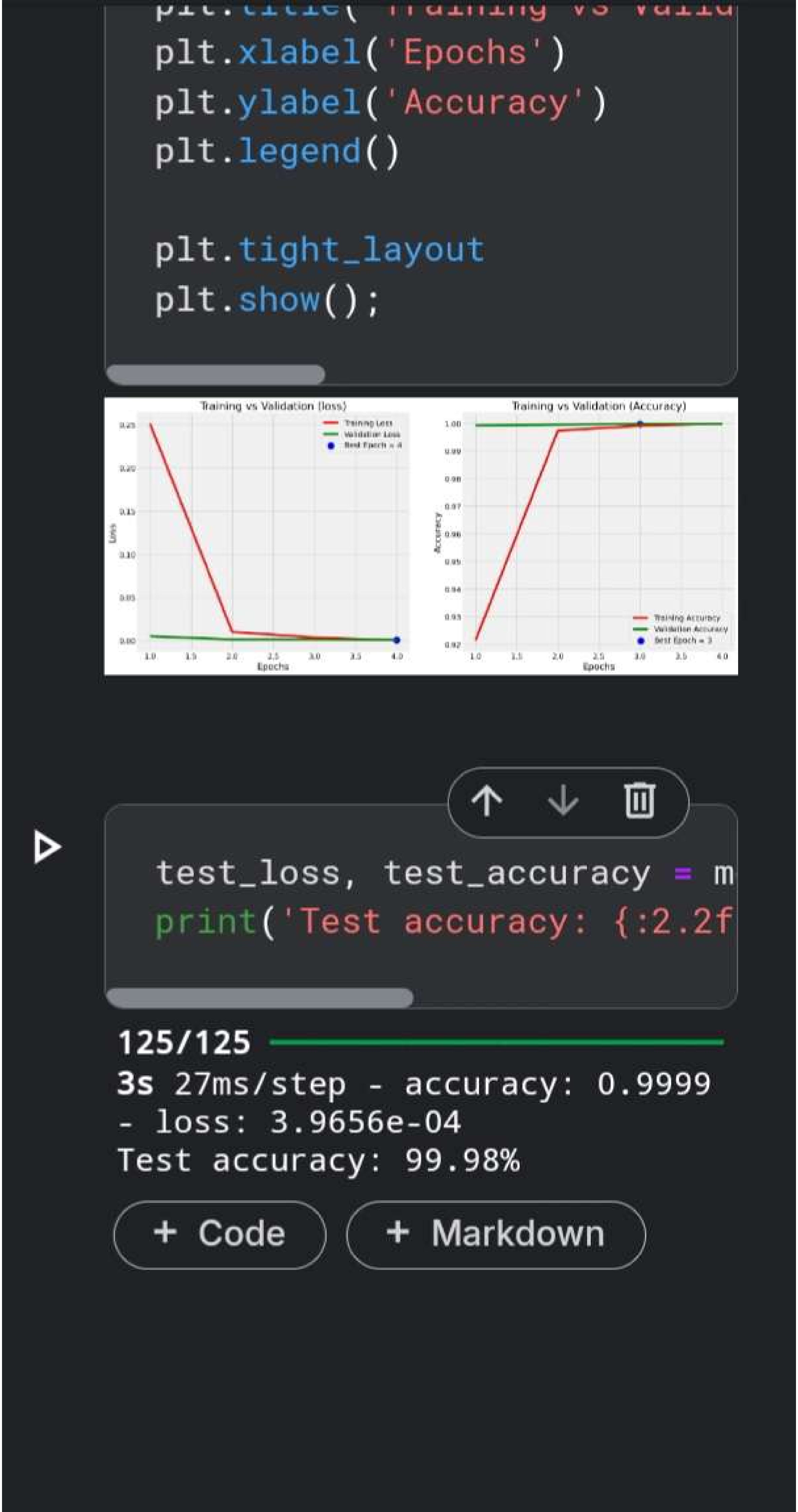


```
[17]:
train_acc = history.history[
train_loss = history.history

val_acc = history.history['v
val_loss = history.history['

index_loss = np.argmin(val_l
index_acc = np.argmax(val_ac

val_lowest = val_loss[index_
val_highest = val_acc[index_

Epochs = [i+1 for i in range

loss_label = f'Best Epoch = 
acc_label = f'Best Epoch = {

plt.figure(figsize= (20,8))
plt.style.use('fivethirtyeig

plt.subplot(1,2,1)
plt.plot(Epochs , train_loss
plt.plot(Epochs , val_loss ,
plt.scatter(index_loss +1 ,
plt.title('Training vs Valid
plt.xlabel('Epochs')
plt.ylabel('Loss')
```



```
plt.subplot(1,2,1)
plt.plot(Epochs , train_loss
plt.plot(Epochs , val_loss ,
plt.scatter(index_loss +1 ,
plt.title('Training vs Valid
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

plt.subplot(1,2,2)
plt.plot(Epochs , train_acc
plt.plot(Epochs , val_acc ,
plt.scatter(index_acc + 1 ,
plt.title('Training vs Valid
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.tight_layout
plt.show();
```





```
plt.title('Training vs Valid
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.tight_layout
plt.show();
```

```
test_loss, test_accuracy = m
print('Test accuracy: {:2.2f
```

```
125/125 ━━━━━━━━━━━━━━━━━━━━
3s 27ms/step - accuracy: 0.9999
- loss: 3.9656e-04
Test accuracy: 99.98%
```