In [21]:

```
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import matplotlib.pyplot as plt
import numpy as np
```

In [22]:

```
(X_train , y_train),(X_test , y_test) = datasets.cifar10.load_data()
X_train.shape
```

Out[22]:

```
(50000, 32, 32, 3)
```

50000 - Training Örneği
32x32 - Görsel Boyutları
3 - RGB

In [23]:

```
X_test.shape
```

Out[23]:

```
(10000, 32, 32, 3)
```

10000 - Test Örneği
32x32 - Görsel Boyutları
3 - RGB

In [24]:

```
y_train[:5]
```

Out[24]:

```
array([[6],
       [9],
       [9],
       [4],
       [1]], dtype=uint8)
```

In [25]:

```
y_train = y_train.reshape(-1,)
y_train[:5]
```

Out[25]:

```
array([6, 9, 9, 4, 1], dtype=uint8)
```

In [26]:

```
classes = ["airplane","automobile","bird","cat","deer","dog","frog","horse","ship","truc
k"]
```

In [27]:

```
classes[0]
```

Out[27]:

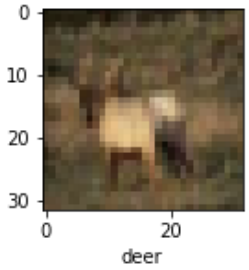
```
'airplane'
```

In [28]:

```
def gorsel_goster(X, y, index):
    plt.figure(figsize = (15,2))
    plt.imshow(X[index])
    plt.xlabel(classes[y[index]])
```

In [29]:

```
gorsel_goster(X_train, y_train, 3)
```



RGB kanalı 3 bölümden oluşur. R(kırmızı), G(yeşil) ve B(mavi).
Bu üç ayrı kanalın her biri 0 ile 255 arası bir değer alabilir ve böylece renkler oluşturulur.
Verisetimizdeki her bir gorselin değerlerini 255'e bölersek, 0 ile 1 arasında normalizasyon yapmış oluruz.

In [30]:

```
X_train = X_train / 255
X_test = X_test / 255
```

In [31]:

```
cnn = models.Sequential([
    layers.Conv2D(filters=32, kernel_size=(3, 3), activation='relu', input_shape=(32, 32
, 3)),
    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(filters=64, kernel_size=(3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(10, activation='softmax')
])
```

In [32]:

```
cnn.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

In [33]:

```
history1 = cnn.fit(X_train, y_train, epochs=100, steps_per_epoch = 50, batch_size = 3)
```

```
Epoch 1/100
50/50 [==============================] - 0s 3ms/step - loss: 2.3232 - accuracy: 0.0867
Epoch 2/100
50/50 [==============================] - 0s 3ms/step - loss: 2.2927 - accuracy: 0.1267
Epoch 3/100
50/50 [==============================] - 0s 3ms/step - loss: 2.2879 - accuracy: 0.0733
Epoch 4/100
50/50 [==============================] - 0s 3ms/step - loss: 2.2453 - accuracy: 0.1600
Epoch 5/100
50/50 [==============================] - 0s 3ms/step - loss: 2.1293 - accuracy: 0.2000
Epoch 6/100
50/50 [==============================] - 0s 3ms/step - loss: 2.0958 - accuracy: 0.1867
Epoch 7/100
50/50 [==============================] - 0s 3ms/step - loss: 2.0906 - accuracy: 0.2400
Epoch 8/100
```

```
50/50 [==============================] - 0s 3ms/step - loss: 2.0841 - accuracy: 0.2400
Epoch 9/100
50/50 [==============================] - 0s 3ms/step - loss: 2.0778 - accuracy: 0.1733
Epoch 10/100
50/50 [==============================] - 0s 3ms/step - loss: 1.9997 - accuracy: 0.2400
Epoch 11/100
50/50 [==============================] - 0s 3ms/step - loss: 1.9548 - accuracy: 0.2933
Epoch 12/100
50/50 [==============================] - 0s 3ms/step - loss: 1.9678 - accuracy: 0.2933
Epoch 13/100
50/50 [==============================] - 0s 3ms/step - loss: 1.9768 - accuracy: 0.2733
Epoch 14/100
50/50 [==============================] - 0s 3ms/step - loss: 1.8487 - accuracy: 0.2733: 0
s - loss: 1.8618 - accuracy: 0.28
Epoch 15/100
50/50 [==============================] - 0s 3ms/step - loss: 1.8185 - accuracy: 0.2400
Epoch 16/100
50/50 [==============================] - 0s 3ms/step - loss: 1.8966 - accuracy: 0.2867
Epoch 17/100
50/50 [==============================] - 0s 3ms/step - loss: 1.8247 - accuracy: 0.3600
Epoch 18/100
50/50 [==============================] - 0s 3ms/step - loss: 1.8850 - accuracy: 0.3667
Epoch 19/100
50/50 [==============================] - 0s 3ms/step - loss: 1.8713 - accuracy: 0.3267
Epoch 20/100
50/50 [==============================] - 0s 3ms/step - loss: 1.8648 - accuracy: 0.3400
Epoch 21/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6843 - accuracy: 0.3800
Epoch 22/100
50/50 [==============================] - 0s 3ms/step - loss: 1.8152 - accuracy: 0.3000
Epoch 23/100
50/50 [==============================] - 0s 3ms/step - loss: 1.9340 - accuracy: 0.2667
Epoch 24/100
50/50 [==============================] - 0s 3ms/step - loss: 1.7366 - accuracy: 0.3400
Epoch 25/100
50/50 [==============================] - 0s 3ms/step - loss: 1.7417 - accuracy: 0.3867
Epoch 26/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6894 - accuracy: 0.3933
Epoch 27/100
50/50 [==============================] - 0s 3ms/step - loss: 1.8444 - accuracy: 0.2933
Epoch 28/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6634 - accuracy: 0.3467
Epoch 29/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5731 - accuracy: 0.3867
Epoch 30/100
50/50 [==============================] - 0s 3ms/step - loss: 1.7307 - accuracy: 0.3800
Epoch 31/100
50/50 [==============================] - 0s 4ms/step - loss: 1.7811 - accuracy: 0.3333
Epoch 32/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6965 - accuracy: 0.3800
Epoch 33/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6000 - accuracy: 0.4467
Epoch 34/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6356 - accuracy: 0.4600
Epoch 35/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6384 - accuracy: 0.3600
Epoch 36/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6417 - accuracy: 0.3600
Epoch 37/100
50/50 [==============================] - 0s 3ms/step - loss: 1.7579 - accuracy: 0.3133
Epoch 38/100
50/50 [==============================] - 0s 3ms/step - loss: 1.7449 - accuracy: 0.3800
Epoch 39/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5062 - accuracy: 0.4200
Epoch 40/100
50/50 [==============================] - 0s 3ms/step - loss: 1.7103 - accuracy: 0.3467
Epoch 41/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5640 - accuracy: 0.3867
Epoch 42/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4983 - accuracy: 0.4133
Epoch 43/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6881 - accuracy: 0.3867
```

```
Epoch 44/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5130 - accuracy: 0.4333
Epoch 45/100
50/50 [==============================] - 0s 3ms/step - loss: 1.7060 - accuracy: 0.3200
Epoch 46/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6156 - accuracy: 0.4267
Epoch 47/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5270 - accuracy: 0.4933
Epoch 48/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6167 - accuracy: 0.4200
Epoch 49/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6423 - accuracy: 0.4333
Epoch 50/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5880 - accuracy: 0.4200
Epoch 51/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6583 - accuracy: 0.4067
Epoch 52/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5393 - accuracy: 0.4933
Epoch 53/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4796 - accuracy: 0.4533
Epoch 54/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5692 - accuracy: 0.3933
Epoch 55/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5053 - accuracy: 0.4733
Epoch 56/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5005 - accuracy: 0.4467
Epoch 57/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5171 - accuracy: 0.4800
Epoch 58/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5014 - accuracy: 0.4733
Epoch 59/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6768 - accuracy: 0.4133
Epoch 60/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5879 - accuracy: 0.4067
Epoch 61/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4625 - accuracy: 0.4867
Epoch 62/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6781 - accuracy: 0.3533
Epoch 63/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5335 - accuracy: 0.4667
Epoch 64/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5274 - accuracy: 0.4667
Epoch 65/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5733 - accuracy: 0.3667
Epoch 66/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4050 - accuracy: 0.4533
Epoch 67/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4626 - accuracy: 0.4733
Epoch 68/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4594 - accuracy: 0.5267
Epoch 69/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4733 - accuracy: 0.5000
Epoch 70/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4070 - accuracy: 0.4400
Epoch 71/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6137 - accuracy: 0.3600
Epoch 72/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5342 - accuracy: 0.4067
Epoch 73/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5983 - accuracy: 0.4400
Epoch 74/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5155 - accuracy: 0.4667
Epoch 75/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4678 - accuracy: 0.4800
Epoch 76/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5565 - accuracy: 0.4667
Epoch 77/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5661 - accuracy: 0.4667
Epoch 78/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4939 - accuracy: 0.4467
Epoch 79/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4734 - accuracy: 0.5133
```

```
Epoch 80/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5458 - accuracy: 0.4600
Epoch 81/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4594 - accuracy: 0.4467
Epoch 82/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4142 - accuracy: 0.4667
Epoch 83/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3616 - accuracy: 0.5467
Epoch 84/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3426 - accuracy: 0.5067
Epoch 85/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5228 - accuracy: 0.4333
Epoch 86/100
50/50 [==============================] - 0s 3ms/step - loss: 1.7311 - accuracy: 0.3533
Epoch 87/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3942 - accuracy: 0.4867
Epoch 88/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3896 - accuracy: 0.4733
Epoch 89/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3035 - accuracy: 0.5133
Epoch 90/100
50/50 [==============================] - 0s 3ms/step - loss: 1.6444 - accuracy: 0.4533
Epoch 91/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3893 - accuracy: 0.4933
Epoch 92/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3792 - accuracy: 0.4667
Epoch 93/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5225 - accuracy: 0.4867
Epoch 94/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5122 - accuracy: 0.4200
Epoch 95/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3693 - accuracy: 0.4867
Epoch 96/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3129 - accuracy: 0.5400
Epoch 97/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5738 - accuracy: 0.4000
Epoch 98/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4718 - accuracy: 0.4133
Epoch 99/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4433 - accuracy: 0.4867
Epoch 100/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4252 - accuracy: 0.4933
```

In [34]:

```
cnn.evaluate(X_test,y_test)
```

```
313/313 [==============================] - 1s 4ms/step - loss: 1.4207 - accuracy: 0.4818
```

Out[34]:

```
[1.420654058456421, 0.48179998993873596]
```

In [35]:

```
y_pred = cnn.predict(X_test)
y_pred[:5]
```

Out[35]:

```
array([[2.93262098e-02, 2.11419770e-03, 1.50769338e-01, 3.44102323e-01,
        2.65090019e-02, 2.87988842e-01, 9.53981131e-02, 1.85567848e-02,
        3.34677175e-02, 1.17674051e-02],
       [2.30432853e-01, 3.06034293e-02, 2.84814130e-04, 2.27278251e-05,
        7.08096195e-05, 3.59747190e-07, 5.48025014e-07, 9.57089696e-07,
        7.37349272e-01, 1.23428891e-03],
       [2.89973170e-01, 9.70317274e-02, 5.22906426e-03, 2.34699412e-03,
        2.65229191e-03, 2.71224009e-04, 1.15850155e-04, 7.56267633e-04,
        5.69388688e-01, 3.22346464e-02],
       [4.51340854e-01, 1.16449632e-02, 1.16970073e-02, 9.15140030e-04,
        3.01827933e-03, 4.81925599e-05, 7.62591590e-05, 1.93701868e-04,
        5.16228139e-01, 4.83738445e-03],
       [3.79529479e-03, 1.92758814e-03, 9.48157236e-02, 3.40892851e-01,
```

```
        8.60930011e-02, 2.76945055e-01, 1.72627524e-01, 1.75131522e-02,
        3.11403279e-03, 2.27569952e-03]], dtype=float32)
```

In [36]:

```
y_classes = [np.argmax(i) for i in y_pred]
y_classes[:5]
```

Out[36]:

```
[3, 8, 8, 8, 3]
```

In [37]:

```
y_test[:5]
```

Out[37]:

```
array([[3],
       [8],
       [8],
       [0],
       [6]], dtype=uint8)
```

In [38]:

```
from sklearn.metrics import confusion_matrix , classification_report
print("Sınıflandırma Sonucu : \n" , classification_report(y_test , y_classes))
```

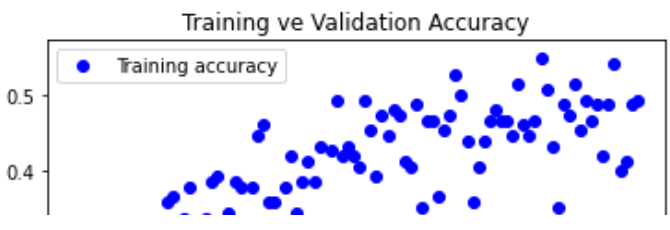
```
Sınıflandırma Sonucu :
              precision    recall  f1-score   support

           0       0.48      0.57      0.52      1000
           1       0.64      0.63      0.63      1000
           2       0.38      0.33      0.35      1000
           3       0.33      0.33      0.33      1000
           4       0.58      0.11      0.18      1000
           5       0.41      0.52      0.46      1000
           6       0.65      0.52      0.58      1000
           7       0.47      0.67      0.55      1000
           8       0.49      0.63      0.55      1000
           9       0.50      0.52      0.51      1000

    accuracy                           0.48     10000
   macro avg       0.49      0.48      0.47     10000
weighted avg       0.49      0.48      0.47     10000
```

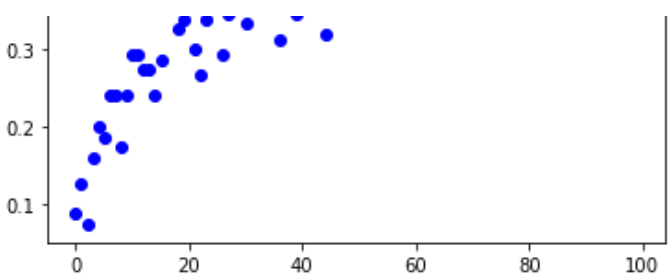
In [39]:

```
import matplotlib.pyplot as plt
%matplotlib inline

accuracy = history1.history['accuracy']
loss = history1.history['loss']
epochs = range(len(accuracy))

plt.plot(epochs, accuracy, 'bo', label='Training accuracy')
plt.title('Training ve Validation Accuracy')
plt.legend()
plt.figure()
```

Out[39]:

```
<Figure size 432x288 with 0 Axes>
```






```
<Figure size 432x288 with 0 Axes>
```

In [ ]:

```
```