In [1]:

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

In [2]:

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

Out[2]:

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

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

In [3]:

```
X_test.shape
```

Out[3]:

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

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

In [4]:

```
y_train[:5]
```

Out[4]:

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

In [5]:

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

Out[5]:

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

In [6]:

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

In [7]:

```
classes[0]
```

Out[7]:

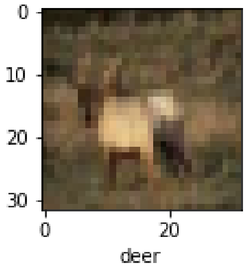
```
'airplane'
```

In [8]:

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

In [9]:

```
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 [10]:

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

In [11]:

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

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

In [12]:

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

In [23]:

```
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: 1.4406 - accuracy: 0.4867
Epoch 2/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4215 - accuracy: 0.4800
Epoch 3/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3824 - accuracy: 0.4733
Epoch 4/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3392 - accuracy: 0.4667
Epoch 5/100
50/50 [==============================] - ETA: 0s - loss: 1.3587 - accuracy: 0.46 - 0s 4ms
/step - loss: 1.3217 - accuracy: 0.4867
Epoch 6/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3229 - accuracy: 0.5333
Epoch 7/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3189 - accuracy: 0.6200
Epoch 8/100
50/50 [==============================] - 0s 4ms/step - loss: 1.5880 - accuracy: 0.4467
Epoch 9/100
```

```
50/50 [==============================] - 0s 4ms/step - loss: 1.2406 - accuracy: 0.5467
Epoch 10/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4371 - accuracy: 0.5267
Epoch 11/100
50/50 [==============================] - 0s 4ms/step - loss: 1.5104 - accuracy: 0.4200
Epoch 12/100
50/50 [==============================] - 0s 4ms/step - loss: 1.5579 - accuracy: 0.3933
Epoch 13/100
50/50 [==============================] - 0s 4ms/step - loss: 1.5395 - accuracy: 0.4933
Epoch 14/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3717 - accuracy: 0.4733
Epoch 15/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4708 - accuracy: 0.4667
Epoch 16/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4808 - accuracy: 0.4667
Epoch 17/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3182 - accuracy: 0.5400
Epoch 18/100
50/50 [==============================] - 0s 4ms/step - loss: 1.6434 - accuracy: 0.4200
Epoch 19/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4549 - accuracy: 0.4933
Epoch 20/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3960 - accuracy: 0.5000
Epoch 21/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4631 - accuracy: 0.4933
Epoch 22/100
50/50 [==============================] - 0s 4ms/step - loss: 1.5755 - accuracy: 0.5000
Epoch 23/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5052 - accuracy: 0.4667
Epoch 24/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3611 - accuracy: 0.5267
Epoch 25/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4826 - accuracy: 0.4600
Epoch 26/100
50/50 [==============================] - 0s 4ms/step - loss: 1.6180 - accuracy: 0.4533
Epoch 27/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5191 - accuracy: 0.4200
Epoch 28/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4487 - accuracy: 0.4667
Epoch 29/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4312 - accuracy: 0.5200
Epoch 30/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4455 - accuracy: 0.4733
Epoch 31/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3228 - accuracy: 0.6067
Epoch 32/100
50/50 [==============================] - ETA: 0s - loss: 1.4879 - accuracy: 0.44 - 0s 4ms
/step - loss: 1.4358 - accuracy: 0.4867
Epoch 33/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4667 - accuracy: 0.5400
Epoch 34/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4491 - accuracy: 0.4800
Epoch 35/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5311 - accuracy: 0.3933
Epoch 36/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3972 - accuracy: 0.4933
Epoch 37/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4099 - accuracy: 0.5600
Epoch 38/100
50/50 [==============================] - 0s 4ms/step - loss: 1.5013 - accuracy: 0.4267
Epoch 39/100
50/50 [==============================] - 0s 3ms/step - loss: 1.2970 - accuracy: 0.5133
Epoch 40/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3323 - accuracy: 0.5467
Epoch 41/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3873 - accuracy: 0.5133
Epoch 42/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4981 - accuracy: 0.5000
Epoch 43/100
50/50 [==============================] - 0s 5ms/step - loss: 1.3222 - accuracy: 0.5000
Epoch 44/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3531 - accuracy: 0.4867
```

```
Epoch 45/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4318 - accuracy: 0.4800
Epoch 46/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3715 - accuracy: 0.4733
Epoch 47/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4491 - accuracy: 0.4800
Epoch 48/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4547 - accuracy: 0.5333
Epoch 49/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4127 - accuracy: 0.5000
Epoch 50/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4938 - accuracy: 0.4533
Epoch 51/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3011 - accuracy: 0.5667
Epoch 52/100
50/50 [==============================] - 0s 3ms/step - loss: 1.2922 - accuracy: 0.5467
Epoch 53/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3783 - accuracy: 0.4467
Epoch 54/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4443 - accuracy: 0.4800
Epoch 55/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4078 - accuracy: 0.4600
Epoch 56/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3725 - accuracy: 0.4867
Epoch 57/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4603 - accuracy: 0.4867
Epoch 58/100
50/50 [==============================] - 0s 3ms/step - loss: 1.2559 - accuracy: 0.6067
Epoch 59/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3957 - accuracy: 0.4800
Epoch 60/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3220 - accuracy: 0.5400
Epoch 61/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3914 - accuracy: 0.4600
Epoch 62/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5410 - accuracy: 0.4333
Epoch 63/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4601 - accuracy: 0.4867
Epoch 64/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3348 - accuracy: 0.5533
Epoch 65/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4849 - accuracy: 0.4333
Epoch 66/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4786 - accuracy: 0.4933
Epoch 67/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3212 - accuracy: 0.5333
Epoch 68/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3409 - accuracy: 0.5267
Epoch 69/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4116 - accuracy: 0.4733
Epoch 70/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3217 - accuracy: 0.5333
Epoch 71/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3602 - accuracy: 0.5467
Epoch 72/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4639 - accuracy: 0.5200
Epoch 73/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3186 - accuracy: 0.4867
Epoch 74/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3899 - accuracy: 0.5600
Epoch 75/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4611 - accuracy: 0.5000
Epoch 76/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4228 - accuracy: 0.4600
Epoch 77/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3953 - accuracy: 0.4933
Epoch 78/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4335 - accuracy: 0.4667
Epoch 79/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4500 - accuracy: 0.4867
Epoch 80/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3856 - accuracy: 0.5533
```

```
Epoch 81/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3879 - accuracy: 0.5000
Epoch 82/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4208 - accuracy: 0.4533
Epoch 83/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4873 - accuracy: 0.4467
Epoch 84/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4071 - accuracy: 0.5067
Epoch 85/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4162 - accuracy: 0.4867
Epoch 86/100
50/50 [==============================] - 0s 4ms/step - loss: 1.2431 - accuracy: 0.5333
Epoch 87/100
50/50 [==============================] - 0s 4ms/step - loss: 1.3362 - accuracy: 0.5400
Epoch 88/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4760 - accuracy: 0.4667
Epoch 89/100
50/50 [==============================] - 0s 3ms/step - loss: 1.5609 - accuracy: 0.4667
Epoch 90/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3624 - accuracy: 0.5267
Epoch 91/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3203 - accuracy: 0.5067
Epoch 92/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4256 - accuracy: 0.5333
Epoch 93/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4702 - accuracy: 0.4667
Epoch 94/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4042 - accuracy: 0.5200
Epoch 95/100
50/50 [==============================] - 0s 3ms/step - loss: 1.2549 - accuracy: 0.5800
Epoch 96/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3594 - accuracy: 0.5133
Epoch 97/100
50/50 [==============================] - 0s 3ms/step - loss: 1.3720 - accuracy: 0.5067
Epoch 98/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4501 - accuracy: 0.5067
Epoch 99/100
50/50 [==============================] - 0s 3ms/step - loss: 1.4092 - accuracy: 0.5267
Epoch 100/100
50/50 [==============================] - 0s 4ms/step - loss: 1.4958 - accuracy: 0.4467
```

In [14]:

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

```
313/313 [==============================] - 1s 3ms/step - loss: 1.4421 - accuracy: 0.4910
```

Out[14]:

```
[1.4420700073242188, 0.4909999966621399]
```

In [15]:

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

Out[15]:

```
array([[8.7722847e-03, 9.1130816e-04, 8.6289629e-02, 4.5914552e-01,
         1.0434080e-01, 2.6656434e-01, 5.6806419e-02, 1.1073265e-02,
         4.7843894e-03, 1.3119834e-03],
        [1.4496610e-01, 1.5510891e-01, 6.3135359e-04, 1.6547216e-04,
         6.9622634e-05, 1.2235076e-05, 3.3398617e-05, 3.8381149e-06,
         6.4120245e-01, 5.7806596e-02],
        [2.6985297e-01, 8.4360853e-02, 4.4607587e-02, 8.9490719e-02,
         1.6844239e-02, 2.5212910e-02, 6.0294289e-03, 5.9630722e-03,
         3.8349441e-01, 7.4143738e-02],
        [4.8296180e-01, 1.4993618e-02, 4.1721411e-02, 8.4517412e-03,
         2.1375585e-02, 1.5486666e-03, 1.4024152e-03, 3.8937782e-03,
         4.0496653e-01, 1.8684402e-02],
        [5.1076603e-03, 2.5342365e-03, 1.2273740e-01, 2.3401138e-01,
         2.1049251e-01, 1.9017768e-01, 1.7104021e-01, 5.4658644e-02,
         3.8776405e-03, 5.3626406e-03]], dtype=float32)
```

In [16]:

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

Out[16]:

```
[3, 8, 8, 0, 3]
```

In [17]:

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

Out[17]:

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

In [27]:

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

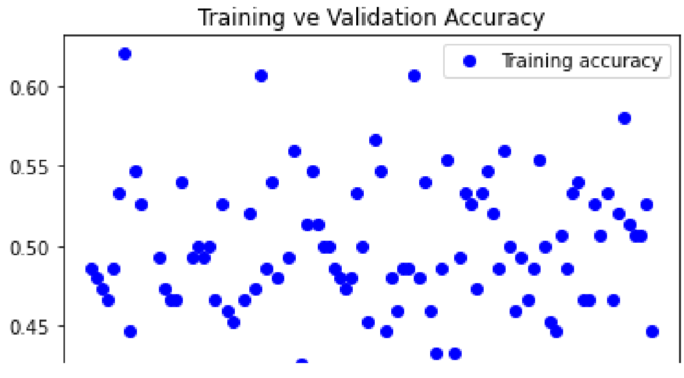
```
Sınıflandırma Sonucu :
               precision    recall  f1-score   support

           0       0.56      0.58      0.57      1000
           1       0.70      0.64      0.67      1000
           2       0.36      0.16      0.22      1000
           3       0.30      0.39      0.34      1000
           4       0.34      0.50      0.40      1000
           5       0.41      0.41      0.41      1000
           6       0.49      0.58      0.53      1000
           7       0.59      0.56      0.57      1000
           8       0.65      0.61      0.63      1000
           9       0.64      0.48      0.55      1000

    accuracy                           0.49     10000
   macro avg       0.50      0.49      0.49     10000
weighted avg       0.50      0.49      0.49     10000
```

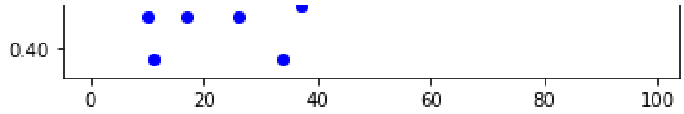
In [30]:

```
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[30]:

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






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

In [ ]: