In [35]:
```
import tensorflow as tf
from tensorflow.keras import layers, models
import numpy as np
import matplotlib.pyplot as plt
```

In [36]:
```
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.cifar10.load_data()
```

In [39]:
```
x_train, x_test = x_train / 255.0, x_test / 255.0
y_train, y_test = y_train.flatten(), y_test.flatten()
```

In [40]:
```
from keras import models,layers
model = models.Sequential([
    layers.Flatten(input_shape=(32,32,3)),
    layers.Dense(512, activation='relu'),
    layers.Dropout(0.2),
    layers.Dense(256, activation='relu'),
    layers.Dropout(0.2),
    layers.Dense(128, activation='relu'),
    layers.Dense(10, activation='softmax')
])
```

In [41]:
```
model.compile(optimizer='adam',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

In [42]:
```
history = model.fit(x_train, y_train, epochs=15,
                    validation_data=(x_test,y_test),
                    batch_size=64)
model.save('cifar10_mlp_model.h5')
```

```
Epoch 1/15
782/782 [==============================] - 11s 13ms/step - loss: 1.9773 - accuracy: 0.2726 - val_loss: 1.8256 - val_accuracy: 0.3386
Epoch 2/15
782/782 [==============================] - 8s 11ms/step - loss: 1.8305 - accuracy: 0.3332 - val_loss: 1.7153 - val_accuracy: 0.3833
Epoch 3/15
782/782 [==============================] - 8s 10ms/step - loss: 1.7777 - accuracy: 0.3550 - val_loss: 1.7182 - val_accuracy: 0.3750
Epoch 4/15
782/782 [==============================] - 8s 11ms/step - loss: 1.7384 - accuracy: 0.3713 - val_loss: 1.6427 - val_accuracy: 0.4182
Epoch 5/15
782/782 [==============================] - 8s 10ms/step - loss: 1.7067 - accuracy: 0.3817 - val_loss: 1.6249 - val_accuracy: 0.4279
Epoch 6/15
782/782 [==============================] - 8s 11ms/step - loss: 1.6853 - accuracy: 0.3895 - val_loss: 1.6370 - val_accuracy: 0.4127
Epoch 7/15
782/782 [==============================] - 8s 10ms/step - loss: 1.6638 - accuracy: 0.3965 - val_loss: 1.6051 - val_accuracy: 0.4271
Epoch 8/15
782/782 [==============================] - 8s 11ms/step - loss: 1.6488 - accuracy: 0.4046 - val_loss: 1.5799 - val_accuracy: 0.4396
Epoch 9/15
782/782 [==============================] - 8s 10ms/step - loss: 1.6370 - accuracy: 0.4091 - val_loss: 1.5892 - val_accuracy: 0.4346
Epoch 10/15
782/782 [==============================] - 8s 11ms/step - loss: 1.6235 - accuracy: 0.4141 - val_loss: 1.5807 - val_accuracy: 0.4329
Epoch 11/15
782/782 [==============================] - 8s 10ms/step - loss: 1.6072 - accuracy: 0.4196 - val_loss: 1.5816 - val_accuracy: 0.4293
Epoch 12/15
782/782 [==============================] - 8s 11ms/step - loss: 1.6011 - accuracy: 0.4234 - val_loss: 1.5542 - val_accuracy: 0.4497
Epoch 13/15
782/782 [==============================] - 8s 10ms/step - loss: 1.5940 - accuracy: 0.4274 - val_loss: 1.5535 - val_accuracy: 0.4549
Epoch 14/15
782/782 [==============================] - 8s 11ms/step - loss: 1.5814 - accuracy: 0.4302 - val_loss: 1.5236 - val_accuracy: 0.4609
Epoch 15/15
782/782 [==============================] - 8s 10ms/step - loss: 1.5726 - accuracy: 0.4347 - val_loss: 1.5160 - val_accuracy: 0.4604
```

In [43]:
```
test_loss, test_acc = model.evaluate(x_test,y_test,verbose=2)
```

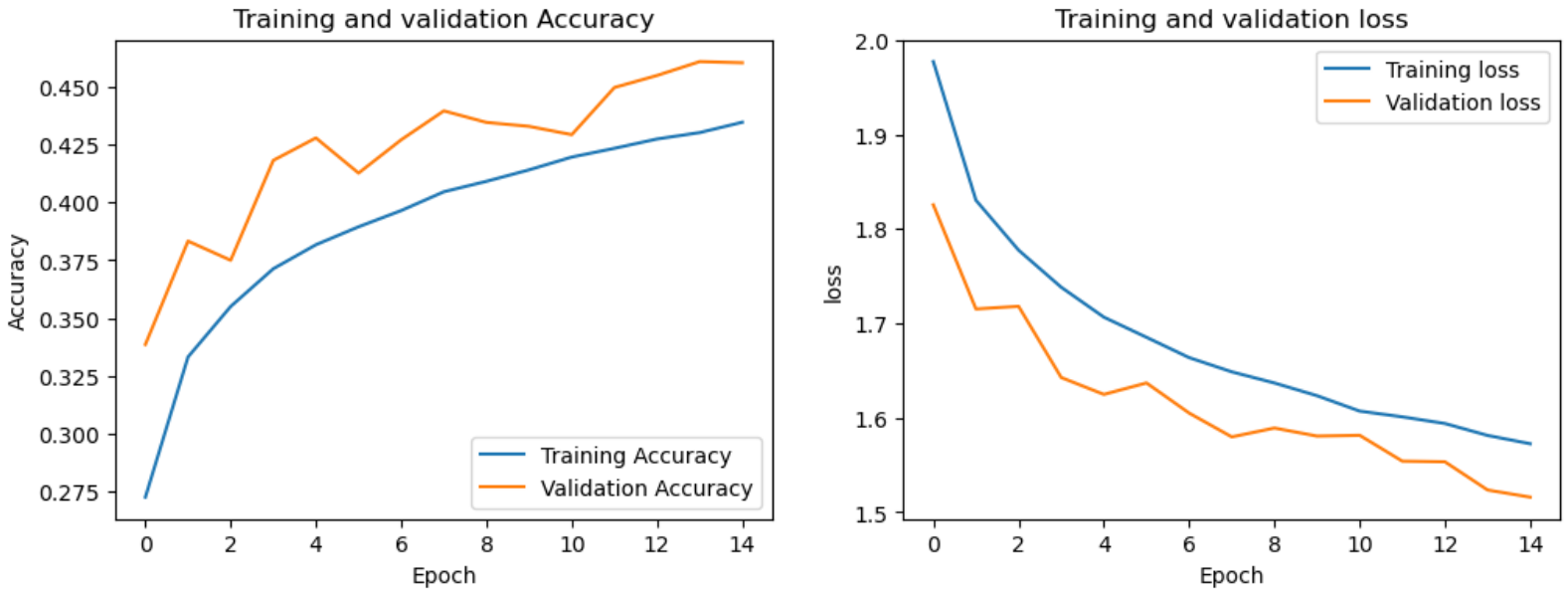
```
313/313 - 0s - loss: 1.5160 - accuracy: 0.4604 - 439ms/epoch - 1ms/step
```

In [45]:
```
#plotting the loss curve
plt.figure(figsize=(12,4))

#plot training and validation loss values
plt.subplot(1,2,1)
plt.plot(history.history['accuracy'], label = 'Training Accuracy')
plt.plot(history.history['val_accuracy'], label = 'Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend(loc='lower right')
plt.title('Training and validation Accuracy')

#plot training and validation accuracy values
plt.subplot(1,2,2)
plt.plot(history.history['loss'], label = 'Training loss')
plt.plot(history.history['val_loss'], label = 'Validation loss')
plt.xlabel('Epoch')
plt.ylabel('loss')
plt.legend(loc='upper right')
plt.title('Training and validation loss')
```

Out[45]: Text(0.5, 1.0, 'Training and validation loss')



In [ ]: