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

# Load CIFAR-10 dataset
(train_images, train_labels), (test_images, test_labels) = datasets.cifar10.load_data()

# Normalize pixel values to be between 0 and 1
train_images = train_images.astype('float32') / 255.0
test_images = test_images.astype('float32') / 255.0

# One-hot encode the labels
train_labels = tf.keras.utils.to_categorical(train_labels, 10)
test_labels = tf.keras.utils.to_categorical(test_labels, 10)

# Build the CNN model
model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(10, activation='softmax')  # 10 classes for CIFAR-10
])

# Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Train the model
history = model.fit(train_images, train_labels, epochs=10, batch_size=64,
                    validation_data=(test_images, test_labels))

# Evaluate the model
test_loss, test_acc = model.evaluate(test_images, test_labels, verbose=2)
print('\nTest accuracy:', test_acc)

# Plot training & validation accuracy and loss values
def plot_history(history):
    # Plot training & validation accuracy values
    plt.figure(figsize=(12, 4))

    plt.subplot(1, 2, 1)
    plt.plot(history.history['accuracy'], label='Train Accuracy')
    plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
    plt.title('Model Accuracy')
    plt.ylabel('Accuracy')
    plt.xlabel('Epoch')
    plt.legend()

    # Plot training & validation loss values
    plt.subplot(1, 2, 2)
    plt.plot(history.history['loss'], label='Train Loss')
    plt.plot(history.history['val_loss'], label='Validation Loss')
    plt.title('Model Loss')
    plt.ylabel('Loss')
    plt.xlabel('Epoch')
    plt.legend()

    plt.show()

# Call the function to plot the history
plot_history(history)
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 ━━━━━━━━━━━━━━━━━━━━ 3s 0us/step
/usr/local/lib/python3.10/dist-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass an `input_shape`/`inpu
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/10
782/782 ━━━━━━━━━━━━━━━━━━━━ 92s 114ms/step - accuracy: 0.3343 - loss: 1.8009 - val_accuracy: 0.5452 - val_loss: 1.2705
Epoch 2/10
782/782 ━━━━━━━━━━━━━━━━━━━━ 130s 99ms/step - accuracy: 0.5515 - loss: 1.2595 - val_accuracy: 0.6104 - val_loss: 1.1144
Epoch 3/10
782/782 ━━━━━━━━━━━━━━━━━━━━ 79s 96ms/step - accuracy: 0.6189 - loss: 1.0772 - val_accuracy: 0.6310 - val_loss: 1.0572
Epoch 4/10
782/782 ━━━━━━━━━━━━━━━━━━━━ 77s 98ms/step - accuracy: 0.6591 - loss: 0.9775 - val_accuracy: 0.6650 - val_loss: 0.9687
Epoch 5/10
782/782 ━━━━━━━━━━━━━━━━━━━━ 83s 100ms/step - accuracy: 0.6903 - loss: 0.8824 - val_accuracy: 0.6824 - val_loss: 0.9032
Epoch 6/10
782/782 ━━━━━━━━━━━━━━━━━━━━ 80s 98ms/step - accuracy: 0.7110 - loss: 0.8246 - val_accuracy: 0.6937 - val_loss: 0.8919
Epoch 7/10
782/782 ━━━━━━━━━━━━━━━━━━━━ 76s 98ms/step - accuracy: 0.7237 - loss: 0.7775 - val_accuracy: 0.6945 - val_loss: 0.8791
Epoch 8/10
782/782 ━━━━━━━━━━━━━━━━━━━━ 84s 100ms/step - accuracy: 0.7425 - loss: 0.7350 - val_accuracy: 0.6931 - val_loss: 0.8900
Epoch 9/10
782/782 ━━━━━━━━━━━━━━━━━━━━ 78s 96ms/step - accuracy: 0.7557 - loss: 0.6990 - val_accuracy: 0.7041 - val_loss: 0.8754
Epoch 10/10
782/782 ━━━━━━━━━━━━━━━━━━━━ 75s 96ms/step - accuracy: 0.7691 - loss: 0.6592 - val_accuracy: 0.7129 - val_loss: 0.8428
313/313 - 6s - 18ms/step - accuracy: 0.7129 - loss: 0.8428

Test accuracy: 0.7128999829292297
```