```python
# STEP 2: Import libraries
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import matplotlib.pyplot as plt
# STEP 3: Load and preprocess CIFAR-10
(X_train, y_train), (X_test, y_test) = datasets.cifar10.load_data()
#  Use only 5,000 training samples for speed
X_train, y_train = X_train[:5000], y_train[:5000]
# Normalize pixel values
X_train, X_test = X_train / 255.0, X_test / 255.0
# One-hot encode labels
y_train = tf.keras.utils.to_categorical(y_train, 10)
y_test = tf.keras.utils.to_categorical(y_test, 10)
# STEP 4: Build a small CNN model
model = models.Sequential()
model.add(layers.Conv2D(16, (3, 3), activation='relu', input_shape=(32, 32, 3)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(32, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dense(32, activation='relu'))
model.add(layers.Dense(10, activation='softmax'))  # 10 classes
# STEP 5: Compile and train the model (only 3 epochs)
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
history = model.fit(
    X_train, y_train,
    epochs=3,
    batch_size=64,
    validation_data=(X_test, y_test),
    verbose=2
)
# STEP 6: Evaluate and plot results
test_loss, test_acc = model.evaluate(X_test, y_test, verbose=0)
print(f"\n Test Accuracy: {test_acc * 100:.2f}%")
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.title('Training vs Validation Accuracy')
plt.legend()
plt.grid(True)
plt.show()
```

OUTPUT

```
Epoch 1/3
79/79 - 11s - 141ms/step - accuracy: 0.2088 - loss: 2.1288 - val_accuracy: 0.2988 - val_loss: 1.9147
Epoch 2/3
79/79 - 7s - 92ms/step - accuracy: 0.3636 - loss: 1.7819 - val_accuracy: 0.3306 - val_loss: 1.8119
Epoch 3/3
79/79 - 10s - 130ms/step - accuracy: 0.4126 - loss: 1.6230 - val_accuracy: 0.3883 - val_loss: 1.6628

 Test Accuracy: 38.83%
```