```
import tensorflow as tf
import keras
import torch
import numpy as np
import matplotlib.pyplot as plt

print(" ✅ Deep Learning Libraries Installed Successfully!\n")
print("TensorFlow version:", tf.__version__)
print("Keras version:", keras.__version__)
print("PyTorch version:", torch.__version__)

# ==========================================
# ⚙️ STEP 3 — SIMPLE DEMO USING TENSORFLOW
# ==========================================
# Generate dummy data
x = np.random.rand(100, 1)
y = 3 * x + 2 + np.random.randn(100, 1) * 0.1

# Define a simple model
model_tf = tf.keras.Sequential([
    tf.keras.layers.Dense(1, input_shape=(1,))
])

model_tf.compile(optimizer='sgd', loss='mse')
history_tf = model_tf.fit(x, y, epochs=10, verbose=0)
```

```
# Predictions
y_pred_tf = model_tf.predict(x)

# Visualization
plt.figure(figsize=(6,4))
plt.scatter(x, y, label='Actual Data', color='blue')
plt.plot(x, y_pred_tf, color='red', label='TensorFlow Prediction')
plt.title("TensorFlow Linear Regression")
plt.xlabel("X values")
plt.ylabel("Y values")
plt.legend()
plt.show()

print("\n✅ TensorFlow Model Trained & Visualized Successfully!\n")

# ==========================================
# ⚙️ STEP 4 — SIMPLE DEMO USING KERAS
# ==========================================
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense

# Build a simple neural network
model_keras = Sequential([
    Dense(8, activation='relu', input_shape=(1,)),
    Dense(1)
])
```

```
model_keras.compile(optimizer='adam', loss='mse')
history_keras = model_keras.fit(x, y, epochs=10, verbose=0)

# Predictions
y_pred_keras = model_keras.predict(x)

# Visualization
plt.figure(figsize=(6,4))
plt.scatter(x, y, label='Actual Data', color='blue')
plt.plot(x, y_pred_keras, color='green', label='Keras Prediction')
plt.title("Keras Neural Network Regression")
plt.xlabel("X values")
plt.ylabel("Y values")
plt.legend()
plt.show()

print("\n✅ Keras Model Trained & Visualized Successfully!\n")

# ==========================================
# ⚙️ STEP 5 — SIMPLE DEMO USING PYTORCH
# ==========================================
import torch.nn as nn
import torch.optim as optim

# Dummy data (convert to tensors)
```

```
X = torch.rand(100, 1)
Y = 3 * X + 2 + 0.1 * torch.randn(100, 1)

# Define simple linear regression model
model_torch = nn.Linear(1, 1)
criterion = nn.MSELoss()
optimizer = optim.SGD(model_torch.parameters(), lr=0.1)

# Train for a few epochs
losses = []
for epoch in range(10):
    optimizer.zero_grad()
    outputs = model_torch(X)
    loss = criterion(outputs, Y)
    loss.backward()
    optimizer.step()
    losses.append(loss.item())
    print(f"Epoch {epoch+1}, Loss: {loss.item():.4f}")

# Plot training loss
plt.figure(figsize=(6,4))
plt.plot(range(1, 11), losses, marker='o', color='purple')
plt.title("PyTorch Training Loss")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.show()
```

```
# Visualize predictions
preds_torch = model_torch(X).detach().numpy()

plt.figure(figsize=(6,4))
plt.scatter(X, Y, color='blue', label='Actual Data')
plt.plot(X, preds_torch, color='orange', label='PyTorch Prediction')
plt.title("PyTorch Linear Regression")
plt.xlabel("X values")
plt.ylabel("Y values")
plt.legend()
plt.show()

print("\n✅ PyTorch Model Trained & Visualized Successfully!\n")
```