```
import tensorflow as tf
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.utils import to_categorical
import matplotlib.pyplot as plt
import numpy as np

(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train, x_test = x_train / 255.0, x_test / 255.0

y_train = to_categorical(y_train, 10)
y_test = to_categorical(y_test, 10)

model = Sequential([
   Flatten(input_shape=(28, 28)),
   Dense(128, activation='relu'),
   Dense(64, activation='relu'),
   Dense(10, activation='softmax')
])

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

history = model.fit(x_train, y_train,
                epochs=10,
                batch_size=32,
                validation_split=0.1,
                verbose=1)

test_loss, test_accuracy = model.evaluate(x_test, y_test, verbose=0)
```

```
print(f"\nTest Accuracy: {test_accuracy:.4f}")
print(f"Test Loss: {test_loss:.4f}")

plt.figure(figsize=(14, 5))

plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Train Accuracy', marker='o')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy', marker='x')
plt.title('Model Accuracy Over Epochs')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.grid(True)

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Train Loss', marker='o')
plt.plot(history.history['val_loss'], label='Validation Loss', marker='x')
plt.title('Model Loss Over Epochs')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.grid(True)

plt.tight_layout()
plt.show()

sample_index = 0
sample_image = x_test[sample_index]
sample_label = np.argmax(y_test[sample_index])
```

```
prediction = model.predict(np.expand_dims(sample_image, axis=0))
predicted_label = np.argmax(prediction)

plt.imshow(sample_image, cmap='gray')
plt.title(f" Predicted: {predicted_label} |  Actual: {sample_label}")
plt.axis('off')
plt.show()
```

Output :

```
Epoch 1/10
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 5s 3ms/step - accuracy: 0.8714 - loss: 0.4372 - val_accuracy: 0.9688 - val_loss: 0.1068
Epoch 2/10
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 4s 2ms/step - accuracy: 0.9660 - loss: 0.1110 - val_accuracy: 0.9745 - val_loss: 0.0896
Epoch 3/10
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 4s 2ms/step - accuracy: 0.9777 - loss: 0.0722 - val_accuracy: 0.9727 - val_loss: 0.0929
Epoch 4/10
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 4s 3ms/step - accuracy: 0.9840 - loss: 0.0514 - val_accuracy: 0.9790 - val_loss: 0.0828
Epoch 5/10
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 4s 3ms/step - accuracy: 0.9865 - loss: 0.0413 - val_accuracy: 0.9737 - val_loss: 0.0975
Epoch 6/10
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 5s 3ms/step - accuracy: 0.9898 - loss: 0.0324 - val_accuracy: 0.9755 - val_loss: 0.0950
Epoch 7/10
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 5s 3ms/step - accuracy: 0.9906 - loss: 0.0279 - val_accuracy: 0.9790 - val_loss: 0.0860
Epoch 8/10
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 5s 3ms/step - accuracy: 0.9920 - loss: 0.0219 - val_accuracy: 0.9803 - val_loss: 0.0800
Epoch 9/10
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 4s 3ms/step - accuracy: 0.9941 - loss: 0.0184 - val_accuracy: 0.9792 - val_loss: 0.0976
Epoch 10/10
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 4s 3ms/step - accuracy: 0.9951 - loss: 0.0149 - val_accuracy: 0.9803 - val_loss: 0.0976
```

Test Accuracy: 0.9784

Test Loss: 0.1031