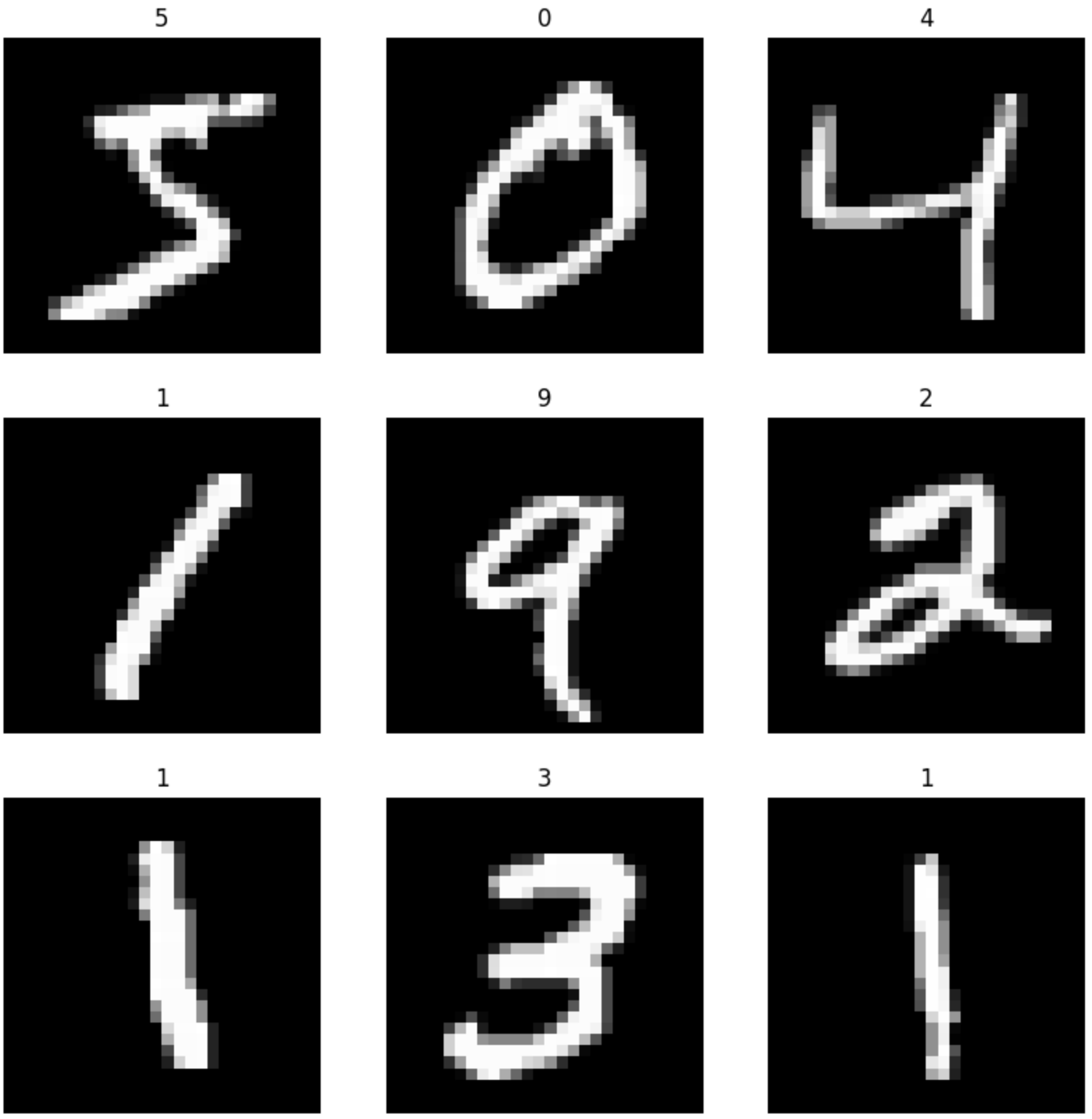
In [ ]:
```
import tensorflow as tf
import matplotlib.pyplot as plt

# Load the MNIST dataset
mnist = tf.keras.datasets.mnist
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
train_images.shape
```

Out[ ]:
```
(60000, 28, 28)
```

In [ ]:
```
# Create a subplot
plt.figure(figsize=(10, 10))
for i in range(9):
      # Display the image
      ax = plt.subplot(3, 3, i + 1)
      plt.imshow(train_images[i], cmap='gray')
      # Print the label
      plt.title(train_labels[i])
      plt.axis('off')

# Show the subplot
plt.show()
```



In [ ]:
```
import tensorflow as tf

# Define the model
model = tf.keras.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dense(10, activation='softmax')
])
```

In [ ]:
```
# Compile the model
model.compile(tf.keras.optimizers.Adam(learning_rate=0.001), loss='sparse_categorical_crossentropy', metrics=['accura
```

In [ ]:
```
model.fit(train_images, train_labels, epochs=3)
```

```
Epoch 1/3
1875/1875 [==============================] - 5s 2ms/step - loss: 2.6410 - accuracy: 0.8510
Epoch 2/3
1875/1875 [==============================] - 4s 2ms/step - loss: 0.4112 - accuracy: 0.9007
Epoch 3/3
1875/1875 [==============================] - 4s 2ms/step - loss: 0.3010 - accuracy: 0.9242
```

Out[ ]: <keras.src.callbacks.History at 0x1799ef22690>

In [ ]:
```
# Evaluate the model
train_accuracy=model.evaluate(train_images , train_labels)
```

```
1875/1875 [==============================] - 3s 1ms/step - loss: 0.2612 - accuracy: 0.9318
```

In [ ]:
```
# Evaluate the model
test_accuracy=model.evaluate(test_images   , test_labels)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 0.3316 - accuracy: 0.9290
```

In [ ]:
```
print("train_accuracy",train_accuracy)
print("test_accuracy",test_accuracy)
```

```
train_accuracy [0.26115748286247253, 0.9318333268165588]
test_accuracy [0.3316153585910797, 0.9290000200271606]
```

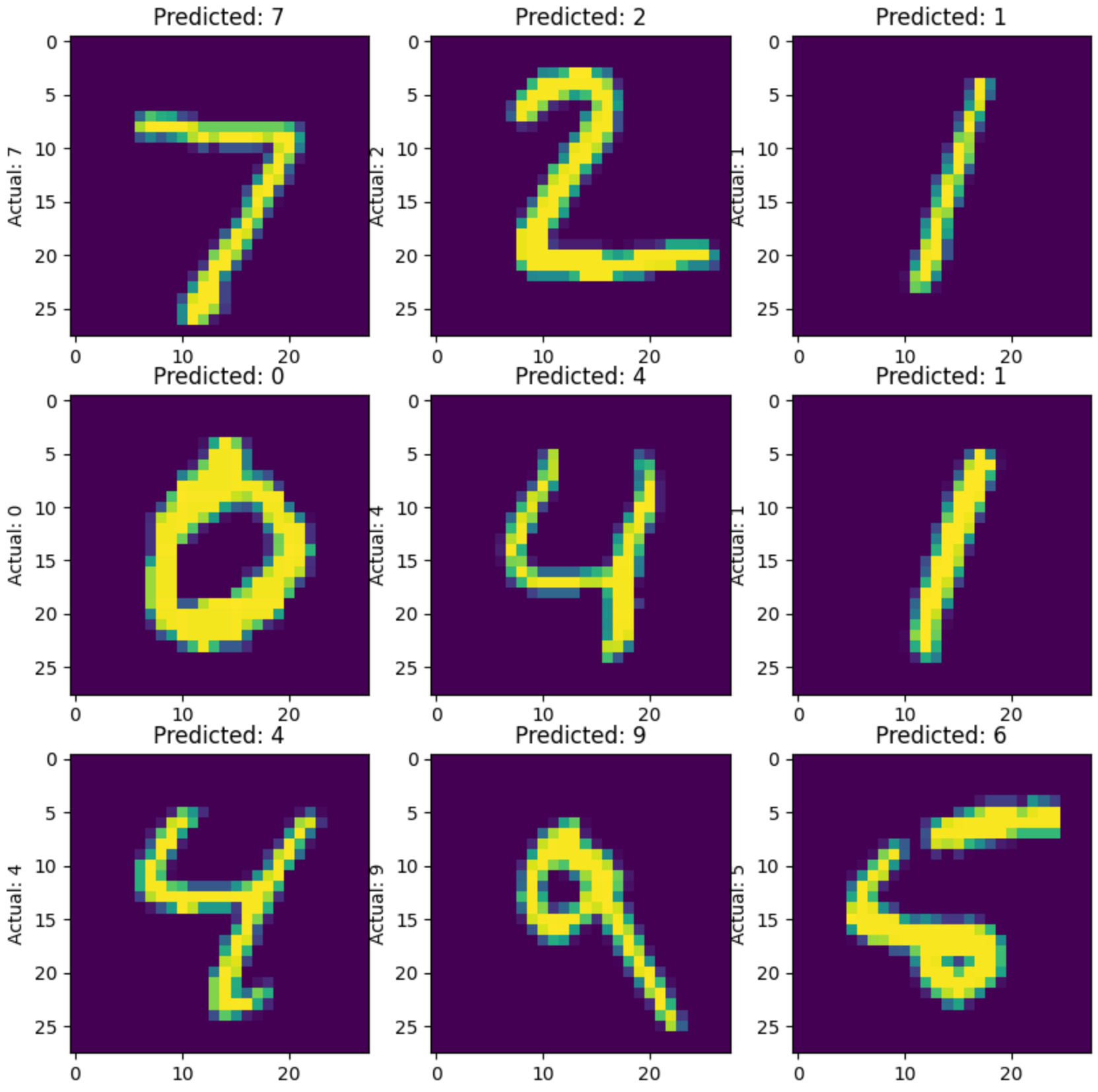
In [ ]:
```
import numpy as np
test_images_9 = test_images[:9]

# Create a subplot
fig, axes = plt.subplots(3, 3, figsize=(10, 10))

for i in range(9):
  test_images_9_1 = np.reshape(test_images_9[i], (1, 28, 28))
  prediction = model.predict(test_images_9_1)
  axes[i // 3, i % 3] = plt.subplot(3, 3, i + 1)
  axes[i // 3, i % 3].imshow(test_images_9[i])
  axes[i // 3, i % 3].set_title(f"Predicted: {prediction.argmax()}")
  axes[i // 3, i % 3].set_ylabel(f"Actual: {test_labels[i]}")

plt.show()
```

```
1/1 [==============================] - 0s 72ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 24ms/step
```


Predicted: 7
Actual: 7
Predicted: 2
Actual: 2
Predicted: 1
Actual: 1
Predicted: 0
Actual: 0
Predicted: 4
Actual: 4
Predicted: 1
Actual: 1
Predicted: 4
Actual: 4
Predicted: 9
Actual: 9
Predicted: 6
Actual: 5