In [1]:
```
import numpy as np
import tensorflow as tf # Import tensorflow library
import matplotlib.pyplot as plt # Import matplotlib library
```

In [2]:
```
mnist = tf.keras.datasets.mnist # Object of the MNIST dataset
(x_train, y_train),(x_test, y_test) = mnist.load_data() # Load data
```

In [3]:
```
x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
input_shape = (28, 28, 1)
```

In [4]:
```
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
```

In [5]:
```
x_train /= 255
x_test /= 255
```

In [6]:
```
#Build the model object
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D
model = Sequential()
model.add(Conv2D(28, kernel_size=(3,3), input_shape=input_shape))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation=tf.nn.relu))
model.add(Dropout(0.2))
model.add(Dense(10,activation=tf.nn.softmax))
```

In [7]:
```
# Compile the model
model.compile(optimizer="adam", loss="sparse_categorical_crossentropy", metrics=["accuracy"])
```

In [8]:
```
model.fit(x=x_train, y=y_train, epochs=5) # Start training process
```

```
Epoch 1/5
1875/1875 [==============================] - 47s 22ms/step - loss: 0.3731 - accuracy: 0.88580s - loss: 0.3732 - accuracy: 0.88
Epoch 2/5
1875/1875 [==============================] - 39s 21ms/step - loss: 0.0885 - accuracy: 0.9723
Epoch 3/5
1875/1875 [==============================] - 40s 21ms/step - loss: 0.0574 - accuracy: 0.9824
Epoch 4/5
1875/1875 [==============================] - 40s 21ms/step - loss: 0.0443 - accuracy: 0.9856
Epoch 5/5
1875/1875 [==============================] - 39s 21ms/step - loss: 0.0329 - accuracy: 0.9894
```

Out[8]:
```
<tensorflow.python.keras.callbacks.History at 0x1eedcfae670>
```

In [9]:
```
# Evaluate the model performance
test_loss, test_acc = model.evaluate(x=x_test, y=y_test)
# Print out the model accuracy
print('\nTest accuracy:', test_acc)
```

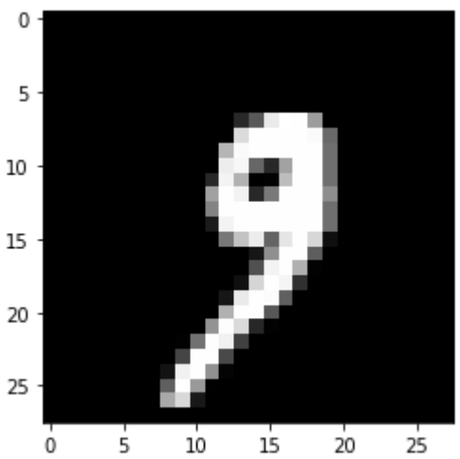
```
313/313 [==============================] - 3s 8ms/step - loss: 0.0505 - accuracy: 0.9839

Test accuracy: 0.9839000105857849
```

In [10]:
```
predictions = model.predict([x_test]) # Make prediction
```

In [11]:
```
print(np.argmax(predictions[1000])) # Print out the number
```

```
9
```

In [12]:
```
plt.imshow(x_test[1000], cmap="gray") # Import the image
plt.show() # Show the image
```



In [ ]:
```
```