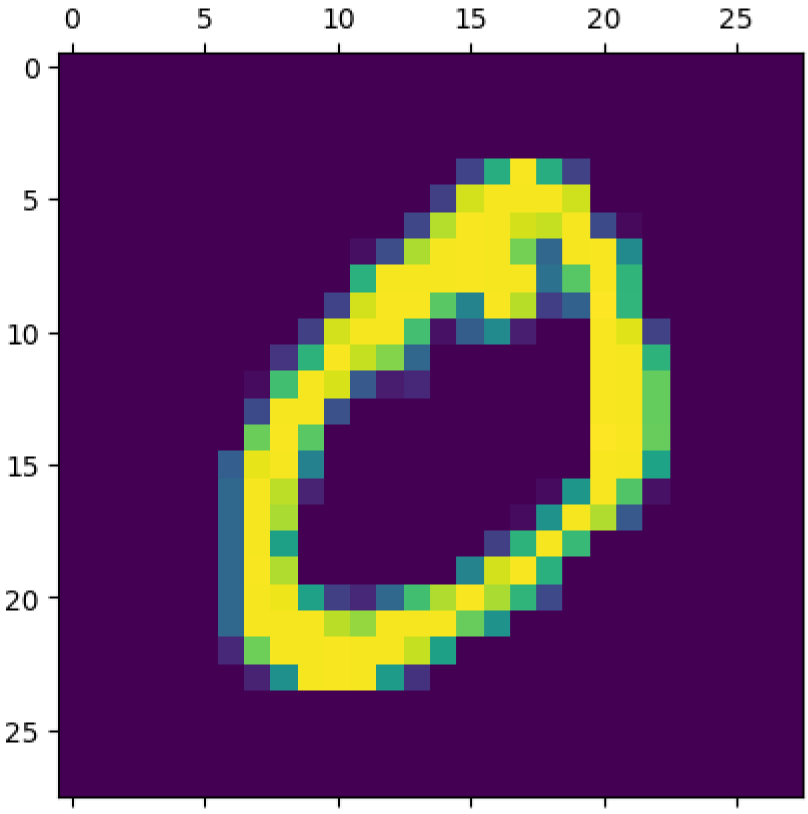
In [1]:
```
import tensorflow as tf
from tensorflow import keras
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import random
%matplotlib inline
```

In [2]:
```
mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mn
ist.npz
11490434/11490434 [==============================] - 9s 1us/step
```

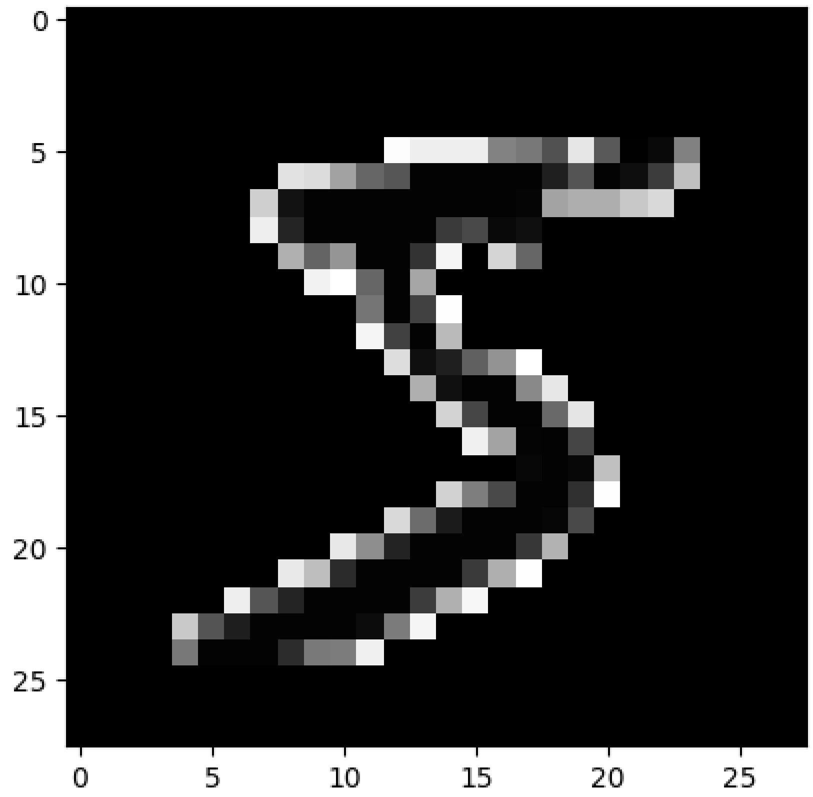
In [3]:
```
plt.matshow(x_train[1])
```

Out[3]:
```
<matplotlib.image.AxesImage at 0x22a528cdde0>
```



In [4]:
```
plt.imshow(-x_train[0], cmap="gray")
```

Out[4]:
```
<matplotlib.image.AxesImage at 0x22a57ddff10>
```



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

In [6]:
```
model = keras.Sequential([
keras.layers.Flatten(input_shape=(28, 28)),
keras.layers.Dense(128, activation="relu"),
keras.layers.Dense(10, activation="softmax")
])

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 flatten (Flatten)           (None, 784)               0         
                                                                 
 dense (Dense)               (None, 128)               100480    
                                                                 
 dense_1 (Dense)             (None, 10)                1290      
                                                                 
=================================================================
Total params: 101770 (397.54 KB)
Trainable params: 101770 (397.54 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [7]:
```
model.compile(optimizer="sgd",
loss="sparse_categorical_crossentropy",
metrics=['accuracy'])
```

In [8]:
```
history=model.fit(x_train,
y_train,validation_data=(x_test,y_test),epochs=10)
```

```
Epoch 1/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.6589 - accuracy:
0.8296 - val_loss: 0.3626 - val_accuracy: 0.9024
Epoch 2/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3420 - accuracy:
0.9047 - val_loss: 0.3027 - val_accuracy: 0.9150
Epoch 3/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.2945 - accuracy:
0.9176 - val_loss: 0.2685 - val_accuracy: 0.9256
Epoch 4/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2642 - accuracy:
0.9262 - val_loss: 0.2457 - val_accuracy: 0.9302
Epoch 5/10
1875/1875 [==============================] - 9s 5ms/step - loss: 0.2406 - accuracy:
0.9321 - val_loss: 0.2247 - val_accuracy: 0.9355
Epoch 6/10
1875/1875 [==============================] - 12s 6ms/step - loss: 0.2211 - accuracy:
0.9381 - val_loss: 0.2124 - val_accuracy: 0.9393
Epoch 7/10
1875/1875 [==============================] - 9s 5ms/step - loss: 0.2052 - accuracy:
0.9427 - val_loss: 0.1996 - val_accuracy: 0.9407
Epoch 8/10
1875/1875 [==============================] - 10s 6ms/step - loss: 0.1914 - accuracy:
0.9460 - val_loss: 0.1855 - val_accuracy: 0.9454
Epoch 9/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.1797 - accuracy:
0.9488 - val_loss: 0.1775 - val_accuracy: 0.9480
Epoch 10/10
1875/1875 [==============================] - 11s 6ms/step - loss: 0.1692 - accuracy:
0.9524 - val_loss: 0.1682 - val_accuracy: 0.9503
```

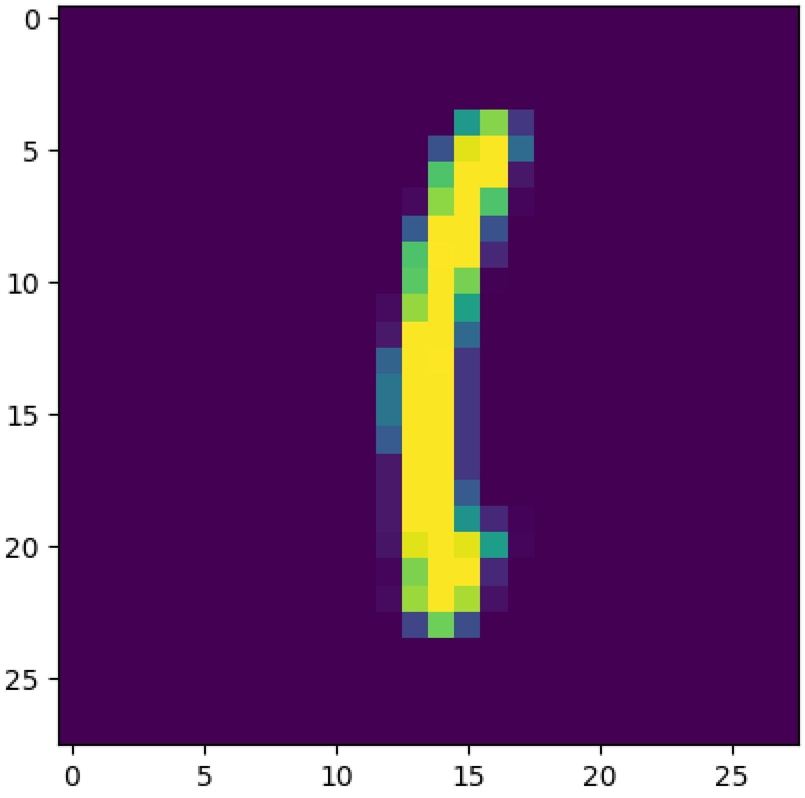
In [9]:

```
test_loss,test_acc=model.evaluate(x_test,y_test)
print("Loss=%.3f" %test_loss)
print("Accuracy=%.3f" %test_acc)
```

```
313/313 [==============================] - 2s 5ms/step - loss: 0.1682 - accuracy: 0.
9503
Loss=0.168
Accuracy=0.950
```

In [10]:

```
n=random.randint(0,9999)
plt.imshow(x_test[n])
plt.show()
```



In [11]:

```
x_train
```

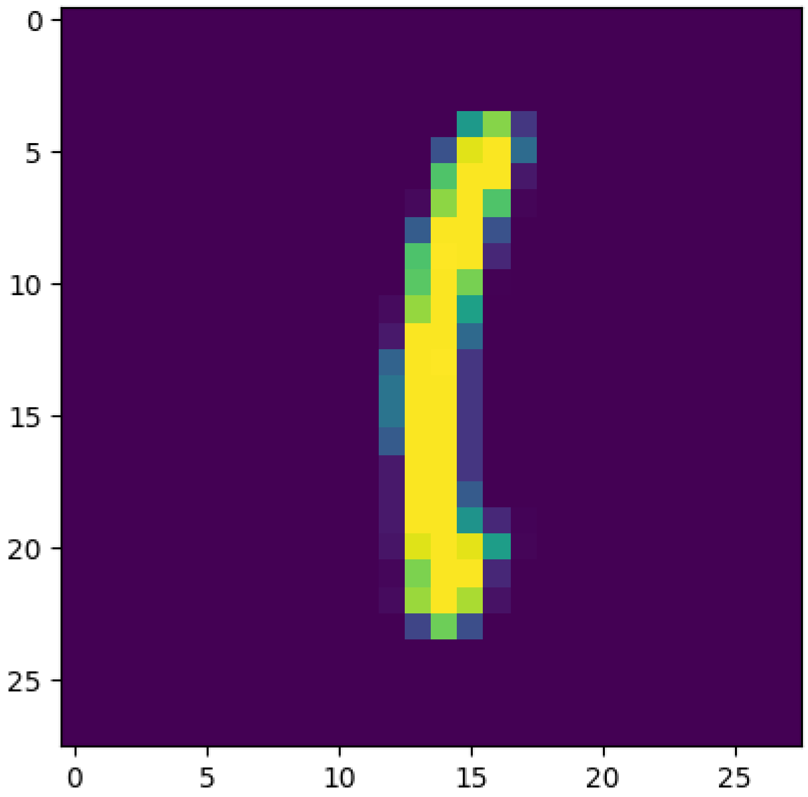
```
Out[11]: array([[[0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                ...,
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.]],

               [[0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                ...,
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.]],

               [[0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                ...,
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.]],

               ...,

               [[0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                ...,
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.]],

               [[0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                ...,
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.]],

               [[0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                ...,
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.],
                [0., 0., 0., ..., 0., 0., 0.]]])
```

```
In [12]: x_test
```

```
Out[12]: array([[[0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 ...,
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.]],

                [[0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 ...,
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.]],

                [[0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 ...,
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.]],

                ...,

                [[0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 ...,
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.]],

                [[0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 ...,
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.]],

                [[0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 ...,
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.],
                 [0., 0., 0., ..., 0., 0., 0.]]])
```

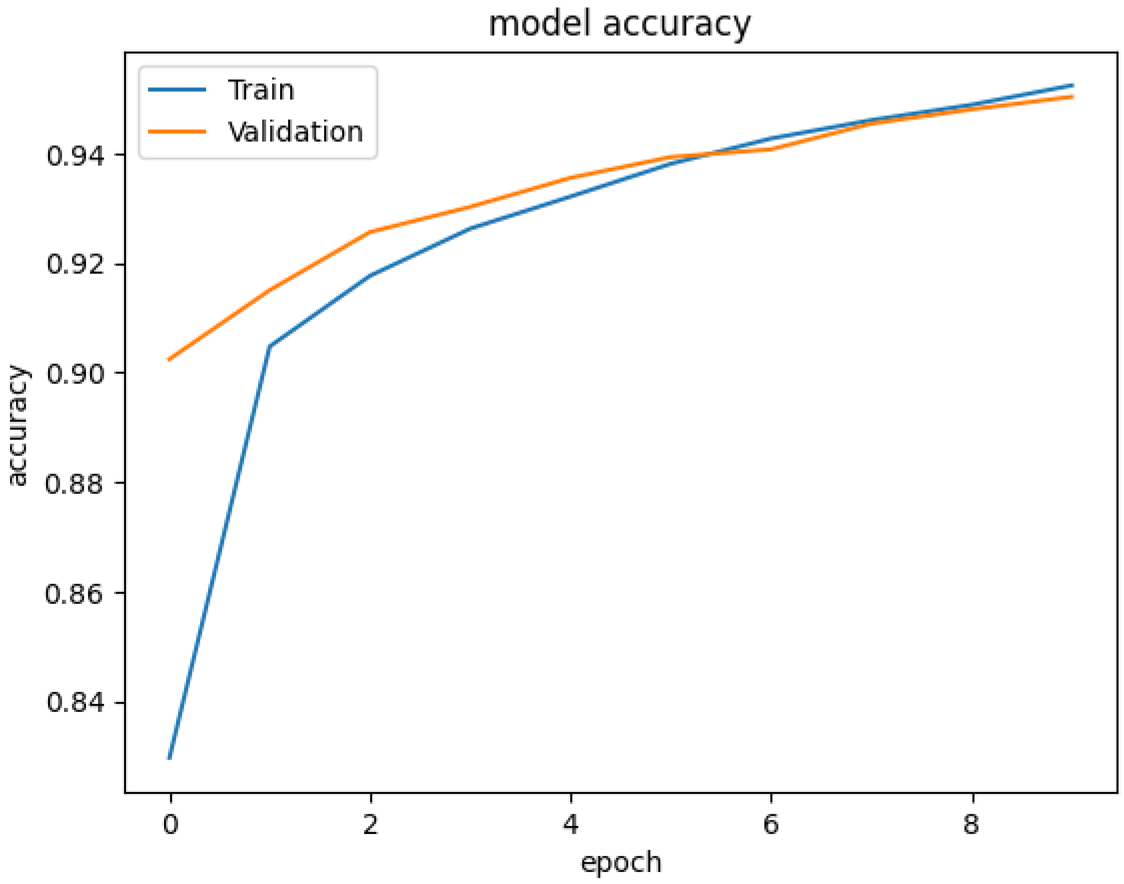
```
In [13]: predicted_value=model.predict(x_test)
         plt.imshow(x_test[n])
         plt.show()

         print(predicted_value[n])
```

```
313/313 [==============================] - 1s 3ms/step
```

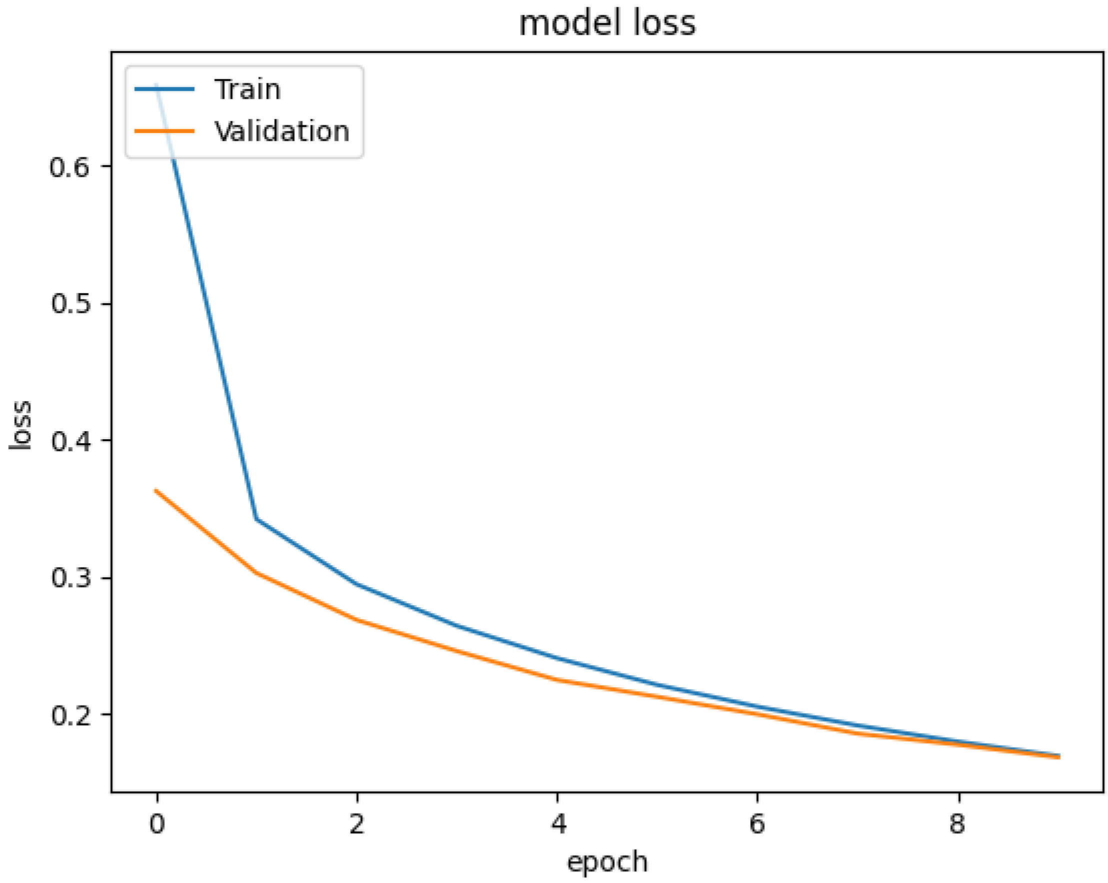


```
[1.2237692e-06 9.9714309e-01 1.4832725e-04 8.8756526e-04 2.6444686e-06
 5.8377063e-04 1.3396455e-04 8.7590515e-06 1.0609174e-03 2.9855037e-05]
```

In [14]:
```
history.history.keys()
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```



In [15]:
```
history.history.keys()
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```


model loss
Train
Validation
loss
0.2
0.3
0.4
0.5
0.6
epoch
0
2
4
6
8