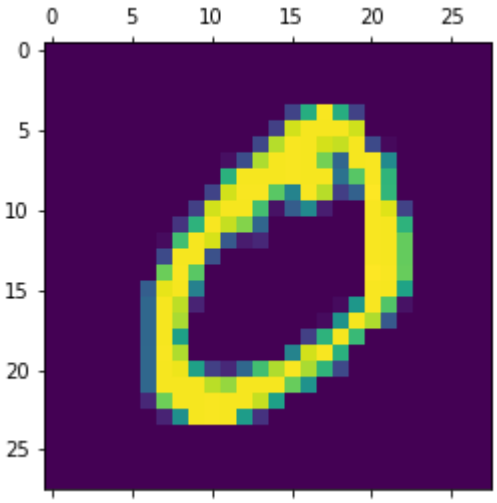
In [1]:
```
#importing necessary libraries
import tensorflow as tf
from tensorflow import keras
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import random
%matplotlib inline
```

```
C:\Users\vikas pawar\anaconda3\lib\site-packages\scipy\__init__.py:146: Us
erWarning: A NumPy version >=1.16.5 and <1.23.0 is required for this versi
on of SciPy (detected version 1.26.1
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [2]:
```
#import dataset and split into train and test data
mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

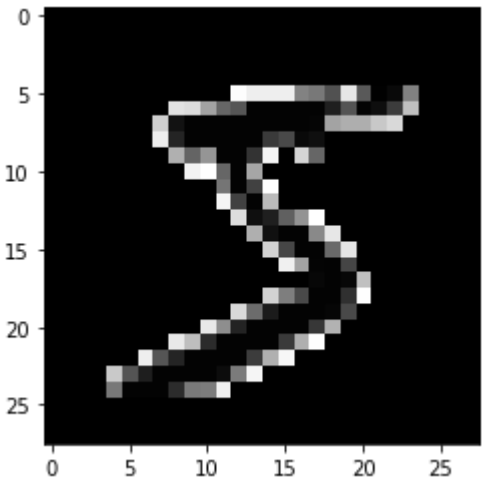
In [3]:
```
plt.matshow(x_train[1])
```

Out[3]:
```
<matplotlib.image.AxesImage at 0x22513f33220>
```



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

Out[4]:
```
<matplotlib.image.AxesImage at 0x22516003a90>
```



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'])
```

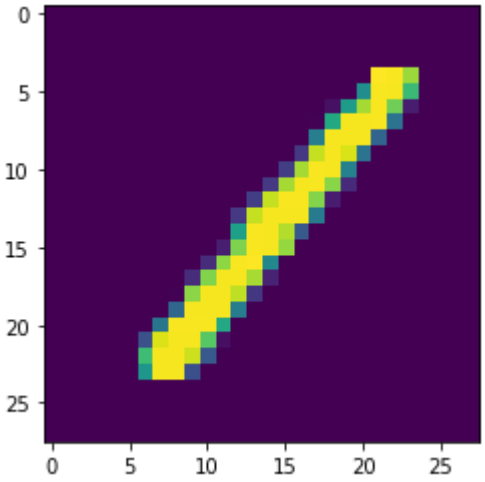
In [8]:
```
history=model.fit(x_train,
y_train,validation_data=(x_test,y_test),epochs=10)
```

```
Epoch 1/10
1875/1875 [==============================] - 13s 6ms/step - loss: 0.6399 -
accuracy: 0.8386 - val_loss: 0.3519 - val_accuracy: 0.9041
Epoch 2/10
1875/1875 [==============================] - 9s 5ms/step - loss: 0.3326 -
accuracy: 0.9071 - val_loss: 0.2870 - val_accuracy: 0.9189
Epoch 3/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2853 -
accuracy: 0.9198 - val_loss: 0.2577 - val_accuracy: 0.9267
Epoch 4/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2556 -
accuracy: 0.9278 - val_loss: 0.2336 - val_accuracy: 0.9347
Epoch 5/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2332 -
accuracy: 0.9345 - val_loss: 0.2154 - val_accuracy: 0.9390
Epoch 6/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2153 -
accuracy: 0.9397 - val_loss: 0.2022 - val_accuracy: 0.9421
Epoch 7/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.2000 -
accuracy: 0.9435 - val_loss: 0.1881 - val_accuracy: 0.9452
Epoch 8/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.1870 -
accuracy: 0.9472 - val_loss: 0.1786 - val_accuracy: 0.9494
Epoch 9/10
1875/1875 [==============================] - 10s 5ms/step - loss: 0.1757 -
accuracy: 0.9507 - val_loss: 0.1707 - val_accuracy: 0.9521
Epoch 10/10
1875/1875 [==============================] - 9s 5ms/step - loss: 0.1656 -
accuracy: 0.9533 - val_loss: 0.1620 - val_accuracy: 0.9525
```

In [9]:
```
test_loss,test_acc=model.evaluate(x_test,y_test)
print("Loss=%.3f" %test_loss)
print("Accuracy=%.3f" %test_acc)
```

```
313/313 [==============================] - 1s 4ms/step - loss: 0.1620 - ac
curacy: 0.9525
Loss=0.162
Accuracy=0.952
```

In [10]:
```
n=random.randint(0,9999)
plt.imshow(x_test[n])
plt.show()
```



In [11]:
```
x_train
```

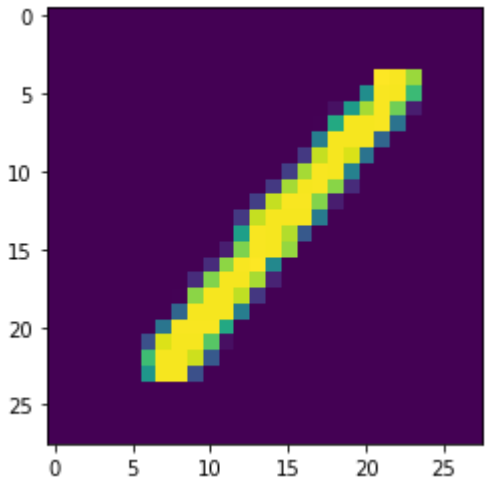
Out[11]:
```
array([[[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       [[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.]]])
```

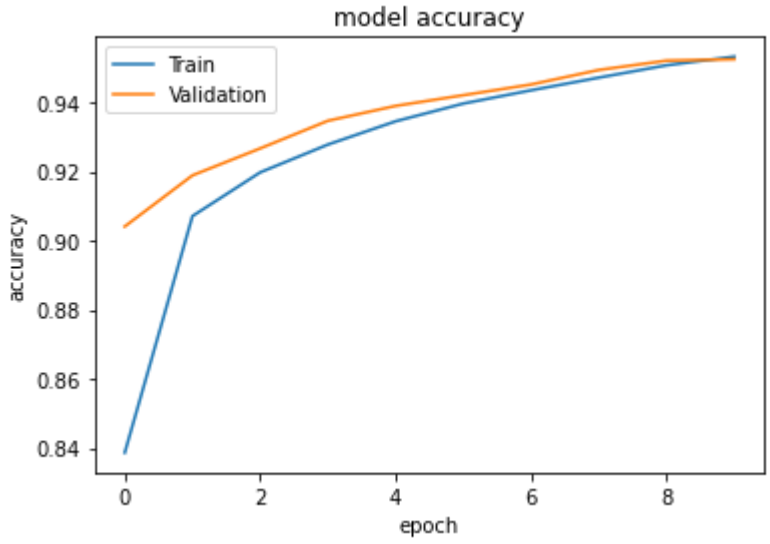
In [13]:
```
predicted_value=model.predict(x_test)
plt.imshow(x_test[n])
plt.show()

print(predicted_value[n])
```

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

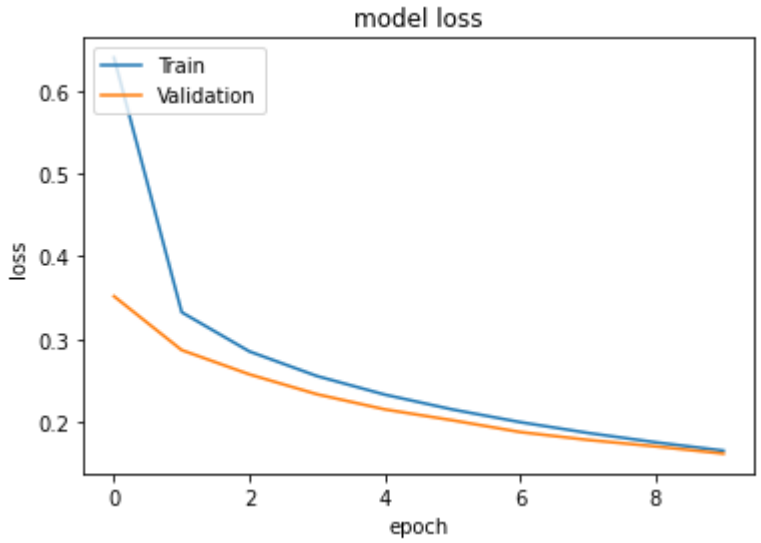



```
[9.3247363e-06 9.9462634e-01 9.4514771e-04 2.9514689e-04 1.3335947e-04
 1.1438889e-04 4.0767285e-05 3.4513610e-04 3.4741354e-03 1.6173248e-05]
```

In [14]:
```
# history.history()
history.history.keys()
# dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])

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()
history.history.keys()
# dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])

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()
```



In [ ]:
```

```