In [4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [5]:
```
class_names=['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat', 'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Anl
```

In [6]:
```
df1 = pd.read_csv(r'C:\Users\PRANAV\Downloads\fashion-mnist_train.csv')
```

In [7]:
```
df1
```

Out[7]:

	label	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel775	pixel776	pixel777	pixel778	pixel779	pi
0	2	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
1	9	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
2	6	0	0	0	0	0	0	0	5	0	...	0	0	0	30	43	
3	0	0	0	0	1	2	0	0	0	0	...	3	0	0	0	0	
4	3	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
59995	9	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
59996	1	0	0	0	0	0	0	0	0	0	...	73	0	0	0	0	
59997	8	0	0	0	0	0	0	0	0	0	...	160	162	163	135	94	
59998	8	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
59999	7	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	

60000 rows × 785 columns

In [8]:
```
x_train = df1.drop("label", axis=1).values
y_train = df1["label"].values
```

In [9]:
```
print("x_train shape: ",x_train.shape)
print("y_train shape: ",y_train.shape)
```

```
x_train shape:  (60000, 784)
y_train shape:  (60000,)
```

In [12]:
```
np.unique(y_train)
```

Out[12]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9], dtype=int64)
```

In [13]:
```
df2 = pd.read_csv(r'C:\Users\PRANAV\Downloads\fashion-mnist_test.csv')
```

In [14]:
```
df2
```

Out[14]:

	label	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel775	pixel776	pixel777	pixel778	pixel779	pix
0	0	0	0	0	0	0	0	0	9	8	...	103	87	56	0	0	
1	1	0	0	0	0	0	0	0	0	0	...	34	0	0	0	0	
2	2	0	0	0	0	0	0	14	53	99	...	0	0	0	0	63	
3	2	0	0	0	0	0	0	0	0	0	...	137	126	140	0	133	
4	3	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
9995	0	0	0	0	0	0	0	0	0	0	...	32	23	14	20	0	
9996	6	0	0	0	0	0	0	0	0	0	...	0	0	0	2	52	
9997	8	0	0	0	0	0	0	0	0	0	...	175	172	172	182	199	
9998	8	0	1	3	0	0	0	0	0	0	...	0	0	0	0	0	
9999	1	0	0	0	0	0	0	0	140	119	...	111	95	75	44	1	

10000 rows × 785 columns

In [15]:
```
x_test = df2.drop("label", axis=1).values
y_test = df2["label"].values
```

In [16]:
```
print("x_test shape: ",x_test.shape)
print("y_test shape: ",y_test.shape)
```

```
x_test shape:  (10000, 784)
y_test shape:  (10000,)
```

In [17]:
```
x_train = x_train.reshape(60000, 28, 28)
x_test = x_test.reshape(10000, 28, 28)
```

In [18]:
```
print(x_train[0])
```

```
[[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   4   0   0   0   0   0  62  61  21  29  23  51 136  61
    0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  88 201 228 225 255 115  62 137 255 235 222
  255 135   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0  47 252 234 238 224 215 215 229 108 180 207 214 224
  231 249 254  45   0   0   0   0   0   0]
 [  0   0   1   0   0 214 222 210 213 224 225 217 220 254 233 219 221 217
  223 221 240 254   0   0   1   0   0   0]
 [  1   0   0   0 128 237 207 224 224 207 216 214 210 208 211 221 208 219
  213 226 211 237 150   0   0   0   0   0]
 [  0   2   0   0 237 222 215 207 210 212 213 206 214 213 214 213 210 215
  214 206 199 218 255  13   0   2   0   0]
 [  0   4   0  85 228 210 218 200 211 208 203 215 210 209 209 210 213 211
  210 217 206 213 231 175   0   0   0   0]
 [  0   0   0 217 224 215 206 205 204 217 230 222 215 224 233 228 232 228
  224 207 212 215 213 229  31   0   4   0]
 [  1   0  21 225 212 212 203 211 225 193 139 136 195 147 156 139 128 162
  197 223 207 220 213 232 177   0   0   0]
 [  0   0 123 226 207 211 209 205 228 158  90 103 186 138 100 121 147 158
  183 226 208 214 209 216 255  13   0   1]
 [  0   0 226 219 202 208 206 205 216 184 156 150 193 170 164 168 188 186
  200 219 216 213 213 211 233 148   0   0]
 [  0  45 227 204 214 211 218 222 221 230 229 221 213 224 233 226 220 219
  221 224 223 217 210 218 213 254   0   0]
 [  0 157 226 203 207 211 209 215 205 198 207 208 201 201 197 203 205 210
  207 213 214 214 214 213 208 234 107   0]
 [  0 235 213 204 211 210 209 213 202 197 204 215 217 213 212 210 206 212
  203 211 218 215 214 208 209 222 230   0]
 [ 52 255 207 200 208 213 210 210 208 207 202 201 209 216 216 216 216 214
  212 205 215 201 228 208 214 212 218  25]
 [118 217 201 206 208 213 208 205 206 210 211 202 199 207 208 209 210 207
  210 210 245 139 119 255 202 203 236 114]
 [171 238 212 203 220 216 217 209 207 205 210 211 206 204 206 209 211 215
  210 206 221 242   0 224 234 230 181  26]
 [ 39 145 201 255 157 115 250 200 207 206 207 213 216 206 205 206 207 206
  215 207 221 238   0   0 188  85   0   0]
 [  0   0   0  31   0 129 253 190 207 208 208 208 209 211 211 209 209 209
```

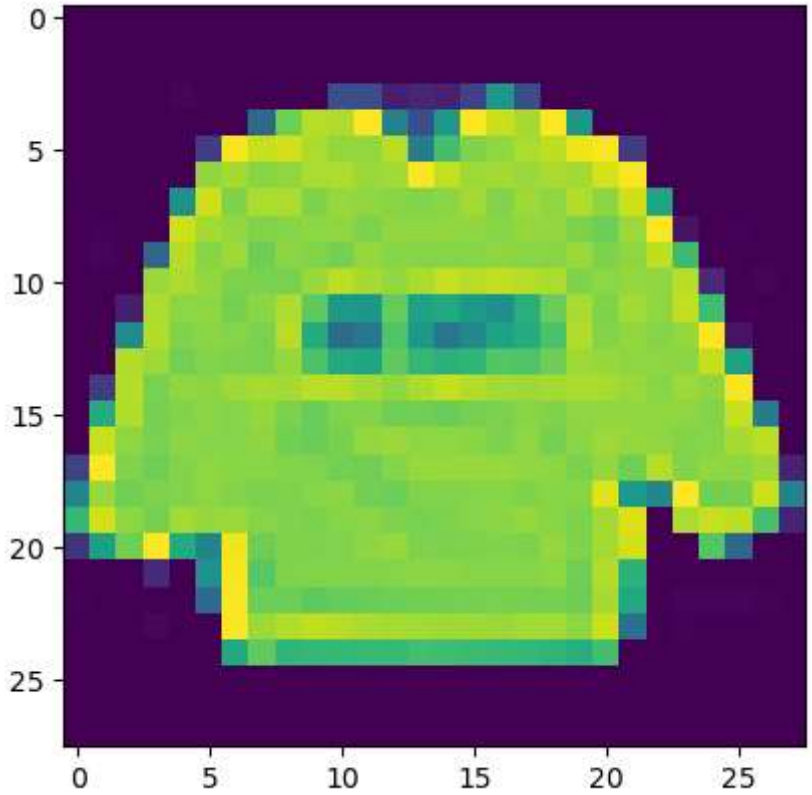
```
 212 201 226 165   0   0   0   0   0   0]
[  2   0   0   0   0  89 254 199 199 192 196 198 199 201 202 203 204 203
 203 200 222 155   0   3   3   3   2   0]
[  0   0   1   5   0   0 255 218 226 232 228 224 222 220 219 219 217 221
 220 212 236  95   0   2   0   0   0   0]
[  0   0   0   0   0   0 155 194 168 170 171 173 173 179 177 175 172 171
 167 161 180   0   0   1   0   1   0   0]
[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0]
[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0]
[  0   0   0   0   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 [19]:
```
y_train[0]
```

Out[19]: 2

In [20]:
```
plt.imshow(x_train[0])
```

Out[20]:
```
<matplotlib.image.AxesImage at 0x1e96408ceb0>
```



In [21]:
```
x_test[10]
```

Out[21]:
```
array([[  0,   0,   0,   0,   0,   0,   0,   1,   0,   0,  83, 142,  50,
          0,   0,   0,   0,  85, 145,  31,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0, 215, 210, 208, 255,
        254, 225, 227, 255, 221, 199, 211, 129,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   2,   0, 105, 213, 187, 187, 204,
        223, 230, 227, 221, 188, 183, 188, 188,   7,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0, 169, 206, 185, 193, 189,
        230, 219, 229, 205, 180, 186, 181, 201,  61,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0, 206, 214, 190, 185, 177,
        204, 244, 215, 174, 181, 177, 187, 209, 118,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   8, 196, 219, 178, 184, 183,
        177, 222, 181, 173, 184, 173, 203, 210, 177,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,  64, 211, 219,  83, 199, 197,
        184, 201, 201, 185, 206, 153, 150, 223, 205,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0, 128, 217, 220,  61, 205, 196,
        188, 194, 211, 199, 203, 159, 112, 226, 194,  30,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0, 165, 222, 253,   0, 203, 197,
        193, 185, 194, 204, 211, 155,  73, 233, 203,  71,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0, 174, 234, 207,   0, 219, 201,
        196, 207, 190, 194, 230, 105,   0, 255, 210,  90,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0, 157, 243, 163,   0, 245, 203,
        215, 209, 215, 182, 231, 142,   0, 255, 223, 109,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0, 150, 241, 142,   0, 230, 192,
        234, 198, 236, 199, 203, 144,   0, 228, 222, 111,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0, 166, 251, 132,  52, 236, 191,
        204, 182, 236, 210, 190, 226,   0, 216, 240, 150,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0, 146, 223,  87, 132, 223, 192,
        196, 186, 215, 201, 184, 231,  55, 122, 218, 112,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 210, 207, 195,
```

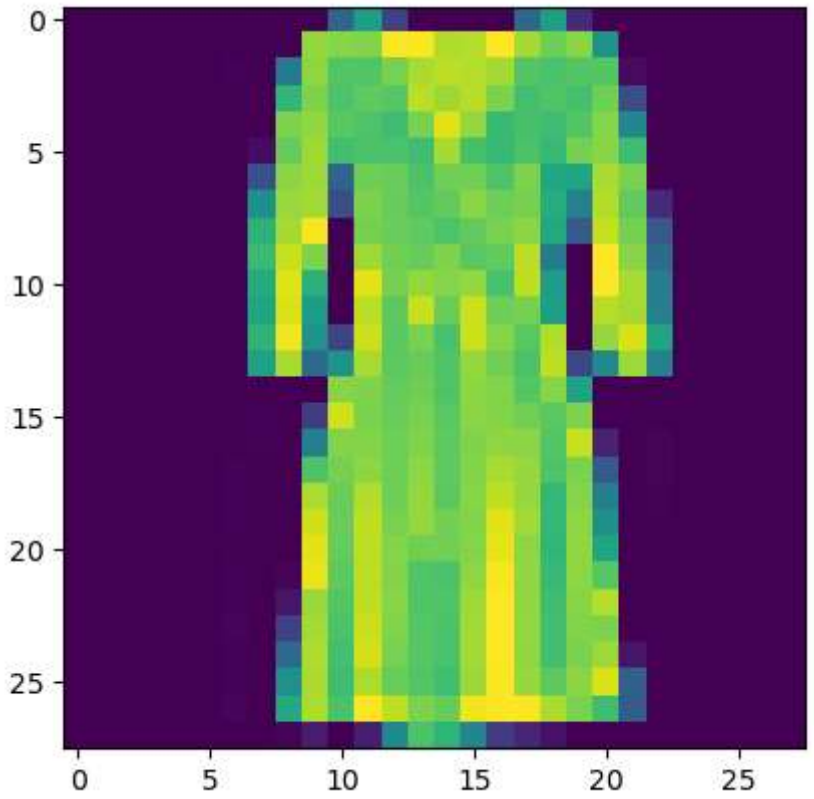
```
 200, 186, 212, 208, 188, 210, 147,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   2,   0,  44, 237, 205, 197,
 204, 190, 211, 208, 201, 191, 207,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   2,   0, 110, 208, 208, 199,
 207, 193, 207, 213, 211, 188, 234,  24,   0,   3,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   2,   0,   0, 184, 203, 212, 199,
 212, 193, 208, 223, 216, 185, 205,  71,   0,   3,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   2,   0,   0, 224, 198, 226, 199,
 215, 191, 210, 231, 216, 170, 209, 110,   0,   2,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   2,   0,   0, 237, 197, 231, 204,
 215, 202, 208, 244, 220, 170, 213, 128,   0,   1,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   1,   0,   0, 245, 196, 230, 209,
 201, 202, 209, 246, 213, 169, 214, 150,   0,   1,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   2,   0,   3, 248, 192, 230, 208,
 186, 184, 213, 253, 214, 173, 212, 189,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   2,   0,  15, 217, 188, 231, 210,
 186, 186, 219, 255, 214, 177, 210, 227,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   3,   0,  49, 222, 183, 235, 207,
 188, 184, 220, 255, 215, 179, 207, 206,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   2,   0,  87, 225, 179, 239, 204,
 189, 183, 221, 255, 214, 180, 205, 218,  15,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   2,   0, 129, 223, 177, 224, 198,
 187, 178, 217, 254, 216, 192, 211, 242,  78,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   3,   0, 156, 224, 183, 255, 231,
 205, 196, 250, 255, 254, 224, 205, 177,  75,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   4,  20,   0,  21, 122,
 184, 167, 118,  45,  27,  12,   0,   0,   0,   0,   0,   0,   0,
   0,   0]], dtype=int64)
```

In [22]:
```
y_test[10]
```

Out[22]:
```
3
```

In [23]:
```
plt.imshow(x_test[10])
```

Out[23]:
```
<matplotlib.image.AxesImage at 0x1e9647db400>
```



In [24]:
```
x_train = x_train/255
x_test = x_test/255
```

In [25]:
```
x_train = x_train.reshape(60000, 28, 28, 1)
x_test = x_test.reshape(10000, 28, 28, 1)
```

In [26]:
```
print("Train Shape :",x_train.shape)
print("Test Shape :",x_test.shape)
print("y_train shape :",y_train.shape)
print("y_test shape :",y_test.shape)
```

```
Train Shape : (60000, 28, 28, 1)
Test Shape : (10000, 28, 28, 1)
y_train shape : (60000,)
y_test shape : (10000,)
```

In [27]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, MaxPooling2D, Flatten
```

In [28]:
```
model=Sequential()

model.add(Conv2D(64, (3,3), activation='relu', input_shape=(28,28,1)))
model.add(MaxPooling2D((2,2)))

model.add(Conv2D(64, (3,3), activation='relu'))
model.add(MaxPooling2D((2,2)))

model.add(Flatten())

model.add(Dense(128,activation='relu'))
model.add(Dense(10,activation='softmax'))

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

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 64)        640

 max_pooling2d (MaxPooling2D  (None, 13, 13, 64)       0
 )

 conv2d_1 (Conv2D)           (None, 11, 11, 64)        36928

 max_pooling2d_1 (MaxPooling  (None, 5, 5, 64)         0
 2D)

 flatten (Flatten)           (None, 1600)              0

 dense (Dense)               (None, 128)               204928

 dense_1 (Dense)             (None, 10)                1290

=================================================================
Total params: 243,786
Trainable params: 243,786
Non-trainable params: 0
_________________________________________________________________
```

In [29]:
```
model.fit(x_train, y_train, epochs=3, verbose=1, validation_data=(x_test,y_test))
```

```
Epoch 1/3
1875/1875 [==============================] - 75s 39ms/step - loss: 0.4372 - accuracy: 0.8416 - val_loss: 0.
3308 - val_accuracy: 0.8764
Epoch 2/3
1875/1875 [==============================] - 72s 38ms/step - loss: 0.2923 - accuracy: 0.8932 - val_loss: 0.
2801 - val_accuracy: 0.8989
Epoch 3/3
1875/1875 [==============================] - 72s 39ms/step - loss: 0.2479 - accuracy: 0.9089 - val_loss: 0.
2396 - val_accuracy: 0.9125
```

Out[29]:
```
<keras.callbacks.History at 0x1e92d6c4e50>
```

In [30]:
```
predictions = model.predict(x_test)
```

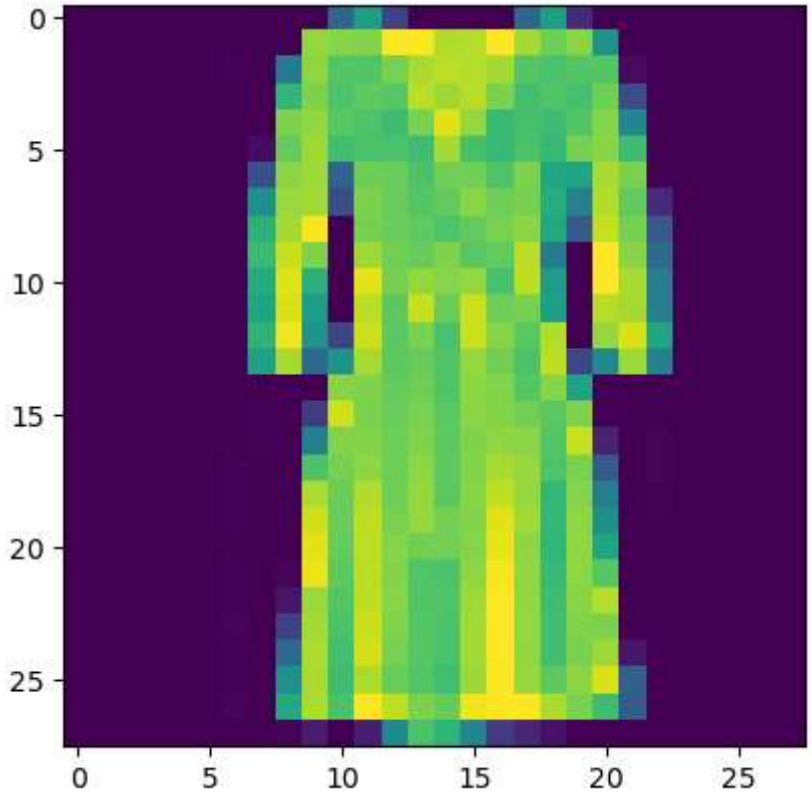
```
313/313 [==============================] - 5s 14ms/step
```

In [31]:
```
import numpy as np

index=10

print(predictions[index])

final_value=np.argmax(predictions[index])

print("Actual label :",y_test[index])
print("Predicted label :",final_value)
print("Class :",class_names[final_value])
```

```
[2.5851827e-04 8.5433959e-07 4.6868139e-05 9.9954766e-01 3.7650629e-05
 4.5716575e-08 1.0573206e-04 4.7540732e-07 1.7320131e-06 4.7126207e-07]
Actual label : 3
Predicted label : 3
Class : Dress
```

In [32]:
```
plt.imshow(x_test[10])
```

Out[32]: <matplotlib.image.AxesImage at 0x1e92d9b9d30>



In [33]:
```
loss, accuracy = model.evaluate(x_test, y_test)

print("Loss :",loss)
print("Accuracy (Test Data) :",accuracy*100)
```

```
313/313 [==============================] - 4s 14ms/step - loss: 0.2396 - accuracy: 0.9125
Loss : 0.23957380652427673
Accuracy (Test Data) : 91.25000238418579
```

In [ ]: