In [6]:
```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# Load the MNIST Fashion Dataset
(x_train, y_train), (x_test, y_test) = keras.datasets.fashion_mnist.load_data
```

In [7]:
```
# Normalize pixel values between 0 and 1
x_train = x_train.astype("float32") / 255.0
x_test = x_test.astype("float32") / 255.0

# Reshape the input images to include a single color channel
x_train = np.expand_dims(x_train, -1)
x_test = np.expand_dims(x_test, -1)

# Convert the labels to categorical format
num_classes = 10
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
```

In [8]:
```
# Define the CNN model
model = keras.Sequential(
    [
        layers.Conv2D(32, kernel_size=(3, 3), activation="relu", input_shape=
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dropout(0.5),
        layers.Dense(num_classes, activation="softmax"),
    ]
)

# Compile the model
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["ac

# Train the model
batch_size = 128
epochs = 10
model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs, validation_
```

```
Epoch 1/10
422/422 [==============================] - 28s 66ms/step - loss: 0.6991 - ac
curacy: 0.7488 - val_loss: 0.4448 - val_accuracy: 0.8395
Epoch 2/10
422/422 [==============================] - 28s 65ms/step - loss: 0.4550 - ac
curacy: 0.8361 - val_loss: 0.3842 - val_accuracy: 0.8602
Epoch 3/10
422/422 [==============================] - 29s 68ms/step - loss: 0.4060 - ac
curacy: 0.8546 - val_loss: 0.3537 - val_accuracy: 0.8697
Epoch 4/10
422/422 [==============================] - 30s 72ms/step - loss: 0.3742 - ac
curacy: 0.8655 - val_loss: 0.3330 - val_accuracy: 0.8827
Epoch 5/10
422/422 [==============================] - 28s 67ms/step - loss: 0.3518 - ac
curacy: 0.8740 - val_loss: 0.3154 - val_accuracy: 0.8867
Epoch 6/10
422/422 [==============================] - 28s 67ms/step - loss: 0.3365 - ac
curacy: 0.8793 - val_loss: 0.3108 - val_accuracy: 0.8855
Epoch 7/10
422/422 [==============================] - 28s 66ms/step - loss: 0.3272 - ac
curacy: 0.8821 - val_loss: 0.2992 - val_accuracy: 0.8928
Epoch 8/10
422/422 [==============================] - 28s 67ms/step - loss: 0.3148 - ac
curacy: 0.8868 - val_loss: 0.2890 - val_accuracy: 0.8975
Epoch 9/10
422/422 [==============================] - 28s 67ms/step - loss: 0.3053 - ac
curacy: 0.8901 - val_loss: 0.2775 - val_accuracy: 0.9007
Epoch 10/10
422/422 [==============================] - 28s 66ms/step - loss: 0.2997 - ac
curacy: 0.8929 - val_loss: 0.2800 - val_accuracy: 0.8970
```

Out[8]: <keras.callbacks.History at 0x2a813957190>

In [9]:
```
# Evaluate the model on the test set
score = model.evaluate(x_test, y_test, verbose=0)
print("Test loss:", score[0])
print("Test accuracy:", score[1])
```

```
Test loss: 0.29885396361351013
Test accuracy: 0.890999972820282
```

In [22]:
```
from PIL import Image
class_names = [
    "T-shirt/top",
    "Trouser",
    "Pullover",
    "Dress",
    "Coat",
    "Sandal",
    "Shirt",
    "Sneaker",
    "Bag",
    "Ankle boot"
]

# Load and preprocess a single image
image = Image.open(r"C:\Users\User\Downloads\Picture1.jpg")  # Replace with t
image = image.resize((28, 28))  # Resize the image to match the input shape o
image = image.convert("L")  # Convert the image to grayscale
image = np.array(image)
image = np.expand_dims(image, 0)
image = np.expand_dims(image, -1)

# Classify the image using the trained model
predictions = model.predict(image)
class_index = np.argmax(predictions[0])
class_name = class_names[class_index]  # Assuming you have a list of class na
print("Predicted class:", class_name)
```

```
1/1 [==============================] - 0s 20ms/step
Predicted class: Pullover
```

In [ ]:
```
```