**PROJECT TITLE: FASHION RECOMMENDER SYSTEM**

**CODE PART:**

import numpy as np

import matplotlib.pyplot as plt

from tensorflow import keras

from tensorflow.keras import layers

# Load the Fashion MNIST dataset

(fashion\_train\_images, fashion\_train\_labels), \_ = keras.datasets.fashion\_mnist.load\_data()

# Normalize the images

fashion\_train\_images = fashion\_train\_images.astype('float32') / 255.0

# Flatten the images for the autoencoder

fashion\_train\_images = fashion\_train\_images.reshape((fashion\_train\_images.shape[0], -1))

# Define the autoencoder model

encoding\_dim = 64 # Size of the encoded representations

input\_img = layers.Input(shape=(28 \* 28,)) # Images are 28x28 pixels

encoded = layers.Dense(encoding\_dim, activation='relu')(input\_img)

decoded = layers.Dense(28 \* 28, activation='sigmoid')(encoded)

autoencoder = keras.models.Model(input\_img, decoded)

# Compile the autoencoder

autoencoder.compile(optimizer='adam', loss='binary\_crossentropy')

# Train the autoencoder

autoencoder.fit(fashion\_train\_images, fashion\_train\_images, epochs=10, batch\_size=256)

# Display original and reconstructed images

n = 10 # Number of images to display

plt.figure(figsize=(20, 4))

for i in range(n):

# Original images

ax = plt.subplot(2, n, i + 1)

plt.imshow(fashion\_train\_images[i].reshape(28, 28), cmap='gray')

plt.axis('off')

# Reconstructed images

reconstructed\_img = autoencoder.predict(fashion\_train\_images[i].reshape(1, -1))

ax = plt.subplot(2, n, i + 1 + n)

plt.imshow(reconstructed\_img.reshape(28, 28), cmap='gray')

plt.axis('off')

plt.show()

**OUTPUT:**