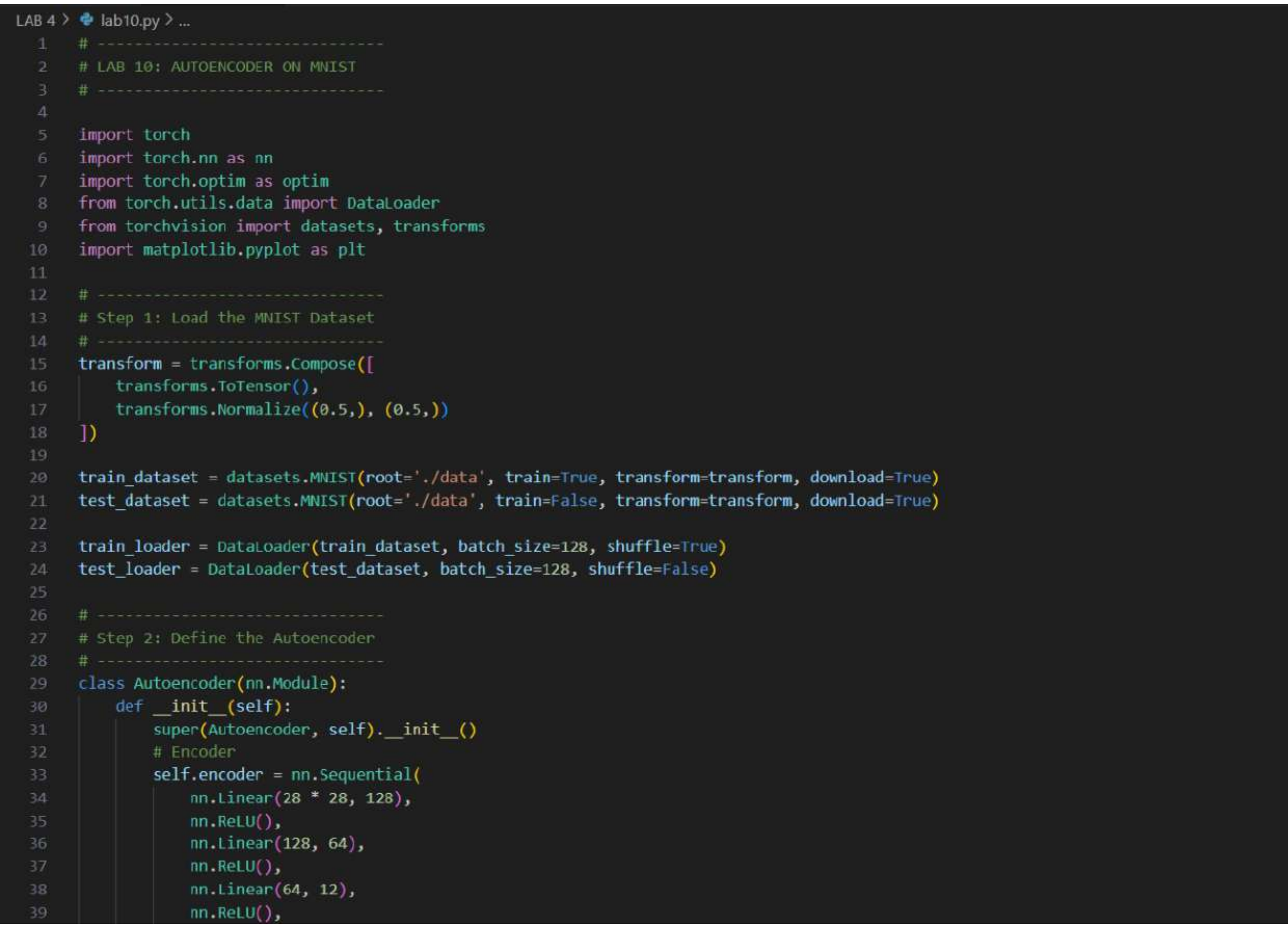
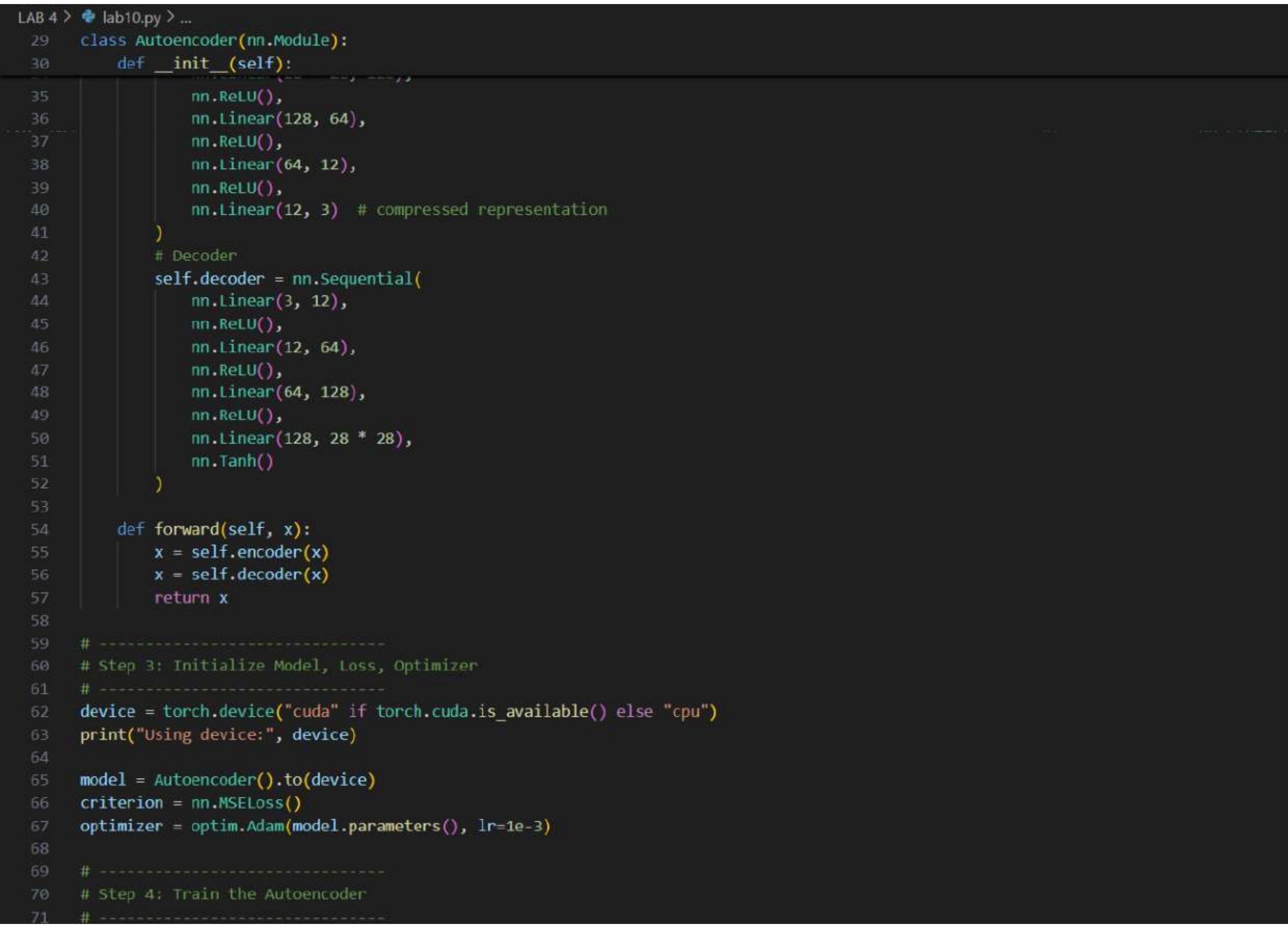
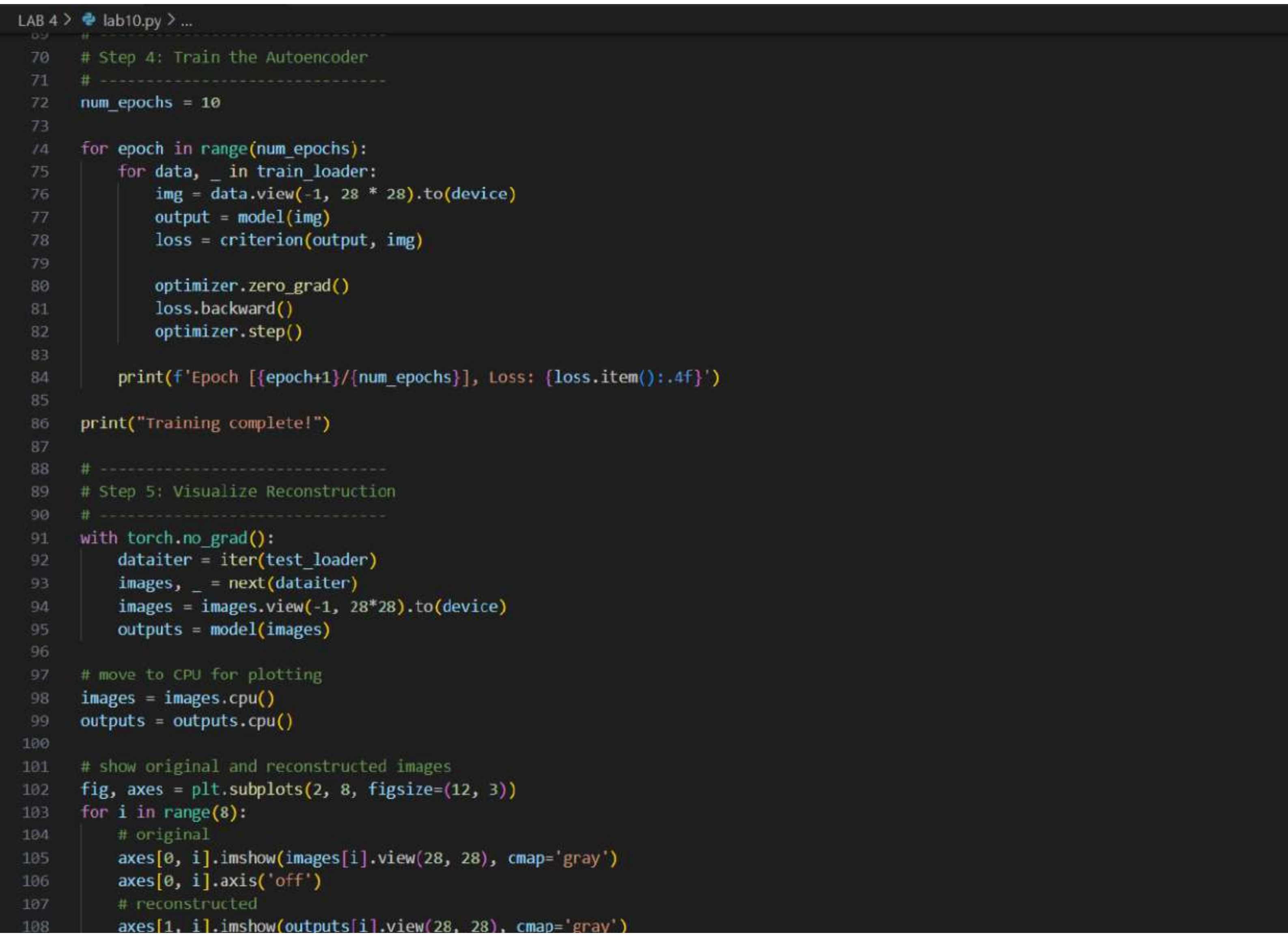


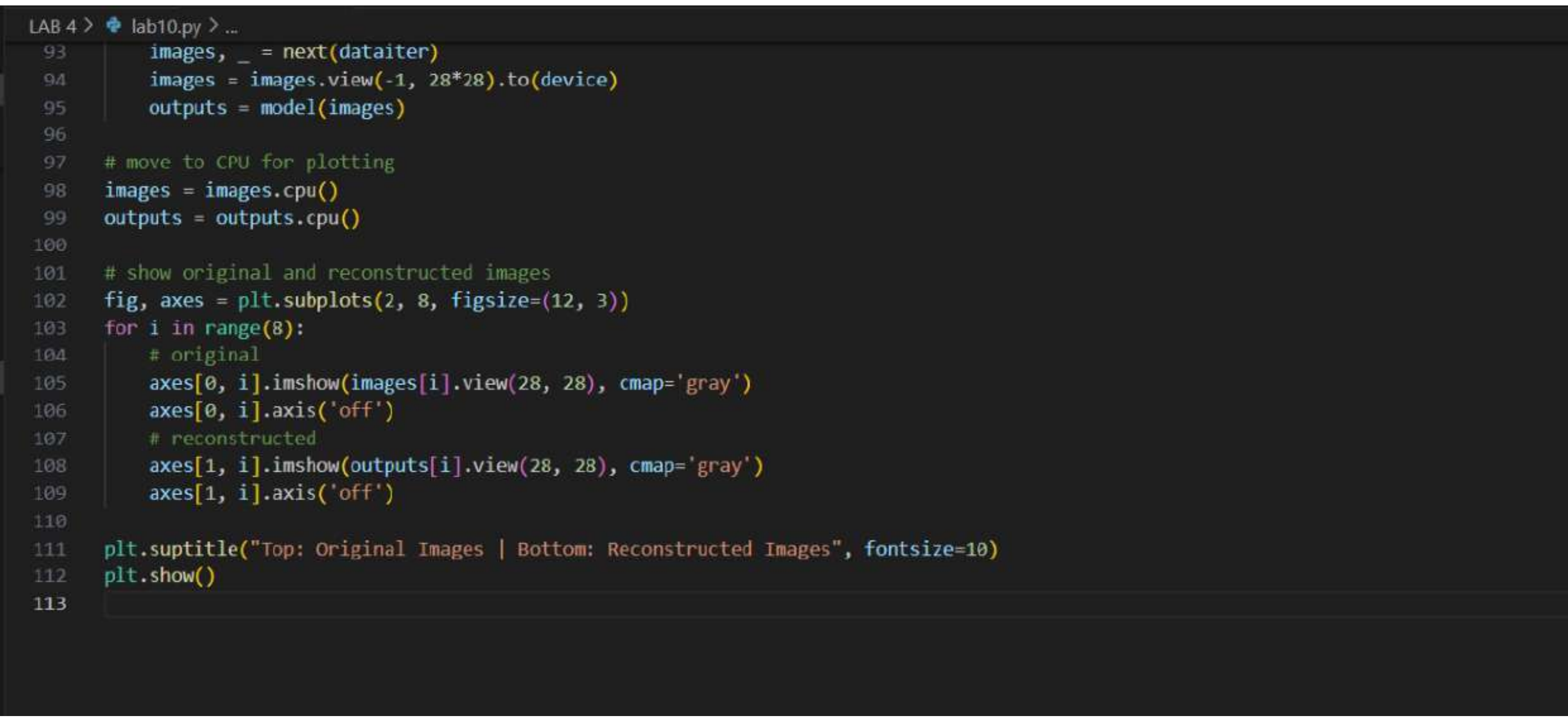
```python
# ------------------------------
# LAB 10: AUTOENCODER ON MNIST
# ------------------------------

import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import DataLoader
from torchvision import datasets, transforms
import matplotlib.pyplot as plt

# ------------------------------
# Step 1: Load the MNIST Dataset
# ------------------------------
transform = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize((0.5,), (0.5,))
])

train_dataset = datasets.MNIST(root='./data', train=True, transform=transform, download=True)
test_dataset = datasets.MNIST(root='./data', train=False, transform=transform, download=True)

train_loader = DataLoader(train_dataset, batch_size=128, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=128, shuffle=False)

# ------------------------------
# Step 2: Define the Autoencoder
# ------------------------------
class Autoencoder(nn.Module):
    def __init__(self):
        super(Autoencoder, self).__init__()
        # Encoder
        self.encoder = nn.Sequential(
            nn.Linear(28 * 28, 128),
            nn.ReLU(),
            nn.Linear(128, 64),
            nn.ReLU(),
            nn.Linear(64, 12),
            nn.ReLU(),
```



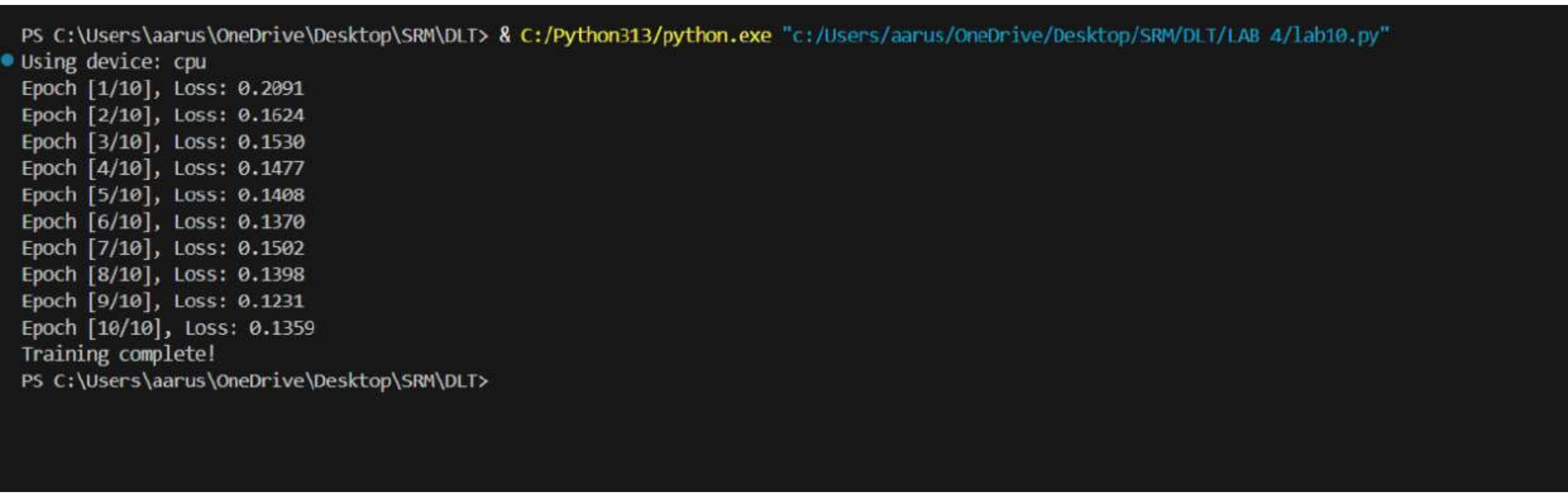
```python
class Autoencoder(nn.Module):
    def __init__(self):
            nn.ReLU(),
            nn.Linear(128, 64),
            nn.ReLU(),
            nn.Linear(64, 12),
            nn.ReLU(),
            nn.Linear(12, 3)  # compressed representation
        )
        # Decoder
        self.decoder = nn.Sequential(
            nn.Linear(3, 12),
            nn.ReLU(),
            nn.Linear(12, 64),
            nn.ReLU(),
            nn.Linear(64, 128),
            nn.ReLU(),
            nn.Linear(128, 28 * 28),
            nn.Tanh()
        )

    def forward(self, x):
        x = self.encoder(x)
        x = self.decoder(x)
        return x

# ------------------------------
# Step 3: Initialize Model, Loss, Optimizer
# ------------------------------
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print("Using device:", device)

model = Autoencoder().to(device)
criterion = nn.MSELoss()
optimizer = optim.Adam(model.parameters(), lr=1e-3)

# ------------------------------
# Step 4: Train the Autoencoder
# ------------------------------
```



```
# Step 4: Train the Autoencoder
# -------------------------------
num_epochs = 10

for epoch in range(num_epochs):
    for data, _ in train_loader:
        img = data.view(-1, 28 * 28).to(device)
        output = model(img)
        loss = criterion(output, img)

        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

    print(f'Epoch [{epoch+1}/{num_epochs}], Loss: {loss.item():.4f}')

print("Training complete!")

# -------------------------------
# Step 5: Visualize Reconstruction
# -------------------------------
with torch.no_grad():
    dataiter = iter(test_loader)
    images, _ = next(dataiter)
    images = images.view(-1, 28*28).to(device)
    outputs = model(images)

# move to CPU for plotting
images = images.cpu()
outputs = outputs.cpu()

# show original and reconstructed images
fig, axes = plt.subplots(2, 8, figsize=(12, 3))
for i in range(8):
    # original
    axes[0, i].imshow(images[i].view(28, 28), cmap='gray')
    axes[0, i].axis('off')
    # reconstructed
    axes[1, i].imshow(outputs[i].view(28, 28), cmap='gray')
```



```python
    images, _ = next(dataiter)
    images = images.view(-1, 28*28).to(device)
    outputs = model(images)

# move to CPU for plotting
images = images.cpu()
outputs = outputs.cpu()

# show original and reconstructed images
fig, axes = plt.subplots(2, 8, figsize=(12, 3))
for i in range(8):
    # original
    axes[0, i].imshow(images[i].view(28, 28), cmap='gray')
    axes[0, i].axis('off')
    # reconstructed
    axes[1, i].imshow(outputs[i].view(28, 28), cmap='gray')
    axes[1, i].axis('off')

plt.suptitle("Top: Original Images | Bottom: Reconstructed Images", fontsize=10)
plt.show()

```



```
PS C:\Users\aarus\OneDrive\Desktop\SRM\DLT> & C:/Python313/python.exe "c:/Users/aarus/OneDrive/Desktop/SRM/DLT/LAB 4/lab10.py"
Using device: cpu
Epoch [1/10], Loss: 0.2091
Epoch [2/10], Loss: 0.1624
Epoch [3/10], Loss: 0.1530
Epoch [4/10], Loss: 0.1477
Epoch [5/10], Loss: 0.1408
Epoch [6/10], Loss: 0.1370
Epoch [7/10], Loss: 0.1502
Epoch [8/10], Loss: 0.1398
Epoch [9/10], Loss: 0.1231
Epoch [10/10], Loss: 0.1359
Training complete!
PS C:\Users\aarus\OneDrive\Desktop\SRM\DLT>
```