In [2]:
```
import mnist_loader
import matplotlib.pyplot as plt
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
```

1

In [38]:
```
training_data,validation_data,test_data= mnist_loader.load_data_wrapper()
training_data =list(training_data)
validation_data =list(validation_data)
test_data =list(test_data)

def plot_digit(digit):
    # A function to plot a vector of length 784 as a 28 x 28 image
    digit_image = digit.reshape(28,28)
    plt.imshow(digit_image, cmap = plt.get_cmap('gray'))
    plt.axis("off")
    plt.show()
```

a

In [39]:
```
def sigmoid(z):
    return 1.0 / (1.0 + np.exp(-z))

def sigmoid_prime(z):
    return sigmoid(z) * (1 - sigmoid(z))

class Network(object):
    def __init__(self, sizes):
        self.num_layers = len(sizes)
        self.sizes = sizes
        self.biases = [np.random.randn(y, 1) for y in sizes[1:]]
        self.weights = [np.random.randn(y, x) for x, y in zip(sizes[:-1], sizes[1:]

    def feedforward(self, a):
        for b, w in zip(self.biases, self.weights):
            a = sigmoid(np.dot(w, a) + b)
        return a

    def SGD(self, training_data, epochs, mini_batch_size, eta, test_data=None):
        training_data = list(training_data)
        n = len(training_data)
        if test_data:
            test_data = list(test_data)
            n_test = len(test_data)
        for j in range(epochs):
            random.shuffle(training_data)
            mini_batches = [training_data[k:k + mini_batch_size] for k in range(0,
            for mini_batch in mini_batches:
                self.update_mini_batch(mini_batch, eta)
            if test_data:
                print("Epoch {} : {} / {}".format(j, self.evaluate(test_data), n_te
            else:
                print("Epoch {} complete".format(j))

    def update_mini_batch(self, mini_batch, eta):
        nabla_b = [np.zeros(b.shape) for b in self.biases]
        nabla_w = [np.zeros(w.shape) for w in self.weights]
```

```
        for x, y in mini_batch:
            delta_nabla_b, delta_nabla_w = self.backprop(x, y)
            nabla_b = [nb + dnb for nb, dnb in zip(nabla_b, delta_nabla_b)]
            nabla_w = [nw + dnw for nw, dnw in zip(nabla_w, delta_nabla_w)]
        self.weights = [w - (eta / len(mini_batch)) * nw for w, nw in zip(self.weig
        self.biases = [b - (eta / len(mini_batch)) * nb for b, nb in zip(self.biase

    def backprop(self, x, y):
        nabla_b = [np.zeros(b.shape) for b in self.biases]
        nabla_w = [np.zeros(w.shape) for w in self.weights]
        activation = x
        activations = [x]
        zs = []
        for b, w in zip(self.biases, self.weights):
            z = np.dot(w, activation) + b
            zs.append(z)
            activation = sigmoid(z)
            activations.append(activation)
        delta = self.cost_derivative(activations[-1], y) * sigmoid_prime(zs[-1])
        nabla_b[-1] = delta
        nabla_w[-1] = np.dot(delta, activations[-2].transpose())
        for l in range(2, self.num_layers):
            z = zs[-l]
            sp = sigmoid_prime(z)
            delta = np.dot(self.weights[-l + 1].transpose(), delta) * sp
            nabla_b[-l] = delta
            nabla_w[-l] = np.dot(delta, activations[-l - 1].transpose())
        return (nabla_b, nabla_w)

    def evaluate(self, test_data):
        test_results = [(np.argmax(self.feedforward(x)), y) for (x, y) in test_data
        return sum(int(x == y) for (x, y) in test_results)

    def cost_derivative(self, output_activations, y):
        return (output_activations - y)
```

In [40]:
```
encoder=Network([784,10])
encoder.SGD(training_data,10,10,3.0,test_data=test_data)
```

```
Epoch 0 : 5799 / 10000
Epoch 1 : 6527 / 10000
Epoch 2 : 7389 / 10000
Epoch 3 : 7493 / 10000
Epoch 4 : 7494 / 10000
Epoch 5 : 7575 / 10000
Epoch 6 : 8369 / 10000
Epoch 7 : 8337 / 10000
Epoch 8 : 8344 / 10000
Epoch 9 : 8387 / 10000
```

b

In [42]:
```
decoder=Network([10, 784])

latent_vectors=[encoder.feedforward(x) for x, _ in training_data]
targets =[x for x, _ in training_data]
training_data_decoder=list(zip(latent_vectors, targets))
decoder.SGD(training_data_decoder,10,10,3.0)
```

```
Epoch 0 complete
Epoch 1 complete
Epoch 2 complete
Epoch 3 complete
Epoch 4 complete
Epoch 5 complete
Epoch 6 complete
Epoch 7 complete
Epoch 8 complete
Epoch 9 complete
```

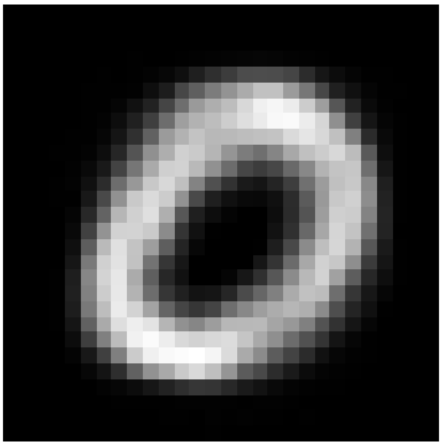
c

Each column of the decoder's weight matrix represents a basis vector in the original 784 dimensional space, capturing features or patterns like strokes or edges that are combined to reconstruct the input image from the latent vector.

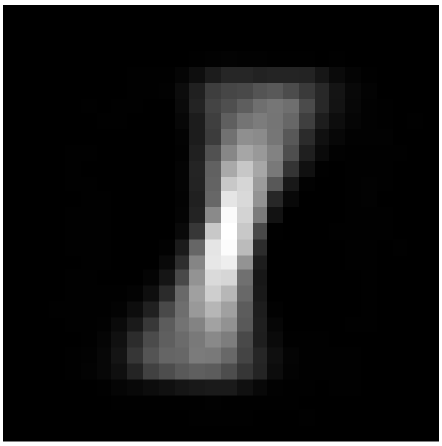
d

In [43]:
```
perfect_vectors=[np.eye(10)[:,k].reshape(10,1) for k in range(10)]

reconstructed_images=[decoder.feedforward(vec) for vec in perfect_vectors]


for digit, image in enumerate(perfect_reconstructed_images):
    print(f"Reconstructed Image for Perfect Example of Digit {digit}:")
    plot_digit(image.reshape(28,28))
```

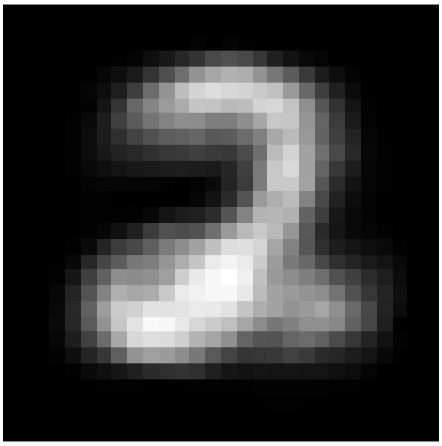
```
Reconstructed Image for Perfect Example of Digit 0:
```



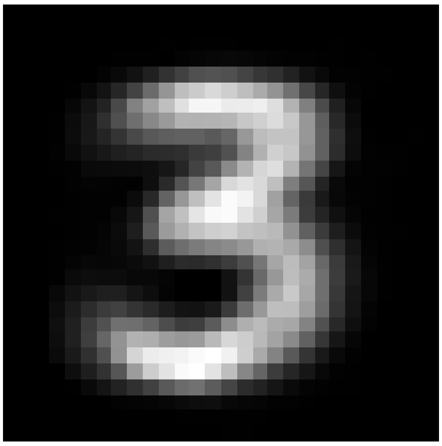
```
Reconstructed Image for Perfect Example of Digit 1:
```



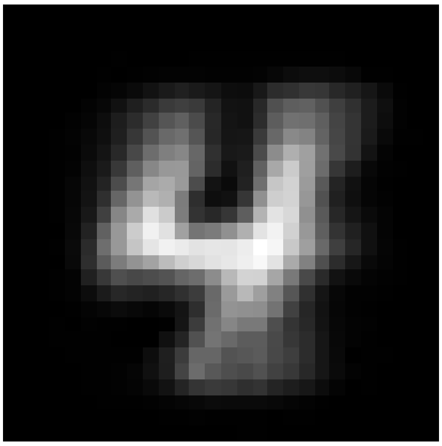
```
Reconstructed Image for Perfect Example of Digit 2:
```



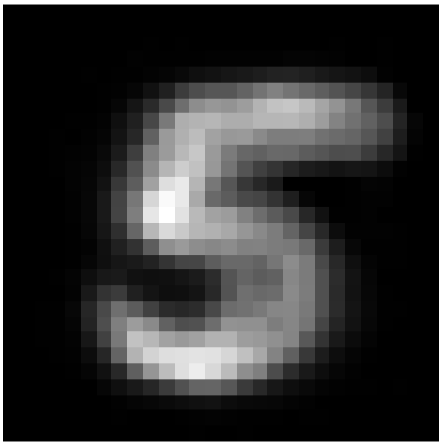
Reconstructed Image for Perfect Example of Digit 3:



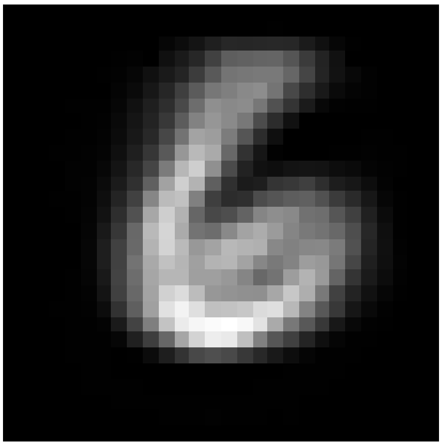
Reconstructed Image for Perfect Example of Digit 4:



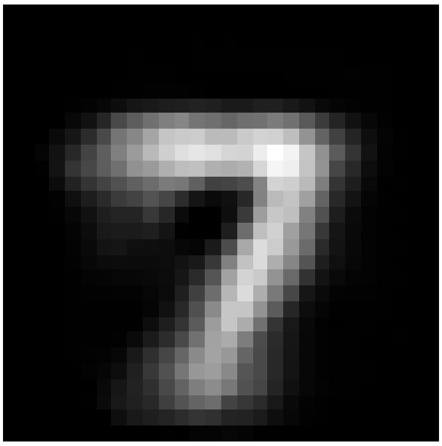
Reconstructed Image for Perfect Example of Digit 5:



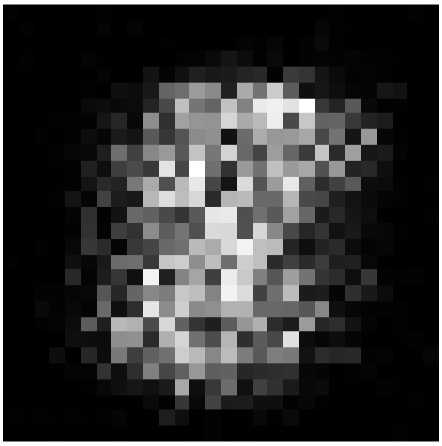
Reconstructed Image for Perfect Example of Digit 6:



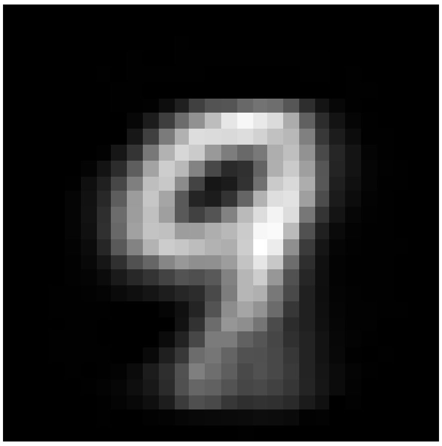
```
Reconstructed Image for Perfect Example of Digit 7:
```



```
Reconstructed Image for Perfect Example of Digit 8:
```



```
Reconstructed Image for Perfect Example of Digit 9:
```



e

In [46]:
```
class Autoencoder:
    def __init__(self, encoder, decoder):
        self.encoder =encoder
        self.decoder =decoder

    def encode(self,input_data):
        return self.encoder.feedforward(input_data)

    def decode(self,latent_representation):
        return self.decoder.feedforward(latent_representation)

    def reconstruct(self,input_data):
        latent_representation =self.encode(input_data)
        reconstructed_output =self.decode(latent_representation)
        return reconstructed_output

    def train(self,training_data,epochs_encoder,epochs_decoder,mini_batch_size,eta,
        #Train encoder
        self.encoder.SGD(training_data,epochs_encoder,mini_batch_size,eta,test_data

        #latent vectors using trained encoder
        latent_vectors =[self.encoder.feedforward(x) for x, _ in training_data]
        targets=[x for x, _ in training_data]
        training_data_decoder =list(zip(latent_vectors, targets))

        #Train decoder
        self.decoder.SGD(training_data_decoder,epochs_decoder,mini_batch_size,eta)
```

f

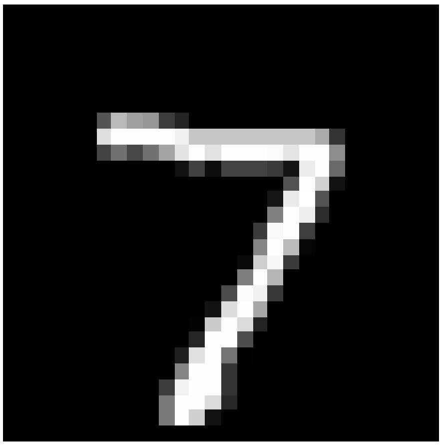
In [104…
```
encoder=Network([784,10])
decoder=Network([10,784])
autoencoder=Autoencoder(encoder,decoder)

autoencoder.train(training_data,10,10,10,3.0,test_data)
```

```
Epoch 0 : 8018 / 10000
Epoch 1 : 8215 / 10000
Epoch 2 : 8308 / 10000
Epoch 3 : 8310 / 10000
Epoch 4 : 8323 / 10000
Epoch 5 : 8356 / 10000
Epoch 6 : 8358 / 10000
Epoch 7 : 8367 / 10000
Epoch 8 : 8359 / 10000
Epoch 9 : 8384 / 10000
Epoch 0 complete
Epoch 1 complete
Epoch 2 complete
Epoch 3 complete
Epoch 4 complete
Epoch 5 complete
Epoch 6 complete
Epoch 7 complete
Epoch 8 complete
Epoch 9 complete
```

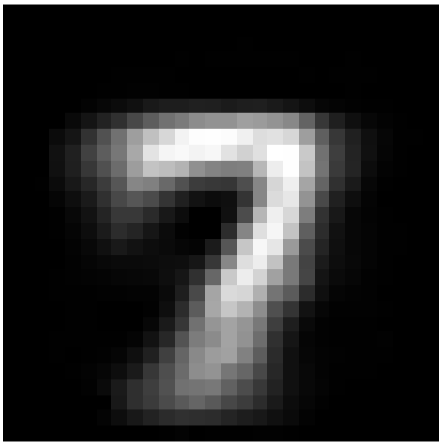
In [106…
```
indices=[0,1,2,3,4]

for i in indices:
    x,y=test_data[i]

    reconstructed_image=autoencoder.reconstruct(x)
```

```
    print(f"Original Image Labeled as {y}:")
    plot_digit(x.reshape(28,28))
    print("Reconstructed Image:")
    plot_digit(reconstructed_image.reshape(28,28))
```

```
Original Image Labeled as 7:
```

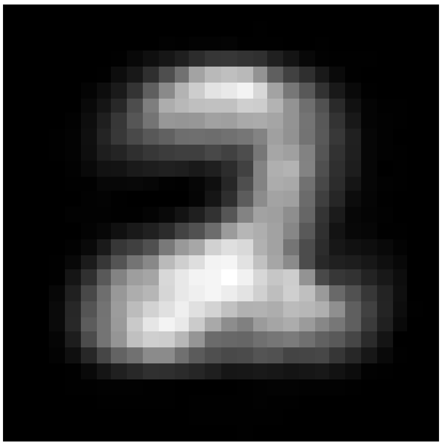


```
Reconstructed Image:
```

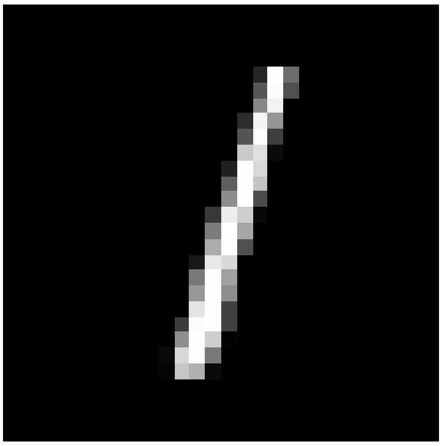


```
Original Image Labeled as 2:
```

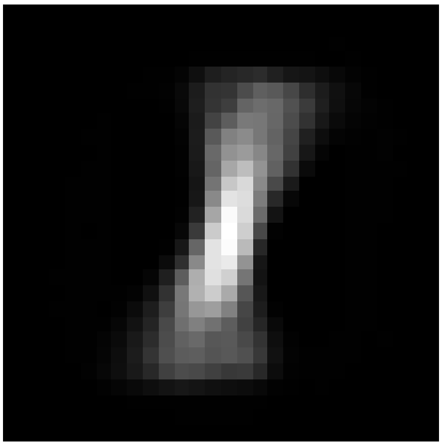
```
Reconstructed Image:
```



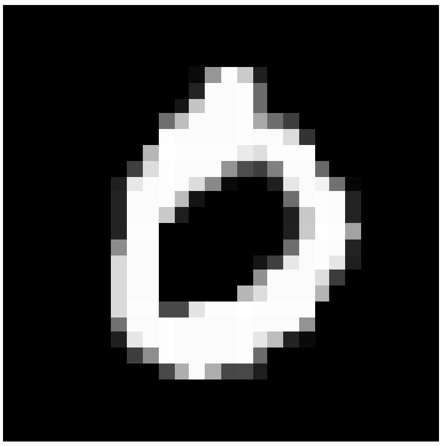
```
Original Image Labeled as 1:
```



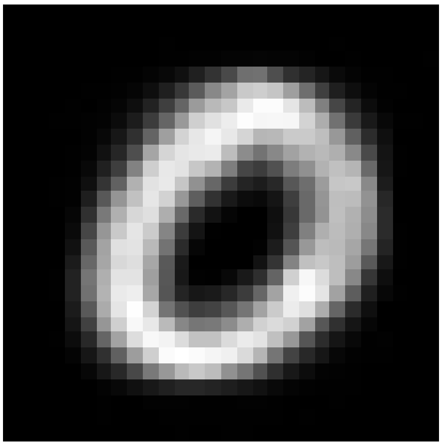
```
Reconstructed Image:
```



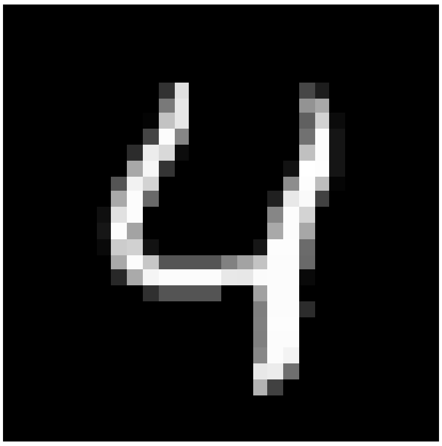
```
Original Image Labeled as 0:
```



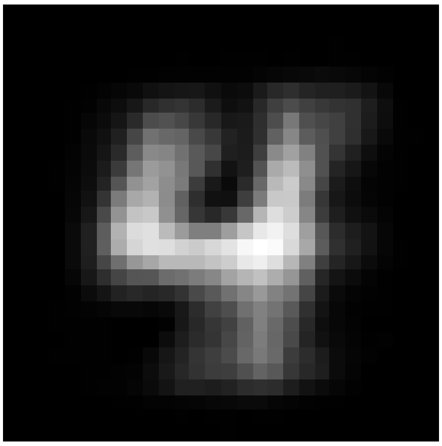
```
Reconstructed Image:
```



```
Original Image Labeled as 4:
```



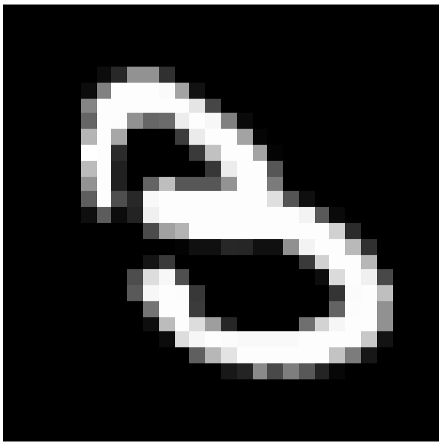
```
Reconstructed Image:
```



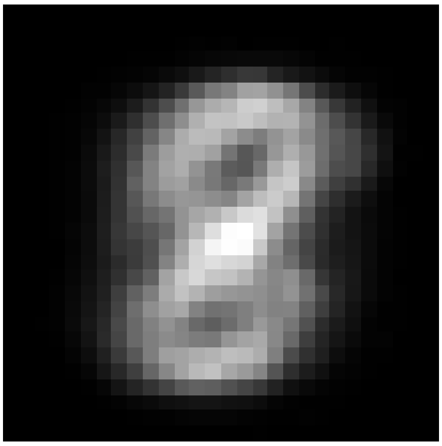
In [107…

```
indices=[18,33,38,66,97]

for i in indices:
    x,y=test_data[i]

    reconstructed_image=autoencoder.reconstruct(x)

    print(f"Original Image Labeled as {y}:")
    plot_digit(x.reshape(28,28))
    print("Reconstructed Image:")
    plot_digit(reconstructed_image.reshape(28,28))
```

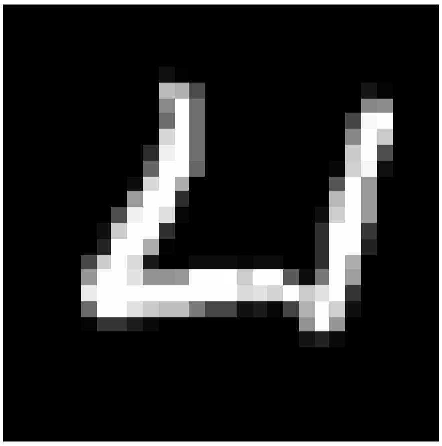
```
Original Image Labeled as 3:
```



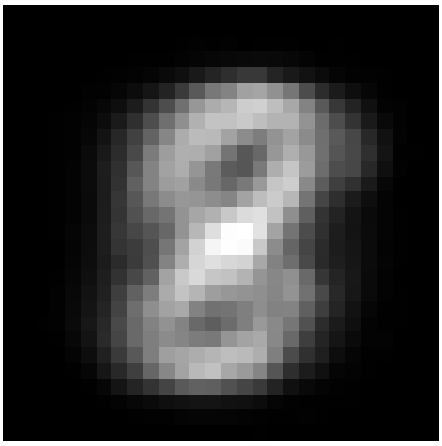
Reconstructed Image:



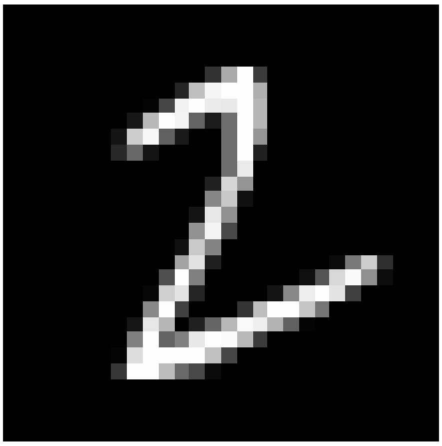
Original Image Labeled as 4:



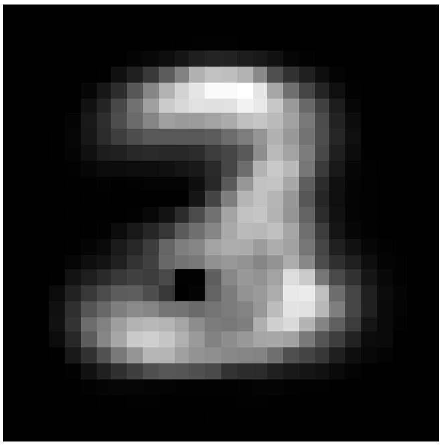
Reconstructed Image:



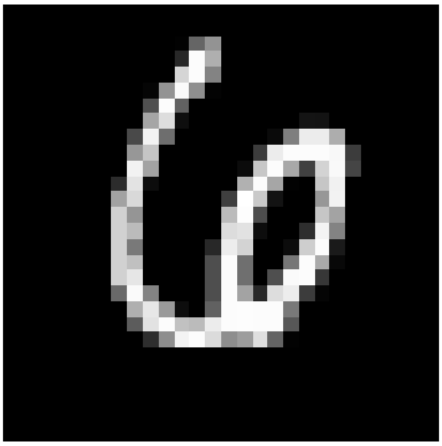
Original Image Labeled as 2:



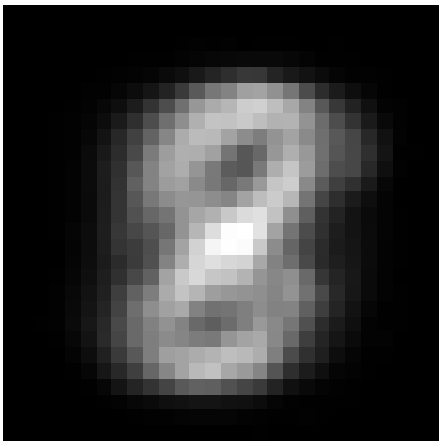
Reconstructed Image:



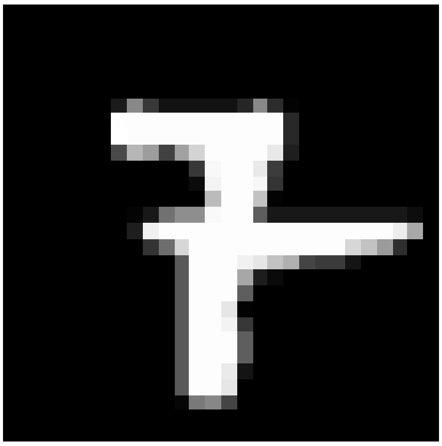
Original Image Labeled as 6:



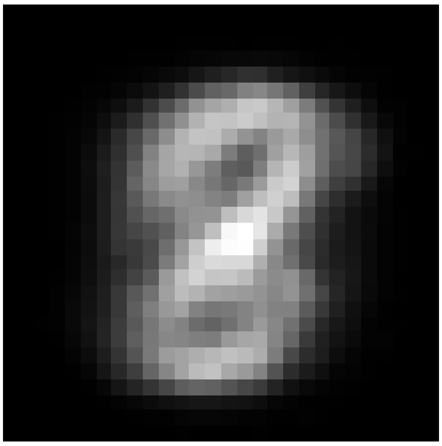
Reconstructed Image:



Original Image Labeled as 7:



Reconstructed Image:



g

a.1

In [108…

```
encoder=Network([784,40,10])
encoder.SGD(training_data,10,10,3.0,test_data=test_data)
```

```
Epoch 0 : 9099 / 10000
Epoch 1 : 9286 / 10000
Epoch 2 : 9303 / 10000
Epoch 3 : 9391 / 10000
Epoch 4 : 9403 / 10000
Epoch 5 : 9429 / 10000
Epoch 6 : 9412 / 10000
Epoch 7 : 9460 / 10000
Epoch 8 : 9464 / 10000
Epoch 9 : 9490 / 10000
```

b.1

In [109…

```
decoder=Network([10,40, 784])

latent_vectors=[encoder.feedforward(x) for x, _ in training_data]
targets =[x for x, _ in training_data]
training_data_decoder=list(zip(latent_vectors, targets))
decoder.SGD(training_data_decoder,10,10,3.0)
```

```
Epoch 0 complete
Epoch 1 complete
Epoch 2 complete
Epoch 3 complete
Epoch 4 complete
Epoch 5 complete
Epoch 6 complete
Epoch 7 complete
Epoch 8 complete
Epoch 9 complete
```

c.1

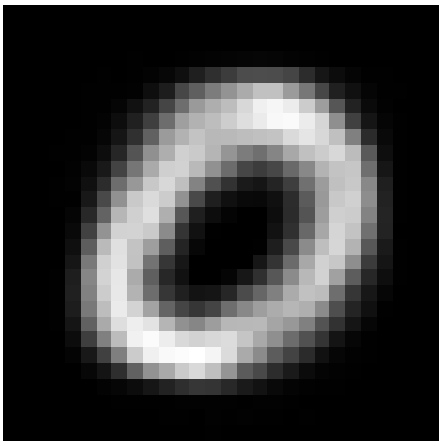
Each column of the decoder's weight matrices represents how the latent dimensions combine to form abstract features in the hidden layer, which are then mapped to reconstruct specific details of the original image.

d.1

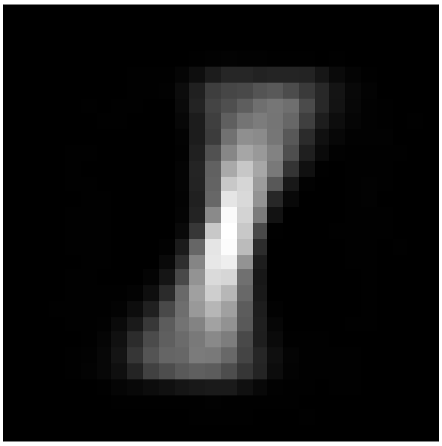
In [110…

```
perfect_vectors=[np.eye(10)[:,k].reshape(10,1) for k in range(10)]

reconstructed_images=[decoder.feedforward(vec) for vec in perfect_vectors]


for digit, image in enumerate(perfect_reconstructed_images):
    print(f"Reconstructed Image for Perfect Example of Digit {digit}:")
    plot_digit(image.reshape(28,28))
```

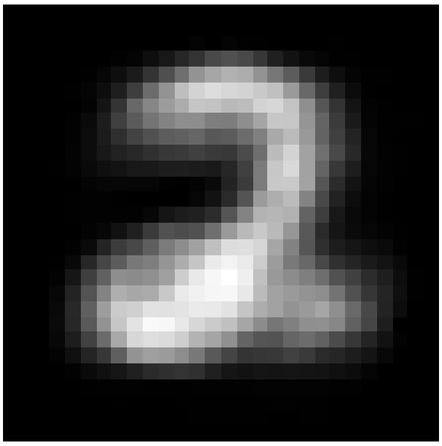
```
Reconstructed Image for Perfect Example of Digit 0:
```



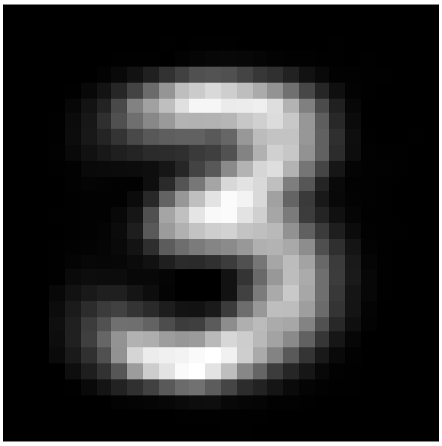
```
Reconstructed Image for Perfect Example of Digit 1:
```



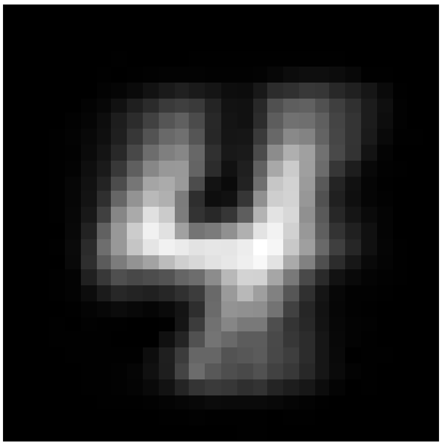
```
Reconstructed Image for Perfect Example of Digit 2:
```



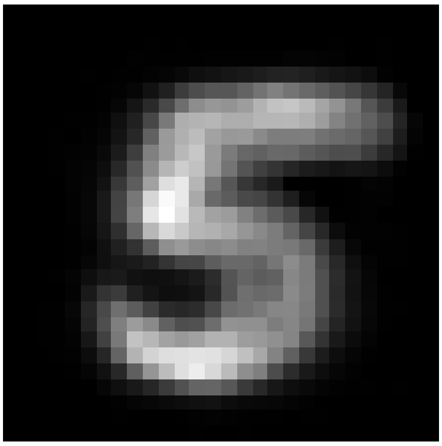
Reconstructed Image for Perfect Example of Digit 3:



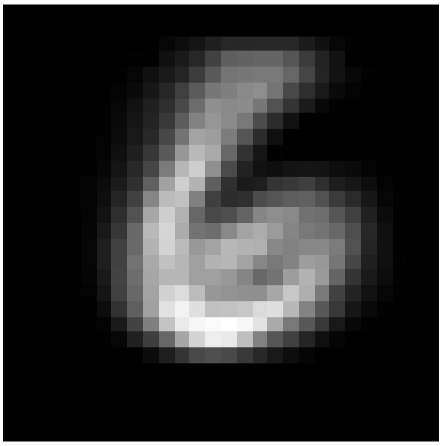
Reconstructed Image for Perfect Example of Digit 4:



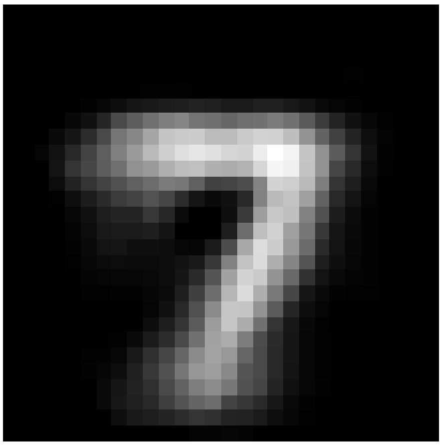
Reconstructed Image for Perfect Example of Digit 5:



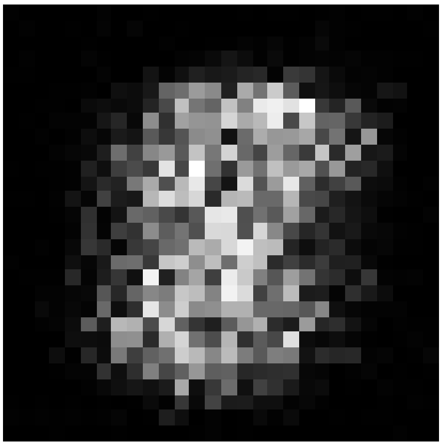
Reconstructed Image for Perfect Example of Digit 6:



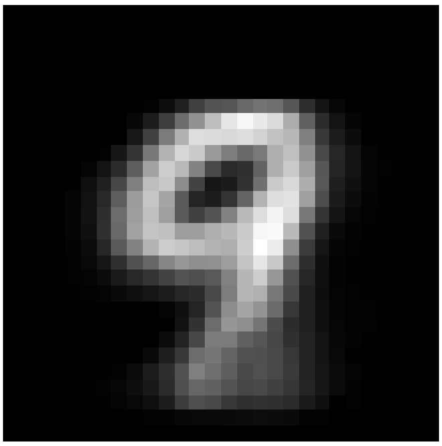
Reconstructed Image for Perfect Example of Digit 7:



Reconstructed Image for Perfect Example of Digit 8:



Reconstructed Image for Perfect Example of Digit 9:



e.1

Same as before

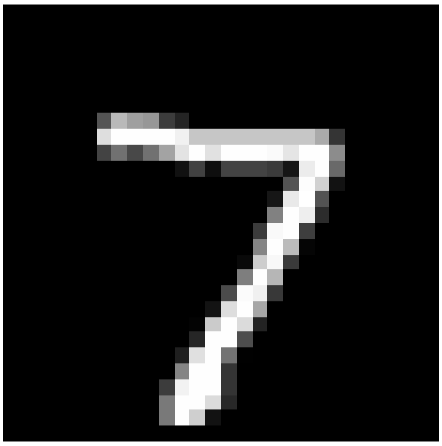
f.1

In [89]:
```
encoder=Network([784,40, 10])
decoder=Network([10, 40,784])
autoencoder=Autoencoder(encoder,decoder)

autoencoder.train(training_data,10,10,10,3.0,test_data)
```

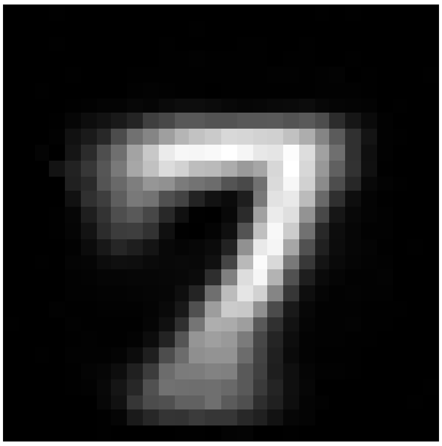
```
Epoch 0 : 7336 / 10000
Epoch 1 : 8284 / 10000
Epoch 2 : 9287 / 10000
Epoch 3 : 9345 / 10000
Epoch 4 : 9360 / 10000
Epoch 5 : 9402 / 10000
Epoch 6 : 9445 / 10000
Epoch 7 : 9459 / 10000
Epoch 8 : 9445 / 10000
Epoch 9 : 9464 / 10000
Epoch 0 complete
Epoch 1 complete
Epoch 2 complete
Epoch 3 complete
Epoch 4 complete
Epoch 5 complete
Epoch 6 complete
Epoch 7 complete
Epoch 8 complete
Epoch 9 complete
```

In [99]:
```
indices=[0,1,2,3,4]

for i in indices:
    x,y=test_data[i]

    reconstructed_image=autoencoder.reconstruct(x)

    print(f"Original Image Labeled as {y}:")
    plot_digit(x.reshape(28,28))
    print("Reconstructed Image:")
    plot_digit(reconstructed_image.reshape(28,28))
```

```
Original Image Labeled as 7:
```

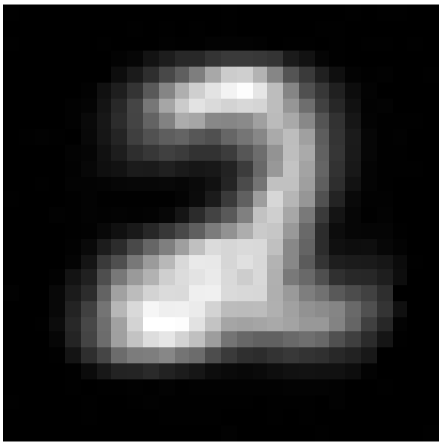


```
Reconstructed Image:
```

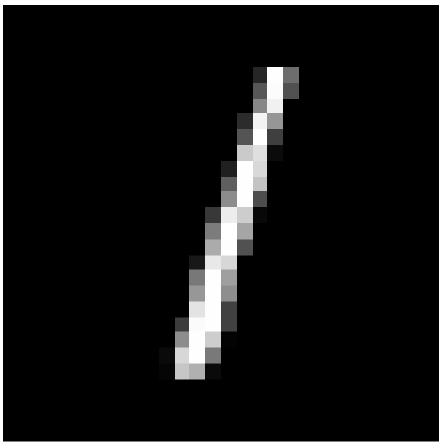


Original Image Labeled as 2:

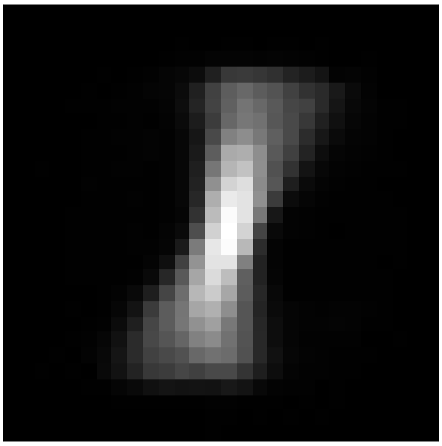
Reconstructed Image:



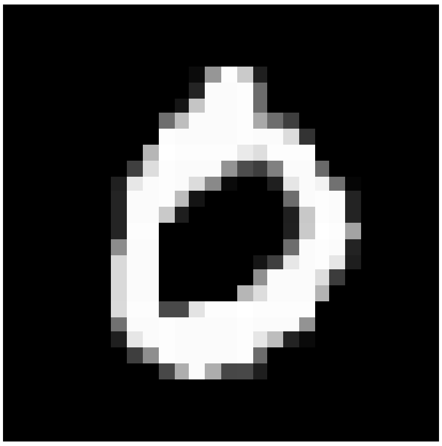
Original Image Labeled as 1:



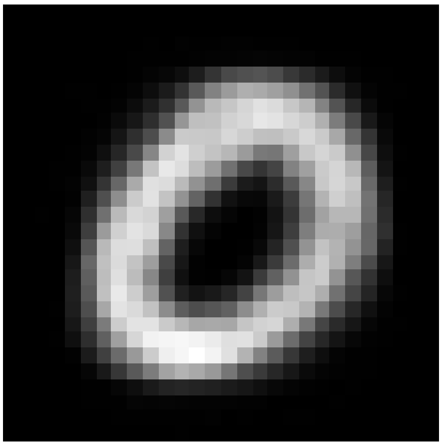
Reconstructed Image:



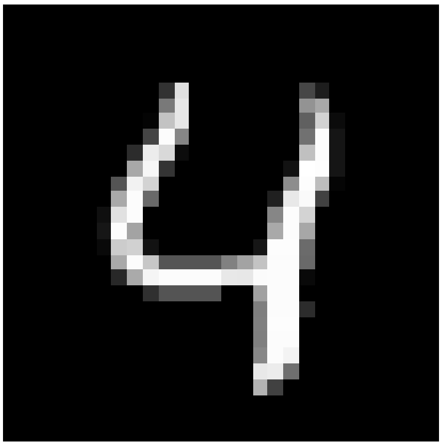
```
Original Image Labeled as 0:
```



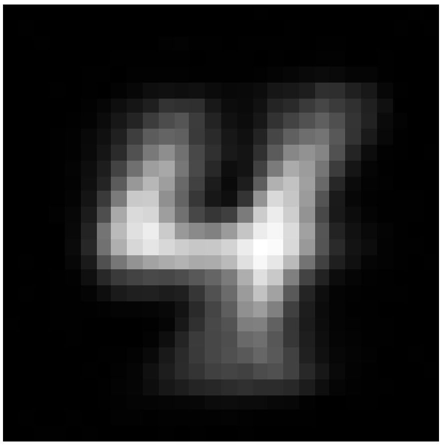
```
Reconstructed Image:
```



```
Original Image Labeled as 4:
```



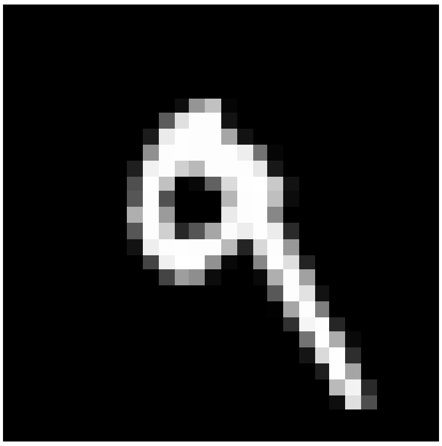
```
Reconstructed Image:
```



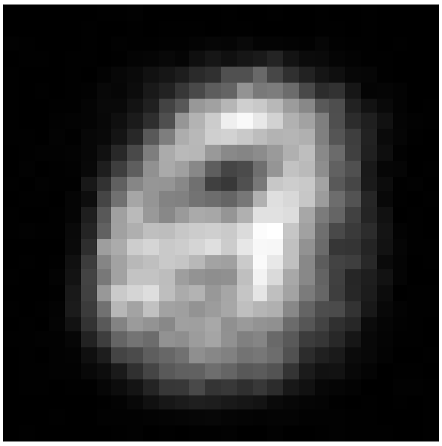
In [100…

```
indices=[7,24,61,149,151]

for i in indices:
    x,y=test_data[i]

    reconstructed_image=autoencoder.reconstruct(x)

    print(f"Original Image Labeled as {y}:")
    plot_digit(x.reshape(28,28))
    print("Reconstructed Image:")
    plot_digit(reconstructed_image.reshape(28,28))
```

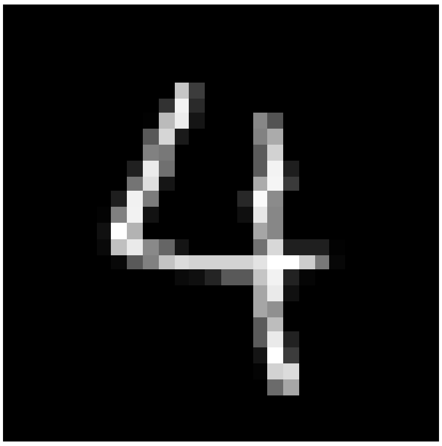
```
Original Image Labeled as 9:
```



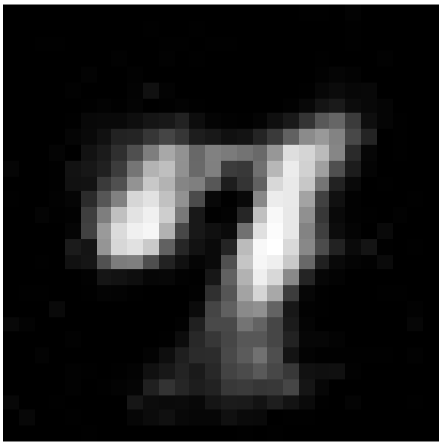
```
Reconstructed Image:
```



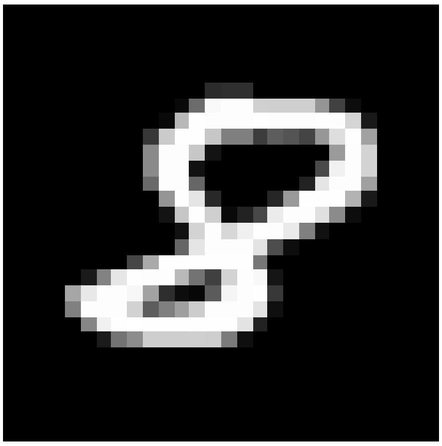
```
Original Image Labeled as 4:
```



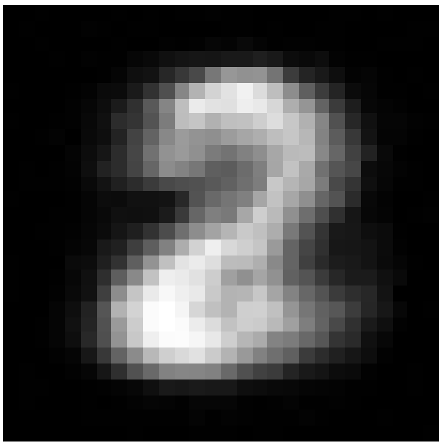
Reconstructed Image:



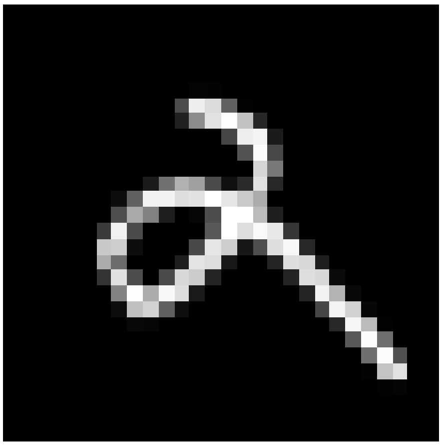
Original Image Labeled as 8:



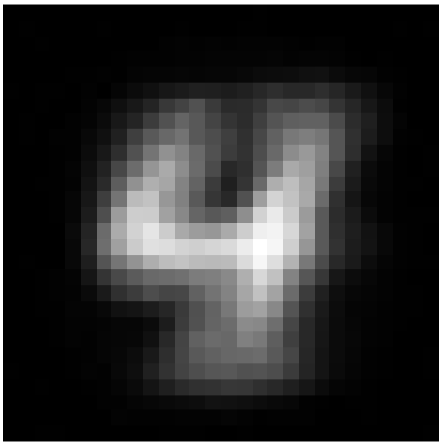
Reconstructed Image:



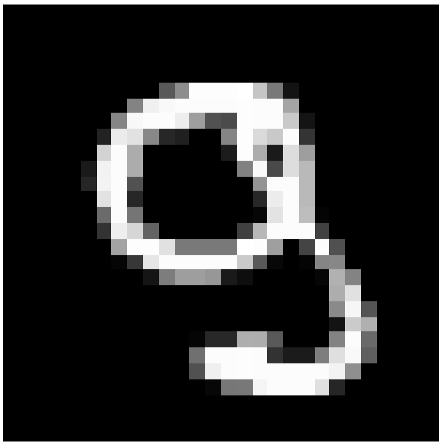
Original Image Labeled as 2:



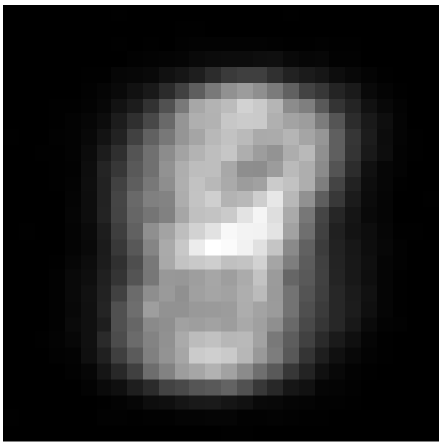
Reconstructed Image:



Original Image Labeled as 9:



Reconstructed Image:



2

a

In [8]:
```
x_vals=np.linspace(0,2*np.pi,10000)

f_vals=np.sin(x_vals)
g_vals=(1+np.sin(x_vals))/2

f_training_data=list(zip(x_vals,f_vals))
g_training_data=list(zip(x_vals,g_vals))
```

b

In [9]:
```
def create_model():
    model=Sequential([
        Dense(3,activation='sigmoid',input_shape=(1,)),
        Dense(1,activation='sigmoid')
    ])
    model.compile(optimizer='adam',loss='mse')
    return model

f_model=create_model()
g_model=create_model()

g_model.summary()
```

```
C:\Users\antho\anaconda3\lib\site-packages\keras\src\layers\core\dense.py:88: User
Warning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using
Sequential models, prefer using an `Input(shape)` object as the first layer in the
model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

Model: "sequential_3"

Layer (type)	Output Shape	
dense_6 (Dense)	(None, 3)	
dense_7 (Dense)	(None, 1)	

Total params: 10 (40.00 B)

Trainable params: 10 (40.00 B)

Non-trainable params: 0 (0.00 B)

c

In [10]:
```
#input shape of (1,) for Keras
x_vals=x_vals.reshape(-1,1)
f_vals=f_vals.reshape(-1,1)
g_vals=g_vals.reshape(-1,1)

f_model.fit(x_vals,f_vals,epochs=200,batch_size=20,verbose=1)
g_model.fit(x_vals,g_vals,epochs=200,batch_size=20,verbose=1)
```

```
Epoch 1/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.6970
Epoch 2/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.5199
Epoch 3/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.4791
Epoch 4/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.4693
Epoch 5/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.4422
Epoch 6/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.4231
Epoch 7/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.3957
Epoch 8/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.3691
Epoch 9/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.3438
Epoch 10/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.3383
Epoch 11/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.3245
Epoch 12/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 993us/step - loss: 0.3126
Epoch 13/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.3090
Epoch 14/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.3065
Epoch 15/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.3031
Epoch 16/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2994
Epoch 17/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.3007
Epoch 18/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2976
Epoch 19/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2950
Epoch 20/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2940
Epoch 21/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2937
Epoch 22/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2979
Epoch 23/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2951
Epoch 24/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2910
Epoch 25/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2892
Epoch 26/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2847
Epoch 27/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2900
Epoch 28/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2826
Epoch 29/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2850
Epoch 30/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2873
Epoch 31/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2874
Epoch 32/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2845
```

```
Epoch 33/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2875
Epoch 34/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2824
Epoch 35/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2825
Epoch 36/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2840
Epoch 37/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2871
Epoch 38/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2782
Epoch 39/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2861
Epoch 40/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2821
Epoch 41/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2858
Epoch 42/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2792
Epoch 43/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2830
Epoch 44/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2801
Epoch 45/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2870
Epoch 46/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2847
Epoch 47/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2793
Epoch 48/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2776
Epoch 49/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2899
Epoch 50/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2792
Epoch 51/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2853
Epoch 52/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2851
Epoch 53/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2802
Epoch 54/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2817
Epoch 55/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2768
Epoch 56/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2822
Epoch 57/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2811
Epoch 58/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2861
Epoch 59/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2820
Epoch 60/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2865
Epoch 61/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2832
Epoch 62/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2744
Epoch 63/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - loss: 0.2678
Epoch 64/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2579
```

```
Epoch 65/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2608
Epoch 66/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2531
Epoch 67/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2653
Epoch 68/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2537
Epoch 69/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2535
Epoch 70/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2547
Epoch 71/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2556
Epoch 72/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2520
Epoch 73/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2543
Epoch 74/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2530
Epoch 75/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2529
Epoch 76/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2613
Epoch 77/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2464
Epoch 78/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2421
Epoch 79/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2501
Epoch 80/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2510
Epoch 81/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - loss: 0.2524
Epoch 82/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - loss: 0.2519
Epoch 83/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - loss: 0.2510
Epoch 84/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - loss: 0.2485
Epoch 85/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - loss: 0.2517
Epoch 86/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - loss: 0.2492
Epoch 87/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - loss: 0.2508
Epoch 88/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - loss: 0.2542
Epoch 89/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2534
Epoch 90/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2604
Epoch 91/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2477
Epoch 92/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2541
Epoch 93/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2465
Epoch 94/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2594
Epoch 95/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2534
Epoch 96/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2523
```

```
Epoch 97/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2542
Epoch 98/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2537
Epoch 99/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2499
Epoch 100/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2507
Epoch 101/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2544
Epoch 102/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2515
Epoch 103/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2540
Epoch 104/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - loss: 0.2554
Epoch 105/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - loss: 0.2549
Epoch 106/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2534
Epoch 107/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2507
Epoch 108/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2566
Epoch 109/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2497
Epoch 110/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2542
Epoch 111/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2553
Epoch 112/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2494
Epoch 113/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2558
Epoch 114/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - loss: 0.2519
Epoch 115/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2509
Epoch 116/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2495
Epoch 117/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2481
Epoch 118/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2544
Epoch 119/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2603
Epoch 120/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2517
Epoch 121/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2510
Epoch 122/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2512
Epoch 123/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2495
Epoch 124/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2498
Epoch 125/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2469
Epoch 126/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2549
Epoch 127/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2581
Epoch 128/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2470
```

```
Epoch 129/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2520
Epoch 130/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2493
Epoch 131/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2458
Epoch 132/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2480
Epoch 133/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2568
Epoch 134/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2514
Epoch 135/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2432
Epoch 136/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2506
Epoch 137/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2535
Epoch 138/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2510
Epoch 139/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2457
Epoch 140/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2511
Epoch 141/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2528
Epoch 142/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2520
Epoch 143/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2457
Epoch 144/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2504
Epoch 145/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2548
Epoch 146/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2480
Epoch 147/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2490
Epoch 148/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2490
Epoch 149/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2502
Epoch 150/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.2494
Epoch 151/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2451
Epoch 152/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2491
Epoch 153/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2525
Epoch 154/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2488
Epoch 155/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2461
Epoch 156/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2564
Epoch 157/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2602
Epoch 158/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2499
Epoch 159/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2509
Epoch 160/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2528
```

```
Epoch 161/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2507
Epoch 162/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2492
Epoch 163/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2562
Epoch 164/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2495
Epoch 165/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2478
Epoch 166/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2526
Epoch 167/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2500
Epoch 168/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2504
Epoch 169/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2514
Epoch 170/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2446
Epoch 171/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2456
Epoch 172/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2481
Epoch 173/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2578
Epoch 174/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2468
Epoch 175/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2468
Epoch 176/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2502
Epoch 177/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2457
Epoch 178/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2513
Epoch 179/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2568
Epoch 180/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2549
Epoch 181/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2569
Epoch 182/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2563
Epoch 183/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2515
Epoch 184/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2471
Epoch 185/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2478
Epoch 186/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2524
Epoch 187/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2512
Epoch 188/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2455
Epoch 189/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2428
Epoch 190/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2475
Epoch 191/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2536
Epoch 192/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2528
```

```
Epoch 193/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2537
Epoch 194/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2457
Epoch 195/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2578
Epoch 196/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2481
Epoch 197/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2498
Epoch 198/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2532
Epoch 199/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2448
Epoch 200/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.2484
Epoch 1/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.1136
Epoch 2/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0869
Epoch 3/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0674
Epoch 4/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0530
Epoch 5/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0431
Epoch 6/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0359
Epoch 7/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0312
Epoch 8/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0277
Epoch 9/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0250
Epoch 10/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0229
Epoch 11/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0199
Epoch 12/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0173
Epoch 13/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0139
Epoch 14/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0127
Epoch 15/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0113
Epoch 16/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0106
Epoch 17/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0102
Epoch 18/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0094
Epoch 19/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0096
Epoch 20/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0092
Epoch 21/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0091
Epoch 22/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0094
Epoch 23/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0091
Epoch 24/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
```

```
Epoch 25/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 26/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0092
Epoch 27/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0091
Epoch 28/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 29/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 30/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0091
Epoch 31/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 32/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 33/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 34/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 35/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 36/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 37/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 38/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
Epoch 39/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0092
Epoch 40/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 41/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 42/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
Epoch 43/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 44/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
Epoch 45/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
Epoch 46/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 47/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 48/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 49/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 50/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 51/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 52/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0093
Epoch 53/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 54/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 55/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
Epoch 56/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
```

```
Epoch 57/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 58/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 59/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0091
Epoch 60/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
Epoch 61/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 62/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 63/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 64/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
Epoch 65/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 66/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 67/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 68/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 69/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 70/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 71/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 72/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 73/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 74/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 75/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 76/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 77/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 78/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
Epoch 79/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
Epoch 80/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 81/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 82/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 83/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 84/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 85/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 86/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 87/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 88/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
```

```
Epoch 89/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 90/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 91/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 92/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 93/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 94/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 95/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 96/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 97/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 98/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 99/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 100/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 101/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0089
Epoch 102/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 103/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0088
Epoch 104/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 105/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 106/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
Epoch 107/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 108/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 109/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 110/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 111/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 112/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 113/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0090
Epoch 114/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 115/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 116/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 117/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 118/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 119/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 120/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0083
```

```
Epoch 121/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 122/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 123/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 124/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 125/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 126/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 127/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
Epoch 128/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 129/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 130/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 131/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0087
Epoch 132/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 133/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 134/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 135/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 136/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 137/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 138/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 139/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 140/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 141/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 142/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 143/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0086
Epoch 144/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 145/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 146/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 147/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 148/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 149/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 150/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 151/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
Epoch 152/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
```

```
Epoch 153/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 154/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0085
Epoch 155/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 156/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0079
Epoch 157/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0079
Epoch 158/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
Epoch 159/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0081
Epoch 160/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0081
Epoch 161/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0081
Epoch 162/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0077
Epoch 163/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 164/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 165/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
Epoch 166/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 167/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 168/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
Epoch 169/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 170/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 171/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 172/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 173/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0084
Epoch 174/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
Epoch 175/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 176/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0083
Epoch 177/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0079
Epoch 178/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 179/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 180/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
Epoch 181/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0079
Epoch 182/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
Epoch 183/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 184/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
```

```
Epoch 185/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0079
Epoch 186/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 187/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0077
Epoch 188/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 189/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 190/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0079
Epoch 191/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 192/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 193/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
Epoch 194/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 195/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0081
Epoch 196/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0083
Epoch 197/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0080
Epoch 198/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.0077
Epoch 199/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0080
Epoch 200/200
500/500 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - loss: 0.0082
```

Out[10]:

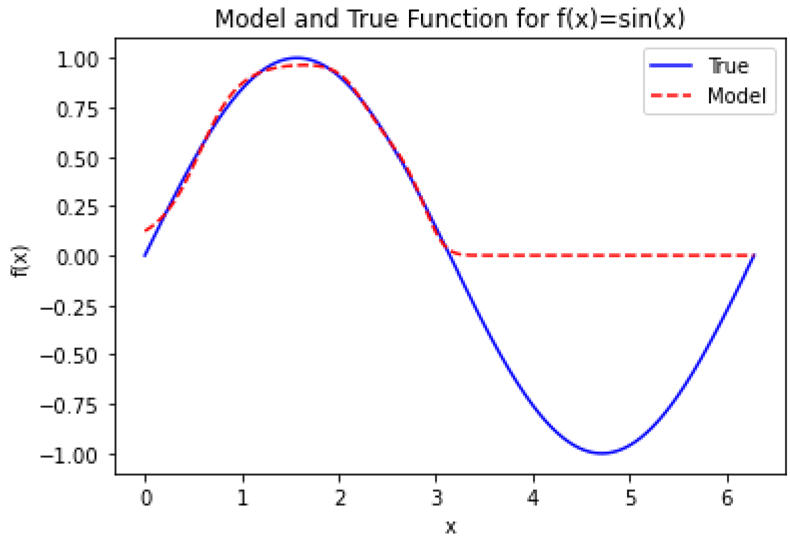
```
<keras.src.callbacks.history.History at 0x286fc88eb50>
```

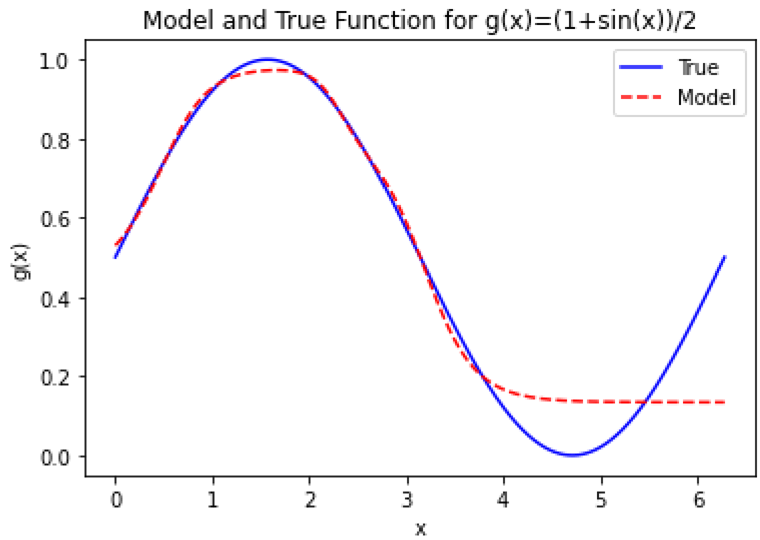
d

In [11]:

```python
x_test=x_vals.reshape(-1,1)
f_true=np.sin(x_test)
g_true=(1+np.sin(x_test))/2

f_pred=f_model.predict(x_test)
g_pred=g_model.predict(x_test)


plt.plot(x_test,f_true,label="True",color='blue')
plt.plot(x_test,f_pred,label="Model",color='red',linestyle='dashed')
plt.xlabel("x")
plt.ylabel("f(x)")
plt.title("Model and True Function for f(x)=sin(x)")
plt.legend()
plt.show()

plt.plot(x_test,g_true,label="True",color='blue')
plt.plot(x_test,g_pred,label="Model",color='red',linestyle='dashed')
plt.xlabel("x")
plt.ylabel("g(x)")
plt.title("Model and True Function for g(x)=(1+sin(x))/2")
plt.legend()
plt.show()
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
313/313 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
```



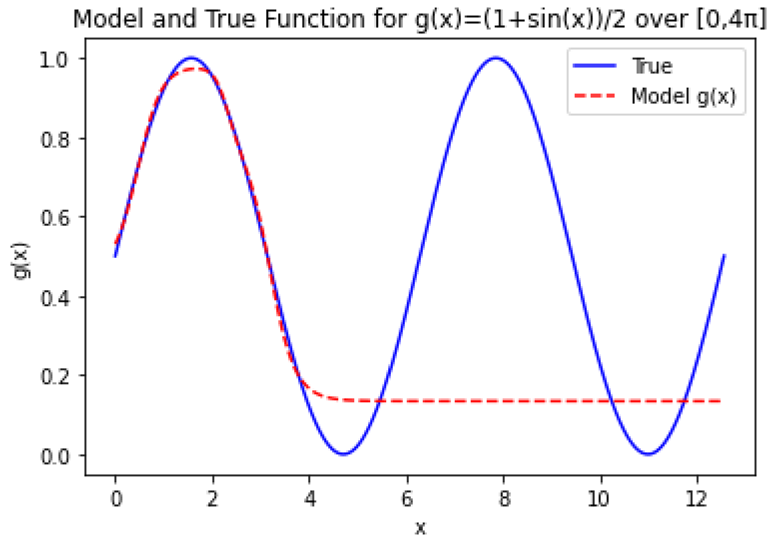


The model for $g(x) = \frac{1+\sin(x)}{2}$ performs better than $f(x) = \sin(x)$ because $g(x)$ exactly aligns with the sigmoid activation's output range ($[0, 1]$), while $f(x)$, ranging from $[-1, 1]$, is harder for the sigmoid to approximate. Scaling or using tanh activations could improve $f(x)$'s performance.

e

In [12]:
```
x_test_ext=np.linspace(0,4*np.pi,1000).reshape(-1,1)

g_true_ext=(1 + np.sin(x_test_ext)) / 2

g_pred_ext=g_model.predict(x_test_ext)


plt.plot(x_test_ext,g_true_ext,label="True",color='blue')
plt.plot(x_test_ext,g_pred_ext,label="Model g(x)",color='red',linestyle='dashed')
plt.xlabel("x")
plt.ylabel("g(x)")
plt.title("Model and True Function for g(x)=(1+sin(x))/2 over [0,4π]")
plt.legend()
plt.show()
```

```
32/32 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```



The model's performance declines significantly outside the training range $x \in [0, 2\pi]$.

Explanations:

1. **Training Data Used**:
   The model was trained only on $x \in [0, 2\pi]$, so it has no knowledge of the function's behavior beyond this interval. Neural networks can learn well within the training range but can struggle to find periodic patterns accurately.

2. **Expressivity of the Network**:
   The network isn't able to fully capture the periodic nature of $g(x)$ over extended intervals with only three hidden neurons, since each neuron in the hidden layer can represent a simple, smooth feature, but periodic functions require multiple features to approximate the oscillations. Additionally, smooth and bounded sigmoid activations, further limit the ability to represent high frequency oscillations.

The model's performance declines significantly outside the training range $x \in [0, 2\pi]$.

Explanations:

1. **Training Data Used**:
   The model was only trained on $x \in [0, 2\pi]$, so it has no knowledge of the function's behavior beyond this interval. Neural networks can learn periodic patterns well if given sufficient training data, but here the limited range of training data prevents the network from modelling the periodicity of $g(x)$.

2. **Expressivity of the Network**:
   The network isn't able to fully capture the periodic nature of $g(x)$ over extended intervals with only three hidden neurons. Each neuron in the hidden layer represents a simple, smooth feature, but periodic functions require the network to combine multiple features to approximate oscillations. Additionally, smooth and bounded sigmoid activations further limit the ability to represent high frequency oscillations effectively.