In [14]:
```
import keras
from keras import layers
# This is the size of our encoded representations
encoding_dim = 32 # 32 floats -> compression of factor 24.5, assuming the
# This is our input image
input_img = keras.Input(shape=(784,))
# "encoded" is the encoded representation of the input
encoded = layers.Dense(encoding_dim, activation='relu')(input_img)
# "decoded" is the lossy reconstruction of the input
decoded = layers.Dense(784, activation='sigmoid')(encoded)
# This model maps an input to its reconstruction
autoencoder = keras.Model(input_img, decoded)
```

In [15]:
```
encoder = keras.Model(input_img, encoded)
```

In [16]:
```
# This is our encoded (32-dimensional) input
encoded_input = keras.Input(shape=(encoding_dim,))
# Retrieve the last layer of the autoencoder model
decoder_layer = autoencoder.layers[-1]
# Create the decoder model
decoder = keras.Model(encoded_input, decoder_layer(encoded_input))
```

In [17]:
```
autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
```

In [18]:
```
from keras.datasets import mnist
import numpy as np
(x_train, _), (x_test, _) = mnist.load_data()
```

In [19]:
```
x_train = x_train.astype('float32') / 255.
x_test = x_test.astype('float32') / 255.
x_train = x_train.reshape((len(x_train), np.prod(x_train.shape[1:])))
x_test = x_test.reshape((len(x_test), np.prod(x_test.shape[1:])))
print(x_train.shape)
print(x_test.shape)
```

```
(60000, 784)
(10000, 784)
```

In [20]:
```
autoencoder.fit(x_train, x_train,
 epochs=50,
 batch_size=256,
 shuffle=True,
 validation_data=(x_test, x_test))
```

```
Epoch 1/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 9ms/step - loss: 0.3861 - val_loss: 0.1938
Epoch 2/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.1835 - val_loss: 0.1558
Epoch 3/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.1512 - val_loss: 0.1348
Epoch 4/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 12ms/step - loss: 0.1327 - val_loss: 0.1221
Epoch 5/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.1214 - val_loss: 0.1137
Epoch 6/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.1137 - val_loss: 0.1077
Epoch 7/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.1080 - val_loss: 0.1031
Epoch 8/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.1035 - val_loss: 0.0998
Epoch 9/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 12ms/step - loss: 0.1002 - val_loss: 0.0975
Epoch 10/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0982 - val_loss: 0.0958
Epoch 11/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0968 - val_loss: 0.0947
Epoch 12/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 12ms/step - loss: 0.0957 - val_loss: 0.0940
Epoch 13/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.0950 - val_loss: 0.0935
Epoch 14/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0947 - val_loss: 0.0932
Epoch 15/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 11ms/step - loss: 0.0945 - val_loss: 0.0929
Epoch 16/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0940 - val_loss: 0.0927
Epoch 17/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0939 - val_loss: 0.0926
Epoch 18/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.0939 - val_loss: 0.0926
Epoch 19/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.0936 - val_loss: 0.0924
Epoch 20/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0937 - val_loss: 0.0923
Epoch 21/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0936 - val_loss: 0.0923
Epoch 22/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0931 - val_loss: 0.0922
Epoch 23/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0932 - val_loss: 0.0921
Epoch 24/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.0932 - val_loss: 0.0920
Epoch 25/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0932 - val_loss: 0.0921
Epoch 26/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 12ms/step - loss: 0.0931 - val_loss: 0.0920
Epoch 27/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.0932 - val_loss: 0.0920
Epoch 28/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0932 - val_loss: 0.0919
```

```
Epoch 29/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0931 - val_loss: 0.0919
Epoch 30/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0930 - val_loss: 0.0919
Epoch 31/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0930 - val_loss: 0.0919
Epoch 32/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 10ms/step - loss: 0.0930 - val_loss: 0.0920
Epoch 33/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 11ms/step - loss: 0.0930 - val_loss: 0.0918
Epoch 34/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.0930 - val_loss: 0.0919
Epoch 35/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.0928 - val_loss: 0.0918
Epoch 36/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0930 - val_loss: 0.0918
Epoch 37/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.0928 - val_loss: 0.0918
Epoch 38/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.0929 - val_loss: 0.0918
Epoch 39/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.0932 - val_loss: 0.0918
Epoch 40/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.0928 - val_loss: 0.0917
Epoch 41/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 11ms/step - loss: 0.0928 - val_loss: 0.0918
Epoch 42/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 5s 9ms/step - loss: 0.0928 - val_loss: 0.0917
Epoch 43/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.0928 - val_loss: 0.0917
Epoch 44/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.0928 - val_loss: 0.0917
Epoch 45/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.0927 - val_loss: 0.0917
Epoch 46/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.0927 - val_loss: 0.0917
Epoch 47/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.0926 - val_loss: 0.0917
Epoch 48/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.0927 - val_loss: 0.0917
Epoch 49/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.0927 - val_loss: 0.0917
Epoch 50/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.0926 - val_loss: 0.0917
```

Out[20]:
```
<keras.src.callbacks.history.History at 0x22345ebc310>
```

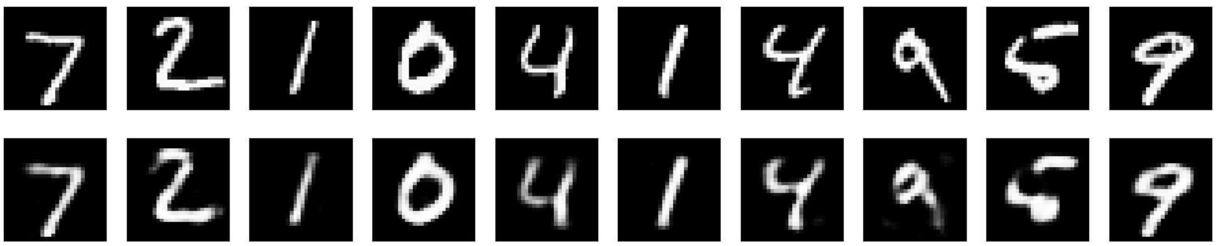
In [21]:
```python
encoded_imgs = encoder.predict(x_test)
decoded_imgs = decoder.predict(encoded_imgs)
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
```

In [23]:
```python
# Display original and reconstructed images
import matplotlib.pyplot as plt
```

In [24]:
```
n = 10  # How many digits we will display
plt.figure(figsize=(20, 4))
for i in range(n):
    # Display original
    ax = plt.subplot(2, n, i + 1)
    plt.imshow(x_test[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)

    # Display reconstruction
    ax = plt.subplot(2, n, i + 1 + n)
    plt.imshow(decoded_imgs[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
```



In [25]:
```
from keras import regularizers

encoding_dim = 32
```

In [26]:
```
input_img = keras.Input(shape=(784,))
# Add a Dense layer with a L1 activity regularizer
encoded = layers.Dense(encoding_dim, activation='relu',
                activity_regularizer=regularizers.l1(10e-5))(input_img)
decoded = layers.Dense(784, activation='sigmoid')(encoded)

autoencoder = keras.Model(input_img, decoded)
```

In [28]:
```
autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
```

In [29]:
```
autoencoder.fit(x_train, x_train,
 epochs=50,
 batch_size=256,
 shuffle=True,
 validation_data=(x_test, x_test))
```

```
Epoch 1/50
235/235  4s 7ms/step - loss: 0.6803 - val_loss: 0.6155
Epoch 2/50
235/235  2s 7ms/step - loss: 0.5987 - val_loss: 0.5535
Epoch 3/50
235/235  2s 9ms/step - loss: 0.5398 - val_loss: 0.5038
Epoch 4/50
235/235  2s 9ms/step - loss: 0.4928 - val_loss: 0.4638
Epoch 5/50
235/235  2s 9ms/step - loss: 0.4548 - val_loss: 0.4314
Epoch 6/50
235/235  3s 9ms/step - loss: 0.4241 - val_loss: 0.4050
Epoch 7/50
235/235  2s 8ms/step - loss: 0.3986 - val_loss: 0.3834
Epoch 8/50
235/235  2s 8ms/step - loss: 0.3784 - val_loss: 0.3656
Epoch 9/50
235/235  2s 8ms/step - loss: 0.3611 - val_loss: 0.3508
Epoch 10/50
235/235  2s 9ms/step - loss: 0.3472 - val_loss: 0.3385
Epoch 11/50
235/235  2s 9ms/step - loss: 0.3355 - val_loss: 0.3281
Epoch 12/50
235/235  2s 9ms/step - loss: 0.3257 - val_loss: 0.3194
Epoch 13/50
235/235  2s 8ms/step - loss: 0.3173 - val_loss: 0.3120
Epoch 14/50
235/235  2s 9ms/step - loss: 0.3104 - val_loss: 0.3056
Epoch 15/50
235/235  2s 9ms/step - loss: 0.3043 - val_loss: 0.3002
Epoch 16/50
235/235  2s 8ms/step - loss: 0.2988 - val_loss: 0.2955
Epoch 17/50
235/235  2s 8ms/step - loss: 0.2949 - val_loss: 0.2915
Epoch 18/50
235/235  2s 9ms/step - loss: 0.2905 - val_loss: 0.2880
Epoch 19/50
235/235  2s 9ms/step - loss: 0.2873 - val_loss: 0.2850
Epoch 20/50
235/235  2s 9ms/step - loss: 0.2843 - val_loss: 0.2823
Epoch 21/50
235/235  2s 9ms/step - loss: 0.2819 - val_loss: 0.2800
Epoch 22/50
235/235  2s 9ms/step - loss: 0.2798 - val_loss: 0.2780
Epoch 23/50
235/235  2s 8ms/step - loss: 0.2776 - val_loss: 0.2762
Epoch 24/50
235/235  2s 8ms/step - loss: 0.2761 - val_loss: 0.2747
Epoch 25/50
235/235  2s 9ms/step - loss: 0.2743 - val_loss: 0.2733
Epoch 26/50
235/235  2s 9ms/step - loss: 0.2734 - val_loss: 0.2721
Epoch 27/50
235/235  2s 9ms/step - loss: 0.2720 - val_loss: 0.2710
Epoch 28/50
235/235  2s 9ms/step - loss: 0.2710 - val_loss: 0.2700
```

```
Epoch 29/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.2701 - val_loss: 0.2692
Epoch 30/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2694 - val_loss: 0.2685
Epoch 31/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.2682 - val_loss: 0.2678
Epoch 32/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.2676 - val_loss: 0.2672
Epoch 33/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2675 - val_loss: 0.2667
Epoch 34/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2670 - val_loss: 0.2663
Epoch 35/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.2668 - val_loss: 0.2658
Epoch 36/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2658 - val_loss: 0.2655
Epoch 37/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2656 - val_loss: 0.2652
Epoch 38/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.2650 - val_loss: 0.2649
Epoch 39/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.2649 - val_loss: 0.2646
Epoch 40/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.2648 - val_loss: 0.2644
Epoch 41/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2642 - val_loss: 0.2642
Epoch 42/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2642 - val_loss: 0.2640
Epoch 43/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - loss: 0.2647 - val_loss: 0.2638
Epoch 44/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2637 - val_loss: 0.2637
Epoch 45/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.2641 - val_loss: 0.2636
Epoch 46/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.2641 - val_loss: 0.2635
Epoch 47/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - loss: 0.2635 - val_loss: 0.2634
Epoch 48/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2632 - val_loss: 0.2633
Epoch 49/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2639 - val_loss: 0.2632
Epoch 50/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.2632 - val_loss: 0.2631
```

Out[29]: `<keras.src.callbacks.history.History at 0x2232f28c310>`

In [33]:

```
decoded_imgs = autoencoder.predict(x_test)

# Set the number of images you want to display
n = 10
plt.figure(figsize=(20, 4))

for i in range(n):
    # Display original images
    ax = plt.subplot(2, n, i + 1)
```

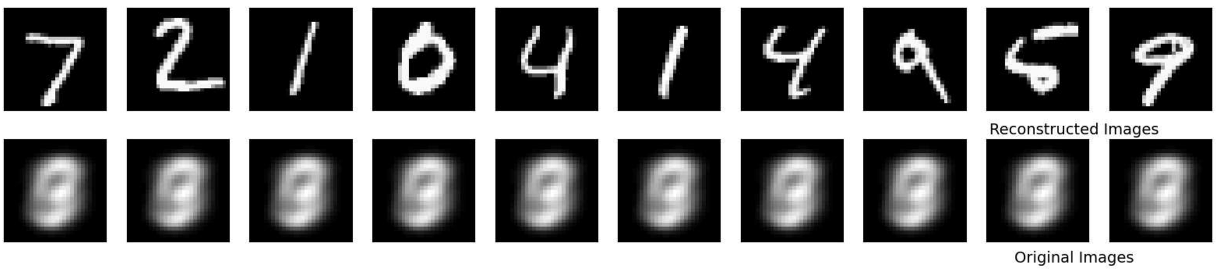
```
    plt.imshow(x_test[i].reshape(28, 28))  # Assuming x_test is reshaped to (28, 28
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)

    # Display reconstructed images
    ax = plt.subplot(2, n, i + 1 + n)
    plt.imshow(decoded_imgs[i].reshape(28, 28))  # Reshape decoded images to (28, 2
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)

# Adding labels for the two rows
plt.text(-10, 30, 'Original Images', fontsize=14, ha='center', va='top')
plt.text(-10, -5, 'Reconstructed Images', fontsize=14, ha='center', va='top')

plt.show()
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
```




```
In [ ]:
```