```
import tensorflow as tf
from tensorflow.keras.layers import Dense, Flatten, Embedding, LSTM
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.datasets import mnist
import numpy as np
import matplotlib.pyplot as plt
# Load MNIST dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Normalize the images
x_train = x_train / 255.0
x_test = x_test / 255.0

# Create dummy text descriptions
texts = ["This is a digit" for _ in range(len(x_train))]  # Simplified for example
# Tokenize text descriptions
tokenizer = Tokenizer(num_words=1000)
tokenizer.fit_on_texts(texts)
sequences = tokenizer.texts_to_sequences(texts)
padded_sequences = pad_sequences(sequences, maxlen=10)
# Define a simple text-to-image model
def build_model():
    model = Sequential([
        Embedding(input_dim=1000, output_dim=64, input_length=10),
        LSTM(128),
        Dense(256, activation='relu'),
        Dense(784, activation='sigmoid'),  # Output layer to match MNIST images
        tf.keras.layers.Reshape((28, 28))
    ])
```

```
    model.compile(optimizer='adam', loss='binary_crossentropy')
    return model

model = build_model()
model.summary()
# Train the model (using dummy data)
model.fit(padded_sequences, x_train, epochs=5, batch_size=32)
# Generate an image from a new text description
new_text = ["This is a digit"]
new_sequence = tokenizer.texts_to_sequences(new_text)
new_padded_sequence = pad_sequences(new_sequence, maxlen=10
# Generate an image
generated_image = model.predict(new_padded_sequence)[0]

# Display the generated image
plt.imshow(generated_image, cmap='gray')
plt.axis('off')
plt.title('Generated Image')
plt.show()
```