```
import tensorflow as tf
from tensorflow import keras
import numpy as np

# Load MNIST dataset
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()

# Preprocess data
x_train = x_train.reshape((-1, 28 * 28))
x_train = x_train.astype('float32') / 255
x_test = x_test.reshape((-1, 28 * 28))
x_test = x_test.astype('float32') / 255
y_train = keras.utils.to_categorical(y_train)
y_test = keras.utils.to_categorical(y_test)

# Define model architecture
model = keras.Sequential([
    keras.layers.Dense(512, activation='relu', input_shape=(28 * 28,)),
    keras.layers.Dropout(0.2),
    keras.layers.Dense(10, activation='softmax')
])

# Compile model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Train model
model.fit(x_train, y_train, epochs=50, batch_size=128, validation_data=(x_test, y_test))

# Save model
# model.save('my_model.h5')
```

```
    Epoch 1/50
    469/469 [==============================] - 4s 5ms/step - loss: 0.2891 - accuracy: 0.9169 - val_
    Epoch 2/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.1223 - accuracy: 0.9650 - val_
    Epoch 3/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0844 - accuracy: 0.9750 - val_
    Epoch 4/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0640 - accuracy: 0.9807 - val_
    Epoch 5/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0501 - accuracy: 0.9851 - val_
    Epoch 6/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0415 - accuracy: 0.9879 - val_
    Epoch 7/50
    469/469 [==============================] - 2s 5ms/step - loss: 0.0337 - accuracy: 0.9899 - val_
    Epoch 8/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0288 - accuracy: 0.9909 - val_
    Epoch 9/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0240 - accuracy: 0.9920 - val_
    Epoch 10/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0205 - accuracy: 0.9935 - val_
    Epoch 11/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0185 - accuracy: 0.9939 - val_
    Epoch 12/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0168 - accuracy: 0.9946 - val_
    Epoch 13/50
    469/469 [==============================] - 2s 5ms/step - loss: 0.0145 - accuracy: 0.9954 - val_
    Epoch 14/50
    469/469 [==============================] - 2s 5ms/step - loss: 0.0135 - accuracy: 0.9955 - val_
    Epoch 15/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0126 - accuracy: 0.9960 - val_
    Epoch 16/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0101 - accuracy: 0.9971 - val_
    Epoch 17/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0110 - accuracy: 0.9961 - val_
    Epoch 18/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0109 - accuracy: 0.9962 - val_
    Epoch 19/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0080 - accuracy: 0.9974 - val_
    Epoch 20/50
    469/469 [==============================] - 2s 5ms/step - loss: 0.0098 - accuracy: 0.9967 - val_
    Epoch 21/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0077 - accuracy: 0.9974 - val_
    Epoch 22/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0079 - accuracy: 0.9976 - val_
    Epoch 23/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0064 - accuracy: 0.9981 - val_
    Epoch 24/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0068 - accuracy: 0.9977 - val_
    Epoch 25/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0066 - accuracy: 0.9978 - val_
    Epoch 26/50
    469/469 [==============================] - 2s 5ms/step - loss: 0.0073 - accuracy: 0.9974 - val_
    Epoch 27/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0070 - accuracy: 0.9977 - val_
    Epoch 28/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0072 - accuracy: 0.9974 - val_
    Epoch 29/50
    469/469 [==============================] - 2s 4ms/step - loss: 0.0067 - accuracy: 0.9976 - val_
```

```
from tensorflow import keras
import numpy as np
from PIL import Image
from google.colab import files

# Upload image
uploaded_file = files.upload()

# Load image
image = Image.open(list(uploaded_file.keys())[0])

# Load trained model
#model = keras.models.load_model('my_model.h5')

# Convert to grayscale and resize
image = image.convert('L').resize((28, 28))

# Convert to numpy array
image_data = np.array(image)

# Preprocess image data
image_data = image_data.reshape((-1, 28 * 28))
image_data = image_data.astype('float32') / 255

# Make prediction
prediction = model.predict(image_data)[0]
```

```
    Choose Files  No file chosen      Upload widget is only available when the cell has been executed in the current brov
    Saving Screenshot_20230304-235815_WhatsApp.jpg to Screenshot_20230304-235815_WhatsApp.jpg
    1/1 [==============================] - 0s 21ms/step
    5
```

```
# Get & Print predicted class label
print("Uploaded image matches to", np.argmax(prediction))
```

```
    Uploaded image matches to 5
```