```python
## Image fitting demonstration using the SIREN model
import tensorflow as tf
from PIL import Image
from image import ImageData
from siren import Siren
from tqdm import trange
import matplotlib.pyplot as plt

def main():
    rng = tf.random.get_global_generator()
    rng.reset_from_seed(0x43966E87BD57227011B5B03B58785EC1)
    img = Image.open("Testcard_F.jpg")
    side_len = 256
    data = ImageData(side_len)
    groundTruthCoords, groundTruthPixels = data(img)

    groundTruthCoords = tf.cast(groundTruthCoords, dtype=tf.float32)

    num_iters = 250
    num_inputs = 2
    num_outputs = 3
    hidden_layer_width = 256
    num_hidden_layers = 6

    model = Siren(num_inputs, hidden_layer_width, num_outputs, num_hidden_layers)

    bar = trange(num_iters)
    optimizer = tf.optimizers.Adam(
        learning_rate=0.0001, beta_1=0.9, beta_2=0.999
    )  # keeping the default parameters

    for i in bar:
        with tf.GradientTape() as tape:
            output, coords = model(groundTruthCoords)
            # breakpoint()
            loss = tf.math.reduce_mean(0.5 * (output - groundTruthPixels) ** 2)

        grads = tape.gradient(loss, model.trainable_variables)
        # should by model outputs wrt the coordinates
        optimizer.apply_gradients(zip(grads, model.trainable_variables))
        bar.set_description(f"Loss @ {i} => {loss.numpy():0.3f}")

    ## AFTER THIS POINT I HAVE MY TRAINED MODEL
    grid = data.get_mgrid(256)
    reflected_grid = tf.reverse(grid, axis=[1])
    ref_grid = tf.cast(reflected_grid, dtype=tf.float32)
    upimg, coords = model(ref_grid)

    fig, axes = plt.subplots(2, 2, figsize=(12, 12))

    axes[0, 0].imshow(img)
    axes[0, 0].set_title("Original Image", fontweight="bold")

    axes[0, 1].imshow(tf.reshape(groundTruthPixels, [side_len, side_len, 3]))
    axes[0, 1].set_title("Ground Truth Image", fontweight="bold")

    axes[1, 0].imshow(tf.reshape(output, [side_len, side_len, 3]))
    axes[1, 0].set_title("SIREN Model", fontweight="bold")

    axes[1, 1].imshow(tf.reshape(upimg, [side_len, side_len, 3]))
    axes[1, 1].set_title("Flipped Image using SIREN", fontweight="bold")

    plt.tight_layout()
    plt.show()

if __name__ == "__main__":
    main()
```