```python
import ipywidgets as widgets
from PIL import Image
from IPython.display import display

# Creating widgets
text = widgets.Text(description='Enter image path:')
button = widgets.Button(description='Detect Emotion')
output = widgets.Output()

def preprocess_image(image_path):
    # Loading the image using PIL (Python Imaging Library)
    image = Image.open(image_path)
    #Converting input image to grayscale as our model accepts only
grayscale input
    image = image.convert('L')
    # Resizing the image to match the input size of your CNN model
    image = image.resize((48,48))
    # Converting the image to a NumPy array
    image_array = np.array(image) / 255.0  # Normalize pixel values to [0,
1]
    # Adding an extra dimension to the array to represent the batch size=1
    image_array = np.expand_dims(image_array, axis=0)
    return image_array

def detect_emotion(image_path):
    # Preprocessing the input image
    image_array = preprocess_image(image_path)
    # Using the CNN model to predict the emotion
    predicted_emotion = np.argmax(model.predict(image_array), axis=-1)
    # Mapping the predicted emotion index to the corresponding emotion
label
    emotion_labels = {0: 'Happy', 1: 'Neutral', 2: 'Sad', 3: 'Surprise'}
    #Mapping the predicted emotion to its label
    predicted_emotion_label = emotion_labels[predicted_emotion[0]]
    #returning the predicted_emotion_label
    return predicted_emotion_label

# Defining callback function for button click
def on_button_click(b):
    with output:
        # Performing emotion detection using the entered image path
        image_path = text.value
        # Calling the detect_emotion function to predict the emotion
        result=detect_emotion(image_path)
        # Printing the result
        print(f'Emotion detected in {image_path}')
        print(result)

# Registering callback function with the button
button.on_click(on_button_click)

# Displaying the widgets
display(text)
display(button)
display(output)
```