In [ ]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import PySimpleGUI as sg
import io
from PIL import Image

# Load CSV files for nominal and non-nominal data
nominal_data = pd.read_csv("D:/Isro1/NOMINAL.csv", index_col=0, parse_dates=True)
non_nominal_data = pd.read_csv("D:/Isro1/AE01.csv", index_col=0, parse_dates=True)

# Function to plot data and detect anomalies
def plot_selected_parameter(selected_parameter, window):
    plt.figure(figsize=(10, 6))

    # Plot nominal data
    plt.plot(nominal_data.index, nominal_data[selected_parameter], label='Nominal Data')

    # Calculate the difference between non-nominal and nominal data
    difference = non_nominal_data[selected_parameter] - nominal_data[selected_parameter]

    # Define a threshold for anomaly detection
    threshold = 0.1  # Adjust as needed

    # Detect anomalies
    anomalies = non_nominal_data[np.abs(difference) > threshold]

    # Plot non-nominal data
    plt.plot(non_nominal_data.index, non_nominal_data[selected_parameter], label='Non-Nominal Data')

    # Plot anomalies
    plt.scatter(anomalies.index, anomalies[selected_parameter], color='red', label='Anomalies')

    plt.title('Sensor Data Plot: {}'.format(selected_parameter))
    plt.xlabel('Time')
    plt.ylabel('Parameter Value')
    plt.legend()

    # Save plot as an image
    img_data = io.BytesIO()
    plt.savefig(img_data, format='png')
    img_data.seek(0)
    window['-PLOT-'].update(data=img_data.read())
    plt.close()

    # Print the number of values deviating beyond the threshold
    num_anomalies = len(anomalies)
    if num_anomalies == 0:
        window['-OUTPUT-'].update("No anomalies detected for parameter: {}".format(selected_parameter))
    else:
        output_text = "Number of values deviating beyond the threshold: {}\n".format(num_anomalies)

        # Sort the anomalies by deviation
        anomalies_sorted = anomalies.copy()
        anomalies_sorted['Deviation'] = np.abs(difference[anomalies.index])
        anomalies_sorted = anomalies_sorted.sort_values(by='Deviation', ascending=False)

        # Print the details of the time stamps where anomalies were detected
        output_text += "\nDetails of anomaly timestamps:\n"
        for timestamp, row in anomalies_sorted.iterrows():
            output_text += "Timestamp: {}, Actual Value: {}, Deviation: {} points\n".format(timestamp, row[selected_parameter], r

        window['-OUTPUT-'].update(output_text)

# Define the layout for PySimpleGUI
layout = [
    [sg.Text("Select Parameter:")],
    [sg.Combo(values=list(nominal_data.columns), key='-PARAMETER-', size=(30, 1), enable_events=True)],
    [sg.Image(key='-PLOT-')],
    [sg.Output(size=(60, 10), key='-OUTPUT-')]
]

# Create the PySimpleGUI window
window = sg.Window("Anomaly Detection System", layout, finalize=True)

# Event loop
while True:
    event, values = window.read()
    if event == sg.WINDOW_CLOSED:
        break
    elif event == '-PARAMETER-':
        selected_param = values['-PARAMETER-']
        if selected_param:
            plot_selected_parameter(selected_param, window)

window.close()
```