Matplotlib Chart Arduino Data

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=480, height=320, initial-scale=1.0">
    <title>Absorbance</title>
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap-icons/font/bootstrap-icons.css"
rel="stylesheet">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css"
rel="stylesheet">
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
    <style>
        body{
            background-color: white;
        }
    </style>

</head>
<body>

<!-- Status Bar -->
{% include 'statSegments/statusBar.html' %}

{#navigation bar#}
{% include 'statSegments/navigation.html' %}


<!-- Main content area -->
<div class="container mt-4 container-custom">

</div>


<!-- Bootstrap JS Bundle with Popper -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>

</body>
</html>
```

i need to add matplotlib chart to display arduino light sensor data that plot include 2048 pixcels data save

To include a Matplotlib chart in your HTML file that displays Arduino light sensor data, you can use Flask and Matplotlib together to render the chart as an image and then embed it in the HTML. Here's a step-by-step guide:

1. **Create a Flask Route for the Chart Image:** This route will generate a Matplotlib chart and return it as a PNG image.
2. **Update the HTML to Display the Chart:** Use an `` `<img>` `` tag in your HTML to display the chart generated by the Flask route.
3. **Modify the Flask Application:** Make sure your Flask app reads the Arduino sensor data, processes it, and plots it using Matplotlib.

1. Flask Route for the Chart Image

In your `` `app.py` `` or main Flask file, add a new route to generate the Matplotlib chart:

```python
from flask import Flask, render_template, Response
import matplotlib.pyplot as plt
import io
import numpy as np

app = Flask(__name__)

@app.route('/plot.png')
def plot_png():
    # Example data: Replace this with actual sensor data
    x = np.linspace(0, 2048, 2048)
    y = np.random.random(2048)  # Replace with actual light sensor data

    # Create the plot
    fig, ax = plt.subplots()
    ax.plot(x, y, label='Sensor Data')
    ax.set_xlabel('Pixel')
    ax.set_ylabel('Value')
    ax.set_title('Arduino Light Sensor Data')
    ax.legend()

    # Save it to a BytesIO object
    output = io.BytesIO()
    plt.savefig(output, format='png')
    plt.close(fig)
    output.seek(0)

    return Response(output, mimetype='image/png')

@app.route('/')
def index():
    return render_template('index.html')

if __name__ == '__main__':
    app.run(debug=True)
```

2. Update the HTML File

In your HTML file (`` `index.html` ``), update the main content area to include the image that will display the Matplotlib chart:

html

```html
<!-- Main content area -->
<div class="container mt-4 container-custom">
    <h2>Arduino Light Sensor Data Plot</h2>
    <img src="{{ url_for('plot_png') }}" alt="Sensor Data Plot" class="img-fluid">
</div>
```

This code includes an `` `<img>` `` tag that references the Flask route `` `plot_png` `` to display the chart.

3. Connecting with Arduino

For connecting with Arduino to get actual sensor data, you'll need to read data from the serial port and use it in the `` `plot_png` `` route. Here's an example of how to modify the route to use actual sensor data:

python

```python
import serial

# Replace with your actual serial port and baud rate
SERIAL_PORT = '/dev/ttyACM0'  # Change to your Arduino port
BAUD_RATE = 9600

# Initialize the serial connection
ser = serial.Serial(SERIAL_PORT, BAUD_RATE)

@app.route('/plot.png')
def plot_png():
    # Read the data from the Arduino
    data = []
    for _ in range(2048):
        line = ser.readline().decode('utf-8').strip()
        data.append(float(line))

    # Create the plot
    x = np.linspace(0, 2048, 2048)
    y = data

    fig, ax = plt.subplots()
    ax.plot(x, y, label='Sensor Data')
    ax.set_xlabel('Pixel')
    ax.set_ylabel('Value')
    ax.set_title('Arduino Light Sensor Data')
    ax.legend()

    output = io.BytesIO()
    plt.savefig(output, format='png')
    plt.close(fig)
    output.seek(0)

    return Response(output, mimetype='image/png')
```

This code snippet reads 2048 data points from the Arduino serial port and plots them.

4. Important Considerations

- **Serial Communication:** Ensure that your Arduino is connected to the correct serial port and that it is sending data in a readable format.
- **Error Handling:** Add error handling to manage issues like serial disconnection.

- **Flask Debug Mode:** In a production environment, disable Flask's debug mode.

By following these steps, you should be able to integrate a Matplotlib chart into your Flask app that visualizes your Arduino light sensor data.