jssss

1 message

pp Pettina <pppettina@gmail.com> Sun, May 4, 2025 at 6:23 AM
To: pp Pettina <pppettina@gmail.com>

```
document.addEventListener('DOMContentLoaded', function() {
    const AUTH_KEY = 'env_monitor_auth';

    function checkAuth() {
      const storedAuth = localStorage.getItem(AUTH_KEY);
      if (!storedAuth) {
        const username = prompt('Enter username:');
        const password = prompt('Enter password:');
        if (username === 'admin' && password === 'admin123') {
          localStorage.setItem(AUTH_KEY, 'authenticated');
        } else {
          alert('Invalid credentials!');
          window.location.reload();
        }
      }
    }

    checkAuth();

    async function fetchData() {
      try {
        const response = await fetch('https://toxyu2j8cc.execute-api.us-east-1.amazonaws.com/dev/data');
        const apiResponse = await response.json();
        const items = JSON.parse(apiResponse.body);
        console.log("API return the data:", items);
        return items;
      } catch (error) {
        console.error("faild to fetch the data:", error);
        return [];
      }
    }


    const ALARM_TEMP = 30;
    let isAlarming = false;

    function updateAlarmIndicator(temp) {
      const statusEl = document.getElementById('status');
      if (temp > ALARM_TEMP) {
        statusEl.className = 'status-panel alert';
        statusEl.innerHTML = `HIGH TEMP ALARM: ${temp}°C`;
        isAlarming = true;
      } else if (isAlarming) {
        statusEl.className = 'status-panel normal';
        statusEl.innerHTML = 'NORMAL CONDITION';
        isAlarming = false;
      }
    }


    const chart = new ApexCharts(document.querySelector("#chart"), {
      chart: {
        type: 'line',
        height: 350,
        events: {
          mounted: function(ctx, config) {
            ctx.addAnnotation({
```

```
                y: ALARM_TEMP,
                y2: ALARM_TEMP + 5,
                borderColor: '#ff0000',
                fillColor: '#ff000033',
                label: {
                    text: 'Danger Zone',
                    style: {
                        color: '#fff',
                        background: '#ff0000'
                    }
                }
            })
        }
    }
  },
  series: [
    {
      name: 'Temperature (&deg;C)',
      data: [],
      color: '#FF4560'
    },
    {
      name: 'Humidity (%)',
      data: [],
      color: '#00E396'
    }
  ],
  xaxis: {
    type: 'datetime',
    labels: {
      datetimeUTC: false
    }
  },
  yaxis: [
    {
      title: { text: 'Temperature (&deg;C)' },
      min: 0,
      max: 50
    },
    {
      opposite: true,
      title: { text: 'Humidity (%)' },
      min: 0,
      max: 100
    }
  ],
  markers: {
    size: 5,
    colors: ['#FF4560'],
    strokeColors: '#fff'
  }
});

chart.render();


async function updateChartData() {
  const items = await fetchData();
  if (items.length === 0) return;


  const latestData = items[0];
  updateAlarmIndicator(latestData.temperature);


  const tempColor = latestData.temperature > ALARM_TEMP ? '#FF0000' : '#FF4560';

  chart.updateSeries([{
    name: 'Temperature (&deg;C)',
```

```
            data: items.map(d => [d.timestamp * 1000, d.temperature]),
            color: tempColor
        }, {
            name: 'Humidity (%)',
            data: items.map(d => [d.timestamp * 1000, d.humidity])
        }]);


        if (latestData.temperature > ALARM_TEMP) {
            chart.addPointAnnotation({
                x: latestData.timestamp * 1000,
                y: latestData.temperature,
                label: {
                    text: 'ALERT',
                    style: {
                        background: '#FF0000',
                        color: '#FFF',
                        fontSize: '12px'
                    }
                }
            });
        }
    }


    updateChartData();


    setInterval(updateChartData, 10000);
});
```