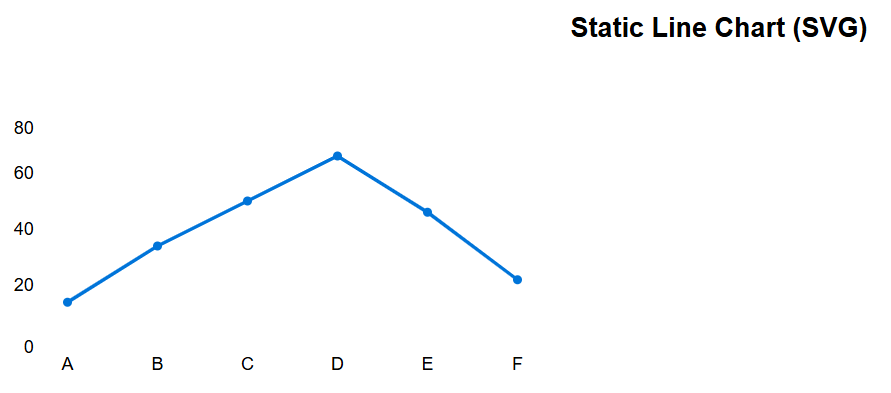
**Q1: Can you create a simple HTML page with a static line chart using only HTML and inline SVG elements?**



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Simple SVG Line Chart</title>

<style>

body { font-family: Arial, sans-serif; }

.chart-title { text-align: center; margin-top: 20px; }

</style>

</head>

<body>

<h2 class="chart-title">Static Line Chart (SVG)</h2>

<svg width="500" height="300" style="display: block; margin: 0 auto; background: #f9f9f9; border: 1px solid #ccc;">

<!-- Axes -->

<line x1="40" y1="250" x2="460" y2="250" stroke="#333" stroke-width="2"/>

<line x1="40" y1="250" x2="40" y2="50" stroke="#333" stroke-width="2"/>

<!-- Y-axis labels -->

<text x="30" y="255" font-size="12" text-anchor="end">0</text>

<text x="30" y="200" font-size="12" text-anchor="end">20</text>

<text x="30" y="150" font-size="12" text-anchor="end">40</text>

<text x="30" y="100" font-size="12" text-anchor="end">60</text>

<text x="30" y="60" font-size="12" text-anchor="end">80</text>

<!-- X-axis labels -->

<text x="60" y="270" font-size="12" text-anchor="middle">A</text>

<text x="140" y="270" font-size="12" text-anchor="middle">B</text>

<text x="220" y="270" font-size="12" text-anchor="middle">C</text>

<text x="300" y="270" font-size="12" text-anchor="middle">D</text>

<text x="380" y="270" font-size="12" text-anchor="middle">E</text>

<text x="460" y="270" font-size="12" text-anchor="middle">F</text>

<!-- Line chart (example data points) -->

<polyline

fill="none"

stroke="#0074d9"

stroke-width="3"

points="

60,210

140,160

220,120

300,80

380,130

460,190

"

/>

<!-- Data points -->

<circle cx="60" cy="210" r="4" fill="#0074d9"/>

<circle cx="140" cy="160" r="4" fill="#0074d9"/>

<circle cx="220" cy="120" r="4" fill="#0074d9"/>

<circle cx="300" cy="80" r="4" fill="#0074d9"/>

<circle cx="380" cy="130" r="4" fill="#0074d9"/>

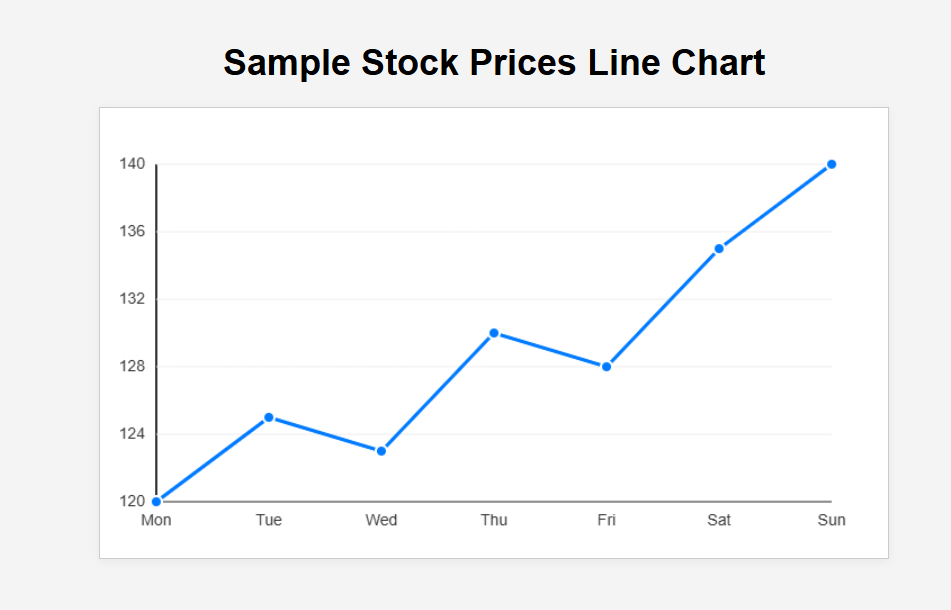
<circle cx="460" cy="190" r="4" fill="#0074d9"/>

</svg>

</body>

</html>

**Q2: Generate an HTML page with a line chart displaying sample stock prices using JavaScript and the HTML <canvas> element.**

****

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Sample Stock Prices Line Chart</title>

<style>

body {

font-family: Arial, sans-serif;

background: #f4f4f4;

display: flex;

flex-direction: column;

align-items: center;

margin: 0;

padding: 40px 0;

}

h1 {

margin-bottom: 20px;

}

canvas {

background: #fff;

border: 1px solid #ccc;

box-shadow: 0 2px 8px rgba(0,0,0,0.05);

}

</style>

</head>

<body>

<h1>Sample Stock Prices Line Chart</h1>

<canvas id="stockChart" width="700" height="400"></canvas>

<script>

// Sample stock price data

const labels = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"];

const prices = [120, 125, 123, 130, 128, 135, 140];

const canvas = document.getElementById('stockChart');

const ctx = canvas.getContext('2d');

// Chart dimensions

const padding = 50;

const chartWidth = canvas.width - 2 \* padding;

const chartHeight = canvas.height - 2 \* padding;

// Find min and max prices

const minPrice = Math.min(...prices);

const maxPrice = Math.max(...prices);

const priceRange = maxPrice - minPrice;

// Draw axes

ctx.beginPath();

ctx.moveTo(padding, padding);

ctx.lineTo(padding, canvas.height - padding);

ctx.lineTo(canvas.width - padding, canvas.height - padding);

ctx.strokeStyle = '#333';

ctx.lineWidth = 2;

ctx.stroke();

// Draw Y-axis labels and grid lines

ctx.font = '14px Arial';

ctx.fillStyle = '#333';

ctx.textAlign = 'right';

ctx.textBaseline = 'middle';

const steps = 5;

for (let i = 0; i <= steps; i++) {

const y = padding + (chartHeight \* i) / steps;

const price = maxPrice - (priceRange \* i) / steps;

ctx.fillText(price.toFixed(0), padding - 10, y);

// Grid line

ctx.beginPath();

ctx.moveTo(padding, y);

ctx.lineTo(canvas.width - padding, y);

ctx.strokeStyle = '#eee';

ctx.lineWidth = 1;

ctx.stroke();

}

// Draw X-axis labels

ctx.textAlign = 'center';

ctx.textBaseline = 'top';

for (let i = 0; i < labels.length; i++) {

const x = padding + (chartWidth \* i) / (labels.length - 1);

ctx.fillText(labels[i], x, canvas.height - padding + 10);

}

// Draw line chart

ctx.beginPath();

for (let i = 0; i < prices.length; i++) {

const x = padding + (chartWidth \* i) / (prices.length - 1);

const y = padding + ((maxPrice - prices[i]) / priceRange) \* chartHeight;

if (i === 0) {

ctx.moveTo(x, y);

} else {

ctx.lineTo(x, y);

}

}

ctx.strokeStyle = '#007bff';

ctx.lineWidth = 3;

ctx.stroke();

// Draw data points

for (let i = 0; i < prices.length; i++) {

const x = padding + (chartWidth \* i) / (prices.length - 1);

const y = padding + ((maxPrice - prices[i]) / priceRange) \* chartHeight;

ctx.beginPath();

ctx.arc(x, y, 5, 0, 2 \* Math.PI);

ctx.fillStyle = '#007bff';

ctx.fill();

ctx.strokeStyle = '#fff';

ctx.lineWidth = 2;

ctx.stroke();

}

</script>

</body>

</html>

**Q3: Build a responsive dashboard layout with a section that displays a customizable line chart for financial data (e.g., stock prices), including a simple form to update the dataset manually.**

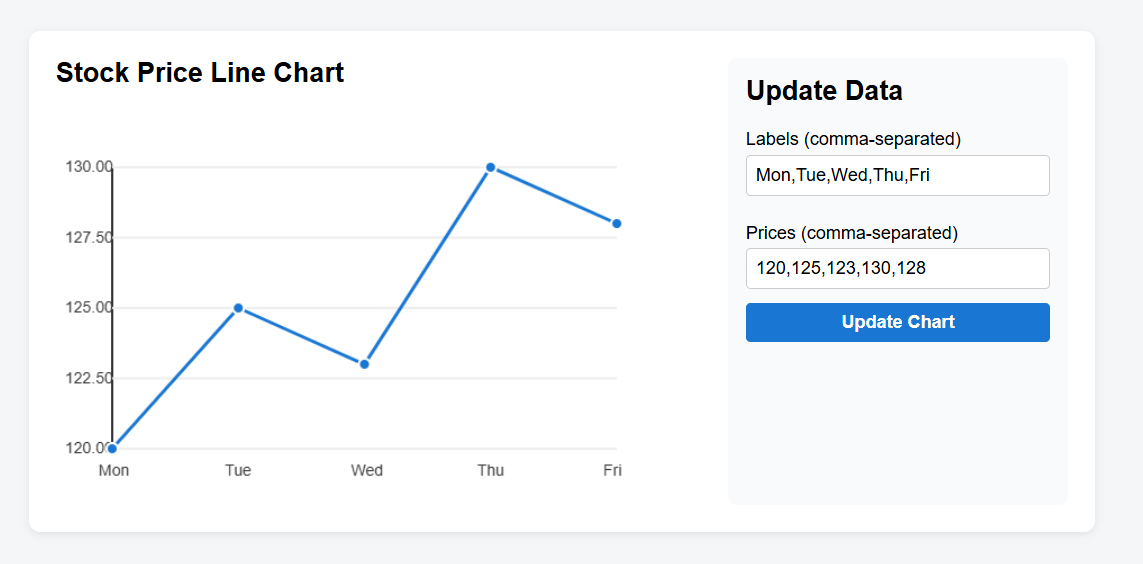
*No se ha mostrado*

npx create-react-app dashboard --template cra-template-pwa

**Consulta adicional:**

Generate an HTML page with a line chart displaying sample stock prices using JavaScript and the HTML <canvas> element.

Build a responsive dashboard layout with a section that displays a customizable line chart for financial data (e.g., stock prices), including a simple form to update the dataset manually.



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Stock Price Dashboard</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

background: #f4f6f8;

}

.dashboard {

max-width: 900px;

margin: 40px auto;

background: #fff;

border-radius: 10px;

box-shadow: 0 2px 8px rgba(0,0,0,0.08);

padding: 24px;

display: flex;

flex-wrap: wrap;

gap: 32px;

}

.chart-section {

flex: 2 1 400px;

min-width: 300px;

}

.form-section {

flex: 1 1 220px;

min-width: 220px;

background: #f9fafb;

border-radius: 8px;

padding: 16px;

box-sizing: border-box;

}

h2 {

margin-top: 0;

}

label {

display: block;

margin: 12px 0 4px;

font-weight: 500;

}

input, textarea, button {

width: 100%;

padding: 8px;

margin-bottom: 12px;

border: 1px solid #ccc;

border-radius: 4px;

font-size: 1em;

}

button {

background: #1976d2;

color: #fff;

border: none;

cursor: pointer;

font-weight: bold;

transition: background 0.2s;

}

button:hover {

background: #125ea7;

}

@media (max-width: 700px) {

.dashboard {

flex-direction: column;

gap: 16px;

}

}

</style>

</head>

<body>

<div class="dashboard">

<section class="chart-section">

<h2>Stock Price Line Chart</h2>

<canvas id="stockChart" width="700" height="350" style="max-width:100%;"></canvas>

</section>

<section class="form-section">

<h2>Update Data</h2>

<form id="dataForm">

<label for="labels">Labels (comma-separated)</label>

<input type="text" id="labels" name="labels" value="Mon,Tue,Wed,Thu,Fri">

<label for="prices">Prices (comma-separated)</label>

<input type="text" id="prices" name="prices" value="120,125,123,130,128">

<button type="submit">Update Chart</button>

</form>

</section>

</div>

<script>

// Initial sample data

let chartLabels = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri'];

let chartData = [120, 125, 123, 130, 128];

const canvas = document.getElementById('stockChart');

const ctx = canvas.getContext('2d');

function drawLineChart(labels, data) {

ctx.clearRect(0, 0, canvas.width, canvas.height);

// Chart area

const padding = 50;

const w = canvas.width - 2 \* padding;

const h = canvas.height - 2 \* padding;

// Find min/max

const minY = Math.min(...data);

const maxY = Math.max(...data);

const yRange = maxY - minY || 1;

// Draw axes

ctx.strokeStyle = '#333';

ctx.lineWidth = 2;

ctx.beginPath();

ctx.moveTo(padding, padding);

ctx.lineTo(padding, padding + h);

ctx.lineTo(padding + w, padding + h);

ctx.stroke();

// Draw Y axis labels

ctx.fillStyle = '#333';

ctx.font = '14px Arial';

for (let i = 0; i <= 4; i++) {

const y = padding + h - (h \* i / 4);

const value = (minY + yRange \* i / 4).toFixed(2);

ctx.fillText(value, 8, y + 4);

ctx.beginPath();

ctx.strokeStyle = '#eee';

ctx.moveTo(padding, y);

ctx.lineTo(padding + w, y);

ctx.stroke();

}

// Draw X axis labels

const stepX = w / (labels.length - 1);

labels.forEach((label, i) => {

const x = padding + i \* stepX;

ctx.fillStyle = '#333';

ctx.font = '14px Arial';

ctx.fillText(label, x - 12, padding + h + 24);

});

// Draw line

ctx.beginPath();

data.forEach((val, i) => {

const x = padding + i \* stepX;

const y = padding + h - ((val - minY) / yRange) \* h;

if (i === 0) ctx.moveTo(x, y);

else ctx.lineTo(x, y);

});

ctx.strokeStyle = '#1976d2';

ctx.lineWidth = 3;

ctx.stroke();

// Draw points

data.forEach((val, i) => {

const x = padding + i \* stepX;

const y = padding + h - ((val - minY) / yRange) \* h;

ctx.beginPath();

ctx.arc(x, y, 5, 0, 2 \* Math.PI);

ctx.fillStyle = '#1976d2';

ctx.fill();

ctx.strokeStyle = '#fff';

ctx.lineWidth = 2;

ctx.stroke();

});

}

// Initial draw

drawLineChart(chartLabels, chartData);

// Handle form submission

document.getElementById('dataForm').addEventListener('submit', function(e) {

e.preventDefault();

const labelsInput = document.getElementById('labels').value.split(',').map(s => s.trim());

const pricesInput = document.getElementById('prices').value.split(',').map(s => parseFloat(s.trim()));

if (labelsInput.length !== pricesInput.length || pricesInput.some(isNaN)) {

alert('Please enter the same number of labels and prices, and ensure all prices are numbers.');

return;

}

chartLabels = labelsInput;

chartData = pricesInput;

drawLineChart(chartLabels, chartData);

});

// Responsive canvas

function resizeCanvas() {

const section = document.querySelector('.chart-section');

const width = section.offsetWidth - 16;

canvas.width = width > 350 ? width : 350;

drawLineChart(chartLabels, chartData);

}

window.addEventListener('resize', resizeCanvas);

resizeCanvas();

</script>

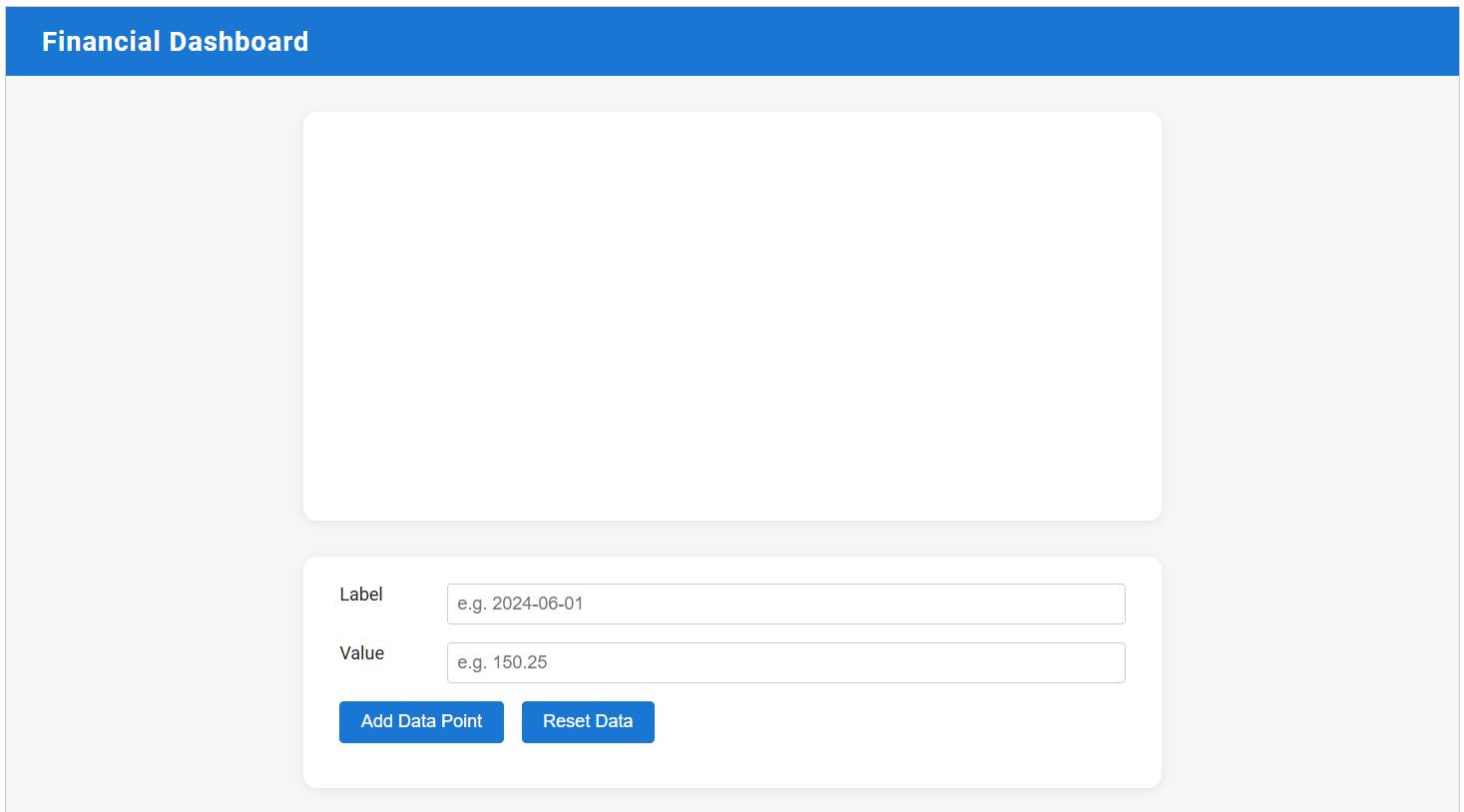
</body>

</html>

**Consulta adicional:**

Build a responsive dashboard layout with a section that displays a customizable line chart for financial data (e.g., stock prices), including a simple form to update the dataset manually.

JavaScript must be placed inside a regular <script> tag — do not use DOMContentLoaded or window.onload. Use addEventListener(...) in your JavaScript to handle events instead of inline onclick attributes. Avoid using onclick="..." or any global function calls from HTML elements.



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Financial Dashboard</title>

<link href="https://fonts.googleapis.com/css?family=Roboto:400,700&display=swap" rel="stylesheet">

<style>

body {

font-family: 'Roboto', sans-serif;

margin: 0;

background: #f4f6f8;

color: #222;

}

.dashboard {

display: flex;

flex-direction: column;

min-height: 100vh;

}

header {

background: #1976d2;

color: #fff;

padding: 1rem 2rem;

font-size: 1.5rem;

font-weight: 700;

letter-spacing: 1px;

}

.main-content {

flex: 1;

display: flex;

flex-direction: column;

align-items: center;

justify-content: flex-start;

padding: 2rem 1rem;

max-width: 900px;

margin: 0 auto;

width: 100%;

}

.chart-section {

background: #fff;

border-radius: 12px;

box-shadow: 0 2px 8px rgba(0,0,0,0.07);

padding: 2rem;

width: 100%;

max-width: 700px;

margin-bottom: 2rem;

}

.form-section {

background: #fff;

border-radius: 12px;

box-shadow: 0 2px 8px rgba(0,0,0,0.07);

padding: 1.5rem 2rem;

width: 100%;

max-width: 700px;

}

.form-row {

display: flex;

gap: 1rem;

margin-bottom: 1rem;

flex-wrap: wrap;

}

.form-row label {

flex: 0 0 80px;

font-weight: 500;

}

.form-row input {

flex: 1 1 120px;

padding: 0.5rem;

border: 1px solid #ccc;

border-radius: 4px;

font-size: 1rem;

}

.form-row button {

background: #1976d2;

color: #fff;

border: none;

padding: 0.6rem 1.2rem;

border-radius: 4px;

font-size: 1rem;

cursor: pointer;

transition: background 0.2s;

}

.form-row button:hover {

background: #125ea7;

}

@media (max-width: 600px) {

.main-content, .chart-section, .form-section {

padding: 1rem;

}

.form-row {

flex-direction: column;

gap: 0.5rem;

}

.form-row label {

flex: none;

}

}

</style>

</head>

<body>

<div class="dashboard">

<header>Financial Dashboard</header>

<div class="main-content">

<section class="chart-section">

<canvas id="lineChart" width="600" height="300"></canvas>

</section>

<section class="form-section">

<form id="dataForm" autocomplete="off">

<div class="form-row">

<label for="labelInput">Label</label>

<input type="text" id="labelInput" required placeholder="e.g. 2024-06-01">

</div>

<div class="form-row">

<label for="valueInput">Value</label>

<input type="number" id="valueInput" required placeholder="e.g. 150.25" step="any">

</div>

<div class="form-row">

<button type="submit">Add Data Point</button>

<button type="button" id="resetBtn">Reset Data</button>

</div>

</form>

</section>

</div>

</div>

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

<script>

// Initial dataset

let chartData = {

labels: ["2024-06-01", "2024-06-02", "2024-06-03", "2024-06-04", "2024-06-05"],

datasets: [{

label: "Stock Price ($)",

data: [150.25, 152.10, 149.80, 153.50, 155.00],

borderColor: "#1976d2",

backgroundColor: "rgba(25, 118, 210, 0.1)",

tension: 0.3,

fill: true,

pointRadius: 4,

pointHoverRadius: 6

}]

};

// Chart.js config

const ctx = document.getElementById('lineChart').getContext('2d');

const lineChart = new Chart(ctx, {

type: 'line',

data: chartData,

options: {

responsive: true,

plugins: {

legend: {

display: true,

position: 'top',

},

title: {

display: true,

text: 'Stock Price Over Time',

font: { size: 18 }

}

},

scales: {

x: {

title: {

display: true,

text: 'Date'

}

},

y: {

title: {

display: true,

text: 'Price ($)'

},

beginAtZero: false

}

}

}

});

// Add event listeners for the form

document.getElementById('dataForm').addEventListener('submit', function(e) {

e.preventDefault();

const label = document.getElementById('labelInput').value.trim();

const value = parseFloat(document.getElementById('valueInput').value);

if (label && !isNaN(value)) {

chartData.labels.push(label);

chartData.datasets[0].data.push(value);

lineChart.update();

this.reset();

}

});

document.getElementById('resetBtn').addEventListener('click', function() {

chartData.labels = [];

chartData.datasets[0].data = [];

lineChart.update();

});

</script>

</body>

</html>