```
// bar-chart.js - Interactive Bar Chart for Traffic Violations


async function createBarChart() {
    // Parse CSV data
    const parsedData = await d3.csv('preproccessed_data/detectionMethod_method_FCA.csv');

    // Clean and process data
    const processedData = parsedData.map(d => ({
        metric: d.METRIC,
        detectionMethod: d.DETECTION_METHOD,
        fines: +d["Sum(FINES)"] || 0,
    })).filter(d => d.fines > 0 || d.arrests > 0 || d.charges > 0);

    // Calculate total values and sort by total descending
    const chartData = processedData.map(d => ({
        ...d,
        total: d.fines,
        label: `${d.metric.replace(/_/g, ' ').replace(/\b\w/g, l => l.toUpperCase())} - ${d.detectionMethod}`
    })).sort((a, b) => b.total - a.total);

    // Chart dimensions
    const margin = { top: 50, right: 30, bottom: 50, left: 70 },
    width = 1000 - margin.left - margin.right,
    height = 400 - margin.top - margin.bottom;

    // Clear any existing chart
    d3.select(".placeholder-charts-boxchart").selectAll("*").remove();

    // Create SVG
    const svg = d3.select(".placeholder-charts-boxchart")
        .append("svg")
        .attr("width", width + margin.left + margin.right)
        .attr("height", height + margin.top + margin.bottom);

    const g = svg.append("g")
        .attr("transform", `translate(${margin.left},${margin.top})`);

    // Create scales
    const xScale = d3.scaleBand()
        .domain(chartData.map(d => d.label))
        .range([0, width])
        .padding(0.1);

    const yScale = d3.scaleLinear()
        .domain([0, d3.max(chartData, d => d.total)])
        .nice()
        .range([height, 0]);

    // Color scale for different metrics
    const colorScale = d3.scaleOrdinal()
        .domain([...new Set(chartData.map(d => d.metric))])
        .range(['#2E86AB', '#A23B72', '#F18F01', '#C73E1D', '#6A994E']);

    // Create tooltip
    const tooltip = d3.select("body").append("div")
        .attr("class", "bar-tooltip")
        .style("opacity", 0)
        .style("position", "absolute")
        .style("background", "rgba(0, 0, 0, 0.9)")
        .style("color", "white")
        .style("padding", "12px")
        .style("border-radius", "8px")
        .style("pointer-events", "none")
        .style("font-size", "14px")
        .style("z-index", "10")
        .style("max-width", "300px")
        .style("box-shadow", "0 4px 6px rgba(0, 0, 0, 0.1)");

    g.append("g")
        .attr("class", "x-axis")
        .attr("transform", `translate(0,${height})`)
        .call(d3.axisBottom(xScale))
        .selectAll("text")
        .style("text-anchor", "middle")
        .style("font-size", "12px")
        .each(function (d) {
            const text = d3.select(this);
            const words = d.split(' - ');
            text.text('');

            text.append("tspan")
                .attr("x", 0)
                .attr("dy", "1.0em")
                .style("font-size", "12px")
                .style("font-weight", "600")
                .text(words[1]);

        });
```

```
g.append("g")
    .attr("class", "y-axis")
    .call(d3.axisLeft(yScale).tickFormat(d3.format(".2s")))
    .selectAll("text")
    .style("font-size", "12px");

// Add axis labels
g.append("text")
    .attr("transform", "rotate(-90)")
    .attr("y", 0 - margin.left)
    .attr("x", 0 - (height / 2))
    .attr("dy", "1em")
    .style("text-anchor", "middle")
    .style("font-size", "12px")
    .style("font-weight", "bold")
    .text("Total Violations");

// Create bars
const bars = g.selectAll(".bar")
    .data(chartData)
    .enter()
    .append("rect")
    .attr("class", "bar")
    .attr("x", d => xScale(d.label))
    .attr("width", xScale.bandwidth())
    .attr("y", height)
    .attr("height", 0)
    .style("fill", d => colorScale(d.metric))
    .style("opacity", 0.8)
    .style("cursor", "pointer");

// Add interactivity to bars
bars
    .on("mouseover", function (event, d) {
        // Highlight bar
        d3.select(this)
            .transition()
            .duration(200)
            .style("opacity", 1)
            .style("stroke", "#333")
            .style("stroke-width", 2);

        // Show tooltip
        const tooltipContent = `
    <div style="border-bottom: 1px solid #555; padding-bottom: 8px; margin-bottom: 8px;">
      <strong>${d.metric.replace(/_/g, ' ').replace(/\b\w/g, l => l.toUpperCase())}</strong><br/>
      <em>${d.detectionMethod}</em>
    </div>
    <div>
      <strong>Fines: ${d3.format(",d")(d.fines)}</strong>
    </div>
  `;

        tooltip.transition()
            .duration(200)
            .style("opacity", 1);

        tooltip.html(tooltipContent)
            .style("left", (event.pageX + 10) + "px")
            .style("top", (event.pageY - 10) + "px");

    })
    .on("mouseout", function (event, d) {
        // Reset bar
        d3.select(this)
            .transition()
            .duration(200)
            .style("opacity", 0.8)
            .style("stroke", "none");

        // Hide tooltip
        tooltip.transition()
            .duration(300)
            .style("opacity", 0);

    })

// Animate bars
bars.transition()
    .duration(1000)
    .delay((d, i) => i * 50)
    .attr("y", d => yScale(d.total))
    .attr("height", d => height - yScale(d.total));

// Add title
svg.append("text")
    .attr("x", (width + margin.left + margin.right) / 2)
    .attr("y", 30)
    .attr("text-anchor", "middle")
    .style("font-size", "20px")
    .style("font-weight", "bold")
    .style("fill", "#333")
    .text("Traffic Violations by Detection Method");

// Add subtitle
svg.append("text")
```

```
        .attr("x", (width + margin.left + margin.right) / 2)
        .attr("y", 50)
        .attr("text-anchor", "middle")
        .style("font-size", "14px")
        .style("fill", "#666")
        .text("Total fines sorted by frequency");

}

createBarChart();
```