# Krunchbox 2.0 - Senior Frontend Developer Code Test

## Overview

Build a React component that displays retail analytics data - a simplified version of our "Insight Card" feature that will replace traditional dashboards in Krunchbox 2.0.

## Time Allocation

* **Setup**: Untimed (create React app, install dependencies)
* **Coding**: 90 minutes
* **Discussion**: 30 minutes

## The Challenge

Create a **Sales Performance Insight Card** that fetches retail data from our API and presents it in a clear, actionable format for retail managers who need quick insights.

### Requirements

**Core Functionality:**

* Fetch data from an api endpoint. JSON included in this email.
* Display key metrics with appropriate formatting (currency, percentages)
* Show performance indicators (up/down arrows, color coding)
* Include one data visualization (chart)
* Include one data table showing top performing stores
* Handle loading and error states gracefully

**Technical Requirements:**

* React with TypeScript
* AG Grid for the data table
* Plotly.js for visualization
* shadcn/ui for UI components (buttons, cards)
* Responsive design (desktop and tablet)
* Clean component structure

**Deliverables:**

1. Working React application
2. At minimum: One complete Insight Card component
3. Basic error handling and loading states
4. Clean, production-ready code

### Evaluation Criteria

* **Code Quality** - TypeScript usage, component structure, maintainability
* **UI/UX** - Clear data presentation, responsive design, loading/error states
* **Business Understanding** - Appropriate metric selection and display
* **Performance** - Efficient rendering, proper React patterns
* **Problem Solving** - Handling edge cases, data formatting

### What We Provide

* Mock API endpoint with sample retail data
* Basic design specifications (colors, spacing)
* Access to all documentation during the test

### For Lead Developer Candidates (Additional 30 min)

* Code review exercise
* Architecture discussion: scaling to 50+ cards
* Team process and migration strategy questions

### Sample Output Expected

┌─────────────────────────────────────────┐  
│ Sales Performance - Week 37, 2025 │  
│ │  
│ $245,678 ↑ 12.2% vs last week │  
│ │  
│ [Line chart showing 7-day trend] │  
│ │  
│ Top Performing Stores: │  
│ ┌─────────────────────────────────────┐ │  
│ │ Store Sales Units Margin % │ │  
│ │ WMT1234 $45.6K 680 ↑15.2% │ │  
│ │ TGT5678 $38.9K 545 ↑8.5% │ │  
│ └─────────────────────────────────────┘ │  
│ │  
│ [Export CSV] [View Full Report] │  
└─────────────────────────────────────────┘

## For Internal Review

### Why This Test Works

1. **Directly Relevant** - Insight Cards are core to Krunchbox 2.0
2. **Real Tech Stack** - Uses our exact tools (React, TypeScript, AG Grid, Plotly)
3. **Appropriate Complexity** - Combines data fetching, visualization, and state management
4. **Business Context** - Tests understanding of retail metrics and user needs
5. **Scalable Thinking** - Discussion reveals enterprise SaaS capability

### Red Flags to Watch For

* Over-engineering the solution
* No error boundaries or fallback UI
* Poor TypeScript usage (any everywhere)
* Ignoring responsive design
* Inefficient re-renders
* No loading states
* Poor data formatting (showing 245678.5 instead of $245,678)

### Green Flags to Look For

* Clean component composition
* Proper TypeScript interfaces
* Memoization where appropriate
* Accessible UI patterns
* Clear code organization
* Thoughtful error handling
* Business metric understanding

### Interview Discussion Topics

**Technical Deep Dive (15 min)**

* Walk through their solution architecture
* Component composition decisions
* State management approach
* Performance optimizations made

**Scaling & Architecture (15 min)**

* How would this scale to 50+ cards on screen?
* Real-time updates via WebSockets approach
* Handling 100k+ row exports
* Multi-tenant customization strategy

**For Lead Candidates - Additional Topics**

* Team structure for frontend development
* Migration strategy from legacy dashboards
* Code review best practices
* Collaboration with Python/FastAPI backend team

### Alignment with Krunchbox 2.0 Goals

This test evaluates candidates against our key initiatives:

* **Simplified UI/UX** - Can they create intuitive interfaces?
* **Performance at Scale** - Do they think about enterprise requirements?
* **Modern Architecture** - Component-based, maintainable code
* **Business Understanding** - Appropriate metric selection and display

### Setup Instructions for HR/Recruiting

1. Create mock API at mockapi.io with provided JSON
2. Send test document 24 hours before interview
3. Provide Zoom/Teams link for screen sharing during coding
4. Ensure candidate has Node.js and npm installed
5. Allow full internet access during test

*This test is designed to take 2 hours total and directly reflects actual Krunchbox 2.0 development work.*