Vector Remote Care

Staff Engineer Take-Home Assessment

Overview

Design a scalable system architecture that demonstrates strategic technical leadership, system thinking, and the ability to guide teams through complex problems. This assessment emphasizes architecture, decision-making, and communication over implementation.

Expected Time Investment: 4-6 hours

Focus: 60% design/architecture, 40% selective implementation

Time Box Guidance: This assessment is designed for no more than 4-6 hours of focused work. We value strategic thinking and discussions over comprehensive documentation. Show us how you think about the hardest problems, not everything you could build.

The Challenge

Vector Remote Care is scaling from 100,000 to 300,000 patients over the next 18 months. Our remote patient monitoring platform processes continuous streams of physiological data from various monitoring devices and generates alerts when clinical intervention may be needed. As we scale, we need to evolve our architecture to handle increased volume while maintaining the clinical accuracy that our customers depend on.

Your Task: Design and partially implement a system that improves how we process, prioritize, and route patient monitoring alerts at scale.

Business Context

Vector serves healthcare organizations that remotely monitor patients with chronic conditions. Patient data arrives throughout the day from various monitoring devices, and our system must identify which data points require clinical attention and route them to the appropriate care team. The platform must handle multiple types of monitoring data, each with different clinical significance and urgency levels.

System Requirements

- **Scale:** Support 3x growth in patient volume (100K → 300K patients) with proportional increase in daily data transmissions
- **Clinical Accuracy:** Maintain 95% sensitivity in identifying alerts that require clinical review (minimize both false positives and false negatives)
- **Data Diversity:** Handle multiple types of monitoring data with different formats, frequencies, and clinical rules

- **Intelligent Routing:** Direct alerts to appropriate clinical teams based on severity, data type, and patient context
- **Extensibility:** Allow for future expansion to new monitoring types and clinical protocols

Your Approach: You have flexibility in how you solve this problem. You might focus on alert classification, intelligent routing, system reliability, data processing pipelines, or another critical aspect. The key is demonstrating how you think about complex systems problems and lead technical solutions.

Part 1: Architecture & Strategy (2-3 hours)

Deliverable: Architecture Design Document (2-3 pages)

Combine the following into one comprehensive document:

- **Problem Analysis:** What's the core technical challenge you're solving? What are the critical failure modes?
- **Architecture Diagram:** High-level system design showing key components and data flows
- **Data Model:** Core entities and relationships needed to support your solution
- **Architectural Decision Record (ADR):** Document ONE critical decision (e.g., database choice, processing model, routing strategy). Include: Context, Decision, Consequences, Alternatives Considered
- **Scalability Strategy:** How does your system handle 3x growth? What are your graceful degradation strategies?
- **Observability:** What metrics matter most and why?

Part 2: Implementation & Technical Depth (1.5-2 hours)

Deliverable: Core Component Implementation

Choose ONE component that's critical to your architecture and implement it. This could be:

- Alert classification logic that applies clinical rules to monitoring data
- Routing engine that distributes alerts to appropriate care teams
- Data processing pipeline that handles high-volume streaming data
- Queue management system for handling burst traffic
- Another core component central to your design

Requirements:

- Production-quality code structure with clear separation of concerns
- Core business logic that demonstrates your approach
- Key unit tests showing your testing philosophy (focus on critical paths)

- Basic error handling for expected failure modes
- README with API contract, design decisions, and setup instructions

Tech Stack: Use what you know best. Node.js/JavaScript/TypeScript preferred. Your choice of database and testing framework—justify in your documentation.

Part 3: Technical Leadership & Strategy (1-1.5 hours)

Deliverable: Engineering Leadership Brief (1.5-2 pages)

Combine the following into a single document with three sections:

Section 1: Technical Trade-offs (0.5 page)

- What shortcuts did you take in your implementation and why?
- What would you build differently with 6 months vs. 2 weeks?
- How do you balance speed to market with technical quality in a healthcare system?
- What's your testing strategy philosophy for systems where accuracy directly impacts patient safety?

Section 2: Team & Process (0.5-1 page)

You're leading a team of 4 engineers (mix of senior and mid-level) to build this system:

- How would you break down the work and structure it across the team?
- What code quality standards and practices would you establish?
- What is one key process you'd implement to maintain quality as the system evolves?
- How would you ensure the team maintains the 95% sensitivity requirement over time?

Section 3: Cross-functional Collaboration (0.5 page)

Describe how you'd work with:

- **Product:** How do you help them understand technical constraints and trade-offs?
- **Clinical Operations:** How do you ensure the system meets real clinical workflow needs?
- **Data Team:** How do you coordinate on data pipelines, quality, and shared infrastructure?

Evaluation Criteria

We'll evaluate your submission on:

- **System Design (30%)** - Can you design for scale, reliability, and future evolution? Do you identify the right problems to solve?
- **Technical Depth (25%)** - Does your implementation demonstrate production-quality practices and sound engineering judgment?
- **Strategic Thinking (20%)** - Do you connect technical decisions to business outcomes and patient safety?
- **Communication (15%)** - Can you explain complex technical concepts clearly to different audiences?
- **Leadership (10%)** - Do you think about team enablement, sustainable practices, and quality at scale?

Submission Guidelines

Submit via GitHub repository including:

1. **README.md** - Setup instructions, overview, and design decisions
2. **/docs folder** containing:
   - Architecture design document
   - Engineering leadership brief
3. **/src folder** - Your core component implementation
4. **/tests folder** - Test suite demonstrating your testing philosophy

Evaluation Timeline: We'll review as a team as quickly as possible and schedule a follow-up technical discussion to dive deeper into your design decisions.

Questions?

This assessment is designed to simulate real work at Vector. If you have clarifying questions (as you certainly would with stakeholders), please document them and your assumptions in your submission. We value engineers who seek clarity and communicate constraints.

Key Differentiator for Staff Level: We're evaluating your ability to identify the right problems, lead architecture, make strategic trade-offs, and elevate team capabilities—not just your coding skills. Show us how you think about systems, scale, and sustainable engineering practices in a healthcare context.