Product Requirement and Low-Fidelity Wireframes

1. Product Requirements

Objective:
To provide users with a comprehensive dashboard and tools to:
- View container images in their repositories.
- Identify and prioritize vulnerabilities based on severity.
- Take action to fix or mitigate critical/high vulnerabilities.

Target Users:
- DevOps Engineers
- Security Engineers
- Platform Engineers
- Site Reliability Engineers (SREs)

Core Features:
A. Dashboard Overview
- Total Images Scanned
- Vulnerable Images Count
- Severity Breakdown (Critical, High, Medium, Low, Info)
- Trend graph of vulnerabilities over time

B. Image List View
- Search and filter (by name, tag, severity, date)
- Columns: Image Name, Tag, Last Scanned Date, Total Vulnerabilities, Severity Breakdown, Action Required, Fixable Issues Count

C. Image Details Page
- Metadata: Image Name, Tag, Created Time, Size
- Vulnerability Breakdown (ID, Severity, Component, Version, Fix Version, Description, Fix Status)
- Filter/Sort by severity or fix availability

D. Fix Suggestions
- List of vulnerabilities with available patches or version upgrades
- Export or generate a fix plan

E. Notifications & Alerts
- Configure alert rules for critical/high vulnerabilities
- Email, Slack, or webhook integrations

Non-Functional Requirements:
- Scalability to support thousands of images
- Real-time or scheduled scanning support
- Role-Based Access Control (RBAC)
- Audit logs for activity tracking

Success Metrics:
- % of fixable vulnerabilities identified and resolved
- Mean Time to Detect (MTTD) & Mean Time to Resolve (MTTR)
- User engagement (dashboard visits, fixes triggered)

2. Low-Fidelity Wireframes

Below is a sample wireframe illustrating the vulnerability dashboard layout:

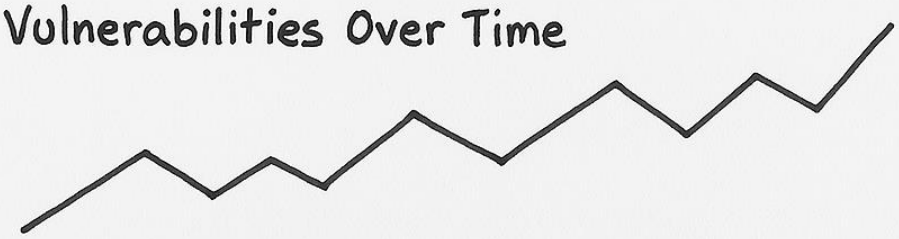
Vulnerability Dashboard

Total Images: 1,204

Vulnerable Images: 542

Critical 47 High 183 Medium Low 121

Vulnerabilities Over Time



Search images

Filter

Image Name	Tag	Last Scanned	Vulnerabilities
backend	v2	04/03/205	7 Critical
frontend	v5	04/01/2025	2 High
frontend	v4	03/27/2025	10 Medium
api-core	v1	03/30/2025	12 Critical

3. Development Action Items

Back-End:
- Integrate container image scanning tool (e.g., Trivy, Clair, Anchore)
- Set up scheduler for periodic scans
- Design database schema for storing image metadata and scan results
- API endpoints for: Dashboard stats, Image list and filters, Vulnerability details, Notifications configuration

Front-End:
- Dashboard UI (charts, tables, filters)
- Image detail and vulnerability viewer
- Notification setup and alerts UI

DevOps & Infrastructure:
- CI/CD setup for scanning and deployment
- Storage and caching for scanned results
- User authentication & RBAC implementation