Product Requirements Document

1. Background & Problem Statement

A security product needs to scan container images for vulnerabilities and present the findings to users.

Container images contain applications along with dependencies, and any of these components may have known vulnerabilities.

User Pain Points:

- Users need to quickly identify which container images have vulnerabilities and their severity levels.
- Critical and high-severity vulnerabilities need immediate attention.
- Users manage thousands of container images, so efficient filtering and sorting mechanisms are crucial.

2. Goals & Objectives

- Provide a dashboard displaying an overview of container image vulnerabilities.
- Allow users to filter and sort images by severity, name, and date.
- Enable users to drill down into details of an image's vulnerabilities.
- Provide actionable insights to fix vulnerabilities.
- Offer an intuitive and scalable UI to handle thousands of images.

3. Features & Functional Requirements

3.1 Core Features

1. **Dashboard View**: Displays a summary of scanned container images, highlighting critical and high-severity vulnerabilities.
2. **Search & Filter**: Users can filter images based on:
   - Severity (Critical, High, Medium, Low, None)
   - Image Name
   - Last Scan Date
3. **Vulnerability Details Page**: Clicking an image opens a detailed view showing:
   - List of vulnerabilities

   - CVE ID, description, and severity
   - Affected package and recommended fixes
4. **Bulk Actions**:
   - Users can select multiple images and trigger a re-scan.
   - Option to export vulnerability reports.
5. **Notifications & Alerts**:
   - Alerts for newly discovered critical/high vulnerabilities.
   - Email/Slack notifications for security teams.

3.2 Non-Functional Requirements

- **Scalability**: Should handle thousands of images efficiently.
- **Performance**: Fast search and filtering.
- **Security**: Ensure secure API access and data protection.

4. User Workflow

1. User logs in and accesses the dashboard.
2. User sees a list of container images with vulnerability summaries.
3. User filters by high/critical severity to find the most urgent issues.
4. User clicks an image to view details.
5. User takes action:
   - Apply suggested fixes.
   - Mark vulnerabilities as acknowledged.
   - Trigger re-scan.
   - Export the report.

5. Wireframes

5.1 Dashboard View
- A table showing a list of container images.
- Columns: Image Name, Last Scan Date, Severity Summary, Actions.
- A filter panel on the left side to filter by severity, name, and date.
- A summary widget on top showing total scanned images and critical vulnerabilities.

5.2 Vulnerability Details Page

- Header with Image Name, Last Scan Date, and Overall Severity.
- Table listing vulnerabilities with columns: CVE ID, Severity, Affected Package, Fix Available.
- Action buttons: Re-Scan, Export Report, Acknowledge Issues.

5.3 Bulk Actions

- Checkboxes to select multiple images.
- Action dropdown for "Re-Scan Selected" and "Export Selected Reports".

5.4 Notifications & Alerts

- A notification bell icon in the top navigation bar.
- Alerts for newly detected critical vulnerabilities.
- Option to enable email/Slack notifications.