The proposed solution involves creating a vulnerability scanning tool for container images, aimed at helping users identify, prioritize, and fix security issues. Here's a detailed breakdown:

Key Features:

1.Vulnerability Scanning:

Automatically scan container images for vulnerabilities and categorize them by severity (Critical, High, Medium, Low, Info).
Continuous or periodic scanning of repositories to keep vulnerability data up to date.

2.Dashboard:

A high-level overview of scanned images with key metrics such as the total number of images, and a breakdown of vulnerabilities by severity.
Highlights container images with Critical and High vulnerabilities, enabling users to prioritize these issues.

3.Image Repository Management:

A searchable and filterable table that displays thousands of container images, with columns for Image Name, Last Scan Date, and the number of vulnerabilities per severity level.
Allows users to filter and sort images by different parameters, including severity and tags, for easier navigation.

4.Detailed Vulnerability View:

Provides detailed information about vulnerabilities in each image, such as severity, affected packages, CVE details, and remediation steps.
Allows users to filter vulnerabilities by severity level within each image.

5.Fix Recommendations:

For each identified vulnerability, the tool suggests remediation steps, such as upgrading to a newer version of the affected package.
Provides details about which package versions contain fixes for specific vulnerabilities.

6.Bulk Actions:

Users can select multiple images to trigger re-scans or export vulnerability reports in bulk, simplifying the workflow for large repositories.

6.Notifications and Alerts:

Configurable alerts to notify users when new vulnerabilities are discovered, especially Critical and High-severity vulnerabilities.

7.Export Reports:

Users can generate and export detailed vulnerability reports in formats like PDF or CSV for documentation or sharing with other teams.

WIREFRAMES (Low-Fidelity)

1. Dashboard View

A top bar with a search/filter option for image repositories.
Cards showing overall metrics like "Total Images Scanned," "Critical Vulnerabilities," "High Vulnerabilities," etc.
A table with sortable columns (Image Name, Last Scan Date, Critical/High/Medium/Low Vulnerabilities).
A filter dropdown to filter by severity levels or image tag.

2. Image Details View

Breadcrumb navigation to go back to the main dashboard.
A header displaying image name and metadata (last scan date, repository, tags, etc.).
A list of vulnerabilities with a column for severity, affected package, and remediation steps.
A filter option to show vulnerabilities by severity (Critical, High, Medium, etc.).

3. Fix and Recommendation Modal

Modal showing a summary of critical vulnerabilities for a specific image.
For each vulnerability, display the affected package, current version, and recommended fixed version.
Button to initiate a re-scan after a fix is applied

DEVELOPMENT ACTION ITEMS:

1.Vulnerability Database Integration: Connect the tool to vulnerability databases (e.g., CVE, Clair, Trivy) to retrieve vulnerability information for scanned images.

2.Image Repository API Integration: Build API calls to manage and retrieve metadata from container image repositories (e.g., image name, tag, last scan date).

3.UI Components: Develop the user interface for both the dashboard and detailed vulnerability views, enabling users to easily navigate the data.

4.Bulk Actions: Implement backend services to support bulk operations like re-scanning multiple images and exporting reports.

5.Notification System: Create a system that sends alerts to users when specific severity thresholds are met (e.g., Critical vulnerabilities).

6.Fix Recommendations Logic: Implement logic to suggest remediations, such as identifying package versions that have fixed vulnerabilities.