Accuknox Assignment

By Ravi Prakash

Problem Statement 1:
Title: Product Requirement and Low-Fidelity Wireframes

Product Requirements Document

Title: Container Image Vulnerability Scanner

1. Background

A security product is required to scan container images, detect vulnerabilities, and display findings to users. These container images may contain applications with dependencies that might have known vulnerabilities. The product should help users quickly identify, assess, and fix vulnerabilities in their repositories.

2. Objective

The goal is to build a simple tool that helps users to find out which container images have security risks. Understand how serious each issue is (Critical, High, Medium, Low) and Take action to fix the most important problems first with capacity to handle large numbers of images smoothly and efficiently.

3. User Personas

- **DevOps Engineers:** Need to ensure container images are secure before deployment to Production.
- **Security Analysts:** Require detailed insights to proactively manage risks.
- **Developers:** Want quick access to vulnerability details to fix issues faster.

4. User Stories

- **As a DevOps Engineer:** I want to see an overview of all container images along with their security risks so that I can evaluate potential threats.
- **As a Security Analyst:** I need to sort and filter images by vulnerability severity to ensure that the most critical risks are addressed first.
- **As a Developer:** I need a way to explore specific vulnerabilities in depth, including their root causes and recommended fixes, to implement security patches effectively.
- **As a Platform Administrator:** I need visibility into which container images have security issues and their severity levels to maintain compliance and security. When critical or high-severity vulnerabilities are found, I need an efficient way to pinpoint affected images and take corrective actions.

5. Functionals Requirements and Must Have Features

- **Image Scanning :** Users can scan a container image for vulnerabilities.The scan results show a list of vulnerabilities categorized by severity with an option to rescan after updates.
- **Dashboard Overview:** Displays total images scanned with vulnerability breakdown by severity and Trend of vulnerabilities over time and a List of most vulnerable images as well.
- **Search & Filtering:** Users can filter vulnerabilities by severity (Critical, High, Medium, Low) and also perform Search by image name, tag, or CVE identifier.
- **Vulnerability Details & Remediation:** On Clicking on a vulnerability it provides Description and impact, Affected components and CVE details with links. Also Suggest remediation (Like: patch/update/ignore).
- **Notifications:** Alerts users when new vulnerabilities are detected in existing images
- **Bulk Actions:** Users can select multiple images and initiate actions such as: "Re-scan" to check for updated vulnerability information. "Mark as reviewed" to indicate that an image has been assessed.
- **User Interface :**The UI should be intuitive and easy to navigate.The Critical and high vulnerabilities should be visually highlighted. The design should support both light and dark modes for user preferences.

6. Non Functional Requirements

- **Security:** Implement secure access with role-based permissions to ensure only authorized users can view or manage vulnerabilities.

- **Performance:** The tool should be able to scan thousands of images efficiently without significant delays.

- **Scalability:** The system should support growing image repositories and increased scan requests as user needs expand.

7. Success Matrix

- Percentage of images scanned.

- Reduction in unresolved Critical/High vulnerabilities.

- Time taken from detection to remediation.

- User engagement with dashboard insights.

8. Development Action Items **(Bonus Task)**

Backend Development

- Implement an API to trigger vulnerability scans for container images.
- Develop a database schema to store scan results, images, and metadata.
- Implement role-based access control (RBAC) for user authentication.
- Enable CI/CD pipeline integration for automated security scanning.

Frontend Development

- Design and implement a user-friendly dashboard displaying scan summaries.
- Develop filtering and search functionalities for images and vulnerabilities.
- Create detailed vulnerability pages showing CVE details and remediation steps.
- Implement bulk action capabilities for rescanning and marking images as reviewed.

Performance & Scalability

- Optimize scanning processes for minimal latency.
- Ensure the system can handle thousands of images efficiently.
- Implement a scalable architecture to support increasing repository sizes.

Security & Compliance

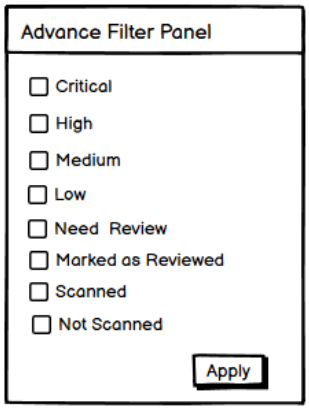
- Implement encryption for stored scan results and user credentials.
- Ensure compliance with security best practices (Like: OWASP Top 10)
- Provide audit logs for all scan activities and user interactions.

Low-Fidelity Wireframes

Image Dashboard:

It helps users to quickly identify images with critical vulnerabilities and prioritize fixes.

- Purpose: Shows a list of container images with their vulnerability counts and severity.
- Components:
    - Header: Search bar, filter options,profile settings,Notifications.
    - Image List: Displays image name, vulnerability counts, and severity distribution with visual indicators.
    - Sort & Pagination: Options to sort images and navigate through pages.
    - Details and Export Option to directly export details in Excel.



Detailed View:

It helps users to Understand specific vulnerabilities and take immediate action to resolve them.

- Purpose: Provides an in-depth view of vulnerabilities for a selected image.
- Components: Header: Displays image name, scan date, and summary.
- Vulnerability List: Detailed rows with ID, description, severity, impact, and remediation steps. Action Buttons: Options like “Mark as Fixed” or “Re-scan.”

Image Repository: It helps users manage and scan container images for vulnerabilities efficiently.

Purpose:
Provides an overview of all container images with filtering and detailed history for each image.

Components:

- **Header:** Displays the application name "Container Image Vulnerability Scanner" with user profile, notification, and settings icons.
- **Search Bar:** Allows users to search for specific container images.
- **Advanced Filter:** Provides filtering options based on vulnerability type, severity, date, or container details.
- **Image Grid:** Displays container images in a structured format, where each image contains details such as vulnerability history, date, and container specifications.
- **Information Panel:** Shows metadata and history of each image, including vulnerability details, date & time of scans, and container specifics.
- **Action Buttons:**
    - **Export:** Allows users to export image vulnerability reports.
    - **Help:** Provides guidance on using the repository and understanding vulnerability reports.