AccuKnox Product Management Assignment

1️⃣ Product Overview

This product helps teams identify security vulnerabilities in their container images quickly. It scans all images within a repository, identifies security issues, and indicates their severity levels. The goal is to streamline the process of addressing critical security concerns, allowing teams to prioritize fixes and maintain system security effectively.

2️⃣ Target Users

This tool is mainly for DevOps teams, security engineers, and IT admins who work with containerized apps. They need a fast and easy way to find and fix security issues in container images—especially when they’re handling hundreds or thousands of them.

3️⃣ Key Goals

- Help users **quickly find container images** that have security problems.
- Show how **serious each issue is** — whether it's critical, high, medium, or low.
- Let users **filter and sort** the list to focus on the most important ones first.
- Give **clear instructions** to fix the major problems (especially critical and high ones).
- Make sure the tool works well even if the user has **thousands of images** to scan.

4️⃣ Core Features

- **Scans Images for Problems**

  The tool automatically checks container images for known security issues using tools like **Trivy** or **Clair**.

- **Shows How Serious the Issues Are**

It clearly marks each issue as **critical, high, medium, or low**, so users know what to fix first.

- **Works with Many Images at Once**

  Users can filter, sort, and take action on **multiple images** in one go — super helpful when managing thousands.

- **Helps You Fix the Problems**

  For each issue, it gives **easy-to-follow guides or links** to help fix them.

- **Connects to Your Image Repositories**

  It can connect with popular image storage platforms like **DockerHub, AWS ECR**, or **Google Container Registry**.

5️⃣ User Workflow

- **Start Scanning**

  Users kick off a scan to check their container images for security issues.

- **See the Results**

  The tool shows a list of images with their security problems and how serious each one is.

- **Focus on the Big Issues**

  Users can filter the list to zoom in on **critical and high-risk** issues.

- **Fix the Problems**

  The tool gives easy guides to help users **fix** the security issues.

- **Check Again**

  After fixing the problems, users can **re-scan** the images to make sure everything's secure.

6️⃣ Low-Fidelity Wireframes

Wireframe Sections:

- **Dashboard Page**
  - Shows a **list of container images** with:

        - Image Name
        - Number of Vulnerabilities
        - **Severity Level** (Critical, High, Medium, Low)
    - **Filters** and **Search Bar** at the top to find specific images.
- **Vulnerability Details Page**
    - When an image is clicked, shows a **detailed list of vulnerabilities** for that image.
    - **Links to remediation guides** to help fix the issues.
    - **Options to either fix or ignore** the vulnerabilities.
- **Bulk Action Page**
    - Allows users to **filter multiple images** by severity or repository.
    - Option to **re-scan** images after fixing issues.

7️⃣ Development Action Items

- **Container Image Scanning**: Use tools like **Trivy**, **Clair**, or **Anchore** to scan container images for vulnerabilities and generate security reports.
- **Severity Calculation**: Implement a severity classification system (Critical, High, Medium, Low) to help users prioritize urgent security issues.
- **UI Development**: Create a clean, user-friendly interface that allows users to view images, apply filters, and access vulnerability details. Ensure the interface works well across all devices.
- **Backend Development**: Build APIs to integrate with container registries like Docker Hub and AWS ECR for image fetching and vulnerability scanning.
- **Security**: Implement user authentication to ensure only authorized users can access and manage container images and vulnerability data.
- **Testing**: Conduct thorough unit and integration testing to verify the reliability of image scanning and vulnerability tracking features.

8️⃣ Bonus/Optional Task: Development Action Items to Discuss with the Development Team

Clear communication with the development team is essential. Here are the key action items for building the product:

1. Container Image Scanning

- **Action Item**: **Select and Integrate Scanning Tools**
  - **Objective**: Integrate security scanning tools like **Trivy**, **Clair**, or **Anchore** to perform vulnerability analysis.
  - **Discussion Points**:
    - Select the most suitable tool between **Trivy**, **Clair**, or **Anchore** based on scanning speed, accuracy, and integration ease.
    - Design scanning processes for both manual and automated image analysis.
    - Determine whether scanning should be asynchronous for better user experience or synchronous for immediate results.
    - Consider implementing a **queue system** for managing lengthy scans.

---

2. Severity Calculation and Reporting

- **Action Item**: **Categorize Vulnerabilities by Severity**
  - **Objective**: Create a system to classify vulnerabilities into **Critical**, **High**, **Medium**, and **Low** severity levels.
  - **Discussion Points**:
    - Establish clear criteria for severity levels using CVE scores or internal standards.
    - Evaluate the need for customizable severity levels and filters.
    - Plan how severity levels will influence UI/UX, such as using **color coding** for quick identification.

---

3. User Interface (UI) Development

- **Action Item**: **Design and Implement User-Friendly UI**
  - **Objective**: Create an intuitive, responsive UI for clear presentation of images, vulnerabilities, and fixes.
  - **Discussion Points**:
    - Select an appropriate UI framework (React, Angular, Vue.js) considering scalability and backend integration.
    - Create efficient user flows for common tasks like scanning images and managing vulnerabilities.
    - Determine the most effective way to display vulnerability information through sortable tables or expandable lists.
    - Consider adding a **progress indicator** for scan monitoring.

---

4. Backend Development

- **Action Item**: **Develop API Services for Container Repositories**
  - **Objective**: Build backend services to interact with **Docker Hub**, **AWS ECR**, and **Google Container Registry**.
  - **Discussion Points**:
    - Establish reliable connections to registries for image metadata retrieval.
    - Design a scalable database architecture for large repository management.
    - Evaluate the need for periodic scans versus **real-time updates** for new images.

---

5. User Authentication and Security

- **Action Item**: **Implement User Authentication and Authorization**
  - **Objective**: Secure access to the tool and sensitive image data through user authentication.
  - **Discussion Points**:

    - Compare authentication methods: **session-based authentication**, **JWT (JSON Web Tokens)**, or **OAuth**.
    - Define clear user roles (Admin, User, Viewer) and their permissions.
    - Implement secure frontend-backend communication using **HTTPS**.

---

6. Testing and Quality Assurance

- **Action Item**: **Implement Unit and Integration Testing**
    - **Objective**: Ensure system reliability through comprehensive testing.
    - **Discussion Points**:
        - Develop thorough **unit tests** for core components like scanning and severity calculation.
        - Create comprehensive **integration tests** to verify system-wide functionality.
        - Define test coverage requirements and include stress testing for large datasets.

---

7. Performance Optimization

- **Action Item**: **Optimize Performance for Large-Scale Repositories**
- **Objective**: Create an efficient system capable of handling thousands of container images.
- **Discussion Points**:
    - Implement **parallel processing** and **asynchronous tasks** for simultaneous image scanning.
    - Develop a strategic **caching** system to minimize redundant operations.
    - Optimize database performance through proper indexing and query optimization.

---

8. External Tool Integration

- **Action Item**: **Integrate with External Vulnerability Databases**

- **Objective**: Enhance scanning accuracy through integration with external vulnerability databases (e.g., **CVE**, **NVD**).
- **Discussion Points**:
  - What's the optimal update strategy—periodic syncs or on-demand fetches from external databases?
  - What's our approach for managing outdated or unsupported CVEs?
  - How should we standardize various CVE formats within our tool?

Conclusion

By discussing these development action items with your team, you'll set clear expectations for what needs to be built and allow for efficient development and implementation. Each action item ensures that the product will be secure, scalable, user-friendly, and maintainable.

Low-Fidelity Wireframes

Vulnerability Details - nginx:latest

Bulk Action Dashboard