**Product Requirements Document :**

**1. Introduction**

**1.1 Purpose**

This document outlines the features of a security solution that scans container images for vulnerabilities and provides actionable results. The solution will assist users in identifying and prioritizing vulnerabilities so that they can secure containerized applications in an efficient manner.

**1.2 Stakeholders**

|  |  |  |  |
| --- | --- | --- | --- |
| Security Engineers | DevOps Teams | Software Developers | IT Administrators |

**2. Objectives**

Efficiently provide insights on vulnerable container images. Focus on high and critical vulnerabilities first. Give the ability to take actions on impacted images immediately. Enable use at scale on thousands of images.

**3. Features & Functionality**

**3.1 Core Features**

1. Image Scanning

Image scanning includes the automated scanning of container images and supports multiple registries (DockerHub, AWS ECR, GCR, Azure Container Registry).

2. Vulnerability Assessment

Images will be categorized by CVE - Critical, High, Medium, Low. Identification of CVEs (Common Vulnerabilities and Exposures). Old and vulnerable dependency detection. Outdated dependency identification.

3. Dashboard & Reports

Dashboard features include summary view of all scanned images and an affected images list sorted by rating. Search and filter functionalities.

4. Detailed Findings

Vulnerabilities are reported by images which include: CVE with details like fix availability, Named CVE - detail level fixes present, and recommendations for remediation.

5. Notifications & Alerts

Critical new vulnerabilities will issue alerts. Email and webhook notification integration. Configurable alerts

6. Filtering & Sorting

Sort by id severity, image name, last date scanned. Provide filters to scanned images with available fixes.

7. API & CI/CD Incorporation

Automation via REST API.

Integration into CI/CD for continuous pipeline scanning.

**4. User Roles & Permissions**

* Administrator: Configure settings, manage users, and has full access.
* Developer: Scan results are visible to them, and fixes are implemented.
* Security Analyst: Vulnerabilities are analysed and reports are generated.

1. **Development Action Items**

**5.1 Backend Development**

* Vulnerability scanning is implemented through open-source scanners (Trivy, Clair).
* Scan result retrieval has dedicated API endpoints developed.
* Schema design of the database to house scan history is included**.**

**5.2 Frontend Development**

* Summary of vulnerabilities have detailed dashboard UI viewed and developed.
* Filtering and sorting functionalities are implemented.
* Detailed view design for vulnerability findings.

**5.3 DevOps & Security**

* Scanning for images that are registered need to be scheduled.
* RBAC (Role Based Access Control) needs to be implemented.
* Protection and validation of secure authentication and authorization needs to be ensured.

1. **Wireframes**

**Below are the low-fidelity wireframes for key interfaces:**

**6.1 Dashboard**

* All scanned images are covered in overview.
* Vulnerability distribution pie chart.
* Recent scans with critical finding list.

**6.2 Image List Page**

* Images scanned table with severity markers.
* Functionality includes search and filter.
* Name, date, and severity sorting.

**6.3 Image Details Page**

* Vulnerabilities infractions of an image diverse list.
* CVE details of severity, and recommend fixes.
* Buttons where pressing will lead to patches on issues.

**6.4 Scan Settings Page**

* Frequency with which scans need to be done are set.
* Preference setting types notifications.
* Container registry integrates that are managed.

1. **Solution Approach**

The security product will be built with a microservices architecture to allow for greater scaling and flexibility. The backend of the system will use an open-source scanning tool for vulnerability detection, while the frontend will contain a user-friendly dashboard to display results. Automated scanning will be implemented through a CI/CD pipeline, providing ongoing security monitoring for container images.