Kalila Lin Cen

Cyber Security Analysis Report

LabMan Management System

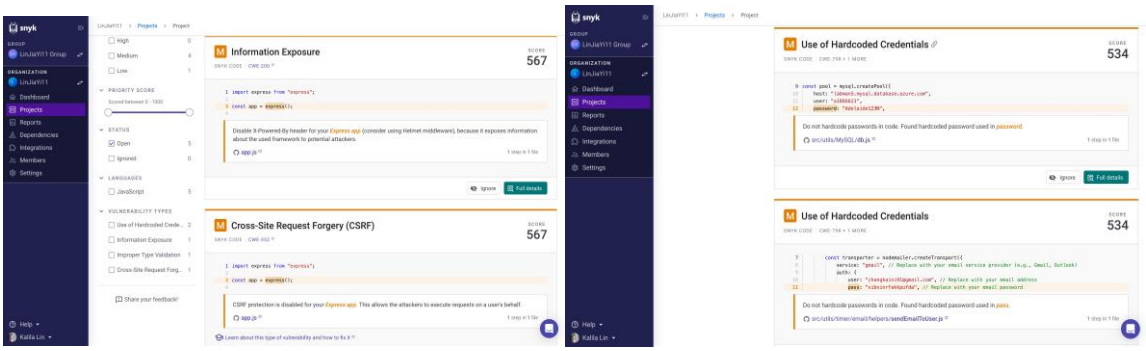
The LabMan project is a web application developed by the University of Adelaide, serving as a laboratory equipment management system for University of Adelaide students and lab managers. The primary purpose of the project is to streamline the administrative process of student equipment requests and ensure timely returns, it was originated due to the process was initially involved paper requests, which led to extended waiting times for approvals. The old process was troublesome, and lab managers faced difficulties in tracking tool usage and ensuring the on-time return of borrowed tools. Therefore, LabMan was designed specifically to streamline the process and resolve the problems mentioned. Its key features include equipment request management to reduce waiting times for students borrowing equipment, borrow and return management, and overdue email alerts. Lab managers can easily track and identify those who haven't returned equipment on time, and the system will automatically send notification emails to alert students.

Static Code Scanning with Snyk

Vulnerabilities	Severity	Vulnerabilities Exploitation	Solution
Information exposure (Low frequency)	Because it simply shows the framework used to potential attackers, the severity is medium.	The attacker can carry out the attack by exploiting known flaws or weaknesses in the framework.	Use a Helmet middleware and disable the X-Powered-By header would be enough.
Cross-Site Request Forgery (CSRF) (Low frequency)	The severity is medium because it allows attackers to perform requests on behalf of users without their agreement, however the app is only for lab tool maintenance owing to its nature.	Attackers can seek the use of certain laboratory tools on behalf of students without their authorisation and never return the tools to the University, putting the students in jeopardy and causing the university to lose pricey laboratory tools.	Enable CSRF protection such as using CSRF Middleware, Implement unique Anti- CSRF Tokens for each user session.
Use of Hardcoded Credentials (Medium frequency)	The severity is medium because the source code poses a high danger of unauthorised system access.	The attacker can utilise the hardcoded credentials to gain access to the system and do the previously specified harmful operations.	Implement a proper access controls and user authentication feature in the app to ensure that authorised user can access to the system.

Kalila Lin Cen

Screenshots Evidence



Static Code Scanning with OX Security

Vulnerabilities	Severity	Vulnerabilities Exploitation	Solution
Code security: Always use a valid certificate, even during testing. (Medium frequency)	The severity is low because there is not much the project is not launch on use yet.	The attacker can intercept, modify, or eavesdrop on data delivered across an insecure connection by taking advantage of the lack of effective certificate validation.	Possibly using valid certificates such as SSL/TLS during testing and development would be better.
Open-source security: eslint@8.35.0 is a JavaScript development dependency having 1 vulnerability. CVE- 2023-26115 (CVSS:7.5, Incorrect Regular Expression) is the most severe vulnerability. (Medium frequency)	The severity is low/info because due to the nature of the project is not launch on use yet.	The vulnerability indicates that an attacker might simply exploit it, resulting in a denial of service, data leakage, or remote code execution.	Because the vulnerability is caused using open-source technology, the only way to remedy it is to find a better replacement for ESLint or to discover updated versions of ESLint that address this vulnerability.
Open-source security: nodemon@2.0.21 is a JavaScript development dependency having 2 vulnerabilities. CVE-2022-25883 (CVSS:7.5, Incorrect Regular Expression) is the most severe vulnerability. (Medium frequency)	The severity is low/info because due to the nature of the project is not launch on use yet.	The vulnerability indicates that an attacker might simply exploit it, resulting in a denial of service, data leakage, or remote code execution.	Because the vulnerability is caused using open-source code, the only way to remedy it is to find a better replacement for nodemon or to upgrade versions of nodemon that address this vulnerability.

Kalila Lin Cen

Screenshots Evidence