OWASP Top 10

Issue Record

Version	Date	Notes
0.1	26/5/21	First draft.

Table of Contents

Security Risks:

1. Injection ✔️

Because I am using JPA, and only using JPA generated queries, I am safe from SQL Injection attacks.

2. Broken Authentication ❌

My application is vulnerable to this risk because there is not password validation preventing well-known, weak or default passwords, The Spring Boot api currently has no protection against brute force attacks.

3. Sensitive Data Exposure ✔️

No sensitive data is sent in plain text, passwords are encrypted using the hashing function BCrypt. BCrypt is good because it salts all the inputs, meaning inputting the same string 5 times would produce 5 different outputs.

4. XML External Entities (XXE) ✔️

Due to the fact that there is no place to upload XML files, and SSO and SAML aren't used either, this risk does not apply meaning my application is safe from external entities.

5. Broken Access Control ❌

My application is at risk because changing the URL of the React application in the browser can open/access components normally not accessible. Once the retrieved JWT Token received from logging in, is used in all requests and components in the React application verify the role of the logged in user, the application should no longer be at risk.

6. Security Misconfiguration ❌

Spring Security comes with many options and configurations, one of which is csrf (Cross-site request forgery) protection. Due to a lack of knowledge on the topic I chose to just disable csrf protection. However, a Cors policy has been configured to only allow requests from my React application.

7. Cross-site Scripting (XSS)✔️

React was made by design to automatically escape XSS therefore my application is safe

8. Insecure Deserialization ❌

Due to the fact that there are not strict constraints during deserialization and no integrity checks such as digital signatures, the api is vulnerable.

9. Components with Known Vulnerabilities ✔️

Since the project is very recent all dependencies, node modules, packages and frameworks are up-to-date. This means that currently the application has no known vulnerabilities in this department, but without a client and server-side dependency checking tool vulnerabilities could arise without me realising or knowing about it.

10. Insufficient Logging & Monitoring ❌

The api has minimal logging, only logging sql queries executed, errors and warnings. With no logging of potentially suspicious activity and only local temporary logs the application is vulnerable.