Security report

**Table of Contents**

1. Introduction
2. Risk Overview
3. Risk Explanation

**Introduction**

This is a Security report that shows the top 10 OWASP risks and how they are tackled in my individual project “Babysita”. For every risk the likelihood, impact, risk, what I have done to prevent them and what could be done is described. For the taken risks there is a small explanation why I have done so (those risks are marked with colors for visual support).

**Risk overview**

A chart with different colored squares

Description automatically generated

**Risk Explanation**

A01:2021-Broken Access Control:

Users should not be able to do modifications to data they do not own. Accessing protected resources without being logged in.

Currently “Babysita” makes use of JWT token that is valid for a short period of time to prevent data stealing incase of token being leaked. Tokens are also created with a secret key. In the BE there are service methods that check if the User can modify a data or if it is an admin. The server accepts request only from localhost:5173.

I think this risk is HIGH because currently the access token is stored in a session which can be easily taken by a hacker (**Vulnerability to Cross-Site Scripting (XSS) Attacks**). Another reason for that is the fact that I have not implemented refresh token because it is out of the scope for the semester.

As a prevention I would want to introduce refresh token and use OAuth such as Google because it is made by professionals and is way more secure. A monitoring system that tracks user activity will be great. I am not planning to implement that due to limited time/technical knowledge.

A02:2021-Cryptographic Failures:

Sensitive data should be protected (passwords). The website should make use of a security certificate and avoid deprecated features in both the FE and BE.

I have implemented a Password hasher and the user data that is being send as a response does not contain sensitive data (in my case the password). I have marked this risk as MODERATE because I make use of HTTP instead of HTTPS. I am taking this risk since the possible solution that I mentioned is out of the scope for this semester.

If I had the technical knowledge, I would have implemented that, so all my server responses are encrypted and harder to access by a hacker.

A03:2021-Injection:  
User-supplied data should be always validated. The SQL can be modified by a hacker to extract data or modify it. Users should be able to make request only from authorized origins.

I have implemented ORM in the server and all the input data is validated by the service classes and by Spring boot beans such as @Email, @NotNull…

A04:2021-Insecure Design:

Data should be validated. The logic flow should be tested. There should be an additional security measures if something continuously fails.

As mentioned, before I have implemented data validation in both FE and BE. Currently my services are almost 100% covered and I have also covered most of the controller’s workflow.

For the future I am planning to implement E2E testing. A way to enhance the current state of the application is to implement a Login attempt tries and a time delay. This risked is marked as Low because I do not expect problems due to my validation.

A05:2021-Security Misconfiguration:

Passwords should be encrypted, there should be a security handling, the software is up to date and does not use deprecated features. Users do not have administrative rights in the database.

I cannot do much about this risk that is why I take it. If I could make my system prone to this risk, I would have created a user account in the database that the applications user will make use of to prevent table/database creation or dropping. I would change from developing environment to production one. The risk is HIGH because I am completely unprotected. This is since some of the requirements for semester 3 have deprecated features and they are mandatory so I cannot avoid that, as for the database user account it will require me to modify some of my application properties which may lead to unexpected bugs that I might find too late and could result into a penalty for my final grade.

A06:2021-Vulnerable and Outdated Components:

The system should be monitored for deprecated features and components/frameworks.

I think this risk is LOW because I use the Spring boot lates versions. I also make use of SonarQube which keeps track of my code quality.

As a prevention measure, I can only think about regularly checking the dependencies I use for their lates versions.

A07:2021-Identification and Authentication Failures:

This risk is related to the authentication part of the application and what the user can do.

Currently I have implemented a password hashing and password complexity in the FE. My JWT token is sored in the browser session. I have marked this risk as MODERATE since the token can be stolen from the session quickly and the fact that I do not check for password complexity on the BE. I take this risk for the token due to my tech knowledge limitation and as for the password complexity and login attempts is because of time limitations (I do not want to risk having bugs in the last Sprint).

For the future I would implement that in the BE and add a Login attempts for the Users.

A08:2021-Software and Data Integrity Failures:

The application should use a secure libraries and dependencies (such as Gradle or Maven in my case). The CI/CD pipeline should not provide sensitive data.

This risk is Low for me because I make use of Gradle. As for my pipeline all the sensitive data is protected by the GitLab variables.

As additional measures I would love to have a monitoring system that validates data integrity.

A09:2021-Security Logging and Monitoring Failures:

The application should have a Monitoring system that checks for suspicious activities. A good add would be to establish or adopt an incident response and recovery plan in case of being hacked or the server going down.

This risk for me is HIGH because I cannot do anything for this risk due to lack of technical knowledge. If I could have implemented such system and plan.

A10:2021-Server-Side Request Forgery:

Web application should not fetch a remote resource without validating the user-supplied URL because of crafted request to an unexpected destination.

This risk is Low for “Babysita” since I do not use external API.

**Conclusion:**

The current state of the project is solid for the semester 3 needs. I have implemented some additional measures such as integration tests and I have almost full Unit test coverage (planning to do E2E testing). My pipeline is secured. However, if I had the time and technical knowledge, I would want to implement a refresh token, OAuth and make use of HTTPS so I can have a production ready application.

Overall, I think I did a decent job for this semester requirements.