Trifonov,Dobri D.V.

dobri.trifono1@gmail.com

owasp top 10 security risks

And how to I prevent them

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Likelihood | Impact | Risk | Actions Possible | Planned |
| A1: Broken Access Control | LOW | HIGH | MEDIUM | Security, Roles, Implement RBAC | YES |
| A2: Cryptographic  Failures | MEDIUM | LOW | LOW | N/A | NO |
| A3: Injection | MEDIUM | HIGH | HIGH | Front-End Regex | NO |
| A4: Insecure Design | MEDIUM | LOW | LOW | Sufficient Unit and Integration tests | YES |
| A5: Security Misconfiguration | LOW | LOW | Note | N/A | YES |
| A6: Vulnerable and Outdated Components | LOW | LOW | Note | Remove all unused dependencies, components, files | YES |
| A7: Identification and Authentication Failures | MEDIUM | MEDIUM | MEDIUM | Implement Front-End Regex credential check, Server side credential check | NO |
| A8: Software and Data Integrity Failures | HIGH | HIGH | CRITICAL | Proper CI/CD deployment | YES |
| A9: Security Logging and Monitoring Failures | MEDIUM | MEDIUM | MEDIUM | Setup software to insure that suspicious activities are detected | NO |
| A10: Server-Side Request Forgery | MEDIUM | HIGH | HIGH | Validate All client-supplied data | NO |

# Broken Access Control

To ensure that a user’s access privileges aren’t tempered with I have included a JWT token. The token keeps the user’s authorities in a claim. The REST API check the token on every API requests, if the token has been tampered with the request will be denied.

# Cryptographic Failures

My application does not use a lot of sensitive information. Personal information such as email is never sent to the front end and passwords are properly encrypted. The only thing that is passed around is the logged-in’s user public username.

# Injection

Hibernate prevents DBMS specific syntax, but it is possible to break out of the HQL syntax and exploit DBMS functions.

# Insecure Design

My application is made to be used for personal/individual use. There isn’t any way that a user could exploit the design of my application since all user content (except credentials) will be publicly available. The only thing that I need to ensure is that a user

# Security Misconfiguration

I have ensured that spring-security is properly implemented and configured

# Vulnerable and Outdated Components

From my project I have removed all unused dependencies, node-modules to ensure that there is a minimal chance for a exploitable component vulnerabilities to occur.

# Identification and Authentication Failures

To prevent this there should be implementation of a multi-factor authentication to prevent automated credential stuffing, brute force, and stolen credential reuse attacks. I will try to improve upon this since it is lacking in my project.

# Software and Data Integrity Failures

In my project I have integrated Continuous Integration and Continuous Deployment with minimal configuration that could be introduced into my pipeline

# Security Logging and Monitoring Failures

I have not added logging into my App.

# Server-Side Request Forgery

All client-supplied data will be sanitized in the final version of my application

# Conclusion

In conclusion my app is decently secured at the moment, but a lot of improvements could be made and I’m planning to make them from RBAC to input validation I hope I manage to do all of them before the final version.