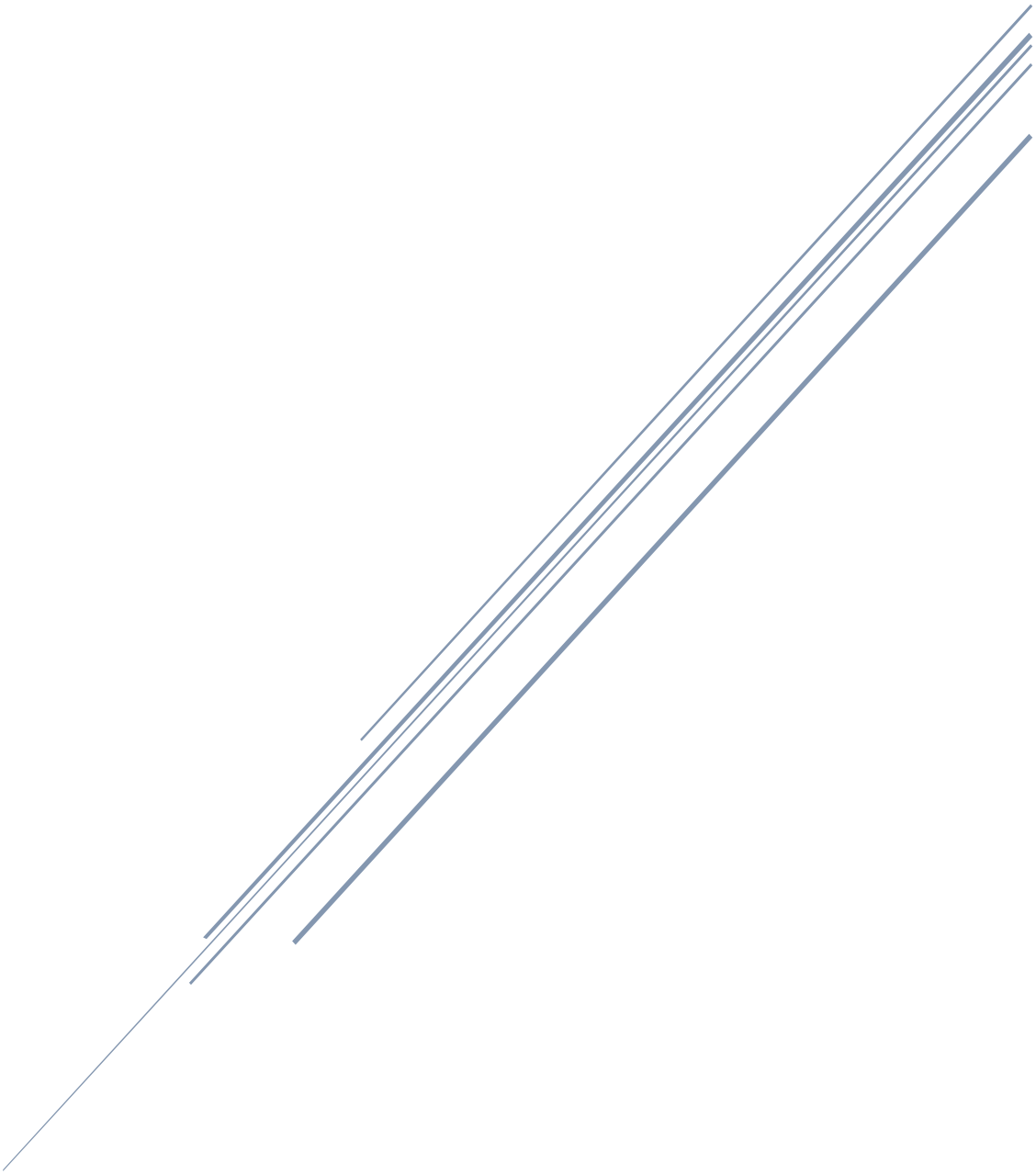
OWASP Report

Safar Project



Name: Omar Abou Dehn

Contents

Introduction: 2

Project Security Analysis: 2

A1: Broken Access Control: 3

A2: Cryptographic Failure: 3

A3: Injection: 3

A4: Insecure Design: 3

A5: Security Misconfiguration: 3

A6: Vulnerable and Outdated Components: 4

A7: Identification and Authentication Failures 4

A8: Software and Data Integrity Failures: 4

A9: Security Logging and Monitoring Failures: 4

A10: Server-Side Request Forgery: 5

Conclusion: 5

References 5

Introduction:

This security document will cover the Open Web Application Security Project (OWASP), which helps to improve software security, and how Safar is secured to prevent the relevant foreseeable vulnerabilities, as well as how it might be improved further.

Project Security Analysis:

	Likelihood	Impact	Risk	Actions possible	Planned
A1: broken access control	Low	High	Low	N/A, fixed	No
A2: Cryptographic failure	Moderate	Moderate	Moderate	Using Https instead of Http	No
A3: Injection	Low	High	Low	N/A	No
A4: Insecure Design	Low	High	Low	Better secret management (connection string of DB)	Yes
A5: Security Misconfiguration	Low	Moderate	Low	Custom Errors notifying the users without revealing sensitive info	No
A6: Vulnerable and Outdated Components	Low	Moderate	Low	Automated vulnerability scanning	No
A7: Identification and Authentication Failures	Moderate	Moderate	Moderate	Check for password complexity	No
A8: Software and Data Integrity Failures	Low	High	Low	N/A	No
A9: Security Logging and Monitoring Failures	High	Moderate	High	Error monitoring and logging implementation	No
A10: Server-Side Request Forgery	Low	High	Low	N/A	No

A1: Broken Access Control:

Broken access control is the risk of failure or vulnerability in the authentication and authorization system, for example, a user performing actions that only administrators are permitted to perform, or a non-registered user (guest) having access to resources that should not be available to him.

This could have a big impact as it might lead to unauthorized information disclosure, modification, or destruction of data.

However, the likelihood of this happening in the project is very low, due to the fact that SpringBoot Security and JWT was used to handle authentication and authorization.

A2: Cryptographic Failure:

Cryptographic failure is about transiting and storing data insecurely, such as financial data, personal information, health records, or any data that falls under privacy laws.

The impact of this risk is usually high, but in relation to Safar project it is moderate, as the amount of sensitive data being stored is limited.

The likelihood of this happening is moderate, because although the password is being encoded before it is stored in the database and is never retrieved, it is being sent as plain text for the registration and login, so someone might intercept the request and have access to the sensitive data, this can be further secured using HTTPs protocol.

A3: Injection:

SQL injection is the insertion of malicious code into SQL statements. It typically occurs when you ask a user for input, such as their username/user-id, and instead of a name/id, the user provides you with a SQL statement that you will inadvertently run on your database.

This could have a huge impact as it might lead to loss/exposure of data or to users acting outside of their permitted authorities.

The likelihood of this happening in Safar is low, because no user input is being directly inserted into an SQL query, because parameterized queries are being used, and on top of that the data is being validated in the frontend and backend.

A4: Insecure Design:

Insecure design flaws occur when developers, quality assurance (QA), and/or security teams fail to foresee and analyze threats during the code design process.

The impact of this risk is high, such application security flaws would allow attackers to assume legitimate user accounts and obtain unauthorized access to password-protected resources for deeper system exploitation.

The likelihood of this happening in project is low, because the secret keys are stored safely and are not pushed to version control system.

A5: Security Misconfiguration:

Security misconfiguration happens when security settings are not properly set during the configuration process or deployed and maintained with defaults settings.

Common misconfiguration vulnerabilities arise from the use of default passwords, open database instances, error messages revealing sensitive information, directory listing enabled, default certificates, misconfigured cloud settings.

The likelihood of this happening is low, as there is no usage of default passwords and directory listing is disabled.

A6: Vulnerable and Outdated Components:

Component-based vulnerabilities occur when a software component is unsupported, out of date, or vulnerable to a known exploit.

The impact of this may vary based on the vulnerability, and it may result in accidental or intentional errors.

The likelihood of this accruing is low, as the latest versions of the technologies and libraries are being used.

A7: Identification and Authentication Failures

Exploiting a broken authentication, an attack is typically initiated by taking advantage of poorly managed credentials and login sessions to masquerade as authenticated users.

The impact of this is moderate, as it my exposure of sensitive data on a large scale

The likelihood of this happening is moderate, because although the passwords are encoded before they are saved, their complexity is not being checked.
Moreover, JWT is being used with an expiration date, so even if it got exposed, it will be only for a limited time.

A8: Software and Data Integrity Failures:

An application that relies on plugins, libraries, or modules from unverified and untrusted sources, repositories, or content delivery networks (CDNs) may be exposed to such a type of failure.

The impact of this may vary based on the vulnerability, it can introduce the potential for unauthorized access malicious code or system compromise.

The likelihood of this happening in the project is low, as all libraries and plugins being used are up to date and from trusted parties.

A9: Security Logging and Monitoring Failures:

Without logging and monitoring, breaches cannot be detected. Insufficient logging, detection, monitoring, and active response occurs any time.

This has a moderate impact, as it makes it more difficult to detect and tackle breaches and errors.

The likelihood of this happening in Safar is high, as no logging/monitoring functionality has been implemented.

A10: Server-Side Request Forgery:

Those issues arise when a web application does not validate the user-supplied URL when fetching a remote resource. This enables attackers to force the application to send a crafted request to an unexpected destination.

(Owasp Top 10, 2021)

(OWASP Top 10 – The Ultimate Vulnerability Guide, 2021)

Conclusion:

Safar Application is secure, and it can be further secured by for example doing error logging and monitoring, checking for password complexity and better secret management.

References

OWASP Top 10 – The Ultimate Vulnerability Guide. (2021). From Crash Test Security: https://crashtest-security.com/owasp-top-10-2021/

Owasp Top 10. (2021). From Owasp: https://owasp.org/Top10/