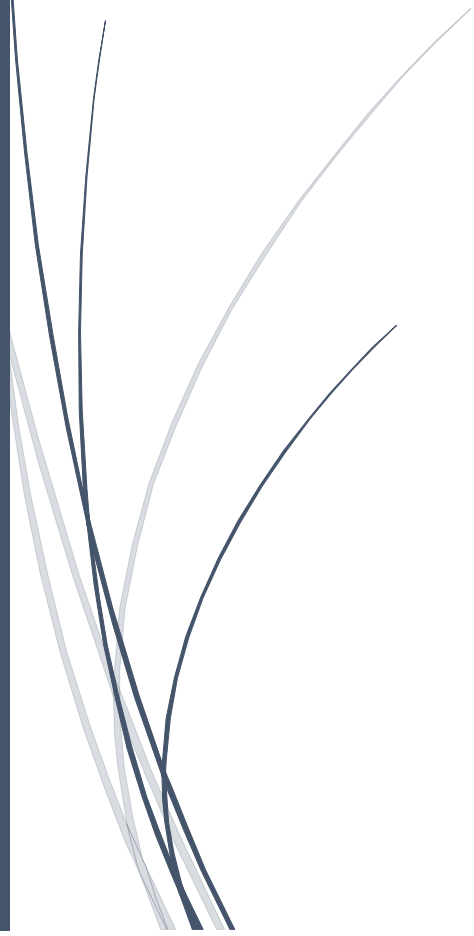
1/13/2022

OWASP Report

belongs to project: Margarita



Stoychev,Stoycho S.D.– 4292723
Class: S3-CB-01

Table of Contents

Diagram

	Likelihood	Impact	Risk
A01 -Broken Access Control	High	Medium	HIGH
A02:2021-Cryptographic Failures	Low	Medium	LOW
A03 -Injection	Medium	High	HIGH
A04 -Insecure Design	High	High	CRITICAL
A05 -Security Misconfiguration	High	Low	MEDIUM
A06 -Vulnerable and Outdated Components	Medium	Low	LOW
A07 -Identification and Authentication Failures	Medium	Low	LOW
A08 -Software and Data Integrity Failures	Low	High	MEDIUM
A09 -Security Logging and Monitoring Failures	Medium	Medium	MEDIUM
A10 -Server-Side Request Forgery	High	High	CRITICAL

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

Likelihood and Impact Levels	
0 to <3	LOW
3 to <6	MEDIUM
6 to 9	HIGH

OWASP Top 10

A01 -Broken Access Control

In this project

The project is using JWT/JSON Web Tokens/ which are blocking unauthorized access to any valuable data. Only authorized requests can do changes to the data.

To keep in mind

In this project, we are manually giving access to the different roles /Admin, Employee, User/. We should always be careful to not give any more access to the users like changing the products!

A02:2021-Cryptographic Failures

In this project

There are two places where cryptographic failure can be an issue.

User's password

For encrypting the password of the user, we are using

BCryptPasswordEncoder

Bcrypt has provided adequate security for a very long time because it was designed to be adaptable by providing a flexible key setup that could be adjusted to make the algorithm harder to crack (to keep up with hackers.

JWT/JSON Web Tokens/

For encrypting the secret key of the JWT we are using Algorithm HMAC256.

To keep in mind

Every year to check for better and stronger encryptions.

Currently, the JWT secret key is never changed. A good addition will be to repeatedly change the secret! The idea is to change the secret key of the JWT every 20h to prevent someone from breaking it and taking unlimited control over the software.

A03 -Injection

In this project

In our case, the front side of the project cannot detect injections, but the backend server is using Parameterized Queries which will prevent them.

To keep in mind

However, there is still what needs to be done. We should not allow any special characters and use LIMIT and other SQL controls within queries to prevent mass disclosure of records in case of SQL injection.

A04 -Insecure Design

In this project

In this project, we have an insecure design!

Example: Every registered user can create as many orders as he wants because they are supposed to pay on delivery. This may become a big issue because the employee role in the company will have to verify that every new order is real, and the person will pay for the product/s.

To keep in mind

The example above should be discussed with all stakeholders and in the end, come up with the best solution for people making orders they will not pay. This is a big issue because the products, the company sells, have a low expiration date.

A05 -Security Misconfiguration

In this project

With the lack of security knowledge of the developer in this project, it is not looked deep into possible security misconfiguration.

To keep in mind

When we update the systems check if the security features are enabled.

WHEN GOING INTO PRODUCTION ALLAYS DELETE THE DEFAULT ACCOUNTS!

A06 -Vulnerable and Outdated Components

In this project

We are not keeping track of our software versions. This may become a serious problem if a component is too outdated. But when we update a component, we always check the compatibility.

To keep in mind

An additional feature that can be added in the future is to subscribe to security bulletins related to the components we use.

A07 -Identification and Authentication Failures

In this project

The frontend and the backend are checking for too short passwords. This is preventing the users from inputting too weak passwords.

To keep in mind

A feature to add is when the user is registering or changing his password. The password should be checked and if it is in a list of the top 10,000 worst passwords to require the user to write a more complex password.

A08 -Software and Data Integrity Failures

In this project

We are always taking our software libraries, plugins, etc. from the original source.

To keep in mind

When we integrate new software, library, plugin, or code we should check for its identity or possible malware!

A09 -Security Logging and Monitoring Failures

In this project

In our software, we are not logging or monitoring. This may be a problem in the future, it is already a user requirement for the next delivery.

To keep in mind

Logging and monitoring are really important parts of any big application. It can identify a bridge in the beginning and prevent it.

A10 -Server-Side Request Forgery

In this project

In this project, the only place where the SSRF can happen is when the admin is adding a new product, he should upload a picture. The problem is that any file will be accepted and saved in the backends' filesystem.

To keep in mind

This is a big issue because if someone gains access to the admin's account, he can upload his software directly to the server.