OWASP Security Report

Stefan-Nikola Stanev

Application: Blackjack royal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Likelihood | Impact | Risk | Actions possible | Planned |
| A1:  Broker access control | High | Severe | High | N/A | Yes |
| A2: Cryptoghraphic failure | Very Unlikely | Severe | Low | Password Encryption | Yes |
| A3:  Injection | Very Unlikely | Moderate | Low | JPA Repository used | Yes |
| A4:  Insecure Design | Moderate | Severe | Moderate | More testing | No |
| A5:  Security Misconfiguration | Unlikely | High | High | CORS should be done correctly | Yes |
| A6:  Vulnerable and Outdated Components | Moderate | High | High | Either Docker usage or manually updating libraries | No |
| A7: Identification and Authentication Failures | Likely | High | High | multi-factor authentication | No |
| A8:  Software and Data Integrity Failure | Unlikely | High | Moderate | Ensure libraries and dependencies are consuming trusted repositories | No |
| A9:  Security Logging and Monitoring | High | Low | High | Ensure log data is encoded correctly to prevent injections or attacks on the logging or monitoring systems | No |
| A10:  SSRF | Unlikely | Severe | High | Using CORS to use specific urls | Yes |

Context explanations:

* A1: JWT user authentication is used with different roles and the user’s sensitive information is never stored in the JWT claims.
* A2: Passwords use SHA256 hashing and salting method that removes the risk of people getting their hands on other people’s passwords.
* A3: JPARepository ORM is used which makes it impossible for SQL injections and their malicious scripts.
* A4: More testing is required and HTTPS would be better since it’s more secured than HTTP.
* A5: CORS configurations allow only specific endpoints.
* A6: Using outdated or vulnerable components within the application might lead to exploitable vulnerabilities, so the code and the dependencies need to be constantly updated. One fix could be to use Docker containers since they work with the version they are set on one version only if the container is not deleted and created anew.
* A7: System only requires the password to be at least 8 characters long, so choosing a strong password is a responsibility given to the user that uses the website.
* A8: No sensitive information should be extracted by the normal user’s interaction to the system.
* A9: No matter what the error could be, there should be no way to extract sensitive information with injections or other means.
* A10: CORS allows only urls that are approved to be able to interact with the system.

Conclusion:  
The security assessment conducted on the "Blackjack Royal" application reveals a robust framework with several established security measures. Each identified risk has been contextualized with corresponding mitigations already in place. Despite these commendable security measures, there exist areas for potential enhancement.

Here are a few of them:

1. Ongoing Testing (A4): Prioritizing extensive testing and migrating to HTTPS could elevate the application's security posture further, reducing vulnerabilities associated with insecure design.
2. Vigilance in Updates (A6): Continual vigilance in updating code and dependencies, especially through Docker containers, stands as a key factor in fortifying against evolving threats.
3. Expanded Security Monitoring (A9): Enhancing security logging and monitoring systems could provide comprehensive oversight, fortifying against unforeseen attacks or exploitation attempts.