# A close-up of a logo Description automatically generated

# Security Report: OWASP Top 10 Security Risks Assessment

### This report provides an analysis of how our application deals with the OWASP Top 10 security risks.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Security Risk | Likelihood | Impact | Risk | Current Mitigation Actions |
| A01:  Broken Access Control | Very Likely | Severe | High | Implemented CORS, Authentication via Outlook, and AuthProvider (checks for token identity) |
| A02:  Cryptographic Failures | Likely | Severe | Moderate | Oulook API for the user account data, we don’t make use of HTTPS |
| A03:  Injection | Unlikely | Severe | Low | No-SQL database (MongoDB) |
| A04:  Insecure Design | Likely | Severe | Moderate | Unit Test and Integration tests implemented, stick to SOLID, Outlook Authentication |
| A05:  Security Misconfiguration | Unlikely | Moderate | Low | Express and React libraries, don’t send security headers, no use of deprecated features |
| A06:  Vulnerable and Outdated Components | Unlikely | Moderate | Low | No use of deprecated features and NPM as module manager |
| A07: Identification and Authentication Failures | Likely | Severe | Moderate | Outlook API for authentication (don’t use sessions, routes not protected), token is stored in local storage and currently we don’t have roles |
| A08:  Software and Data Integrity Failures | Likely | Severe | Moderate to High | NPM is consuming trusted repositories (Open-Sourced packages), no use of CI/CD pipeline |
| A09:  Security Logging and Monitoring Failures | Very Likely | Severe | High | Risk taken due to limited technical knowledge |
| A10:  Server-Side Request Forgery | Likely | Severe | Moderate | React framework takes care of HTTP redirect, no encrypted responses, use of CORS for localhost |

# Explanation

# References

<https://owasp.org/Top10/>