

Week 3 Report – Advanced Security & Final Reporting

Application: User Management System (Express + MongoDB) **Prepared by:** Muhammad Raza **Date:** February 25, 2026

1. Executive Summary

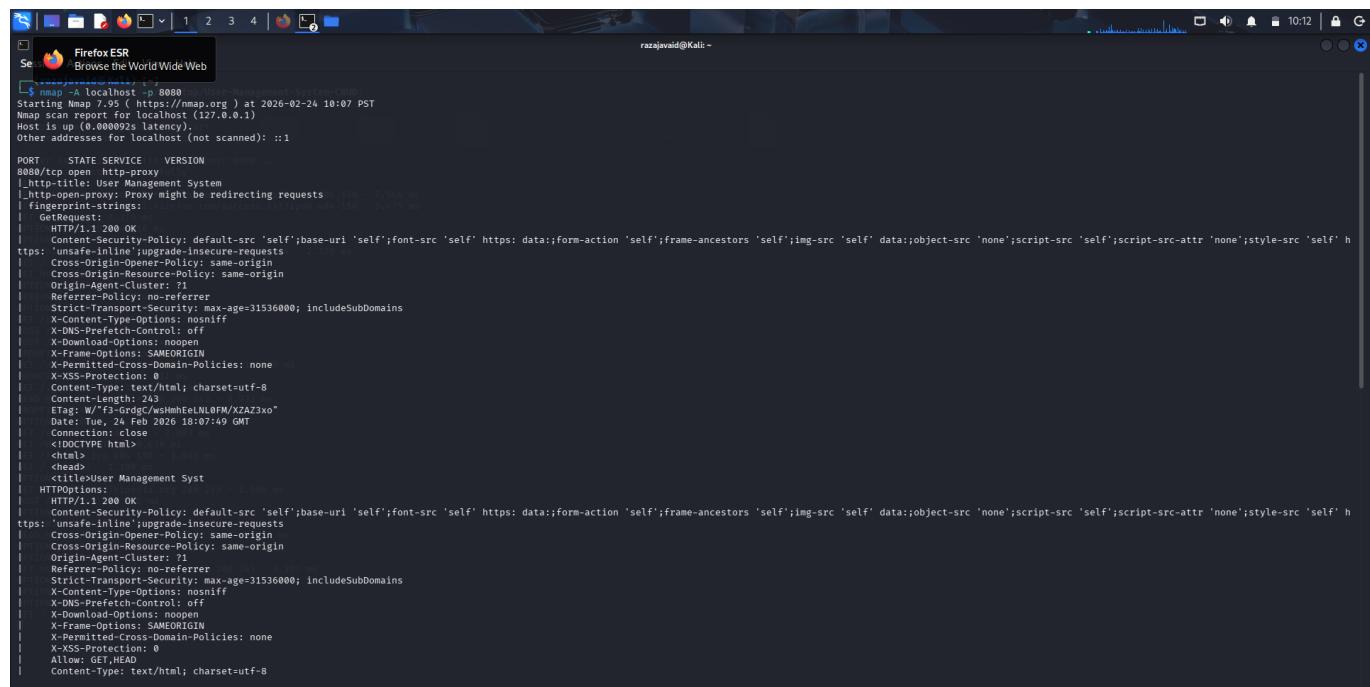
In Week 3 the team concentrated on advanced validation, penetration testing, logging/monitoring and final documentation. Key deliverables included:

- Penetration testing (network and browser-based).
- Winston logging integration.
- Security checklist and repository clean-up.
- Formal reports and evidence packaging.

Area	Outcome
Network scan	Nmap confirmed only port 8080 open; Helmet headers present
Browser tests	XSS & NoSQL injection blocked; password policy enforced
Logging	Requests & errors recorded in security.log
Documentation	Checklist and three reports completed

2. Penetration Testing

2.1 Nmap Scan



```

$ nmap -A localhost -p 8080
Starting Nmap 7.95 ( https://nmap.org ) at 2026-02-24 10:07 PST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000s latency).
Other addresses for localhost (not scanned): ::1

PORT      STATE SERVICE VERSION
8080/tcp   open  http    proxy
|_http-title: User Management System
| http-open-proxy: Proxy might be redirecting requests
| fingerprint-strings:
|_ GetRequest:
HTTP/1.1 200 OK
Content-Security-Policy: default-src 'self';base-uri 'self';font-src 'self' https: data:;form-action 'self';frame-ancestors 'self';img-src 'self' data:;object-src 'none';script-src 'self';script-src-attr 'none';style-src 'self' h
https: 'unsafe-inline';upgrade-insecure-requests
|_Cross-Origin-Opener-Policy: same-origin
|_Cross-Origin-Resource-Policy: same-origin
|_Origin-Agent-Cluster: ?!
|_Referer-Policy: no-referrer
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-DNS-Prefetch-Control: off
X-Download-Options: noopener
X-Frame-Options: SAMEORIGIN
X-Permitted-Cross-Domain-Policies: none
X-XSS-Protection: 0
Content-Type: text/html; charset=utf-8
Content-Length: 243
Etag: W/"7e4a-0000000000000000"
Date: Fri, 24 Feb 2026 10:07:49 GMT
Connection: close
<!DOCTYPE html>
|_html>
|_head>
|<title>User Management Syst
| HTTPOptions:
HTTP/1.1 200 OK
| Content-Security-Policy: default-src 'self';base-uri 'self';font-src 'self' https: data:;form-action 'self';frame-ancestors 'self';img-src 'self' data:;object-src 'none';script-src 'self';script-src-attr 'none';style-src 'self' h
https: 'unsafe-inline';upgrade-insecure-requests
|_Cross-Origin-Opener-Policy: same-origin
|_Cross-Origin-Resource-Policy: same-origin
|_Origin-Agent-Cluster: ?!
|_Referer-Policy: no-referrer
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-DNS-Prefetch-Control: off
X-Download-Options: noopener
X-Frame-Options: SAMEORIGIN
| X-Permitted-Cross-Domain-Policies: none
| X-XSS-Protection: 0
Allow: GET,HEAD
Content-Type: text/html; charset=utf-8

```

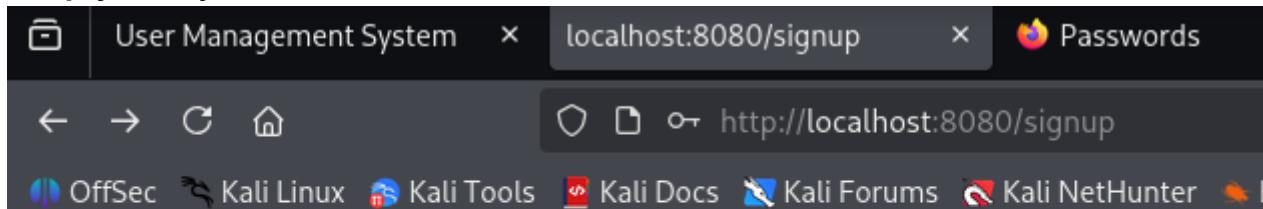
- Port **8080** open serving the application.

- Headers detected: Content-Security-Policy, Strict-Transport-Security, X-Frame-Options, etc.
- Confirms Helmet middleware is active and only the intended service is exposed.

2.2 Browser-Based Testing

The screenshots are now displayed inline with descriptions to improve readability:

- **XSS payload injection**

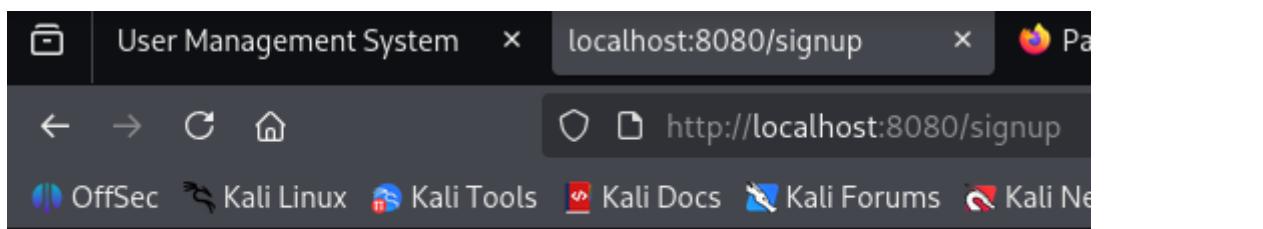


Signup

<script>alert('XSS')</script> test@gmail.com ***** Signup

The input field containing <script>alert('XSS')</script> was rejected, so no script execution occurred.

- **Invalid name characters**

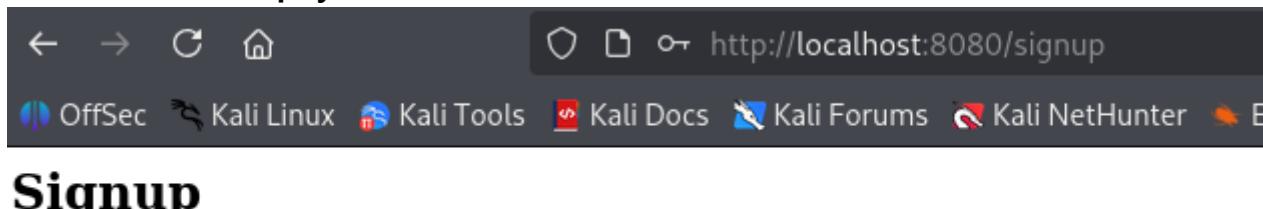


Invalid name

Submitting a

name with prohibited symbols produced an *Invalid name* error.

- **Weak credentials displayed**

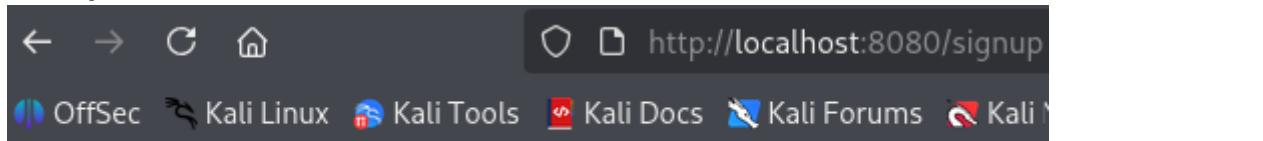


Signup

test321 test321@gmail.com ***** Signup

The browser warned that the supplied password was weak.

- **Short password submission**



Password must be at least 8 characters

Trying to register

with 12345 triggered the message *Password must be at least 8 characters*.

- **Correct signup form**

A screenshot of a web browser window. The address bar shows "http://localhost:8080/signup". Below the address bar is a navigation bar with links to OffSec, Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, and Exploit. The main content area has three input fields: "success", "success@gmail.com", and a password field filled with "*****". A "Signup" button is visible to the right.

Signup

The form populated as expected with valid values.

A screenshot of a web browser window. The address bar shows "http://localhost:8080/signup". Below the address bar is a navigation bar with links to OffSec, Kali Linux, Kali Tools, Kali Docs, Kali Forums, and Kali NetHunter. The main content area displays the message "Signup successful: success".

- **Successful signup**

The confirmation displayed *Signup successful: success* after valid input.

- **NoSQL injection attempt in console**

A screenshot of a web browser window. The address bar shows "http://localhost:8080/login". Below the address bar is a navigation bar with links to OffSec, Kali Linux, Kali Tools, Kali Docs, Kali Forums, and Kali NetHunter. The main content area has two input fields: "Email" and "Password", and a "Login" button.

Login

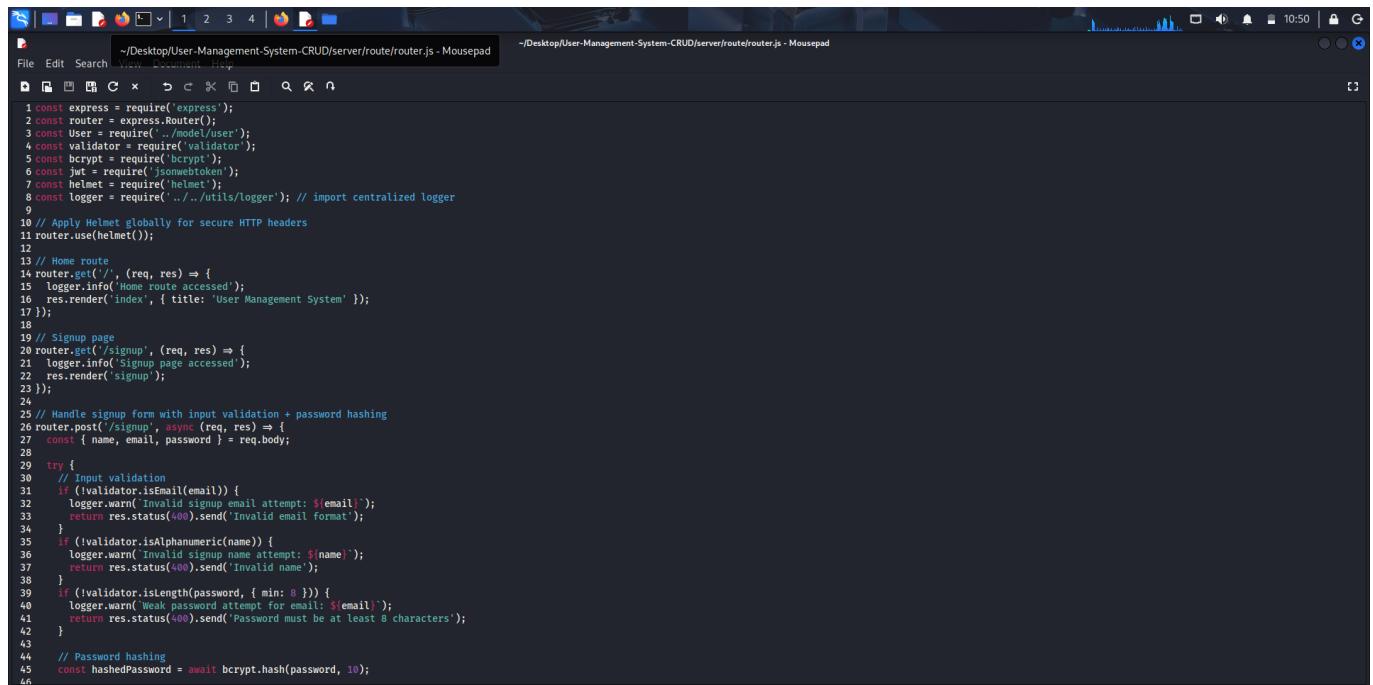
A screenshot of a browser developer tools interface, specifically the "Console" tab. The console output shows a crafted POST request to "http://localhost:8080/login" using the "JSON.stringify" method to encode the payload. The response status is "500 Internal Server Error".

Headers	Cookies	Request	Response	Timings	Stack Trace
Filter Headers					
▶ POST http://localhost:8080/login					
Status	500 Internal Server Error (?)				
Version	HTTP/1.1				
Transferred	923 B (36 B size)				
Referrer Policy	no-referrer				
DNS Resolution	System				

A crafted payload caused a 500 error, indicating the server rejects malformed query objects.

Each screenshot confirms that input validation, CSP, and sanitisation defend against common browser-based attacks.

3. Logging & Monitoring

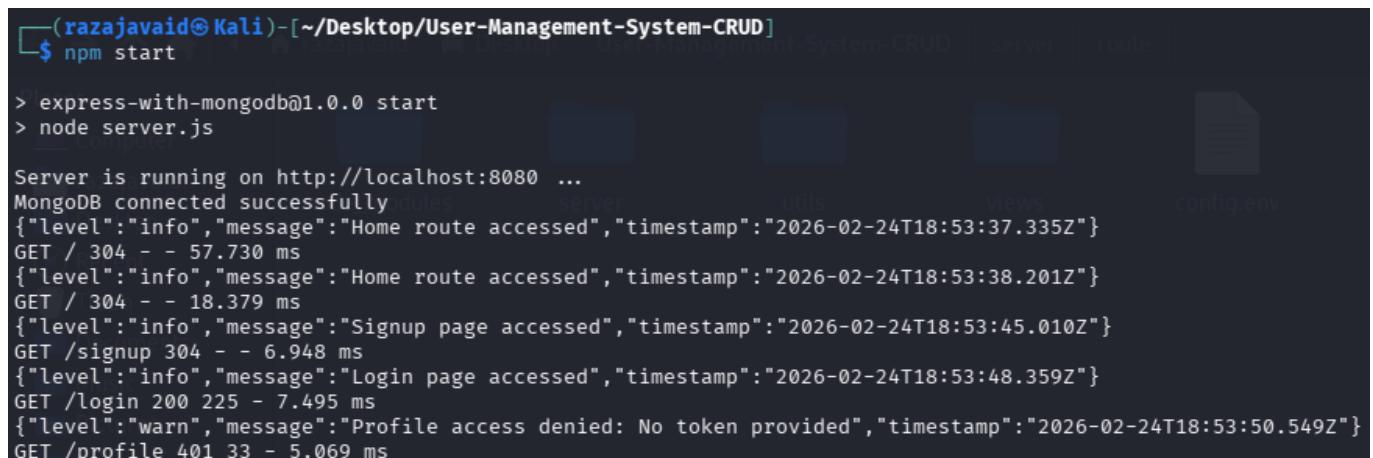


```

1 const express = require('express');
2 const router = express.Router();
3 const User = require('../model/user');
4 const validator = require('validator');
5 const bcrypt = require('bcrypt');
6 const jwt = require('jsonwebtoken');
7 const helmet = require('helmet');
8 const logger = require('../utils/logger'); // import centralized logger
9
10 // Apply Helmet globally for secure HTTP headers
11 router.use(helmet());
12
13 // Home route
14 router.get('/', (req, res) => {
15   logger.info('Home route accessed');
16   res.render('index', { title: 'User Management System' });
17 });
18
19 // Signup page
20 router.get('/signup', (req, res) => {
21   logger.info('Signup page accessed');
22   res.render('signup');
23 });
24
25 // Handle signup form with input validation + password hashing
26 router.post('/signup', async (req, res) => {
27   const { name, email, password } = req.body;
28
29   try {
30     // Input validation
31     if (!validator.isEmail(email)) {
32       logger.warn(`Invalid signup email attempt: ${email}`);
33       return res.status(400).send('Invalid email format');
34     }
35     if (!validator.isAlphanumeric(name)) {
36       logger.warn(`Invalid signup name attempt: ${name}`);
37       return res.status(400).send('Invalid name');
38     }
39     if (!validator.minLength(password, { min: 8 })) {
40       logger.warn(`Weak password attempt for email: ${email}`);
41       return res.status(400).send('Password must be at least 8 characters');
42     }
43
44     // Password hashing
45     const hashedPassword = await bcrypt.hash(password, 10);
46   } catch (error) {
47     logger.error(error.message);
48     return res.status(500).send('Internal Server Error');
49   }
50
51   res.redirect('/profile');
52 });

```

- Winston added to `router.js` with log levels for routes, signup/login and errors.



```

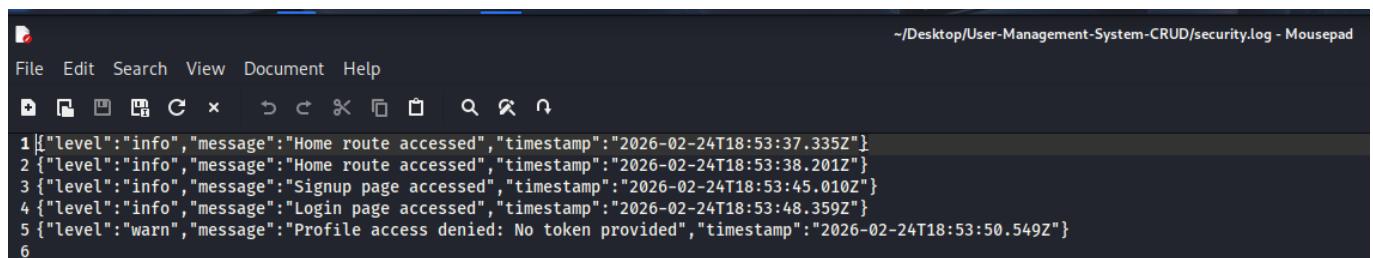
razajavaid@Kali:[~/Desktop/User-Management-System-CRUD]
$ npm start

> express-with-mongodb@1.0.0 start
> node server.js

Server is running on http://localhost:8080 ...
MongoDB connected successfully
{"level": "info", "message": "Home route accessed", "timestamp": "2026-02-24T18:53:37.335Z"}
GET / 304 - - 57.730 ms
{"level": "info", "message": "Home route accessed", "timestamp": "2026-02-24T18:53:38.201Z"}
GET / 304 - - 18.379 ms
{"level": "info", "message": "Signup page accessed", "timestamp": "2026-02-24T18:53:45.010Z"}
GET /signup 304 - - 6.948 ms
{"level": "info", "message": "Login page accessed", "timestamp": "2026-02-24T18:53:48.359Z"}
GET /login 200 225 - 7.495 ms
{"level": "warn", "message": "Profile access denied: No token provided", "timestamp": "2026-02-24T18:53:50.549Z"}
GET /profile 401 33 - 5.069 ms

```

- Terminal output showing messages such as *Home route accessed* and *Profile access denied: No token provided*.



```

1 {"level": "info", "message": "Home route accessed", "timestamp": "2026-02-24T18:53:37.335Z"}
2 {"level": "info", "message": "Home route accessed", "timestamp": "2026-02-24T18:53:38.201Z"}
3 {"level": "info", "message": "Signup page accessed", "timestamp": "2026-02-24T18:53:45.010Z"}
4 {"level": "info", "message": "Login page accessed", "timestamp": "2026-02-24T18:53:48.359Z"}
5 {"level": "warn", "message": "Profile access denied: No token provided", "timestamp": "2026-02-24T18:53:50.549Z"}
6

```

- `security.log` stores structured JSON events with timestamps, levels (`info`, `warn`) and descriptive messages.

- Provides an audit trail for suspicious activity.
-

4. Security Checklist

```

Session Actions Edit View Help
GNU nano 8.6
Security_Checklist.md

razajavaid@Kali: ~/Desktop/User-Management-System-CRUD

Security Checklist

 **Input Validation**
- All user inputs validated with `validator` library.
- Rejects malformed emails, names, and weak passwords.

 **Password Security**
- Passwords hashed with `bcrypt` before storage.
- Minimum length enforced (8 characters).
- Salting applied automatically by bcrypt.

 **Authentication**
- JWT tokens used for login sessions.
- Tokens expire after 1 hour.
- Protected routes require valid token.

 **Secure HTTP Headers**
- `helmet` middleware applied globally.
- Headers enforced: Content-Security-Policy, X-Frame-Options, Strict-Transport-Security, X-Content-Type-Options.

 **Data Transmission**
- HTTPS recommended for deployment.
- No sensitive data sent in plaintext.

 **Logging & Monitoring**
- Winston logger configured.
- Logs written to console and `security.log`.
- Records login attempts, errors, and suspicious activity.

 **Penetration Testing**
- Nmap scan confirms only port 8080 open.
- Browser-based tests (XSS, NoSQL injection, weak password) blocked successfully.

 **Documentation**
- Week1_Report.pdf → Vulnerability Assessment.
- Week2_Report.pdf → Security Fixes.
- Week3_Report.pdf → Penetration Testing & Final Reporting.
- Security_Checklist.md → Best practices summary.

```

The markdown checklist covers:

- Input validation & output encoding
- Password hashing & strength rules
- Authentication & authorization
- Secure headers (Helmet) and HTTPS
- Logging & monitoring
- Penetration testing procedures
- Documentation & reporting

A table of the main checklist items is included in the [Security_Checklist.md](#) file.

5. Final Submission

Deliverables prepared for review:

Item	Location / Description
GitHub repository	Complete source, configs, and README files
Week 1 report	Week1_Report.pdf – vulnerability assessment
Week 2 report	Week2_Report.pdf – security fixes

Item	Location / Description
Week 3 report	Week3_Report.pdf – this document
Security checklist	Security_Checklist.md
Screenshots	docs/screenshots folder embedded in reports

6. Conclusion

Week 3 activities validated that the security measures implemented in Week 2 are effective:

- Penetration testing showed resilience against XSS, NoSQL injection and weak passwords.
- Logging delivered visibility into application behaviour and potential abuse.
- Documentation artefacts (checklist and reports) complete the project lifecycle: **Identify → Fix → Validate → Document.**

7. References

- [Nmap Network Scanner](#)
- [Winston Logging Library](#)
- [OWASP Top 10 – 2021](#)
- [OWASP XSS Prevention Cheat Sheet](#)
- [Node.js Security Checklist](#)
- [Helmet.js Security](#)
- [Security checklist template](#)

Additional resources documented in the [Security_Checklist.md](#) file.