README – Week 3: Cybersecurity Intern (SmartGlass Project)

Name: Posaram Dewasi

Team: 14 - Smart Glass

Role: Cybersecurity Intern

Duration: Week 3 (11 July 2025)

# Project Overview

SmartGlass is a collaborative EdTech platform built with React, Node.js, MongoDB, and Firebase. As a cybersecurity intern, I focused on enhancing backend security, auditing sensitive files, and ensuring secure API practices.

# ✅ Week 3 Objectives Completed

1. Code audits of key files such as jwtAuth.js, authController.js, server.js

2. Implemented authentication and authorization checks using JWT

3. Verified route security and token expiration policies

4. Added monitoring and logging for suspicious access

5. Configured secure .env usage and middleware setup (CORS, Helmet)

# 🧰 Tools Used

- Postman for API testing

- VS Code for code reviews

- MongoDB Compass for DB validation

- JWT.io for token inspection

- Snyk CLI & ZAP (suggested for vulnerability scans)

**. Security Instructions**

As part of the security audit and testing for SmartGlass, the following practices were followed and implemented:

* **Validate User Input**  
  Used express-validator or Joi to ensure safe backend input handling.
* **JWT Protection on Routes**  
  All sensitive routes are protected with middleware (jwtAuth.js) that checks for valid JWT in the Authorization header.
* **Environment Security**  
  All secrets like JWT\_SECRET, MONGO\_URI are stored in .env and excluded from GitHub via .gitignore.
* **Helmet & CORS**  
  Implemented helmet to secure headers and cors for safe cross-origin access control.
* **Token Expiry**  
  JWT tokens expire after 15 minutes to reduce risk of misuse.
* **Rate Limiting**  
  Set up express-rate-limit to limit login requests and prevent brute-force attacks.

**2. Bugs and Errors Found**

| **Bug / Issue** | **Description** | **Action Taken** |
| --- | --- | --- |
| ❌ Missing JWT on protected routes | Route allowed access without authentication | Added jwtAuth.js middleware |
| ❌ No role check | Anyone could access educator routes | Implemented role-based access |
| ❌ Secrets in code | JWT key was visible in code | Moved all credentials to .env |
| ❌ No logs for token issues | Difficult to debug invalid access | Added console.log statements in jwtAuth.js |

# Security Implementation Summary

- JWT-based route protection and middleware verification

- Firebase Admin token validation for user authentication

- Protected MongoDB schemas for users, sessions, and chats

- Input validation and secure error handling (401 for unauthorized access)

# Final Note

All key tasks for Week 3 were implemented successfully. Screenshots of Postman tests, code views, and authentication responses have been included in the report and pull request. This README serves as a summary of the security work performed.