Building to Break A "Vulnerable by Design" Approach to Web Security

COMP6841

Security Engineering

Project Check-in poster

Author Details

- **Name** - Sagar
- **Course** - COMP6841
- **Tutorial Group** – H11B

The "What" and "Why"

Elevator Pitch

This project involves building a fully functional, yet intentionally insecure, web application to demonstrate and analyse critical OWASP Top 10 vulnerabilities. The goal is to create a hands-on sandbox for understanding how attacks like SQL Injection and Cross-Site Scripting work, in order to learn how to prevent them effectively.

Why This Project is Important

Many developers, including myself, build functional applications without a deep understanding of the inherent security risks. My motivation for this project stems from a real-world application I built for a tutoring business. The realisation that I was responsible for sensitive student and tutor data created an urgent need to move beyond being just a builder and become a defender of that data.

A theoretical understanding of vulnerabilities is not enough. This project bridges the gap between theory and practice by creating a tangible environment to see exactly *how* exploits work. By learning to think like an attacker in a controlled setting, developers can become truly effective defenders, capable of building genuinely secure and trustworthy software from the ground up.

The Project Plan

Project Approach & Timeline

The project is managed in three distinct phases over eight weeks. The approach is to build a basic functional application first, then iteratively introduce, exploit, and document one major vulnerability at a time. This modular approach ensures steady progress and focused learning.

Week(s)	Phase	Milestone / Key Activities	Status
1-3	1. Foundation	Project Ideation, OWASP Research, Application Architecture Planning. Developed basic application - user registration, login, and posting features.	✅ Done
4	2. Implementation	Implement and exploit an insecure search function. Document the attack and remediation.	⏳ In Progress
5	2. Implementation	Implement and exploit a stored XSS vulnerability in the post viewing feature.	☐ To Do
6	2. Implementation	Refine existing exploits and documentation. Begin planning the next vulnerability.	☐ To Do
7	2. Implementation	Implement Broken Access Control (maybe or some other vulnerability)	☐ To Do
8	3. Finalisation	Complete all documentation, assemble the final report, and record a proof-of-concept video.	☐ To Do

Challenges & Outcomes

Main Challenges

- Deliberately writing insecure yet functional code is a unique challenge that requires fighting against ingrained best-practice instincts.
- Accurately documenting the exploits and, more importantly, the *correct* remediations requires deep research beyond just identifying the initial flaw.
- The biggest challenge is focusing on 3-4 high-impact vulnerabilities and exploring them deeply, rather than trying to implement every possible flaw superficially.

Project Deliverables & Outcomes

Upon completion, this project will be demonstrating practical and comprehensive cybersecurity skills.

- A locally hostable app serving as a learning and demonstration tool.
- Well-documented source code for the application.
- This report will be the primary deliverable, containing:
    - An overview of the application and its architecture.
    - In-depth write-ups for each implemented vulnerability (SQLi, XSS, etc.), including an explanation of the flaw, a step-by-step proof-of-concept exploit and "Before and After" code samples showing the remediation.

My goal is to use this project to transform my perspective from a developer who uses security features to an engineer who understands them.