A Mini Project Synopsis on

"VulnQuest: Web Vulnerability Exploration Platform"

Submitted in partial fulfillment of the requirements for the award
of the degree of

Final Year Engineering

in

Information Technology

by

Shreyash Narvekar (22104066)
Pritam Ninganaik (22104095)
Radhika Lakhani (22104078)

Under the Guidance of

Ms. Jayshree Jha

Department of Information Technology
NBA Accredited
A.P. Shah Institute of Technology
G.B.Road,Kasarvadavli, Thane(W)-400615
UNIVERSITY OF MUMBAI

Academic Year 2025-2026

CERTIFICATE

This is to certify that the project entitled ***"VulnQuest: Web Vulnerability Exploration Platform."*** submitted by ***"Shreyash Prashant Narvekar" (22104066), "Pritam Bharat Ninganaik " (22104095), "Radhika Sanjay Lakhani" (22104078)*** for the partial fulfillment of the requirement for award of a degree ***Bachelor of Engineering*** in ***Information Technology***,to the University of Mumbai,is a bonafide work carried out during academic year 2025-2026.

Ms.Jayshree Jha
Guide

External Examiner

1.

Place:A.P.Shah Institute of Technology, Thane
Date:

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature:

Shreyash Narvekar (22104066)

Pritam Ninganaik (22104095)

Radhika Lakhani (22104078)

Date:

Acknowledgement

This project would not have come to fruition without the invaluable help of our guide **Ms. Jayshree Jha**. Expressing gratitude towards our HoD, **Dr. Kiran Deshpande**, and the Department of Information Technology for providing us with the opportunity as well as the support required to pursue this project. We would also like to thank our teacher **Mr. Vishal Badgujar** who gave us her valuable suggestions and ideas when we were in need of them. We would also like to thank our peers for their helpful suggestions.

Contents

Abstract

VulnQuest: Web Vulnerability Exploration Platform is a hands-on cybersecurity learning environment developed to simulate real-world web application vulnerabilities and exploitation scenarios. Built using PHP and MySQL, it provides an interactive set of modular labs covering diverse topics such as SQL Injection, Cross-Site Scripting (XSS), Command Injection, File Upload Bypass, Authentication Flaws, XML External Entity (XXE) attacks, and Insecure Direct Object References (IDOR). Each lab progresses in complexity, enabling users to gradually build their offensive security skills through guided and challenge-based exercises.

The platform emphasizes experiential learning by encouraging users to actively engage with simulated attack surfaces and implement remediation strategies. Its containerized architecture, powered by Docker, ensures portability, easy deployment, and consistent execution across environments. VulnQuest not only bridges the gap between theoretical knowledge and practical cybersecurity application but also prepares learners for real-world penetration testing, Capture-the-Flag (CTF) competitions, and industry-recognized certifications.

By integrating realistic scenarios, progressive challenges, and structured guidance, VulnQuest fosters a deep understanding of modern web vulnerabilities and equips students, educators, and professionals with essential skills in offensive and defensive security.

Chapter 1

Introduction

In today's digital era, where web applications form the backbone of almost every business and service, ensuring their security has become a critical priority. The rapid evolution of online platforms has increased both convenience and vulnerability, exposing organizations to a range of cyber threats that exploit weaknesses in web application code, configuration, and logic. Attacks such as SQL Injection, Cross-Site Scripting (XSS), and Command Injection continue to compromise sensitive data and disrupt digital services. Understanding how these vulnerabilities arise—and how to defend against them—has become an essential skill for cybersecurity professionals.

Despite the growing emphasis on cybersecurity education, a significant gap exists between theoretical understanding and practical application. Many students and early-career professionals are well-versed in the concepts of secure coding and vulnerability classifications but lack the hands-on experience needed to detect, exploit, and remediate vulnerabilities effectively. Classroom learning often falls short of providing an environment where learners can safely experiment with real attack vectors and develop a true attacker's mindset.

VulnQuest: Web Vulnerability Exploration Platform aims to bridge this critical gap by providing a safe, interactive, and realistic web security training environment. Built using PHP and MySQL, VulnQuest simulates real-world web application vulnerabilities through structured lab modules that progress from beginner-friendly tutorials to complex exploitation challenges. Each lab encourages learners to explore the underlying mechanisms of vulnerabilities while also focusing on their corresponding mitigation techniques.

The platform includes multiple modules covering vulnerabilities such as SQL Injection, Cross-Site Scripting (XSS), Command Injection, File Upload Bypass, Authentication Flaws, XML External Entity (XXE) attacks, and Insecure Direct Object References (IDOR). The lab design promotes gradual skill development—starting from guided exercises in early modules to open-ended challenges in advanced stages. By doing so, VulnQuest offers a complete learning experience that reinforces both offensive and defensive security skills.

VulnQuest operates within a containerized environment using Docker Compose, ensuring consistent performance, ease of setup, and isolation for each user. This design enables users to reset environments, test multiple approaches, and safely conduct experiments without risk to external systems. The platform supports both self-paced learning and instructor-led workshops, making it suitable for academic institutions, cybersecurity bootcamps, and professional training programs.

Ultimately, VulnQuest is more than just a cybersecurity lab—it is a complete experiential learning ecosystem. It empowers students, educators, and professionals to transition from passive learning to active engagement, equipping them with the practical skills required for ethical hacking, penetration testing, and real-world security assessments.

1.1 Purpose

The primary purpose of the VulnQuest platform is to provide a hands-on, interactive environment for exploring web application vulnerabilities safely and effectively. Traditional classroom learning often lacks the experiential component needed to truly understand the mechanics of attacks and defenses. VulnQuest fills this gap by simulating realistic scenarios where users can learn by doing—experimenting with exploitation techniques and implementing secure coding practices.

By offering a structured collection of web vulnerability labs, the platform encourages learners to build a deep, practical understanding of how vulnerabilities emerge in poorly designed applications. Additionally, it fosters an appreciation for security best practices and emphasizes the importance of proactive defense in development lifecycles.

The goal of VulnQuest is not only to train users in identifying and exploiting vulnerabilities but also to help them develop a strong foundation in secure coding, ethical hacking, and cybersecurity awareness. Through guided tutorials, progressive challenges, and real-world exploitation exercises, it prepares users for professional certifications such as CEH, OSCP, and practical penetration testing roles.

1.2 Problem Statement

With the ever-growing dependence on web-based applications, the number of cyber-attacks targeting them has risen sharply. Despite the availability of extensive theoretical resources, students and professionals often lack the opportunity to apply their knowledge in real, controlled environments. This gap between conceptual understanding and hands-on application results in limited readiness for real-world cybersecurity scenarios.

Existing learning resources are often either too abstract or too dangerous for experimentation on live systems. Learners need a safe, reproducible, and isolated environment where they can practice identifying and exploiting vulnerabilities without causing harm to production networks. Moreover, traditional academic curricula rarely include interactive exercises that simulate realistic attack vectors, leaving graduates underprepared for modern cybersecurity challenges.

VulnQuest addresses this issue by providing a secure and self-contained platform that mimics real-world vulnerabilities while maintaining complete isolation from external systems. The platform allows users to experiment freely, understand attack mechanisms, and develop mitigation strategies—all within a controlled and educational setting.

1.3 Objectives

- To design and develop a containerized web-based lab environment simulating real-world web vulnerabilities for safe experimentation.

- To create modular lab exercises categorized by vulnerability type such as SQL Injection, XSS, Command Injection, File Upload, Authentication Flaws, XXE, and IDOR.

- To implement a progressive difficulty system that guides users from basic concepts to advanced exploitation scenarios.

- To provide built-in instructions, hints, and reset functionalities for enhanced learning and usability.

- To offer a dedicated capstone section combining multiple vulnerabilities into full-chain exploitation challenges.

- To support educators and trainers by allowing easy deployment via Docker Compose for classroom or workshop setups.

- To promote cybersecurity awareness, skill-building, and readiness for professional certifications and penetration testing roles.

1.4 Scope

- The platform provides a collection of interactive web security labs implemented in PHP and MySQL.

- It supports containerized deployment using Docker Compose for consistency and ease of use.

- The system includes both basic and advanced vulnerability scenarios, suitable for learners at different skill levels.

- A dedicated capstone section integrates multiple vulnerabilities to simulate full-chain real-world attacks.

- The platform can be used in academic institutions, training programs, hackathons, and cybersecurity workshops.

Chapter 2

Literature Survey

Recent research in intelligent automation, secure system design, and energy-efficient technologies has demonstrated the potential of combining IoT, AI, and human-computer interaction (HCI) for optimized system performance. These studies provide a foundation for designing robust, secure, and energy-conscious applications in diverse domains, including web security, DevOps education, and smart management platforms.

Aydos et al. [1] conducted a comprehensive mapping study on web application security testing. They analyzed 80 research papers published between 2005 and 2020, classifying testing tools according to vulnerability type, testing methodology (SAST/DAST), and automation level. Their findings emphasized the importance of adhering to OWASP guidelines and integrating security practices within the software development lifecycle (SDLC). While their study highlighted common vulnerabilities and testing strategies, it also noted the limitations of automated tools, particularly false positives and negatives, and the lack of exploration of modern AI-based hybrid security testing methods. This study provides a critical foundation for developing automated security testing frameworks that incorporate AI to improve detection accuracy.

Tauqeer et al. [2] proposed a taxonomy for security testing techniques, dividing the process into Identification, Testing, and Reporting phases. Their review of 292 publications (2010–2019) covered fuzzing, penetration testing, static and dynamic analysis, and risk assessment methodologies. The work is significant in categorizing existing methods for practical adoption; however, it did not include experimental validation or consider machine learning-based security testing advancements. This provides insights into structured approaches to security testing and highlights the potential for integrating intelligent automation.

Rahman et al. [3] developed an authentic learning framework for DevOps security education using Git Hooks for automated static security analysis. The framework incorporated pre-lab, hands-on, and post-lab modules to engage students in secure coding practices. This approach demonstrated improved understanding and engagement in academic settings but lacked industrial-scale validation. The study emphasizes the role of practical, automated tools in cultivating security-conscious developers and can be extended to real-world DevOps pipelines.

In the domain of HCI and system usability, Joven Landeros [4] designed a club management platform to enhance operational efficiency and student engagement. The system offered features such as smart club discovery, automated membership workflows, event planning tools, and a centralized communication hub. Usability testing indicated improved interaction and organizational efficiency, although the study was limited by a small sample size and

short-term evaluation. Future improvements include integrating AI-driven recommendations and social media connectivity to expand user engagement and accessibility.

Overall, these studies demonstrate the integration of automated intelligence, secure design, and usability-focused systems. They collectively provide the theoretical and practical foundation for implementing secure, energy-efficient, and user-friendly platforms that leverage modern IoT, AI, and HCI technologies.

Table 2.1: Summary of Literature Survey

Title	**Key Contributions**	**Limitations**	**Tech Stack / Tools Used**
Security Testing of Web Applications: A Systematic Mapping [1]	Classified 80 papers (2005–2020) by vulnerability type, testing methodology, and automation level. Highlighted OWASP guidelines and SDLC integration.	Automated tools generated false positives/negatives; lacked AI-based hybrid approaches.	OWASP ZAP, SAST/DAST tools, Testing Frameworks
Analysis of Security Testing Techniques [2]	Proposed a taxonomy dividing testing into Identification, Testing, and Reporting. Reviewed 292 papers covering fuzzing, penetration testing, and risk assessment.	Limited to pre-2019 publications; no experimental validation; ML-based testing not included.	SAST/DAST, Metasploit, Burp Suite, Risk Assessment Models
Authentic Learning on DevOps Security [3]	Developed an educational framework using Git Hooks for automated static analysis. Improved student engagement in secure coding practices.	Limited testing scope; not validated in industrial environments.	Git, Git Hooks, CI/CD Pipelines, SAST tools, Python
Enhancing Student Engagement: Club Management Platform [4]	Developed an HCI-focused platform with smart club discovery, automated workflows, and centralized communication.	Small sample size and short-term evaluation; future work includes AI-driven recommendations.	HTML, CSS, JavaScript, React/Node.js; HCI/UX design principles

Chapter 3

Problem Definition

In the current era of digital transformation, web applications play a crucial role in the operation of organizations, institutions, and businesses. However, this widespread dependence on web technologies has also made them prime targets for cyberattacks. Vulnerabilities such as SQL Injection, Cross-Site Scripting (XSS), Command Injection, File Upload Bypass, and Authentication Flaws continue to be among the most exploited weaknesses in modern systems. These attacks can lead to severe data breaches, financial loss, and reputational damage.

While cybersecurity awareness has grown significantly, a major challenge persists in how individuals learn, understand, and apply vulnerability assessment and exploitation techniques. Theoretical study alone does not equip learners with the necessary practical skills required to recognize, exploit, and mitigate real-world vulnerabilities. Existing online resources are often either overly simplistic or lack safe, interactive environments where users can perform realistic penetration testing without risking live systems.

Students, educators, and entry-level cybersecurity professionals frequently struggle to gain hands-on experience because most academic programs focus on conceptual frameworks rather than interactive learning. Moreover, many learners lack access to properly configured test environments that replicate genuine attack vectors, resulting in a gap between academic knowledge and industry-required practical competence.

The **VulnQuest: Web Vulnerability Exploration Platform** aims to address this gap by providing an immersive and controlled environment where users can safely explore, understand, and exploit common web vulnerabilities. By simulating realistic attack surfaces using PHP and MySQL applications, VulnQuest allows users to experience both offensive and defensive perspectives. Each lab module is designed to replicate vulnerabilities found in production systems, accompanied by guided challenges that progress in complexity.

Therefore, the core problem this project seeks to solve is the lack of accessible, realistic, and safe hands-on environments for learning web application security. VulnQuest bridges this gap by offering a self-contained, Docker-based platform where learners can practice identifying and mitigating vulnerabilities through structured, challenge-oriented exercises. This ensures a balanced understanding of both offensive tactics and defensive countermeasures, preparing users for professional penetration testing and ethical hacking roles.

Chapter 4

Proposed System

The proposed system, **VulnQuest: Web Vulnerability Exploration Platform**, is designed to provide a realistic, hands-on cybersecurity learning environment where users can explore, exploit, and mitigate web application vulnerabilities in a safe, isolated setup. The system offers a series of modular labs developed using PHP and MySQL that simulate real-world security flaws such as SQL Injection, Cross-Site Scripting (XSS), Command Injection, File Upload Bypass, Authentication Flaws, XML External Entity (XXE) attacks, and Insecure Direct Object References (IDOR). Each module includes multiple exercises that increase in difficulty, allowing learners to gradually develop their offensive and defensive security skills.

The platform operates in a containerized environment using Docker Compose, ensuring consistent deployment across systems and simplifying environment setup. Each container runs a lab instance that mimics an intentionally vulnerable web application. This structure allows learners to reset, test, and experiment freely without affecting the base system or other users.

Key Features:

- **Interactive Lab Modules:** The platform includes categorized labs for each vulnerability type, such as Injection, XSS, Command Injection, File Upload, Authentication, XXE, and IDOR, with three escalating sub-levels (e.g., Injection 0x01–0x03).

- **Guided and Challenge-Based Learning:** Early exercises provide hints and structured guidance, while later ones serve as open-ended challenges that require independent problem-solving.

- **Capstone Section:** An advanced lab module integrates multiple vulnerabilities into full-chain exploitation scenarios, simulating real-world penetration testing conditions.

- **Self-Paced Learning:** Users can progress at their own speed, with built-in database reset mechanisms to ensure a clean environment for every new attempt.

- **Containerized Deployment:** The platform uses Docker Compose for easy setup, reproducibility, and system isolation, eliminating dependency issues.

- **User Progress Tracking:** The frontend maintains local progress records using browser storage, helping learners track completed challenges.

- **Cross-Platform Compatibility:** The labs run seamlessly on any system that supports Docker, enabling wide accessibility for students and professionals alike.

System Architecture:

The architecture of VulnQuest is structured around three primary layers:

1. **Frontend Layer:** Developed using HTML, CSS, JavaScript, and PHP templates, it provides an intuitive, card-based interface where each card represents a vulnerability module or challenge.

2. **Backend Layer:** Built using PHP and MySQL, the backend handles request processing, user interaction with vulnerabilities, and database operations for each simulated scenario.

3. **Container Layer:** Managed via Docker Compose, this layer encapsulates both standard labs and advanced capstone challenges in isolated containers, ensuring reproducibility and safe testing.

System Workflow:

The workflow of VulnQuest follows a structured sequence designed for seamless user interaction:

1. **Environment Setup:** The user launches the lab using the command `sudo docker-compose up --build`, which initializes the PHP server and MySQL databases within their containers.

2. **Lab Selection:** Upon accessing the web interface, users are presented with categorized vulnerability modules displayed as interactive cards.

3. **Vulnerability Exploration:** Each lab presents a realistic web application flaw. Users perform exploitation attempts, such as SQL Injection queries or XSS payloads, to understand the vulnerability mechanism.

4. **Reset and Reattempt:** If needed, users can visit the `init.php` page to reset the lab environment, restoring databases to their default state for fresh testing.

5. **Challenge Completion:** Each module concludes with a challenge exercise where users apply previously learned concepts to exploit and secure the given scenario.

6. **Capstone Challenge:** The final section integrates multiple vulnerabilities into a chained exploitation sequence, testing comprehensive problem-solving and analytical skills.

Advantages of the Proposed System:

- Provides a realistic, risk-free environment for learning web security concepts.

- Encourages experiential learning through trial and error rather than theoretical memorization.

- Eliminates setup complexity by using Docker for deployment and reset management.

- Bridges the gap between academic learning and real-world cybersecurity practice.

- Prepares users for penetration testing certifications and professional security assessments.

In summary, the proposed system serves as a unified, interactive, and scalable platform for web security education. It not only enhances user understanding of vulnerabilities and exploit mechanisms but also promotes critical thinking and hands-on expertise essential for professional cybersecurity roles.

Chapter 5

Technology Stack

The **VulnQuest: Web Vulnerability Exploration Platform** is developed using an open-source and lightweight technology stack that ensures scalability, portability, and simplicity of deployment. The selection of each component is based on its ability to simulate realistic web application environments while maintaining modularity and performance. The system primarily utilizes PHP and MySQL for backend operations, with Docker providing environment isolation and reproducibility across various operating systems.

The complete technology stack is divided into three major segments: **Frontend**, **Backend**, and **Deployment Infrastructure.**

1. Frontend Technologies

The frontend serves as the user interface of the VulnQuest platform, allowing learners to interact with individual lab challenges and modules. It provides a simple yet interactive layout to navigate between different vulnerability categories and sub-labs. The frontend is designed with accessibility and usability in mind, using standard web technologies.

Technologies Used:

- **HTML5:** Used for structuring the web content and lab modules, ensuring compatibility across browsers.
- **CSS3:** Provides styling, layout consistency, and responsive design for the lab interface.
- **JavaScript:** Enables interactivity, including form validation, dynamic responses, and hint functionality.
- **Bootstrap:** Simplifies UI design and ensures a clean, mobile-friendly experience.
- **PHP (Frontend Integration):** Used for dynamic rendering of challenge cards and database-driven lab content.

Frontend Features:

- Card-based layout for navigating between vulnerability modules.
- User progress tracking using browser localStorage.

- Integrated instructions and reset buttons for guided learning.
- Support for both light and dark themes for improved accessibility.

2. Backend Technologies

The backend of VulnQuest is responsible for executing vulnerability logic, managing lab data, and handling all user interactions that simulate real-world exploitation. It is designed to be lightweight and modular to allow easy modification of lab challenges without affecting the core framework.

Technologies Used:

- **PHP:** The primary server-side scripting language used to simulate vulnerabilities, handle form submissions, and process user inputs.
- **MySQL:** A relational database used for storing user data, lab records, and configurations for both standard and capstone challenges.
- **Apache HTTP Server:** Serves as the web server to host and run the PHP applications within the Docker container.

Backend Functionalities:

- Executes backend logic for vulnerabilities such as SQL Injection, Command Injection, and Authentication Bypass.
- Interacts with MySQL databases to store and retrieve dynamic data during lab exercises.
- Handles challenge verification logic to confirm successful exploitation by the user.
- Supports automatic database reset and reinitialization through the `init.php` utility.

3. Deployment and Infrastructure

The deployment of VulnQuest relies on Docker Compose for environment orchestration. This ensures that each lab runs within its own container, isolated from the host system and other services. Docker simplifies the process of launching, maintaining, and resetting labs without manual configuration or dependency conflicts.

Technologies Used:

- **Docker:** Containerizes each lab component, providing isolated environments for PHP, MySQL, and Apache services.
- **Docker Compose:** Manages multi-container applications, defining how the web and database containers communicate and operate together.

- **Linux:** Acts as the underlying host operating system for executing Docker containers efficiently.

Deployment Features:

- Simple initialization using a single command: `sudo docker-compose up --build`.
- Auto-configuration of MySQL databases for standard and capstone labs.
- Easy system reset capability for restoring clean environments.
- Scalability to add or remove modules dynamically via Docker configuration.

Summary: The technology stack of VulnQuest has been carefully selected to balance simplicity, flexibility, and realism. By leveraging PHP and MySQL for backend operations and Docker for environment management, the platform ensures that users can perform cybersecurity exercises safely and efficiently. The combination of traditional web technologies with modern containerization enables a practical, modular, and educational experience that mirrors real-world penetration testing environments.

Chapter 6

Implementation

The implementation of **VulnQuest: Web Vulnerability Exploration Platform** is deployed using Docker Compose to provide a consistent, isolated environment for each lab. The environment setup steps are listed below to ensure reproducible deployment across different host systems.

6.1 Environment Setup

To set up and run the VulnQuest platform, follow these steps:

1. Install Docker Engine and Docker Compose on the host machine (Linux recommended).

2. Clone the project repository containing the lab modules and Docker configuration:

   ```
   git clone <your-repo-url> vulnquest
   cd vulnquest
   ```

3. Review and, if necessary, adjust configuration files (e.g., `docker-compose.yml`, `.env`) to match host port mappings and resource limits.

4. Build and start the containers with:

   ```
   sudo docker-compose up --build
   ```

5. On first run, initialize or reset the lab databases by visiting the initialization utility in a browser:

   ```
   http://localhost:8080/init.php
   ```

6. Access the VulnQuest web interface at:

```
http://localhost:8080
```

7. If containers need to be stopped and removed:

```
sudo docker-compose down
```

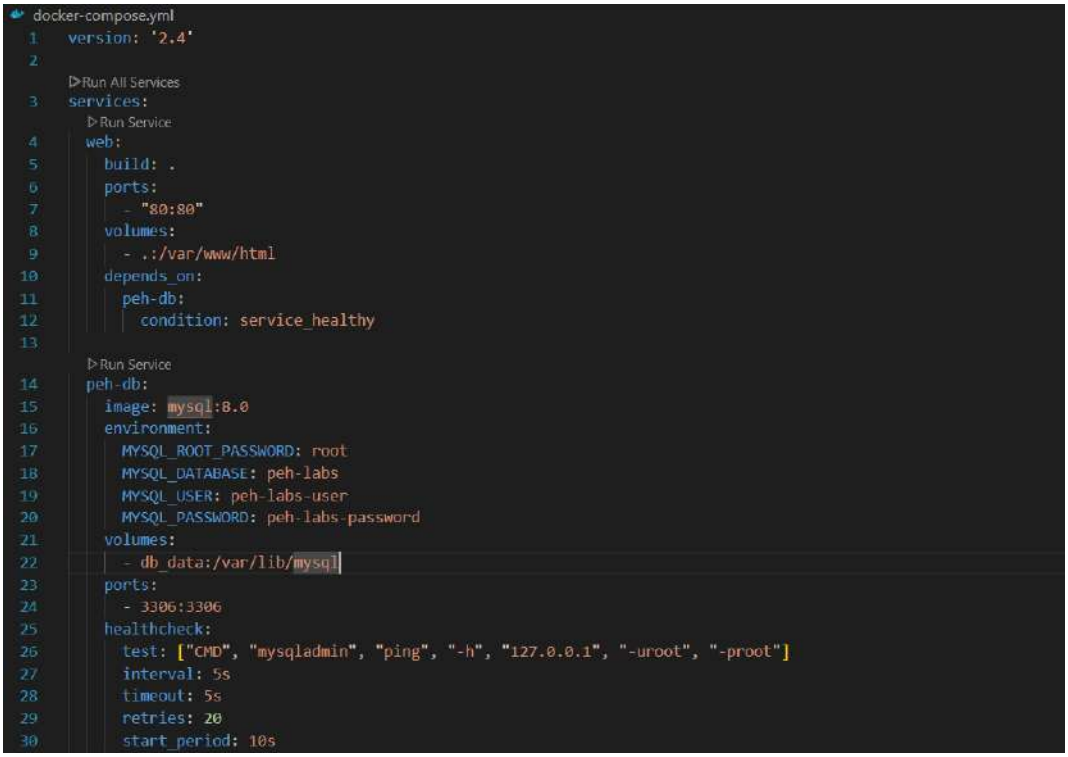

Figure 6.1: Snapshot of `docker-compose.yml` highlighting service definitions, port mappings, volumes, and environment variables.

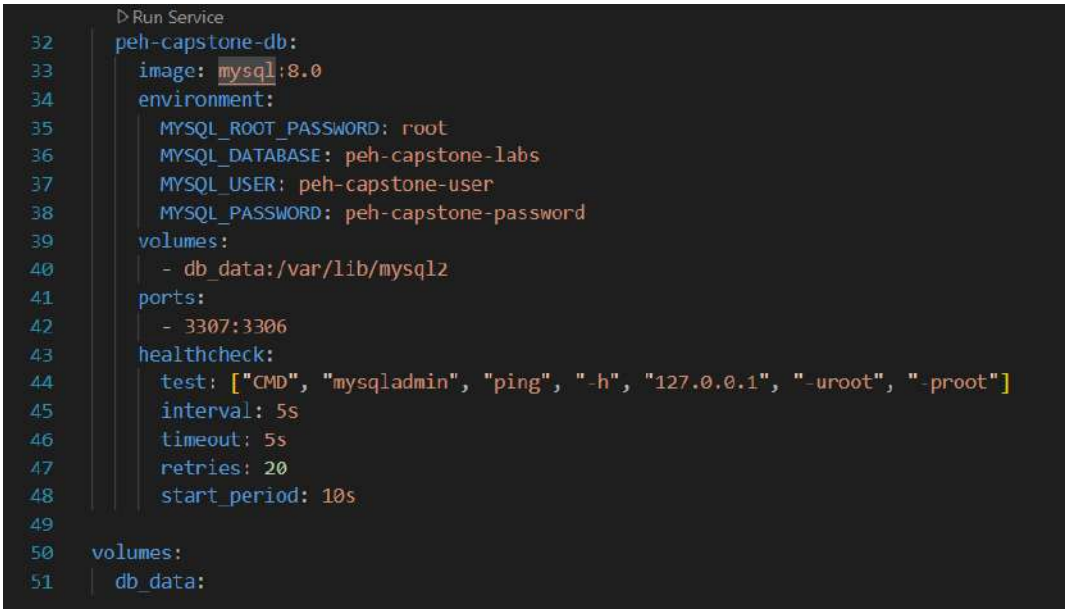

Figure 6.2: Key configuration files (e.g., Dockerfile, relevant Apache/PHP configs) used to build the web container.

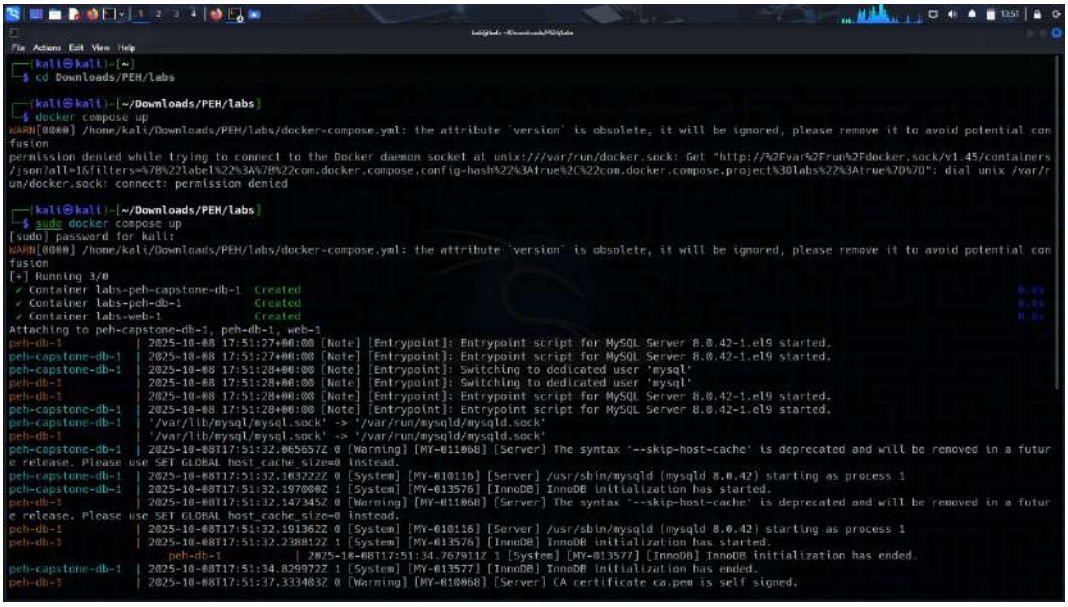

Figure 6.3: Terminal output showing the command `sudo docker-compose up --build` executed successfully with containers starting.

checks.php

```php
<?php

// check injection 0x01
$tableName = "injection0x01";
$checkTable = "SHOW TABLES LIKE '$tableName'";
$result = $conn->query($checkTable);
if ($result->num_rows == 0) {
    $errorMessage = "Error: Could not find injection0x01 table.";
}

// check injection 0x02
$tableName = "injection0x02";
$checkTable = "SHOW TABLES LIKE '$tableName'";
$result = $conn->query($checkTable);
if ($result->num_rows == 0) {
    $errorMessage = "Error: Could not find injection0x02 table.";
}

// check injection 0x03
#$tableName = "injection0x03";
#$checkTable = "SHOW TABLES LIKE '$tableName'";
#$result = $conn->query($checkTable);
#if ($result->num_rows == 0) {
#    $errorMessage = "Error: Could not find injection0x03 table.";
#}
```

Figure 6.4: IDE screenshot showing example vulnerable code snippets used in labs (e.g., SQL injection or XSS vulnerable PHP code).

Chapter 7

Results

This chapter presents the output screenshots captured from the implemented system. The results primarily demonstrate the successful deployment of the Docker-based environment, the working of the platform interface, and the functioning of individual security labs such as SQL Injection and Cross-Site Scripting (XSS), along with the Capstone Project module.

7.1 Screenshots

```
PS E:\ROSPL MAIN> cd labs
PS E:\ROSPL MAIN\labs> docker ps
CONTAINER ID   IMAGE       COMMAND                  CREATED         STATUS                   PORTS
      NAMES
3012876def20   labs-web    "docker-php-entrypoi…"   55 minutes ago   Up 55 minutes            0.0.0.0:8080->80/tcp, [::]:8080->80/tcp
      labs-web-1
37b851314fef   mysql:8.0   "docker-entrypoint.s…"   55 minutes ago   Up 55 minutes (healthy)   0.0.0.0:3307->3306/tcp, [::]:3307->3306/
tcp   labs-peh-capstone-db-1
9e105b6a71c4   mysql:8.0   "docker-entrypoint.s…"   55 minutes ago   Up 55 minutes (healthy)   0.0.0.0:3306->3306/tcp, [::]:3306->3306/
tcp   labs-peh-db-1
PS E:\ROSPL MAIN\labs>
```

Figure 7.1: Output of running containers using `docker ps`, confirming successful deployment of all services.

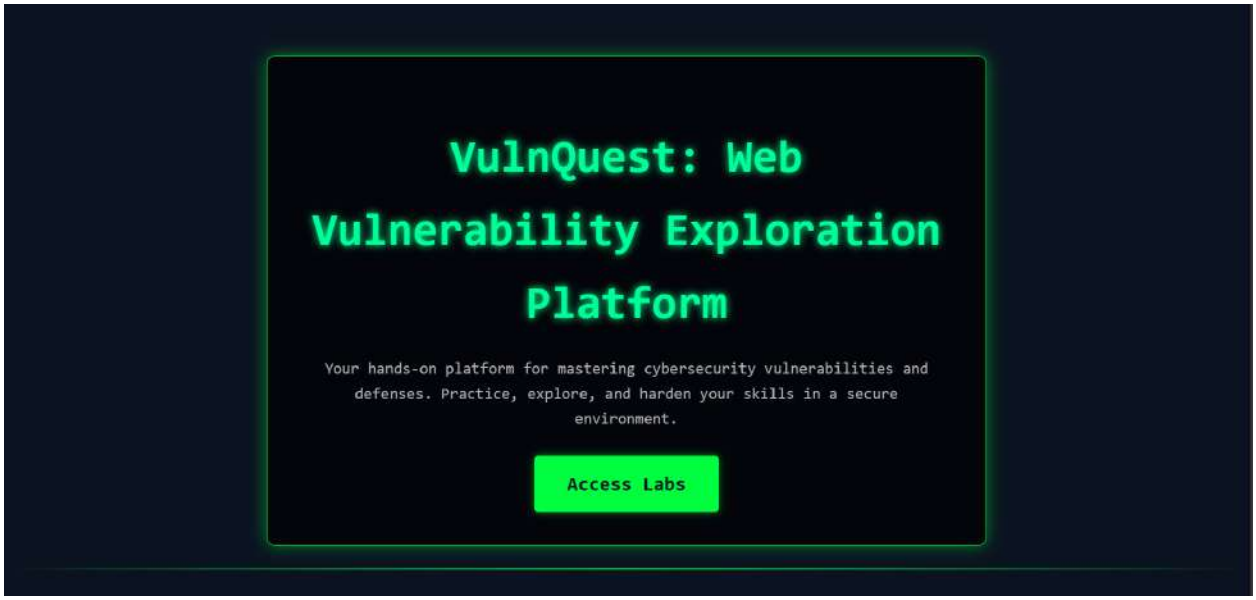

Figure 7.2: Landing page of the platform showing navigation options for different labs and modules.

Figure 7.3: Overview of the Labs section containing multiple vulnerability-based exercises such as SQL Injection and XSS.

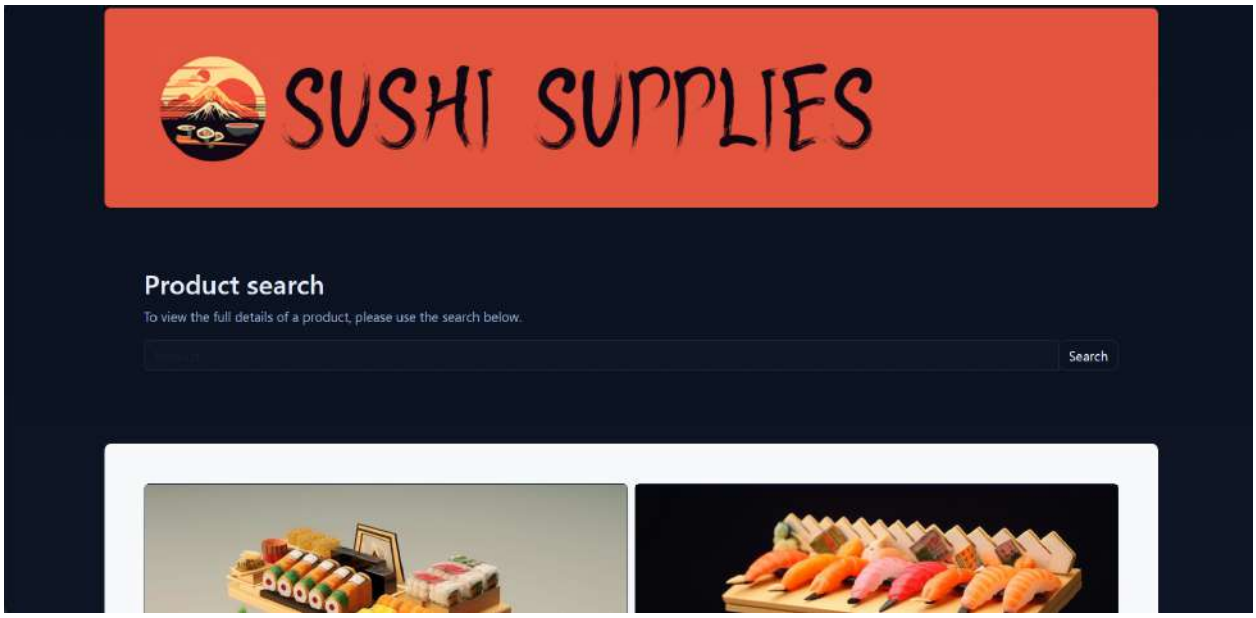

Figure 7.4: First Lab – SQL Injection: Demonstrating exploitation of query-based vulnerabilities in a sample form.

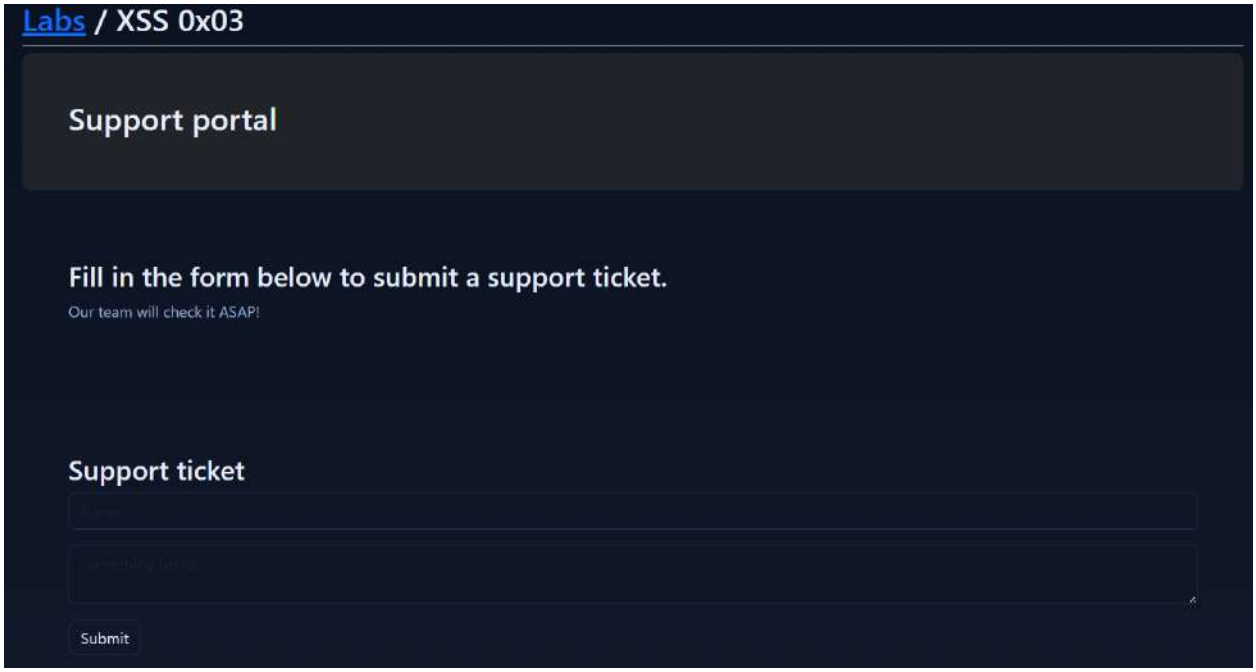

Figure 7.5: Third Lab – Cross-Site Scripting (XSS): Displaying client-side script injection and result visualization.

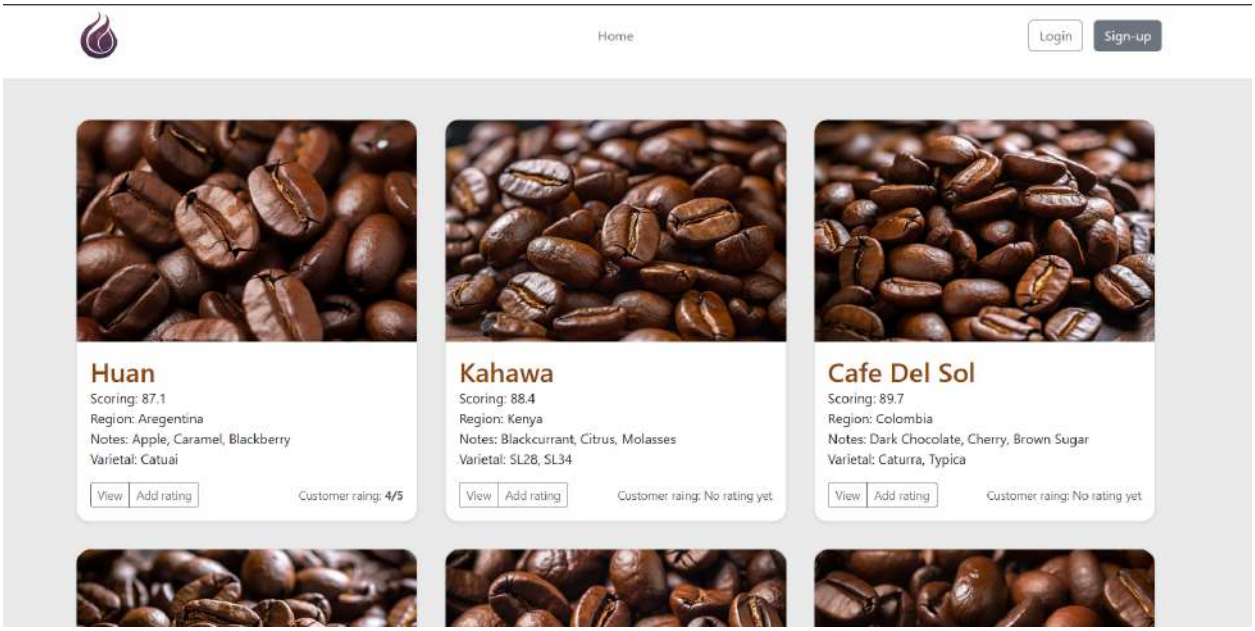

Figure 7.6: Capstone Project Interface: Final integrated environment combining multiple vulnerabilities and defenses.

Chapter 8

Conclusion

The project successfully demonstrates the development of a web application security lab that allows safe experimentation with common web vulnerabilities such as SQL Injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF). The use of Docker-based containerization ensures an isolated, reproducible, and easily deployable environment. This platform provides hands-on learning opportunities for students and professionals, bridging the gap between theoretical knowledge and practical application. Overall, the lab serves as an effective educational tool for understanding web security risks and mitigation techniques, with potential for further expansion and integration with additional security modules in the future.

Chapter 9

Future Scope

The platform developed in this project provides a practical environment for learning and testing web application vulnerabilities. However, there are several avenues for future enhancement:

- **Integration of More Vulnerabilities:** Include additional security flaws such as CSRF, file upload vulnerabilities, and server-side request forgery (SSRF) to make the lab more comprehensive.
- **Automated Assessment:** Implement automated scoring or feedback for users attempting challenges, allowing learners to track progress and understand mistakes in real time.
- **Cloud Deployment:** Deploy the platform on cloud services like AWS or Azure to make it accessible remotely without local setup.
- **Mobile Compatibility:** Adapt the platform for mobile devices, enabling learners to practice on smartphones or tablets.
- **Enhanced Analytics:** Add dashboards to monitor user activity, common mistakes, and performance trends to improve the learning experience.
- **Gamification:** Introduce levels, badges, or leaderboards to motivate users and increase engagement.

These enhancements would increase the usability, accessibility, and educational value of the platform, making it a more robust tool for cybersecurity learning and experimentation.

Bibliography

[1] M. Aydos, Ç. Aldan, E. Coşkun, and A. Soydan, "Security Testing of Web Applications: A Systematic Mapping of the Literature," *Journal of King Saud University – Computer and Information Sciences*, vol. 34, pp. 6775–6792, 2022.

[2] O. B. Tauqeer, S. Jan, A. O. Khadidos, A. Khadidos, F. Q. Khan, and S. Khattak, "Analysis of Security Testing Techniques," *Intelligent Automation & Soft Computing*, vol. 29, no. 1, pp. 291–303, 2021.

[3] M. M. Rahman et al., "Authentic Learning on DevOps Security with Labware: Git Hooks to Facilitate Automated Security Static Analysis," in *Proc. IEEE COMPSAC*, 2024, pp. 2423–2426.

[4] Joven Landeros, "Enhancing Student Engagement: Designing an Intuitive Platform for Clubs and Organizations," Independent Study Report, 2025.

[5] OWASP Foundation, "OWASP Top 10 – The Ten Most Critical Web Application Security Risks," 2023. [Online]. Available: https://owasp.org/www-project-top-ten/

[6] OWASP, "OWASP Vulnerable Web Applications Directory," 2023. [Online]. Available: https://owasp.org/www-project-vulnerable-web-applications-directory/