**APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE**
e-ISSN: 2773-5141

**AITCS**

Vol. 6 No. 2 (2025) 644-660
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

# TryMeOut: A Vulnerability Web Application for Security Students Learning

**Muhammad Muaazin Abdullah[1], Khairul Amin Mohamad Sukri[1]***

[1] *Faculty of Computer Science and Information Technology*
*Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, Malaysia*

*Corresponding Author: khairulm@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2025.06.02.035

**Article Info**

Received: 21 October 2025
Accepted: 19 November 2025
Available online: 30 November 2025

**Keywords**

Cross-Site Scripting (XSS), Command Injection, Search Query Language (SQL), SQL Injection, SQL Attack, SQL Injection Test

**Abstract**

TryMeOut is a web-based platform developed to support cybersecurity students in gaining practical, hands-on experience with common web application vulnerabilities. This platform allows students to safely explore and exploit real-world vulnerabilities such as SQL injection, Cross-Site Scripting (XSS), and Command Injection in a secure, controlled environment. The system features a progression of interactive challenges, starting from basic scenarios and advancing to more complex ones. This structure helps students build their skills gradually, reinforcing core concepts while encouraging exploration and problem-solving. Each vulnerability category includes multiple levels that simulate realistic attacks, giving students valuable experience in recognizing, exploiting, and understanding the impacts of these threats. Targeted primarily at bachelor's degree students in the Information Security program at Universiti Tun Hussein Onn Malaysia (UTHM), the platform aims to offer a user-friendly, educational space that bridges the gap between theory and real-world application. This project is developed using the Waterfall Model methodology, which follows a clear, linear process from planning and analysis to design, implementation, and testing.

## 1. Introduction

As cyber threats continue to evolve in complexity and frequency, there is a growing demand for cybersecurity professionals equipped with practical, real-world skills. The dynamic nature of today's digital environment necessitates that cybersecurity students advance beyond theoretical learning to develop hands-on capabilities that are essential for identifying and mitigating emerging security threats. This project aims to develop a web-based educational platform focused on simulating critical web application vulnerabilities drawn from the OWASP Top 10 list, with particular emphasis on injection flaws such as SQL injection.

These vulnerabilities remain among the most exploited due to poor input validation and insecure coding practices. SQL injection, for example, can allow attackers to manipulate backend database queries, potentially leading to unauthorized data access, data leakage, or complete system compromise. As such, a deep and practical understanding of how these attacks operate and how they can be prevented is vital for any aspiring cybersecurity professional.

The proposed platform offers an interactive, lab-based environment where students can safely practice identifying, exploiting, and remediating a range of vulnerabilities. This hands-on approach is designed to bridge the gap between theoretical coursework and practical application, thereby enhancing student competence in real-world cybersecurity practices. Through structured challenge levels that simulate realistic attack vectors, learners will develop a comprehensive skill set that includes vulnerability detection, exploitation, and secure coding.

Targeted at students of Universiti Tun Hussein Onn Malaysia (UTHM), the platform serves as a valuable tool in fostering a deeper understanding of web application security. It also reinforces essential secure development techniques, ultimately contributing to the preparation of highly skilled and industry-ready cybersecurity professionals.

The three main objectives that need to be achieved are:

i. To design a vulnerability testing platform tailored specifically for cybersecurity students, providing a structured environment for exploring common web application vulnerabilities.
ii. To develop a user-friendly interface that enhances learning experience and allows students of all experience levels to easily navigate and engage with the platform.
iii. To implement alpha and beta testing phases with cybersecurity students, gathering feedback to refine and ensure the platform's effectiveness and usability.

## 2. Literature Review

This section discusses important ideas for creating a platform to help cybersecurity students improve essential skills by reviewing existing tools.

### 2.1 Cybersecurity Education

Cybersecurity education is becoming more critical as digital threats continue to expand rapidly. It focuses on developing skills to safeguard computer systems and data [1]. Conventional classroom methods primarily emphasize theoretical concepts, which can leave students inadequately prepared for real-world situations. Studies highlight the importance of hands-on training in realistic environments. Dedicated platforms for vulnerability testing play a key role in connecting theoretical learning with practical experience. To ensure students are equipped to handle emerging cyber threats, educational institutions should integrate practical exercises, provide internship opportunities, and leverage advanced technology to simulate cyber attacks.

### 2.2 SQL Injection

This project places significant emphasis on injection vulnerabilities, with a particular focus on SQL injection. These vulnerabilities arise when untrusted data is inserted into a command or query, enabling attackers to alter the application's execution. There are three types of SQL injection. First, error-based, it takes advantage of database error [2]. Second, union-based, it pulls information from multiple tables from the database. Third, blind injection, it just randomly tries and error injection. Boolean-based blind SQL injection works by triggering true or false conditions in the database, while time-based blind SQL injection introduces delays to deduce results. Out-of-band SQL injections use indirect channels, such as DNS or HTTP, to retrieve data when direct interaction can't happen. SQL injection can result in unauthorized database access, exposing sensitive data. Such vulnerabilities are both prevalent and highly impactful, posing risks to the confidentiality, integrity, and availability of web applications.

### 2.3 Cross-Site Scripting

Cross-Site Scripting (XSS) is a common web vulnerability that enables attackers to inject malicious scripts into web pages viewed by other users. This can lead to session hijacking, credential theft, or redirection to malicious sites [3]. XSS occurs when input is not properly validated or sanitized and is rendered in the browser. This project includes challenges such as Basic Alert, Image OnError, Stored XSS, Reflected XSS Filter Bypass, DOM-Based XSS, and XSS via File Upload. These scenarios provide hands-on experience in identifying and exploiting XSS attacks, reinforcing secure input handling and output encoding practices.

### 2.4 Command Injection

Command Injection allows attackers to execute arbitrary system commands by exploiting improperly handled user input in system-level operations. This can result in unauthorized access, data leaks, or full system compromise [4]. The platform covers various challenges, including Ping Injection, Reading System Files, Reverse Shells, Code Injection (PHP), LDAP Injection, and Path Traversal. These exercises help students understand how insecure code execution can be exploited and how to prevent it through proper input validation and secure coding techniques.

### 2.5 Existing Similar System

In this section, we analyze existing platforms like the one proposed in this study. These systems serve as vulnerability testing and training grounds for cybersecurity students, providing a foundation of insights for building an effective educational tool [5]. The focus is on platforms that provide hands-on training in web security and vulnerability testing.

### 2.5.1 Hack The Box (HTB)

Hack The Box is an online platform that offers virtual machines (VMs) with various security vulnerabilities for users to practice penetration testing. It includes challenges of differing complexity, allowing both beginners and advanced users to develop their skills. Advanced difficulties and real-world situations are the focus of Hack The Box [6].

### 2.5.2 Damn Vulnerable Web Application (DVWA)

DVWA is a PHP/MySQL web application with various security levels, created to help users practice different types of web vulnerabilities, including SQL injection, XSS, and CSRF.

### 2.5.3 TryHackMe

TryHackMe offers virtual labs and structured learning paths that guide users through various cybersecurity topics, including web application security. It is designed to be accessible to beginners and includes labs focused on popular web vulnerabilities. Training at all levels within the sector is still desperately needed [7].

## 2.6 Limitations of Existing Vulnerability Platform

Existing vulnerability testing platforms, such as Hack The Box (HTB), Damn Vulnerable Web Application (DVWA), TryHackMe offer valuable resources but are often complex to set up, lack specific focus areas, or are outdated. These limitations make them less accessible for beginners and reduce their effectiveness as educational tools [8]. This section will review the challenges associated with current platforms and discuss how this project aims to address these gaps by offering a streamlined, focused, and student-friendly environment.

## 2.7 Comparison Existing System and Proposed System

This comparison table evaluates various cybersecurity training platforms based on several key features. It compares Hack The Box (HTB), DVWA, TryHackMe, and a Proposed System. Table 1 assesses aspects such as focus on beginner friendly usage, ease of setup, progressive learning path, injection vulnerability focus, difficulty levels, realism in challenges, user experience (UX), and feedback mechanisms. This evaluation helps identify the strengths and weaknesses of each platform, guiding users to choose the most suitable one for their cybersecurity learning needs.

**Table 1** *Comparison of Existing Vulnerable Platform*

| Features | Hack The Box (HTB) | DVWA | TryHackMe | TryMeOut |
|---|---|---|---|---|
| Beginner Friendly | Moderate | Moderate | High | High |
| Ease of Setup | Moderate | Moderate | Moderate | High |
| Progressive | No | No | Yes | Yes |
| Focus on Injection | No | No | No | Yes |
| Difficulty Level | Yes | Yes | Yes | Yes |
| Realism | Yes | Yes | Yes | Yes |
| Feedback | Limited | No | Yes | Yes |

## 3. Methodology

This section describes the methodology used in the development of the TryMeOut web application. The project follows the Waterfall Model, which offers a clear and sequential approach through seven main phases: requirements, analysis, design, development, testing, deployment, and maintenance. Each phase is completed before moving on to the next, ensuring that the system is developed in a structured and manageable way.

## 3.1 Waterfall Model

For this project, the Waterfall Model was selected as the development framework due to its structured, sequential approach, which allows each phase to be fully completed before progressing to the next. This model provides clarity and organization, making it ideal for projects with expected goals and requirements, such as this platform. By using the Waterfall Model, the development process is broken down into manageable phases, ensuring thorough analysis, design, and testing before deployment as shown on Fig. 1. Communication as well as every other component is required to complete the project according to the established specifications [9].

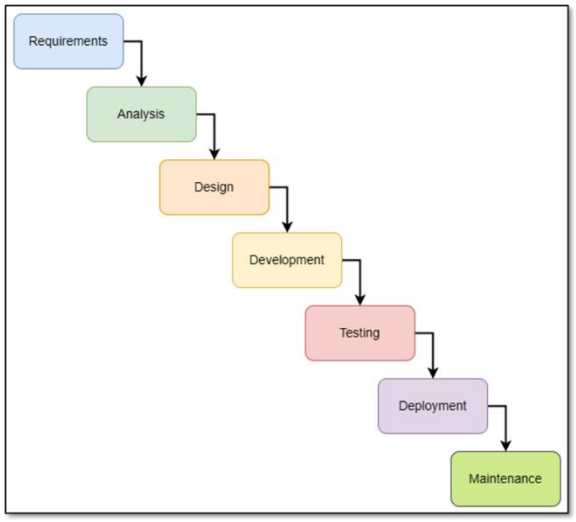



**Fig. 1** *Phases in Waterfall Model*

## 3.2 Waterfall Model Phase

Based on Table 2, it will describe the Waterfall Model phase for this project. The phases are Requirements, Analysis, Design, Development, Testing, Deployment, and Maintenance.

**Table 2** *Waterfall Model Phases*

| Phase | Task | |
|---|---|---|
| Requirements | 1. | Define system requirements |
| | 2. | Identify key features and functionalities |
| Analysis | 1. | Gather information |
| | 2. | Evaluate feasibility |
| | 3. | Identify necessary resources |
| Design | 1. | Create wireframe |
| | 2. | Create system architecture |
| Development | 1. | Develop the system using the resources using front-end and back-end technologies |
| Testing | 1. | Identify Bug during Alpha Testing |
| | 2. | Gather user feedback during Beta Testing |
| Deployment | 1. | Launch the platform on server |
| Maintenance | 1. | Regular update and improvements |

## 4. System Analysis and Design

This section takes a closer look at the analysis and design processes involved in developing the web-based vulnerability testing platform. It starts by outlining the key system requirements necessary to create a secure, functional, and easy-to-use environment.

## 4.1 Functional Requirements

Functional requirements define the specific features and operations the platform must perform. The platform will allow users to interact with simulated environments to test and exploit vulnerabilities. **Table 3** will display all the modules and functionalities.

**Table 3** *Functional Requirements*

| Module | Functionalities |
|---|---|

| | | |
|---|---|---|
| Login Module | 1. | The system should allow the users to log in into the system using username and password. |
| | 2. | The system should allow the user to log in into the system if their credential is valid. |
| | 3. | The system should notify the user if their credential is invalid. |
| | 4. | The system will redirect user to homepage upon the login success. |
| Registration Module | 1. | The system should allow users to register if they don't have any accounts created. |
| Challenge Module | 1. | The system should allow users to perform challenge. |
| | 2. | The system should be able to run challenge by user and proceed to view progress. |
| | 3. | The system should award badge to user after completion of each category. |
| | 4. | The system should award certificate to user after completion of each category. |
| Admin Module | 1. | The system should allow admin to view user information. |
| | 2. | The system should allow admin to view user progress. |
| Security Module | 1. | The system should have a secure login policy such as input sanitization and prepared statements to avoid any bypass or brute force attack. |

## 4.2 Non-Functional Requirement

Non-functional requirements address the quality and performance aspects of the platform. These include scalability, ensuring the system can handle multiple users simultaneously without performance degradation. Security is a top priority, with the platform incorporating encrypted connections, secure authentication methods, and regular updates to prevent potential exploitation. Table 4 will display all the requirements.

**Table 4** *Non-Functional Requirements*

| Requirements | Descriptions |
|---|---|
| Performance | 1. The system should load quickly and without any disruption. |
| | 2. The system should handle up to 30 students at one time. |
| Operational | 1. The system should always be available for access. |
| | 2. The system should support most common browser like, Google Chrome, Mozilla Firefox, and Microsoft Edge. |
| Usability | 1. The system should be user-friendly for students for easy navigation. |
| | 2. The system should provide clarity on the instruction given for the challenge. |
| | 3. The system should have responsive design on desktops and laptops. |
| Security | 1. The system should encrypt critical user information such as password using hashing. |
| | 2. The system should have session management to verify the user authentication. |

## 4.3 System Analysis

The system analysis phase is crucial for understanding the flow of data and the interaction between various components within the platform. In this phase, different models are used to visually represent how the system will work. These models include Data Flow Diagrams (DFD), Context Diagrams, Entity Relationship Diagrams (ERD), and System Flowcharts. They provide a clear, structured approach for designing the system and help identify key relationships and processes that will be integral to the platform's development.

## 4.4 Context Diagram

Fig. 2 presents the context diagram for TryMeOut, a high-level visualization establishing the vulnerability learning application as a central system interacting with two key external entities: Students who access security challenges and submit solutions, and Administrators who configure vulnerabilities and monitor activities. The diagram defines system boundaries by illustrating bidirectional data flows: Students exchange challenge requests and feedback with the application, while Administrators manage vulnerability configurations and receive security alerts. This representation clarifies TryMeOut's operational scope as a self-contained educational platform while highlighting critical interactions with its user ecosystem.

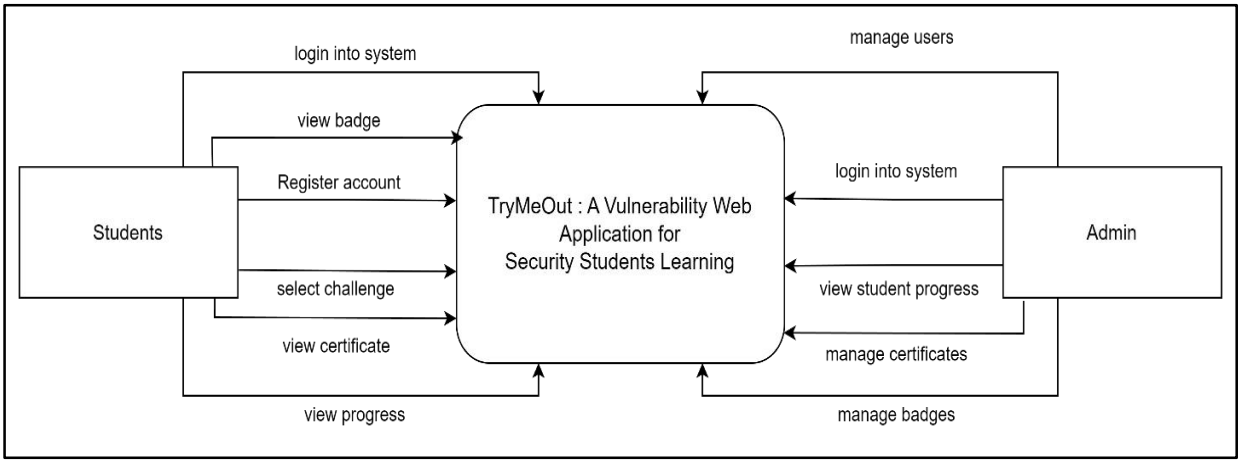



**Fig. 2** *Context Diagram*

## 4.5 Data Flow Diagram (DFD)

An illustration of how data moves through the system and is processed is shown by a data flow diagram (DFD) for the platform on Fig. 3.

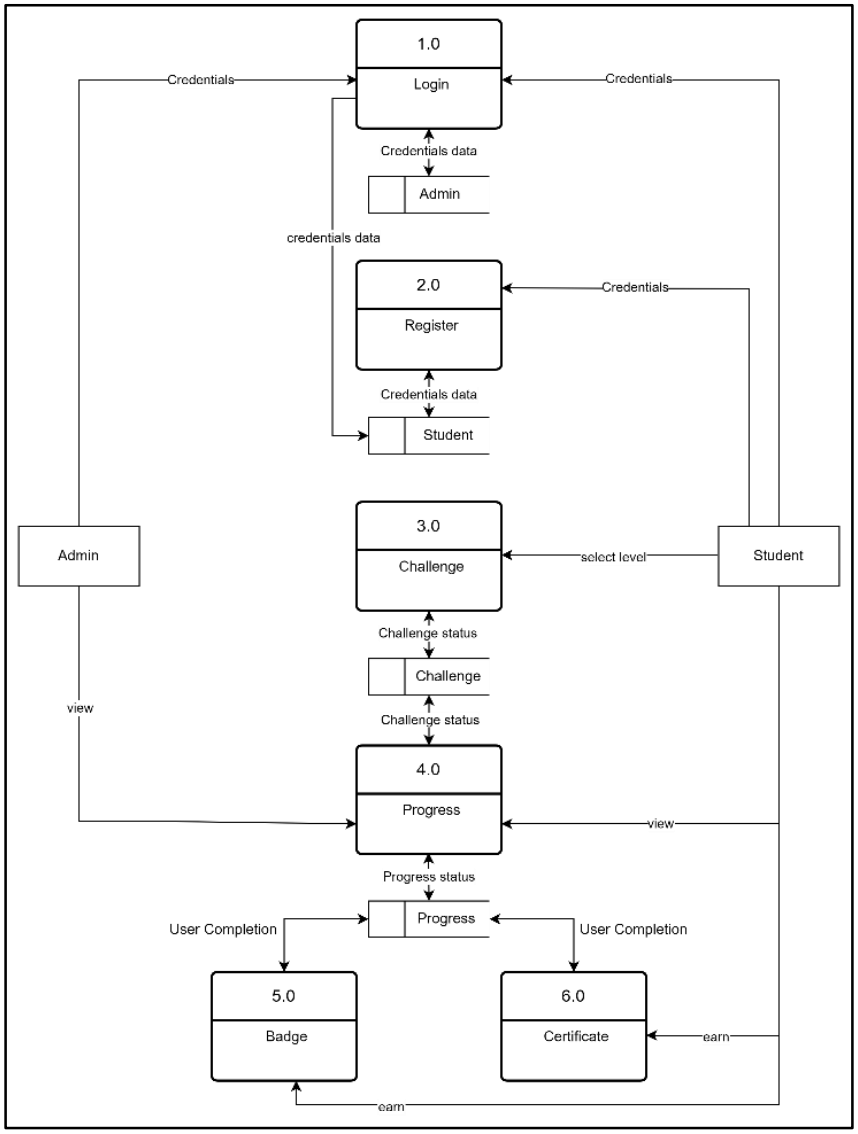



**Fig. 3** *Data Flow Diagram*

## 4.6 Entity Relationship Diagram

Fig. 4 show an Entity Relationship Diagram (ERD) illustrates the relationships between different entities in the system and the data stored within. In the context of the vulnerability testing platform, entities may include users, test results, vulnerability tests, and reports. Each entity relates to relationships that describe how they interact with one another.

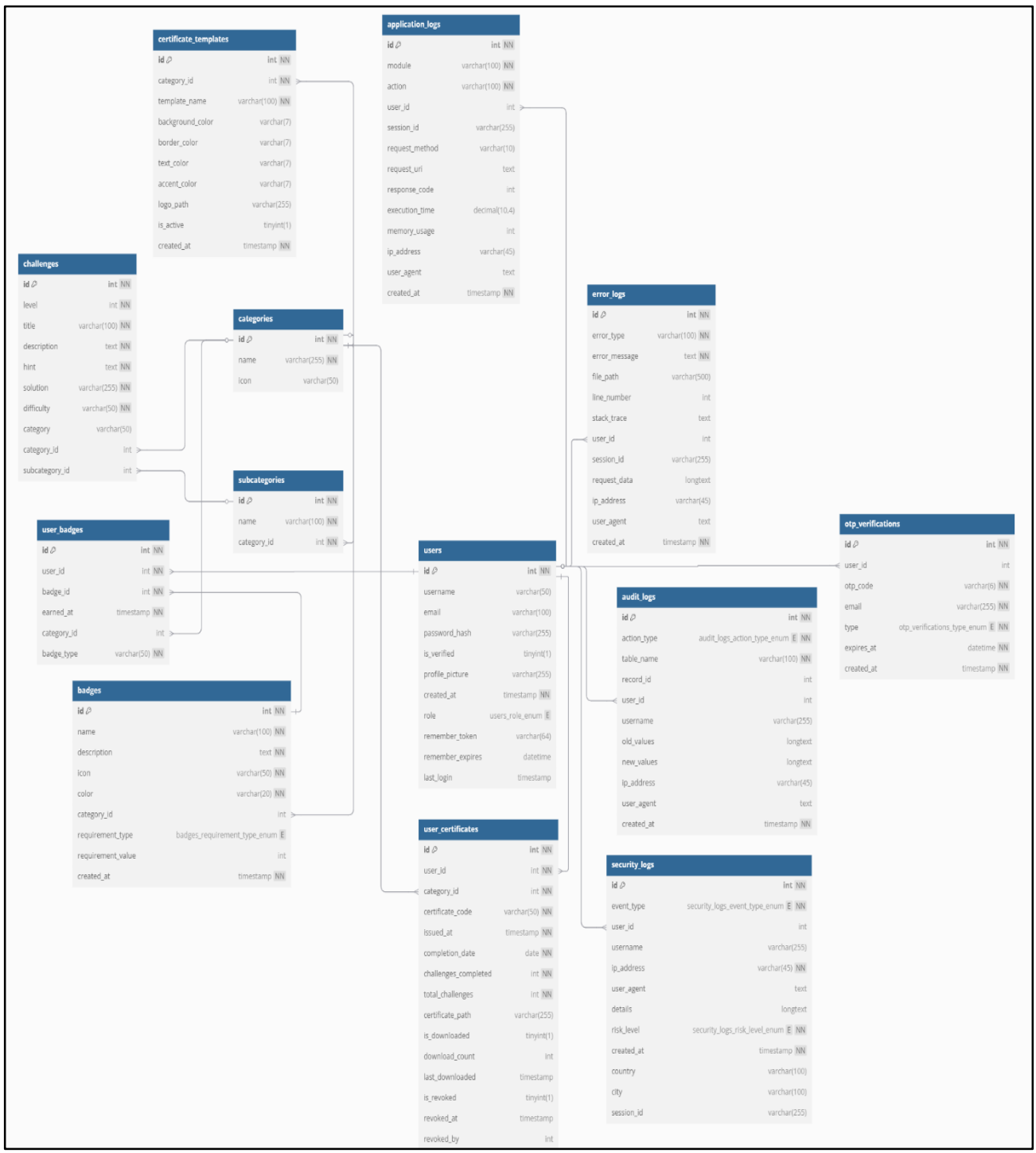



**Fig. 4** *Entity Relationship Diagram*

Applied Information Technology and Computer Science Vol. 6 No. 2 (2025) p. 644-660

## 4.7 System Flowchart

The System Flowchart provides a detailed step-by-step visual representation of the system's processes and their sequences. Fig. 5 shows the logical flow of actions that occur during interactions with the platform.

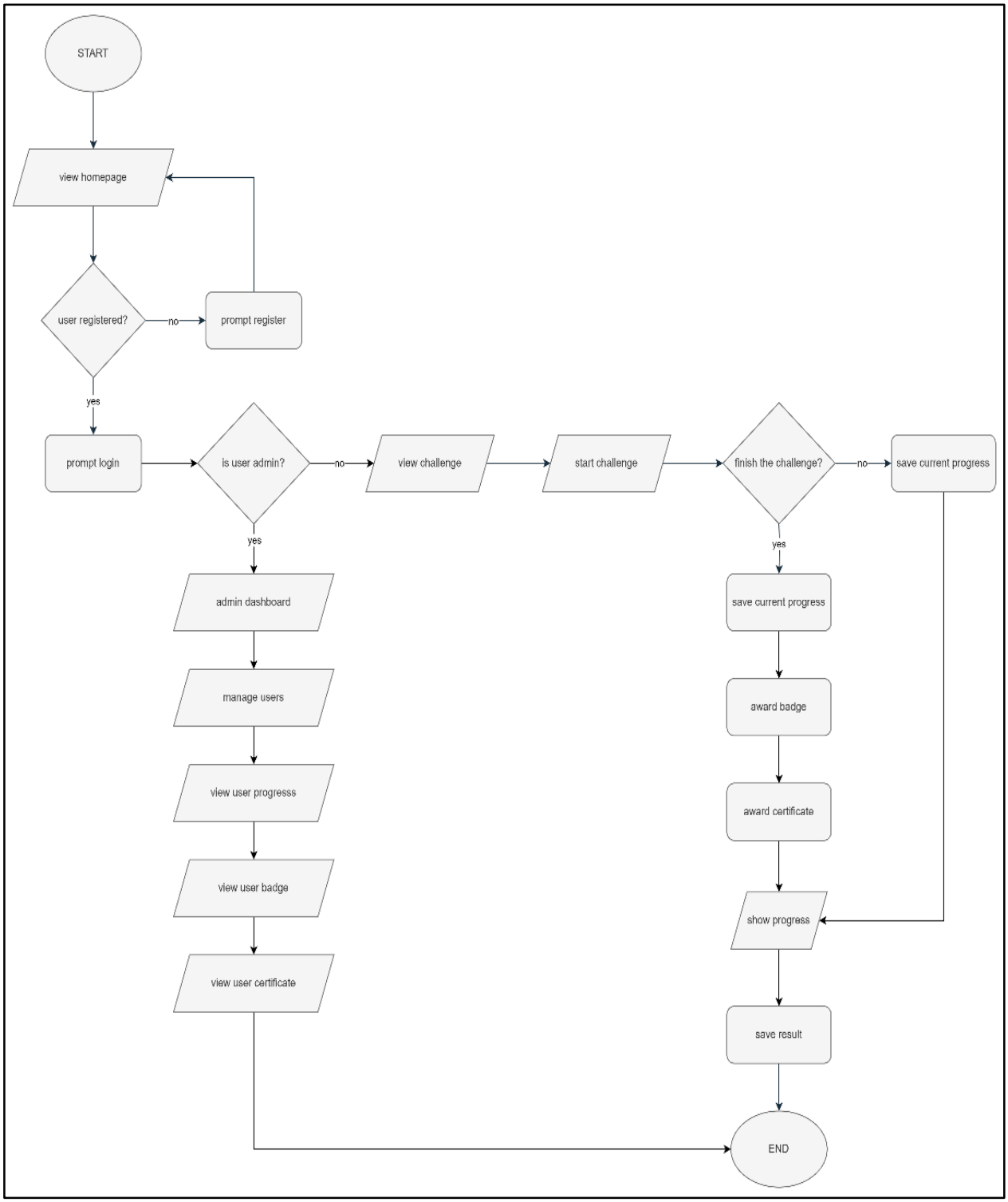



**Fig. 5** *System Flowchart*

## 4.8 System Architecture

The system architecture in Fig. 6 for the vulnerability testing platform is based on a client-server model, where the client (user interface) interacts with the server (backend system) to perform actions like user registration, performing challenge, and saving the progress. The architecture is designed to ensure a smooth interaction between various components.

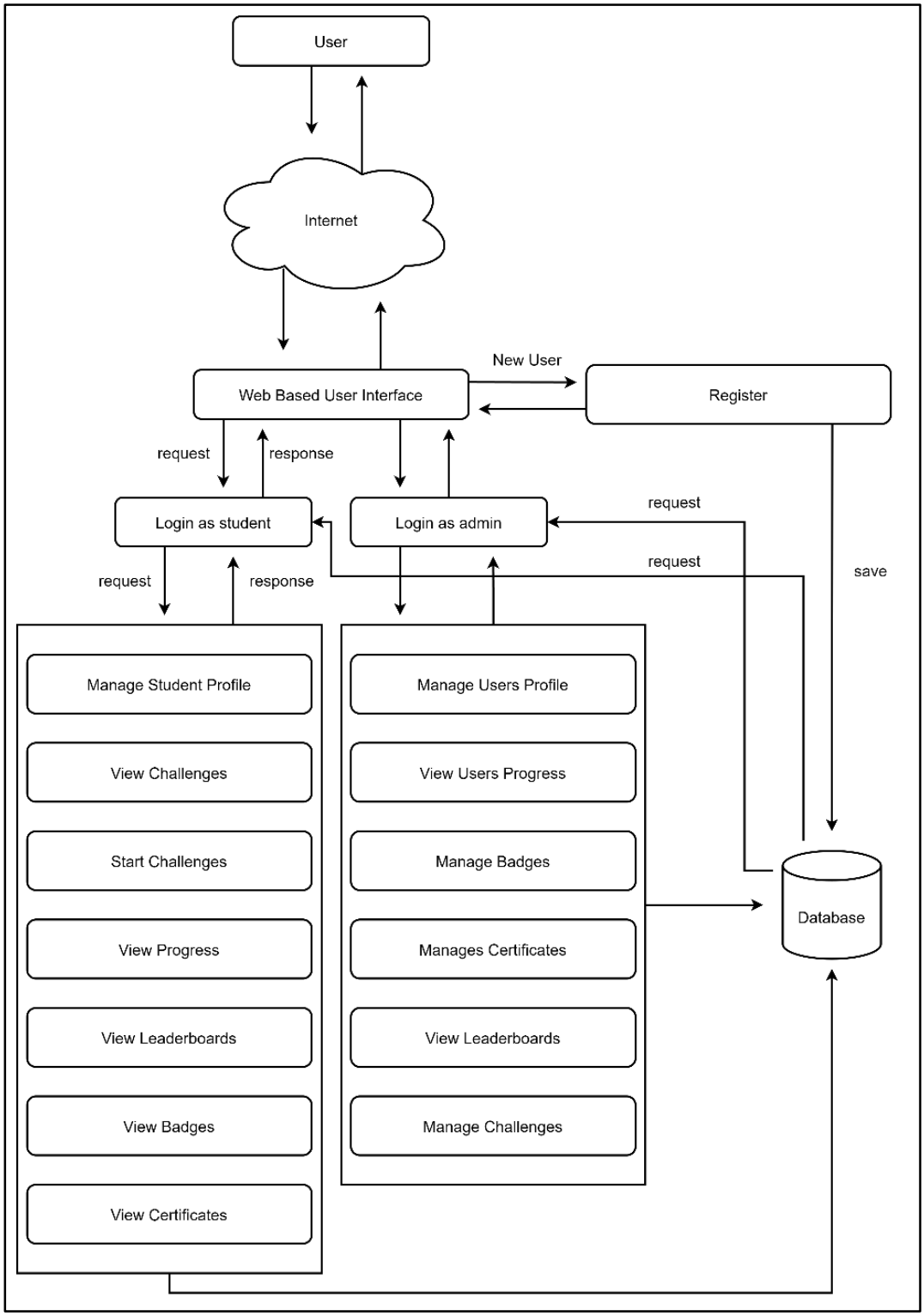



**Fig. 6** *System Architecture*

## 4.9 Interface Design

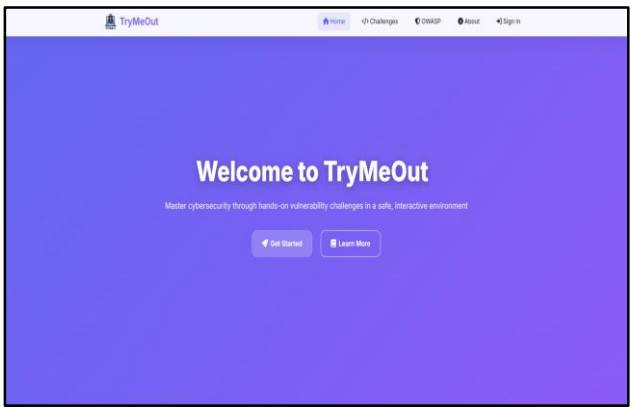
The interface design aims to create a user-friendly, intuitive, and responsive environment that allows cybersecurity students to easily navigate the platform and perform vulnerability exercise. The design prioritizes simplicity, accessibility, and functionality, ensuring users can focus on learning and testing vulnerabilities without unnecessary complexity. A design must include several components to form, integrate, and arrange them in a way that produces an appealing, fulfilling form that adds aesthetic value and a lovely visual experience[10].

## 4.10 Implementation

The TryMeOut platform was implemented with a focus on creating an intuitive, engaging, and educational user experience for cybersecurity students. The interface balances visual appeal with functional efficiency to support effective learning.

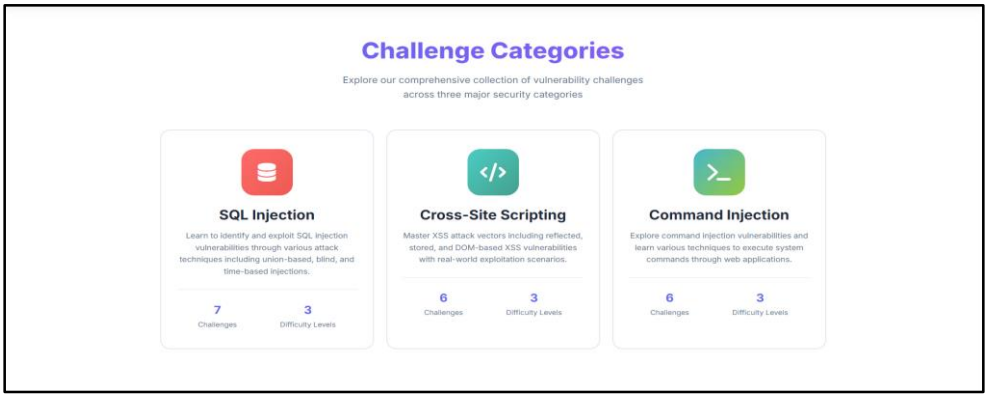
### 4.10.1 Homepage

Homepage implementation features a responsive design with three key sections. Fig. 7 showcases an animated hero section with rotating text phrases that communicate the platform's purpose. Presents core features through visually appealing cards with icons and concise descriptions. Also includes testimonials and call-to-action elements with gradient styling to encourage user registration.




(a)


(b)

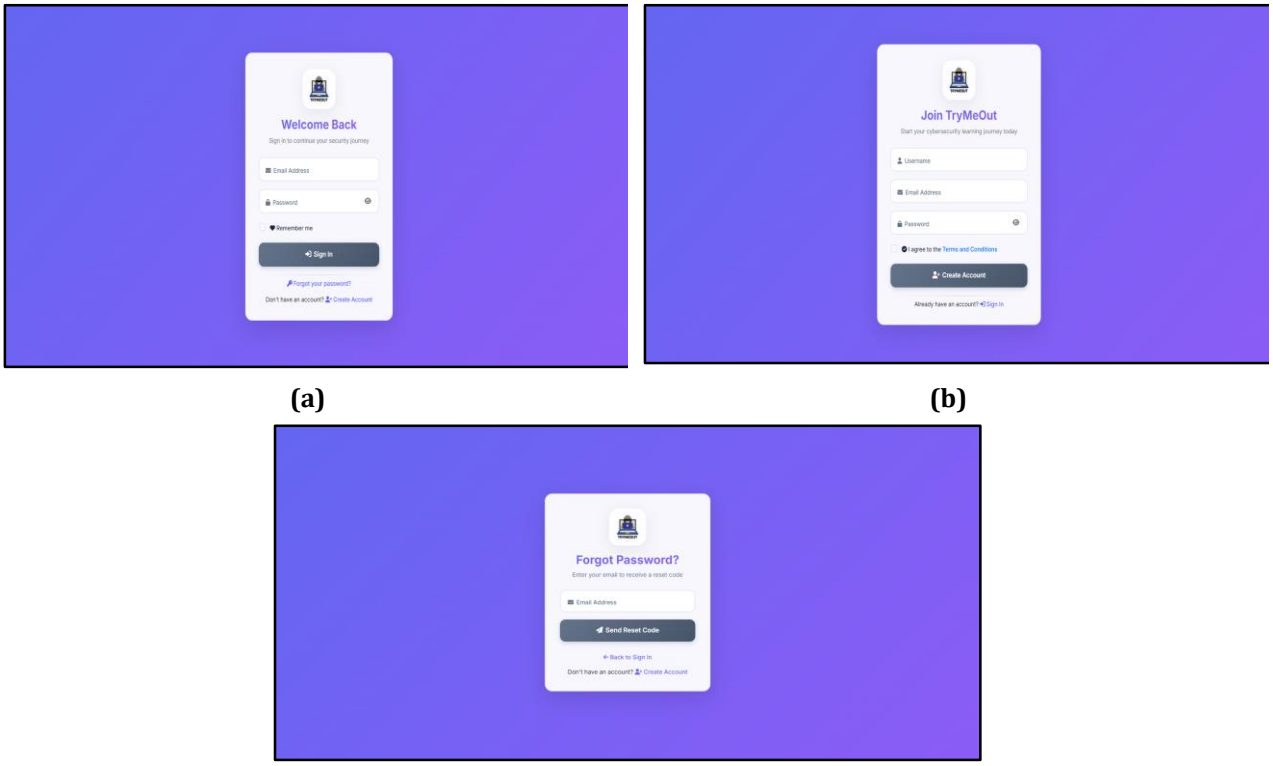



(c)

**Fig. 7** *Homepage(a); Homepage(b); Homepage(c)*

### 4.10.2 User Login and Registration

The authentication system implements security best practices while maintaining usability. Fig. 8 show The Sign In Page features a clean design with form validation and protection against common attacks. The Sign Up Page collects essential information with email validation and password strength requirements. The Forgot Password page implements a secure token-based recovery system that sends verification codes via email.



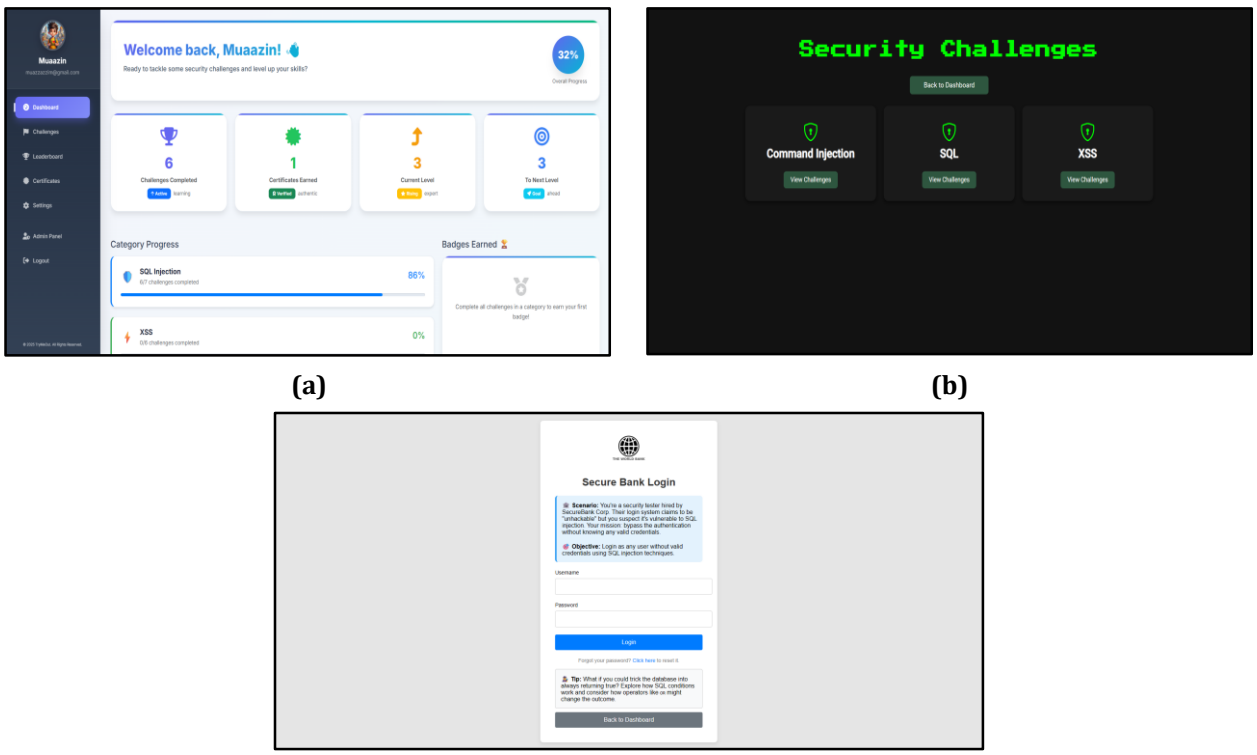

**(a)** **(b)**

**(c)**

**Fig. 8** *Sign In Page(a); Sign Up Page(b); Forgot Password(c)*

### 4.10.3 Dashboard and Challenge Module

The learning environment facilitates effective cybersecurity education through personalized interfaces. Fig. 9 show the Dashboard Page that displays student progress with visual statistics and achievement tracking. The Start Challenge Page organizes challenges by category and difficulty with detailed descriptions. The Challenge Page features a terminal-inspired design with interactive input fields and a progressive hint system.



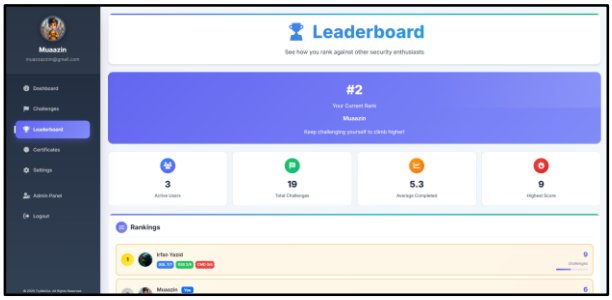

**(a)** **(b)**

**(c)**

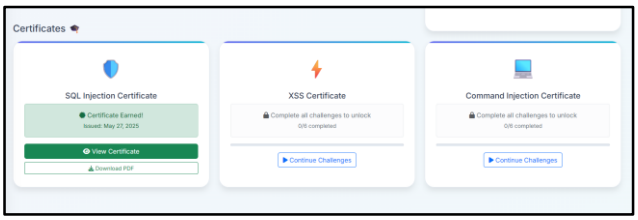
**Fig. 9** *Student Dashboard Page(a); Start Challenge Page(b); Challenge Page(c)*

### 4.10.4 Leaderboard and Certificate

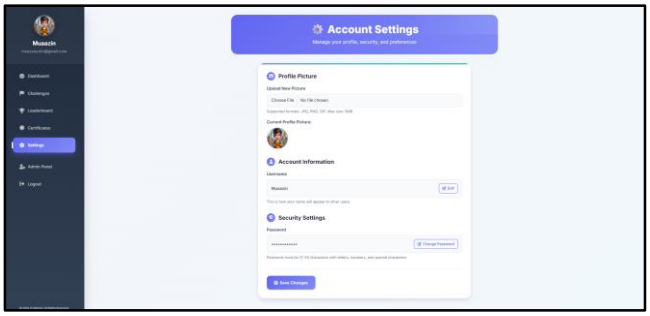
The platform enhances engagement through gamification and personalization features. Fig. 10 show the Leaderboard Page implements a real-time ranking system based on challenge completions. The Certificate Page automatically generates personalized certificates with unique verification codes upon category completion. The Settings Page provides comprehensive account management with secure profile picture uploads and information updates.




(a)




(b)




(c)

**Fig. 10** *Leaderboard Page(a); Certificate Page(b); Settings Page(c)*

### 4.10.5 Admin Module

The Admin Module is designed to facilitate the management and monitoring of the platform's users and their activities as shown on Fig. 11. This module provides administrative users with access to essential functionalities that ensure the smooth operation of the vulnerability testing platform. Administrators could view registered user information, monitor student progress across different challenge categories, and ensure the system is being used effectively.

One of the key features of this module is real-time user progress tracking, which allows admins to oversee each user's challenge completion status and performance. This is useful for evaluating learning outcomes and identifying students who may need additional support. Additionally, the module includes security mechanisms such as session management, access control, and input sanitization to prevent unauthorized access and maintain data integrity.

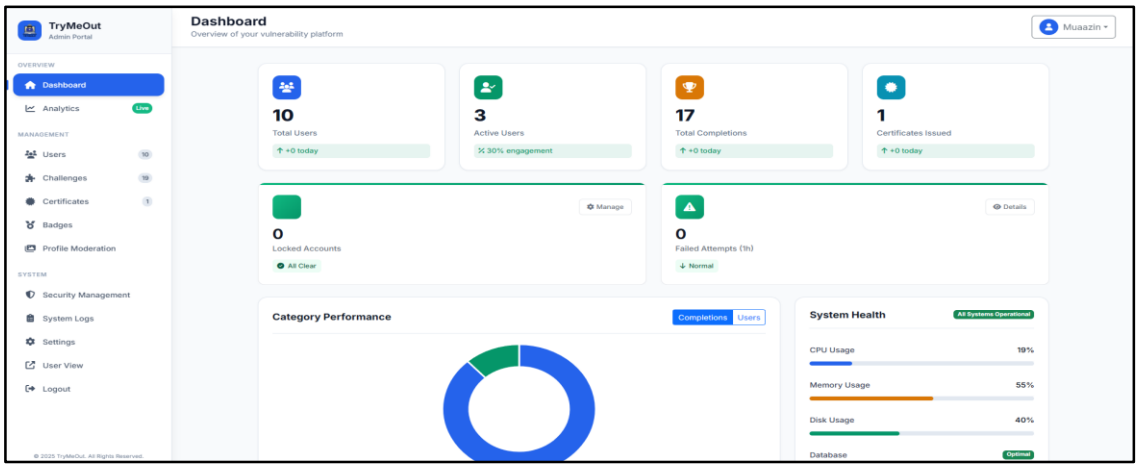



**Fig. 11** *Admin Dashboard*

Fig. 12 shows the dashboard of security management that displays information such as current locked accounts and recent failed attempts.

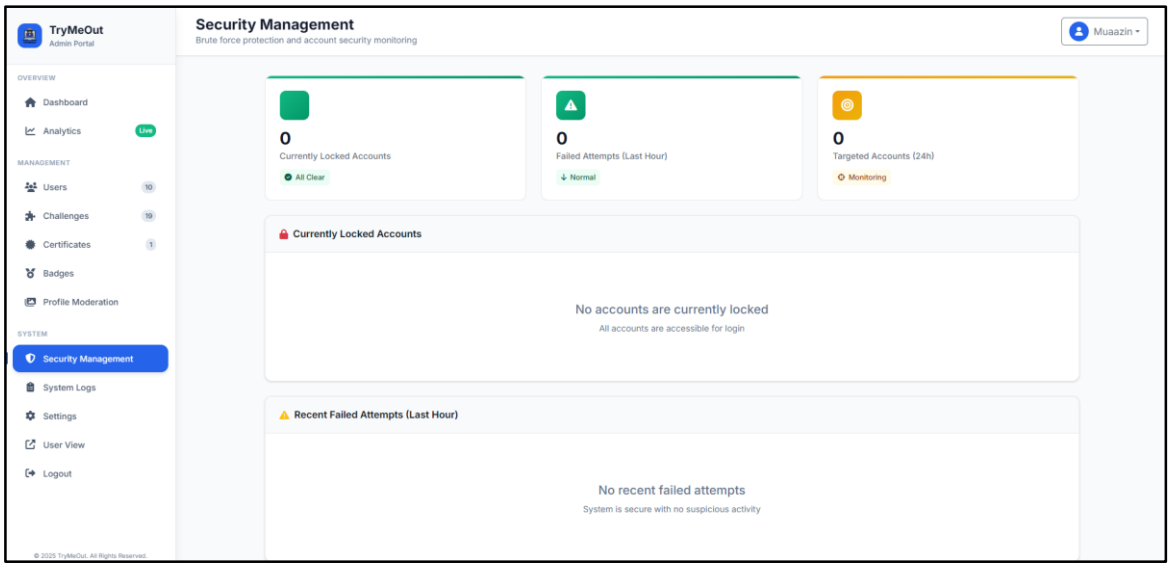



**Fig. 12** *Security Management*

## 4.11 Security Implementation

The TryMeOut platform implements comprehensive security measures to protect user data while providing a controlled environment for learning about vulnerabilities. The security implementation follows industry best practices and creates a clear separation between the educational vulnerability demonstrations and the core platform infrastructure.

User authentication employs secure password hashing with modern algorithms and salt techniques to prevent credential theft as shown on Fig. 13. All sensitive user data is encrypted both in transit using HTTPS and at rest in the database. The platform implements robust input validation and output encoding throughout the application to prevent injection attacks outside of designated challenge areas.

```php
// Hash the password before saving
$hashedPassword = password_hash($password, PASSWORD_DEFAULT);
```

**Fig. 13** *Password Hashing and Salt implemented*

Fig. 14 show a standout security feature is the brute force protection system that tracks failed login attempts and temporarily locks accounts after suspicious activity. This system records IP addresses, timestamps, and user agents for each failed attempt, implementing progressive lockout periods for repeated failures. When an account becomes locked, users receive clear notifications about the lockout duration and remaining time before they can attempt login again.

```php
/**
 * Record a failed login attempt
 */
public function recordFailedAttempt($email, $ip_address = null, $user_agent = null) {
    try {
        $ip_address = $ip_address ?: $this->getClientIp();
        $user_agent = $user_agent ?: ($_SERVER['HTTP_USER_AGENT'] ?? '');

        // Record the failed attempt
        $stmt = $this->pdo->prepare("
            INSERT INTO brute_force_attempts (email, ip_address, user_agent)
            VALUES (?, ?, ?)
        ");
        $stmt->execute([$email, $ip_address, $user_agent]);
```

**Fig. 14** *Brute force protection*

The file upload system incorporates multiple security layers, including file type validation, content scanning for malicious code as shown on Fig. 15, and secure storage practices. Session management implements secure cookie handling with appropriate flags and timeout controls, while CSRF tokens protect all forms against cross-site request forgery attacks.

```php
// Security patterns to detect malicious content
define('MALICIOUS_PATTERNS', [
    '<?php',
    '<?=',
    '<script',
    'javascript:',
    'vbscript:',
    'onload=',
    'onerror=',
    'eval(',
    'base64_decode',
    'shell_exec',
    'system(',
    'exec(',
    'passthru(',
    'file_get_contents',
    'file_put_contents',
    'fopen(',
    'fwrite(',
    'include(',
    'require(',
    'include_once(',
    'require_once('
]);
```

**Fig. 15** *File Content Scanning*

## 5. Result and Discussion

The development of the TryMeOut platform successfully achieved its objective of providing a practical, web-based environment for cybersecurity students to explore and understand common web application vulnerabilities. The system was tested with a series of SQL Injection, Cross-Site Scripting (XSS), and Command Injection challenges, all of which were designed to simulate real-world scenarios in a controlled and educational setting.

### 5.1 System Functionality Testing

The platform was evaluated based on functionality, usability, and educational impact. Functionally, the system operated as intended as shown on Table 5. Each vulnerability scenario was successfully executed, and the corresponding exploitation outcomes were accurately displayed. Students were able to progress through different challenge levels, with each level increasing in complexity. This progression supported deeper learning and maintained engagement throughout the training process.

**Table 5** *System Functionality Testing*

| Description | Pass | Fail |
|---|---|---|
| The system should load quickly and without any disruption. | 5 | 0 |
| The system should handle up to 30 students at one time. | 5 | 0 |
| The system should always be available for access. | 5 | 0 |
| The system should support common browser like, Google Chrome, Mozilla Firefox, Microsoft Edge. | 5 | 0 |
| The system should be user-friendly for students for easy navigation. | 5 | 0 |
| The system should provide clarity on the instruction given for the challenge. | 5 | 0 |
| The system should have responsive design on desktops and laptops. | 5 | 0 |
| The system should encrypt critical user information such as password using hashing. | 5 | 0 |
| The system should have session management to verify the user authentication. | 5 | 0 |
| The system should allow users to register if they don't have any accounts created. | 5 | 0 |
| The system should allow users to perform challenges. | 5 | 0 |
| The system should be able to run challenges by users and proceed to view progress. | 5 | 0 |
| The system should award badges to users after completion of each category. | 5 | 0 |
| The system should award certificates to users after completion of each category. | 5 | 0 |
| The system should allow admin to view user information. | 5 | 0 |
| The system should allow admins to view user progress. | 5 | 0 |
| The system should have input sanitization and prepared statements to avoid any bypass or brute force attack. | 5 | 0 |

### 5.1.1 Admin Functionality Testing

Admin functionality testing was conducted to verify the features available to administrators on the TryMeOut platform. The goal was to ensure that the admin panel worked as intended, allowing full control over user management, challenge monitoring, and system maintenance. Each feature was manually tested by simulating common administrative tasks. The tests confirmed that the admin panel functioned as expected. Admins were able to view user information, monitor completed challenges, and manage content dynamically without any critical issues. Minor improvements were identified, such as enhancing the interface layout for better readability and adding confirmation prompts for sensitive actions like deleting challenges. These will be considered for future updates. Table 6 shows the testing concluded that the admin module is stable, functional, and capable of supporting the platform's operational needs effectively.

**Table 6** *Admin Functionality Testing*

| Descriptions | Pass | Fail |
|---|---|---|
| Admin should be able to login with credentials successfully | 5 | 0 |
| Admin should be redirect to admin dashboard after login | 5 | 0 |
| Admin can navigate to each section on the sidebar | 5 | 0 |
| Admin should be able to view all user list | 5 | 0 |
| Admin should be able to edit user details | 5 | 0 |
| Admin should be able to view user progress | 5 | 0 |
| Admin should be able view user awarded badge | 5 | 0 |
| Admin should be able to view user awarded certificate | 5 | 0 |
| Admin should be able to revoke user badge | 5 | 0 |
| Admin should be able to revoke user certificate | 5 | 0 |
| Admin should be able to delete user | 5 | 0 |
| Admin should be able to generate user list report | 5 | 0 |
| Admin should be able to generate user progress report | 5 | 0 |
| Admin should be able to generate user certificate report | 5 | 0 |

### 5.1.2 User Functionality Testing

User functionality testing focused on evaluating the interactive experience of students using the platform as shown on Table 7. The main objective was to confirm that users could smoothly perform essential tasks such as registering and logging into the system, navigating between challenge levels, attempting SQL injection, XSS, and command injection challenges, receiving immediate feedback on their progress, viewing their personal challenge history and scores.

**Table 7** *User Functionality Testing*

| Descriptions | Pass | Fail |
|---|---|---|
| User should be able to login with credentials successfully | 5 | 0 |
| User should be redirect to user dashboard after login | 5 | 0 |
| User can navigate to each section on the sidebar | 5 | 0 |
| User should be able to view all challenge list | 5 | 0 |
| User should be able to perform each challenge | 5 | 0 |
| User should be awarded badge upon completion of each category | 5 | 0 |
| User should be awarded certificate upon completion of each category | 5 | 0 |
| User should be able to change username | 5 | 0 |
| User should be able to change password | 5 | 0 |
| User should be able to change profile picture | 5 | 0 |
| User should be able to generate certificate upon completion of each category | 5 | 0 |
| User should be able to download certificate | 5 | 0 |
| User should be able to view leaderboard | 5 | 0 |

## 5.2 User Acceptance Testing

User Acceptance Testing (UAT) was conducted to ensure that the TryMeOut platform met its functional, usability, and educational objectives from the perspective of its intended users, cybersecurity students at Universiti Tun Hussein Onn Malaysia (UTHM). The primary goal was to validate whether the system fulfilled its purpose as a hands-on learning tool for web application vulnerabilities, particularly focusing on SQL Injection, XSS, and Command Injection.

A selected group of students, lecturer from the Bachelor of Information Security program participated in the testing. They were asked to register, navigate the platform, attempt various vulnerability challenges, and provide feedback on their overall experience as shown on Fig. 16. Fig. 17 show a structured UAT form was used to assess areas such as system responsiveness, challenge clarity, learning effectiveness, and interface usability.

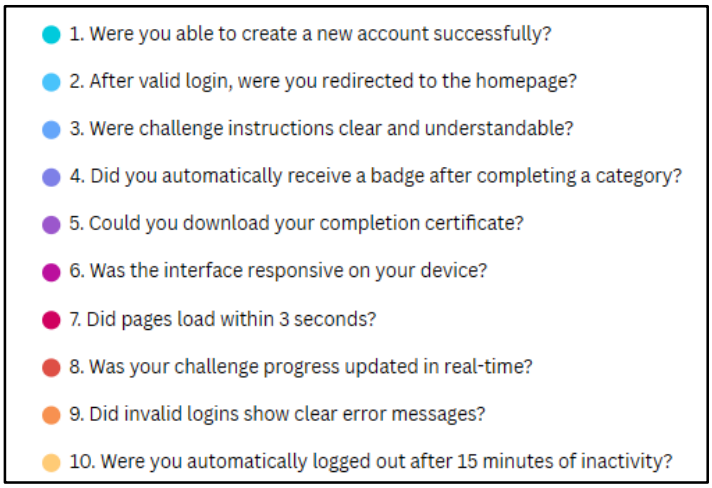



**Fig. 16** *User Acceptance Testing Question*

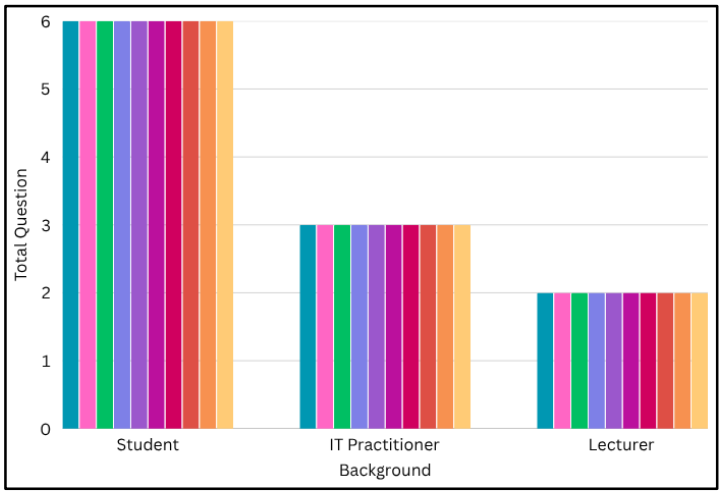



**Fig. 17** *User Acceptance Testing Response Chart*

## 6. Conclusion

To conclude, TryMeOut: A Vulnerability Web Application for Security Students is designed as a practical learning platform that gives students a hands-on approach to understanding web application vulnerabilities. The system is not limited to SQL injection alone but also includes important topics like cross-site scripting and command injection, offering a broader view of real-world security issues commonly found in modern web systems.

This platform provides a guided environment where students can move through different challenges step by step, starting with basic tasks and progressing to more advanced scenarios. It allows them to apply what they have learned in lectures and explore how these vulnerabilities work in practice, all within a safe and controlled setup.

The main goal is to support students at Universiti Tun Hussein Onn Malaysia who are studying information security by giving them a meaningful way to build their technical skills. TryMeOut encourages a deeper understanding of how these attacks work and how they can be prevented, helping students to become more capable and confident in handling cybersecurity threats.

More than just a learning tool, this project aims to close the gap between theoretical study and practical experience. By offering real challenges in a structured format, it helps students prepare for future careers in cybersecurity and strengthens their foundation in web application security.

## Acknowledgement

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support.

## Conflict of Interest

Authors declare that there is no conflict of interest regarding the publication of the paper.

## Author Contribution

*The authors confirm contribution to the paper as follows:* ***study conception and design:*** *M. M. Abdullah, K. A. Mohamad Sukri;* ***data collection:*** *M. M. Abdullah, K. A. Mohamad Sukri,* ***analysis and interpretation of results:*** *M. M. Abdullah, K. A. Mohamad Sukri****; draft manuscript preparation:*** *M. M. Abdullah, K. A. Mohamad Sukri. All authors reviewed the results and approved the final version of the manuscript.*

## References

[1] C. Phuong, N. Saied, and L. Yang, "A Hands-on Education Framework for Cybersecurity," Proceedings Frontiers in Education Conference, FIE, 2023, doi: 10.1109/FIE58773.2023.10343268.
[2] "View of SQL Injection Detection using Machine Learning: A Review." Accessed: Jun. 13, 2025. [Online]. Available: https://mjosht.usim.edu.my/index.php/mjosht/article/view/368/220
[3] J. Harish Kumar and J. J. Godwin Ponsam, "Cross Site Scripting (XSS) vulnerability detection using Machine Learning and Statistical Analysis," 2023 International Conference on Computer Communication and Informatics, ICCCI 2023, 2023, doi: 10.1109/ICCCI56745.2023.10128470.
[4] O. Akinmerese et al., "Defence Against Command Injection Attacks in a Distributed Network Environment," OAlib, vol. 11, no. 05, pp. 1–14, 2024, doi: 10.4236/OALIB.1111491.
[5] "(PDF) Millennial Psychology Towards Hacking Activities." Accessed: Jun. 11, 2025. [Online]. Available: https://www.researchgate.net/publication/359391572_Millennial_Psychology_Towards_Hacking_Activities
[6] "Hack The Box: A Methodical Guide to Ethical Hacking | by Logan Hugli | Medium." Accessed: Jun. 11, 2025. [Online]. Available: https://medium.com/@lhugli/hack-the-box-a-methodical-guide-to-ethical-hacking-187bac1f9670
[7] "TryHackMe | NEW Cyber Security Skills UK Report: Our Expert Roundup!" Accessed: Jun. 11, 2025. [Online]. Available: https://tryhackme.com/resources/blog/cyber-security-skills-report-2024
[8] G. Thaqi, "Gent Thaqi Creating Labs for Ethical Hacking Course Based on WiFi Pineapple and USB Rubber Ducky Title: Creating Labs for Ethical Hacking Course Based on WiFi Pineapple and USB Rubber Ducky Number of Pages: 36 pages + 2 appendices," 2024.
[9] B. Blaskovics, J. Czifra, G. Klimkó, and P. Szontágh, "Impact of the Applied Project Management Methodology on the Perceived Level of Creativity," Acta Polytechnica Hungarica, vol. 20, no. 3, pp. 2023–101.
[10] "View of Implementation Of UI/UX Concepts And Techniques In Web Layout Design With Figma." Accessed: Jun. 11, 2025. [Online]. Available: http://103.241.192.17/~jurnalunidha/index.php/jteksis/article/view/1223/758