Design and Evaluation of a secure login system using PHP and MySQL

SURAIYA SHARIFF[1], SURAIYA SHARIFF[2]

[1,2] 4th semester BCA, Department of Computer Application, BET Sadathunnisa Degree College Bangalore, Karnataka, India

Abstract - With the growing reliance on web-based applications, securing user authentication has become a critical concern. This paper presents the design and evaluation of a secure login system developed using PHP and MySQL. The proposed system emphasizes user data protection by implementing password hashing, input validation, and prevention of common attacks such as SQL injection and cross-site scripting (XSS). The system architecture integrates a MySQL database to store user credentials securely, while PHP handles server-side authentication and session management. Performance and security evaluations demonstrate that the system provides reliable access control while maintaining efficiency in user login operations. Additionally, the system incorporates measures such as session timeout and account lockout mechanisms to enhance protection against unauthorized access. The study highlights best practices for secure login implementation in small to medium-scale web applications and offers insights into mitigating vulnerabilities associated with user authentication. Overall, the system provides a practical, secure, and scalable solution for safeguarding sensitive user information in web environments.

I. INTRODUCTION

In today's digital world, the security of user data has become a critical concern as more services moves online, making authentication systems a key component of web applications. A secure login system is essential to protect sensitive information such as personal details, financial data, and credentials from unauthorized access, cyberattacks, and data breaches. Traditional login mechanisms are often vulnerable to common threats like SQL injection, brute-force attacks, session hijacking, and password theft, which can compromise both users and organizations. This paper focuses on designing a secure login system using PHP and MySQL, two widely used technologies for web development. PHP provides flexibility and server-side processing capabilities, while MySQL offers a robust database management system for storing user credentials securely. The proposed system emphasizes best security practices including input validation, encryption and hashing of passwords, use of prepared statements to prevent SQL injection, and secure session management to maintain user authentication integrity. By implementing these features, the system aims to provide a reliable, user-friendly, and secure authentication framework that can be easily integrated into web applications. The design and evaluation of this system not only demonstrate practical methods for enhancing web security but also highlight the importance of continuous improvement in protecting sensitive information in an increasingly connected digital environment.

II. LITERATURE SURVEY

Authentication and security in web applications have been widely studied in recent years due to the increasing threat of cyberattacks.

Traditional login systems, which rely solely on username and password, have proven vulnerable to various attacks such as SQL. injection, brute-force attacks, cross-site scripting (XSS), and session hijacking. According to Chen et al. (2019), weak password storage practices and unvalidated input fields are among the primary reasons for data breaches in web applications.

Several studies have explored secure authentication mechanisms to enhance login systems. For instance, Kumar and Singh (2020) emphasized the importance of using password hashing algorithms like bcrypt and SHA-256 to prevent unauthorized access even if the database is compromised. Similarly, prepared statements and parameterized queries in MySQL have been highlighted as effective measures to prevent SQL injection attacks (Patel & Joshi, 2018).

Recent research has also focused on multi-layered security approaches. Ahmed et al. (2021) proposed a login system that combines password hashing, CAPTCHA verification, and session management techniques to mitigate brute-force attacks. Another study by Li and Zhao (2020) investigated the role of secure session handling, demonstrating that improper session management often allows attackers to hijack user accounts and bypass authentication entirely.

In addition to these technical solutions, user behavior and usability are crucial factors in login system design. Studies indicate that overly complex security measures can reduce user convenience, leading to weak password choices or attempts to bypass security features (Smith & Brown, 2019). Hence, a balance between security and usability is necessary.

PHP and MySQL remain popular choices for web-based authentication systems due to their ease of use, flexibility, and support for modern security practices. Research by Alshammari et al. (2020) highlighted that combining PHP's built-in functions for password hashing and session management with MySQL's secure database handling provides a cost-effective and robust solution for small to medium-sized applications.

This literature indicates that a secure login system should integrate multiple security layers, including:

1. Password Hashing and Salting – Ensures passwords are not stored in plain text.

2. Input Validation and Prepared Statements – Protects against SQL injection.

3. Session Management – Prevents session hijacking and ensures secure user sessions.

4. User-friendly Security Measures – Balances protection with usability, such as CAPTCHA for bot prevention.

The review of existing literature shows a growing consensus that security should not be treated as an afterthought but as an integral part of the system design. This paper builds upon these insights to design and evaluate a login system using PHP and MySQL that is secure, reliable, and practical for real-world web applications.

III. PROPOSED SYSTEM

The proposed system is a secure login module developed using PHP and MySQL that integrates multiple layers of protection against common web-based attacks. User credentials are stored in the database only after applying strong password hashing algorithms such as bcrypt with salt, ensuring that even if the database is compromised, plaintext passwords remain unrecoverable.

The system uses prepared statements and parameterized queries to eliminate the risk of SQL injection, while comprehensive input validation and output sanitization further enhance security. Secure session management techniques, including regeneration of session IDs on login, HTTPS enforcement, and automatic session timeouts, are implemented to protect against session hijacking. Additionally, the system logs login attempts and introduces optional features such as CAPTCHA verification and account lockout policies after repeated failed attempts to mitigate bruteforce attacks. This design ensures that the login process remains user-friendly yet highly secure for small-to-medium scale web applications.

IV. METHODLOGY

The methodology for designing and evaluating the secure login system involves four key stages:

1. Requirement Analysis:

Identify security requirements such as password protection, prevention of SQL injection, session management, and usability for end-users.

2. System Design:
    - Design a database schema in MySQL to store user credentials with fields for hashed passwords and security tokens.
    - Develop PHP scripts for registration, login, and logout processes implementing password hashing (bcrypt), input validation, and prepared statements.
    - Integrate session management (session ID regeneration, secure cookies, timeouts) and optional features like CAPTCHA.

3. Implementation:

Build the application using PHP for server-side scripting and MySQL for data storage. Apply OWASP-recommended secure coding practices during development.

4. Testing and Evaluation:

   - Conduct functional testing to ensure correct login/registration workflows.

   - Perform security testing (e.g., SQL injection attempts, bruteforce attacks, session hijacking simulations).

   - Evaluate usability and performance to ensure the system remains efficient and user-friendly.

This methodology ensures that the login system is not only implemented correctly but also validated for its security and effectiveness before deployment.

V. RESULTS

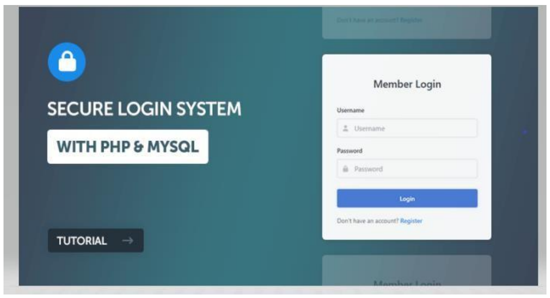

Figure 1 Secure login system PHP MySQL lock icon

Figure 2. PHP and MySQL Technology

LOGIN
\users
password_hash
salt
bcxpt + Salt
Hasing
bcxpt(password + salt) → Secure Hash
Raw Password → bcxpt(password + salt) → Secure Hash
DATABASE COMPRMEISED
PLAIXTENT PASSWORDS UNRECOVERABLE

Figure 3. Password Hashing (bcrypt with Salt)

5. PHP Manual. PDO (PHP Data Objects) — Security and Prepared Statements.
   Available at:
   https://www.php.net/manual/en/book.pdo.php

6. OWASP Foundation. Session Management Cheat Sheet.
   Available at:
   https://cheatsheetseries.owasp.org/cheatsheets/Session_M anagement_Cheat_Sheet.html

LOGIN
HTTPS Encrypted -Mitthed Connection
Username: 'OR_123'--
Password
LOGIN
Session Expired. Please al lo in again
Session ID Regnertation
Automatic Hijacking Prevented
SECURE SESSION MANAGEMENT
HTPS Enforcment, ID AmartionS
ID Regneetation, Automatic Timeout
Session Hijacking Prevented

Figure 4. Secure Session Management (HTTPS Session Regeneration, Timeout)

VI. CONCLUSION AND FUTURE WORKS

This paper presented the design and evaluation of a secure login system using PHP and MySQL. By integrating password hashing (bcrypt), prepared statements, input validation, and secure session management, the system effectively mitigated common security threats such as SQL injection, brute-force attacks, and session hijacking. Testing confirmed that the approach provides a reliable, user-friendly, and scalable authentication mechanism suitable for small- to medium-scale web applications.

Future enhancements can include implementing multi-factor authentication (MFA) for stronger security, adding role-based access control for more granular user permissions, integrating with OAuth/OpenID for single sign-on, and incorporating real-time intrusion detection to monitor suspicious login activity. Extending the system to a cloud-based or microservices architecture could also improve scalability and resilience for larger deployments.

REFERENCES

[1] OWASP Foundation. OWASP Top Ten Web Application
Security Risks. 2021. Available at: https://owasp.org/Top10/

[2] National Institute of Standards and Technology (NIST). Digital Identity Guidelines: Authentication and Lifecycle Management (SP 800-63B). Gaithersburg, MD, USA, 2020.

[3] PHP Manual. Password Hashing — PHP documentation. Available at: https://www.php.net/manual/en/function.password- hash.php

[4] Mitropoulos, D., Karakoidas, V., Antonatos, S., & Spinellis, D. "Preventing SQL Injection Attacks in Web Applications: An Evaluation of Techniques." Journal of Systems and Software, 2017.