Task-2 Findings
Security Enhancements

Prepared by: Abdullah Akram
ID: DHC-3606

Date: July 9, 2025

Findings from Security Implementation in a PHP-Based Web Application

Home Rental System

1 Overview

This document summarizes the findings from implementing security measures in a PHP-based web application, focusing on input validation, password hashing, and HTTP header security.

2 Key Findings

2.1 Input Validation and Sanitization

- **File**: `register.php`
- **Finding**: Inputs were not sanitized, posing risks of injection attacks.
- **Action Taken**: Implemented `filter_var()` for email sanitization and validation, and `trim()` for other inputs.

2.2 Password Storage Security

- **Files**: `register.php`, `login.php`
- **Finding**: Passwords were stored using `md5()`, which is insecure and outdated.
- **Action Taken**: Replaced with `password_hash(PASSWORD_BCRYPT)` and `password_verify()`.
- **Note**: Existing `md5()` passwords required manual upgrades to work with new system.

2.3 HTTP Header Security

- **File**: `header.php`
- **Finding**: Lack of security headers made the application vulnerable to attacks like clickjacking and XSS.
- **Action Taken**: Added headers to prevent MIME-type sniffing, clickjacking, and enforce HTTPS.

3 Issues Encountered

- **Blank Page Issue**: `register.php` displayed a blank page due to missing or incorrect file includes.
- **Resolution**: Added error reporting (`error_reporting(E_ALL)`) to identify issues.
- **Password Compatibility**: Legacy users' passwords in `md5()` format caused login failures.
- **Resolution**: Manual password reset recommended using `password_hash()`.

4 Recommendations

- Implement JWT-based authentication for API endpoints.
- Regularly audit PHP dependencies (e.g., PHPMailer) for updates.

- Add Content Security Policy (CSP) headers for additional protection.