# Penetration Testing Vulnerability Report For page1.html

Target Application: http://[REDACTED]/sql/page1.html

Test Date: 15 August 2025

Tester: Meerjada

## 1. Executive Summary

A critical SQL Injection vulnerability was identified on the login page of the target application. The vulnerability allows attackers to bypass authentication and extract sensitive user data.

## 2. Vulnerability Details

Vulnerability Type: SQL Injection

Affected URL: /sql/page1.php (Login Page)

Impact: Authentication bypass, data extraction, possible further exploitation

Severity: High

## 3. Proof of Concept (PoC)

### 3.1 Login Page Screenshot

The application’s login page where user input is accepted:

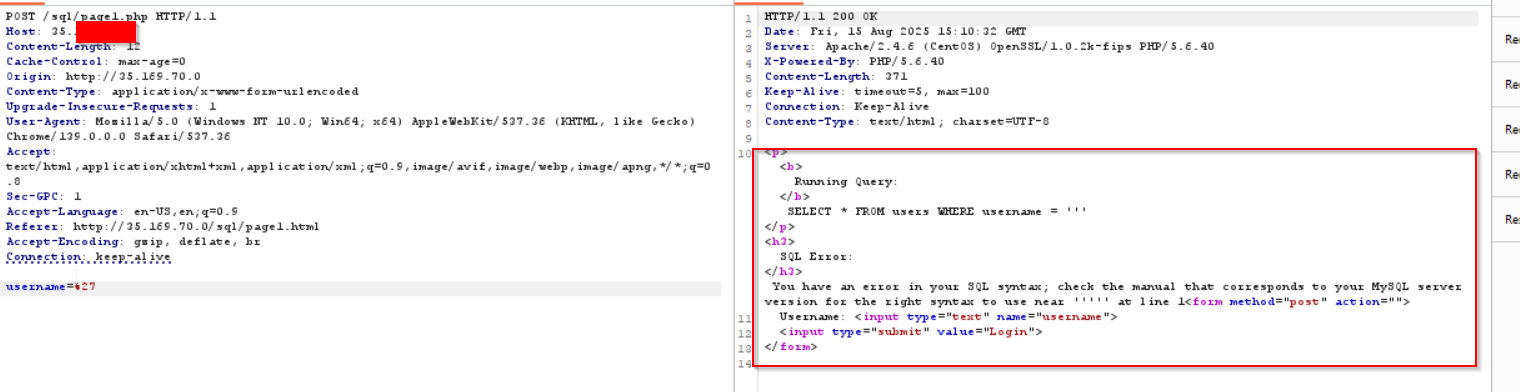
A screenshot of a computer

AI-generated content may be incorrect.

### 3.2 SQL Injection Test with ' (Single Quote – Error-Based)

Request Payload:  
username=’

Observed Behavior:  
The server responds with an SQL syntax error, confirming direct injection into the SQL statement.

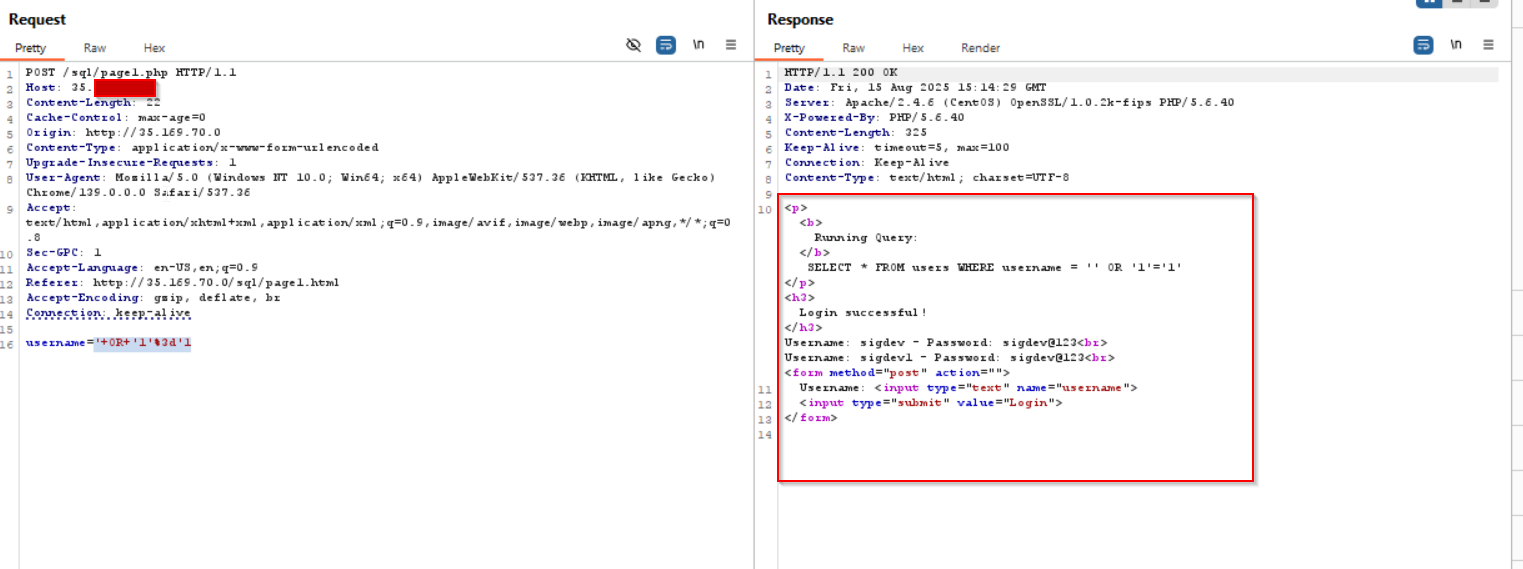


Response Example:  
SELECT \* FROM users WHERE username = ''  
SQL Error: You have an error in your SQL syntax...

### 3.3 Authentication Bypass Exploit (‘ OR ‘1’=‘1’)

Request Payload:  
username=' OR '1'='1'--

Observed Behavior:  
The application authenticates successfully and exposes user credentials in the response.



Response Example:  
SELECT \* FROM users WHERE username = '' OR '1'='1'  
Login successful!  
Username: sigdev - Password: sigdev@123

## 4. Risk Analysis

This vulnerability allows an attacker to:  
- Bypass authentication.  
- Access all users’ data.  
- Potentially escalate privileges and compromise the complete database.

## 5. Recommendations

- Implement parameterized (prepared) statements in all SQL queries.  
- Validate and sanitize all user input.  
- Apply Web Application Firewalls (WAF) with strict SQLi mitigation rules.  
- Regularly perform secure code reviews and vulnerability testing.

# 6. ModSecurity Rule Implementation

Rule ModSec

Example ModSecurity Rule to Block SQL Injection Attempts:  
SecRule REQUEST\_HEADERS|ARGS|ARGS\_NAMES|REQUEST\_URI|QUERY\_STRING "(?i:(union.\*select|select.\*from|insert.\*into|update.\*set|delete.\*from|drop\s+table|or\s+'1'='1))" \  
"phase:2,deny,status:403,id:1001,log,msg:'SQL Injection Attempt Detected'"  
 A black background with many colorful lights

AI-generated content may be incorrect.  
  
Blocked Request-

A screenshot of a computer

AI-generated content may be incorrect.

# Penetration Testing Vulnerability Report For page2.html

* **URL Tested:** <http://35.x.x.x/sql/page2.html>
* **Observed Request/Response:**
  + Attempted SQL injection payloads were submitted via the username field.
  + The application responded with standard error messages or login failures without SQL error leaks or authentication bypass.
* **Screenshot – Secure Login Page:**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.