Penetration Testing Report

Full Name: Rohit Kumar
Program: HCPT
Date: 17-Feb-25

Introduction

This report document hereby describes the proceedings and results of a Black Box security assessment conducted against the **Week 1 Labs**. The report hereby lists the findings and corresponding best practice mitigation actions and recommendations.

1. Objective

The objective of the assessment was to uncover vulnerabilities in the **Week 1 Labs** and provide a final security assessment report comprising vulnerabilities, remediation strategy and recommendation guidelines to help mitigate the identified vulnerabilities and risks during the activity.

2. Scope

This section defines the scope and boundaries of the project.

Application Name	**HTML Injection, Cross Site Scripting (XSS)**

3. Summary

Outlined is a Black Box Application Security assessment for the **Week Labs**.

Total number of Sub-labs: {count} Sub-labs

High	Medium	Low
3	4	7

High - **Number of Sub-labs with hard difficulty level**

Medium - **Number of Sub-labs with Medium difficulty level**

Low - **Number of Sub-labs with Easy difficulty level**

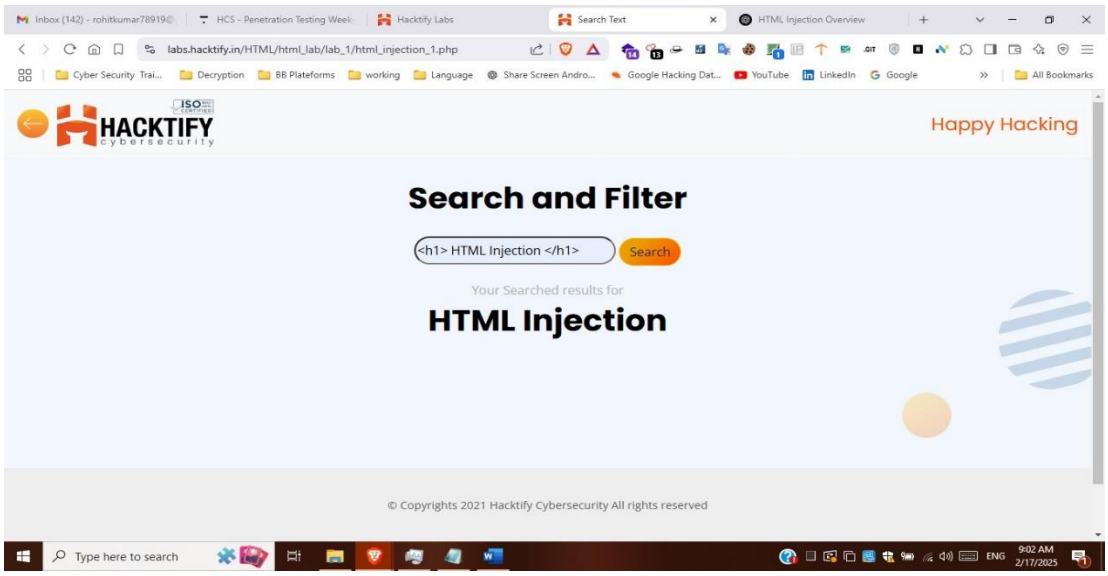
1. HTML Injection

1.1. HTML's are easy!

Reference	Risk Rating
HTML's are easy!	**Low**
Tools Used	
Chrome Browser, HTML	
Vulnerability Description	
This occurs when an attacker injects HTML tags (e.g., <h1> , <script>) into the content of a page. The page may display unexpected styles, images, or even executable scripts.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/html_lab/lab_1/html_injection_1.php	
Consequences of not Fixing the Issue	
If Vulnerability is not patched it can change the appearance of the site, which might confuse or trick users, insert fake login forms to attempt Phishing and it could change the content of the page, inserting offensive or misleading information	
Suggested Countermeasures	
Input Validation and Sanitization, Escape Special Characters	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/pdfs/62394726-week-1-technicalguide	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

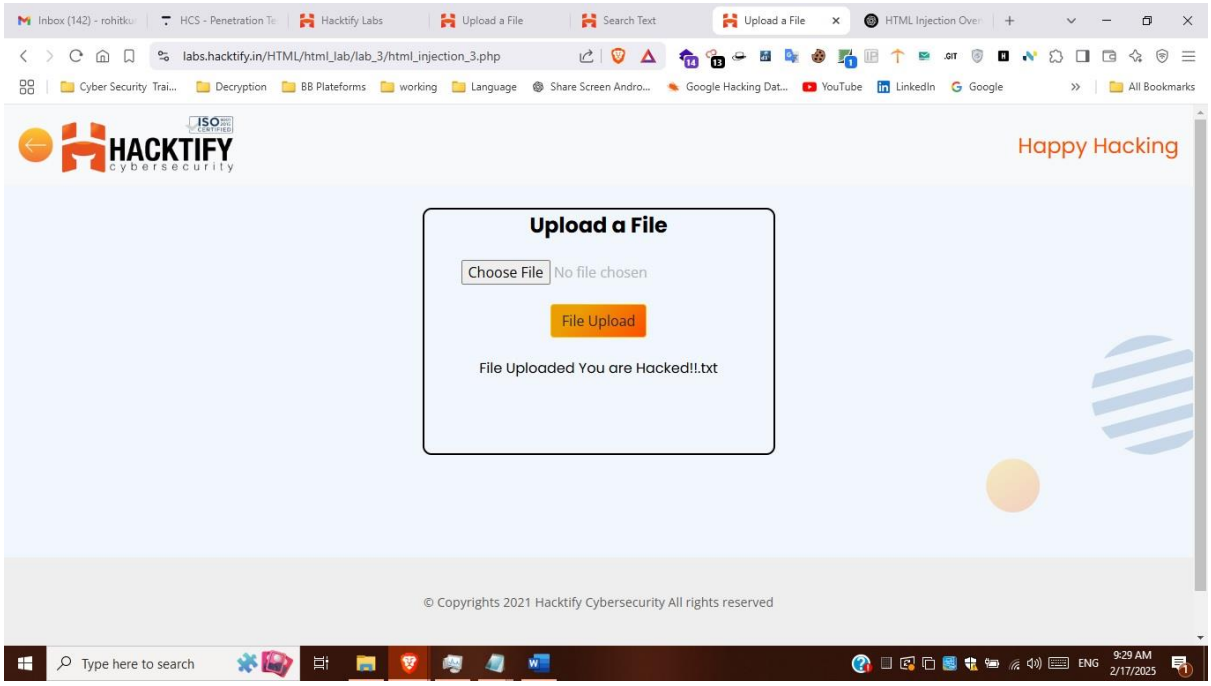


1.2. File names are also vulnerable

Reference	Risk Rating
File names are also vulnerable	**Low**
Tools Used	
Web Browser, HTML	
Vulnerability Description	
In this vulnerability a malicious file with name as HTML Injection payload get stored in application.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
http://labs.hacktify.in/HTML/html_lab/lab_3/html_injection_3.php	
Consequences of not Fixing the Issue	
If it is not patched, user can be triggered to store malicious file by attacker which led to account hijacking.	
Suggested Countermeasures	
Don’t let user to upload filename content as HTML payload	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/pdfs/62394726-week-1-technicalguide	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

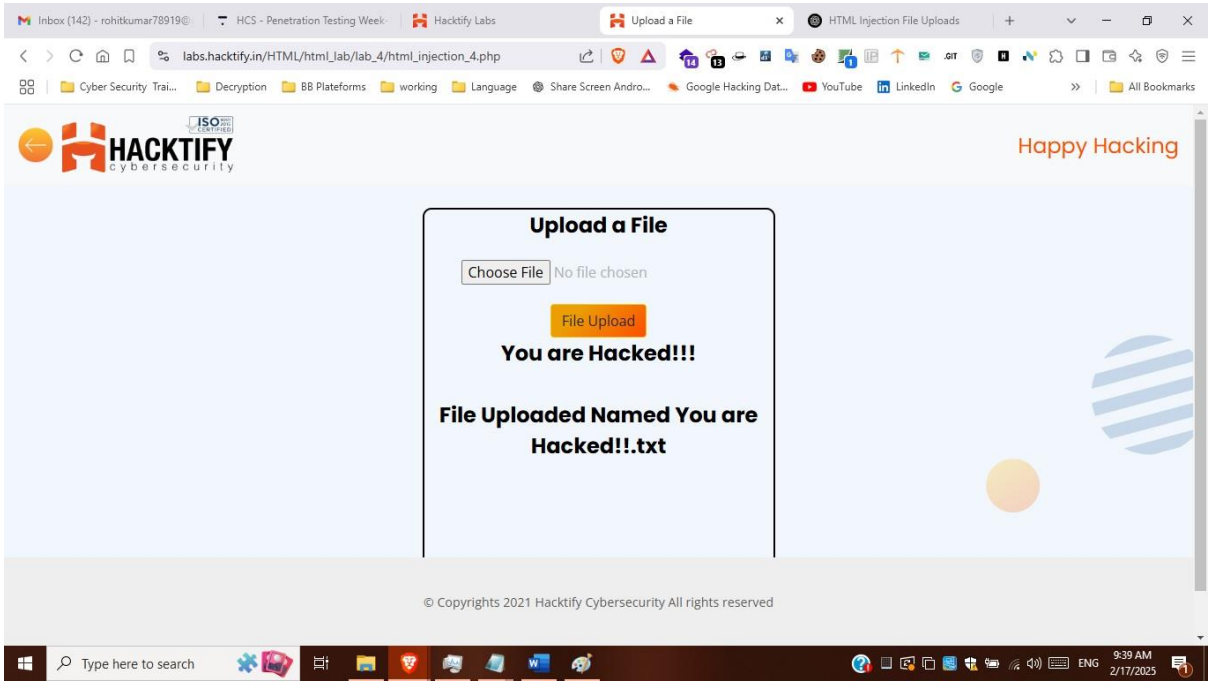


1.3. File content and HTML injection a perfect pair

Reference	Risk Rating
File content and HTML injection a perfect pair	**Medium**
Tools Used	
Web Browser, HTML	
Vulnerability Description	
HTML injection occurs when a web application improperly handles user input, allowing the user to inject HTML.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/html_lab/lab_4/html_injection_4.php	
Consequences of not Fixing the Issue	
When a file upload system doesn't sanitize or validate files properly, it might upload files containing malicious scripts	
Suggested Countermeasures	
Don't let user to upload filename content as HTML payload	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/pdfs/62394726-week-1-technicalguide	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab



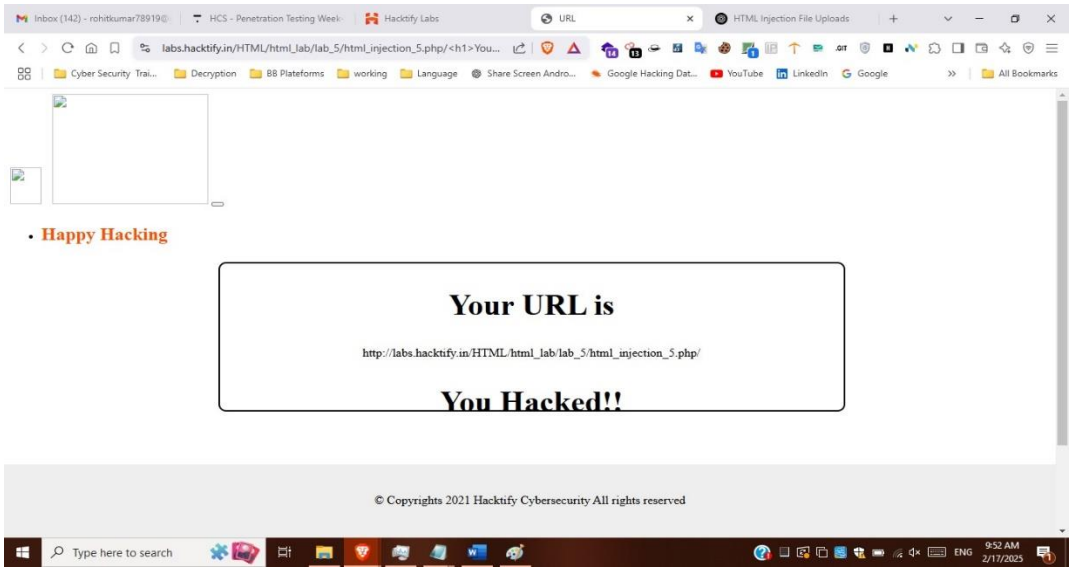
1.4. Injecting HTML using URL

Reference	Risk Rating
Injecting HTML using URL	**Medium**
Tools Used	
Web Browser, HTML	
Vulnerability Description	
Injecting HTML via a URL is a common type of attack known as **URL-based HTML Injection**. It occurs when a web application does not properly sanitize user input that gets inserted into URLs or query parameters.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/html_lab/lab_5/html_injection_5.php	
Consequences of not Fixing the Issue	
If it is not patched, user can be triggered to click and perform any action on malicious URL.	
Suggested Countermeasures	
Restrict allowed characters in URL parameters	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/pdfs/62394726-week-1-technicalguide	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

URL:https://labs.hacktify.in/HTML/html_lab/lab_5/html_injection_5.php/<h1>YouHacked!!<h1>

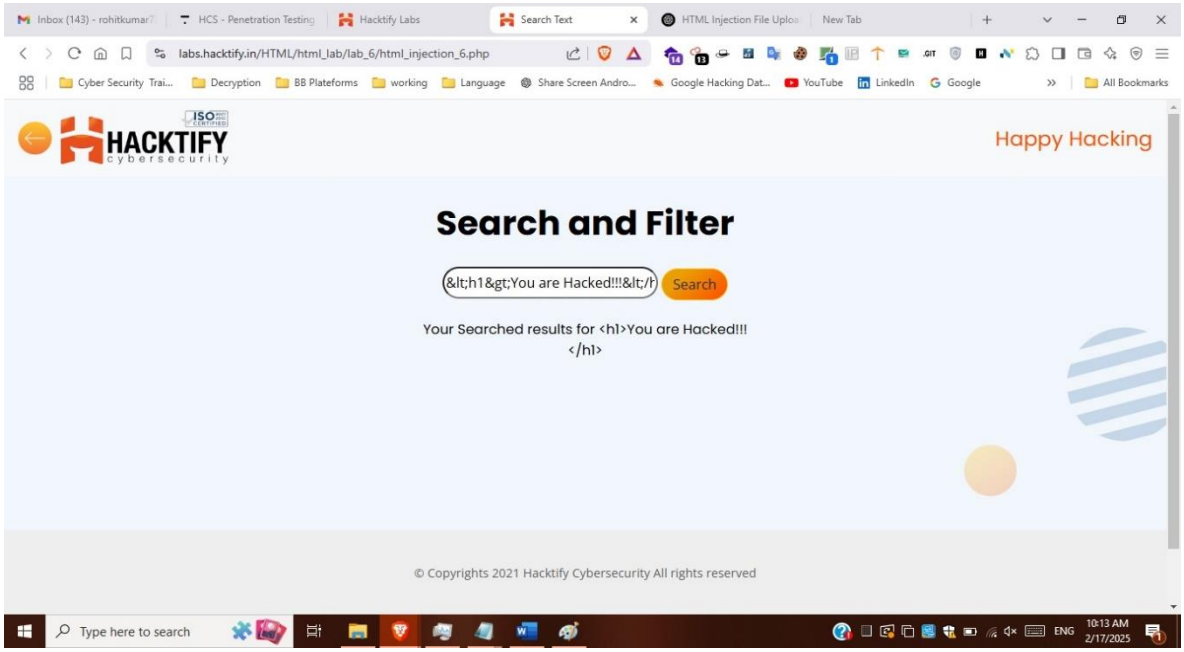


1.5. Encode IT!

Reference	Risk Rating
Encode IT!	**Hard**
Tools Used	
Web Browser, HTML Payload(You are Hacked!!!	
Vulnerability Description	
HTML Injection via HTML encoding is a method of injecting HTML or JavaScript into a web page, often by bypassing some basic input validation mechanisms that simply look for raw HTML characters like <, >, and &. When these characters are **encoded** (for example, < becomes <, and > becomes >)	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/html_lab/lab_6/html_injection_6.php	
Consequences of not Fixing the Issue	
An attacker can still inject malicious content, especially if the application doesn't decode these HTML-encoded values properly before rendering them on the page.	
Suggested Countermeasures	
Properly sanitize user input before displaying it in the page. Use libraries or frameworks that automatically sanitize inputs to prevent HTML	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/pdfs/62394726-week-1-technicalguide	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab



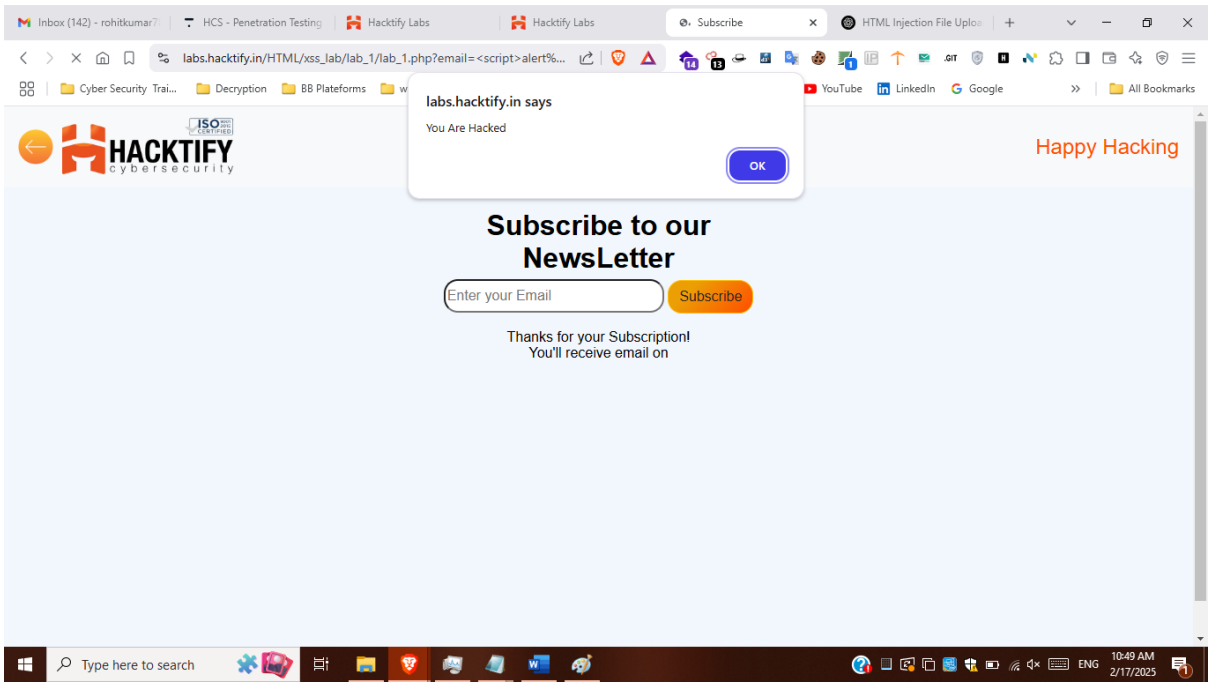
2. Cross Site Scripting (XSS)

2.1. Let's Do IT!

Reference	Risk Rating
Let's Do IT!	Low
Tools Used	
Web Browser, XSS Payload (<script>alert('You are Hacked');</script>)	
Vulnerability Description	
Cross-Site Scripting (XSS) is a type of vulnerability that allows attackers to inject malicious scripts into webpages viewed by other users.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_1/lab_1.php	
Consequences of not Fixing the Issue	
If this vulnerability is not patched user will be victim of XSS attack	
Suggested Countermeasures	
Always encode user input before rendering	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/lessons/62394526-1-xss	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

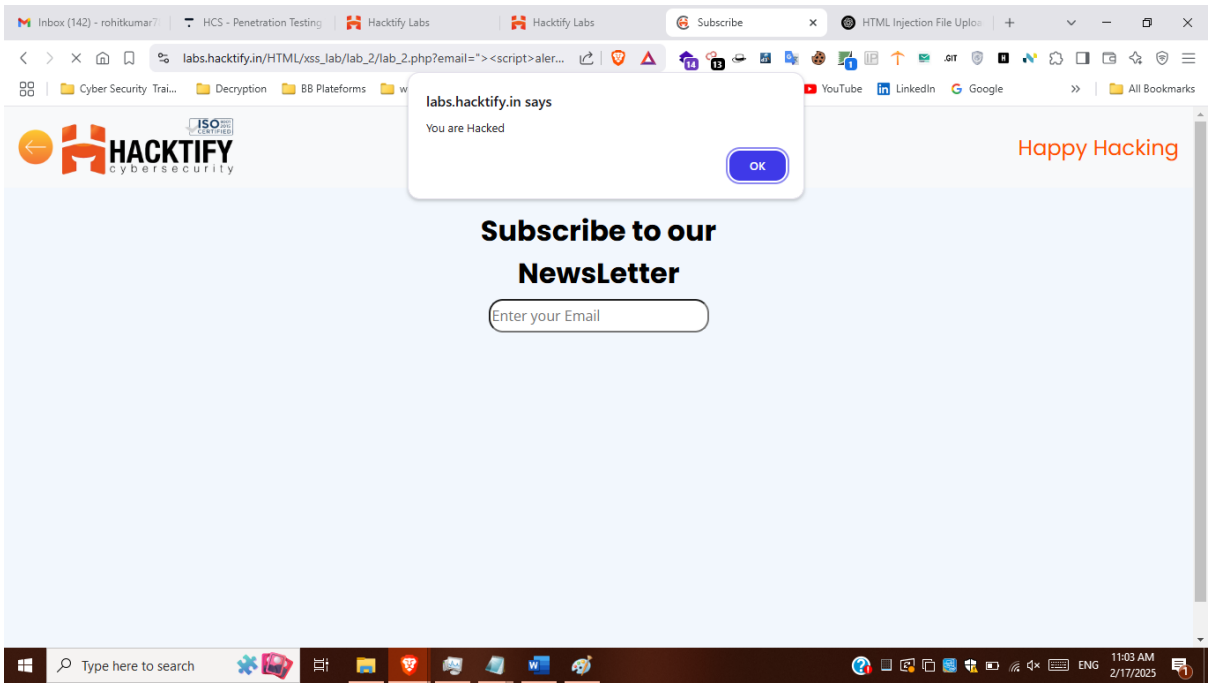


2.2. Balancing is Important in Life!

Reference	Risk Rating
Balancing is Important in Life!	**Low**
Tools Used	
Web Browser, XSS Payload { "><script>alert('You are Hacked');</script> }	
Vulnerability Description	
Cross-Site Scripting (XSS) is a type of vulnerability that allows attackers to inject malicious scripts into webpages viewed by other users	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_2/index.php	
Consequences of not Fixing the Issue	
If this vulnerability is not patched user will be victim of XSS attack	
Suggested Countermeasures	
Do not allow special characters as user input	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/lessons/62405053-2-reflected-xss	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

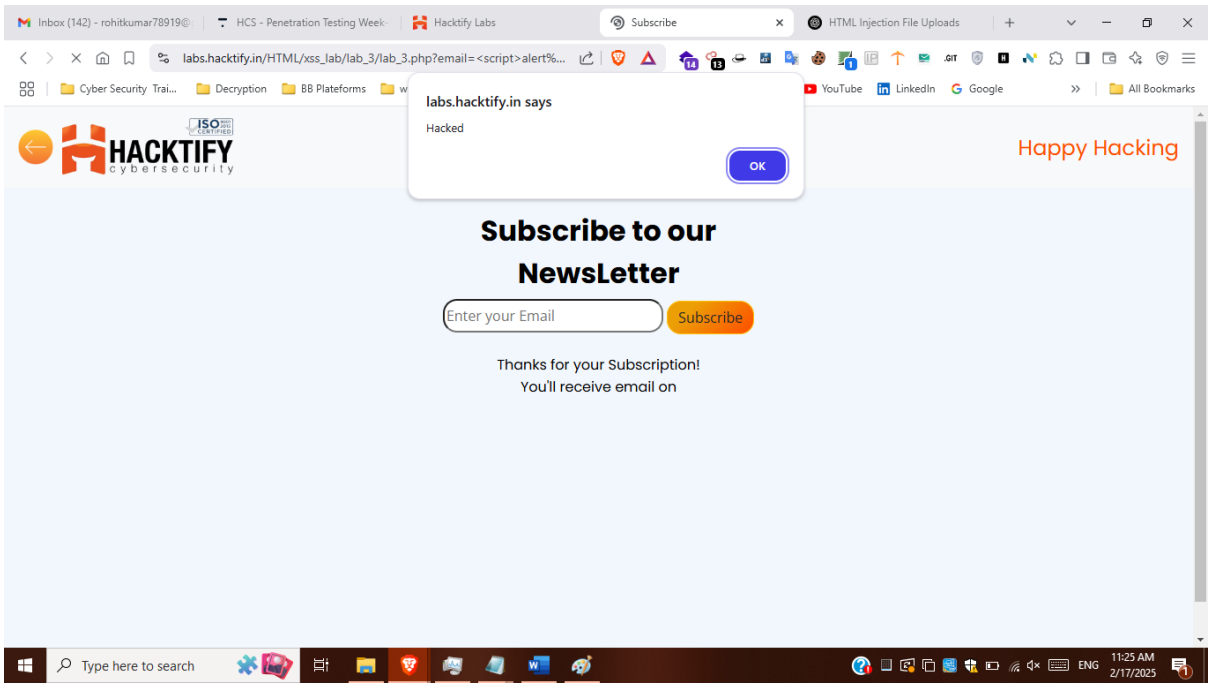


2.3. XSS is everywhere!

Reference	Risk Rating
XSS is everywhere!	Low
Tools Used	
Web Browser, XSS Payload { <script>alert('Hacked');</script>@gmail.com }	
Vulnerability Description	
Cross-Site Scripting (XSS) is a type of vulnerability that allows attackers to inject malicious scripts into webpages viewed by other users	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_3/lab_3.php	
Consequences of not Fixing the Issue	
If this vulnerability is not patched user will be victim of XSS attack	
Suggested Countermeasures	
Do not allow special characters as user input	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/lessons/62405053-2-reflected-xss	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

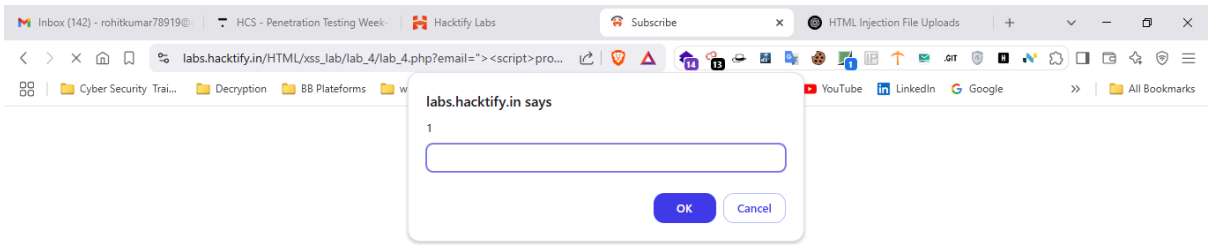


2.4. Alternatives are must!

Reference	Risk Rating
Alternatives are must!	**Medium**
Tools Used	
Web Browser, XSS Payload { "><script>prompt(1);</script>@gmail.com }	
Vulnerability Description	
Cross-Site Scripting (XSS) is a type of vulnerability that allows attackers to inject malicious scripts into webpages viewed by other users	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_4/lab_4.php	
Consequences of not Fixing the Issue	
If this vulnerability is not patched user will be victim of XSS attack	
Suggested Countermeasures	
Do not allow any alternative of alert()	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/lessons/62405053-2-reflected-xss	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab



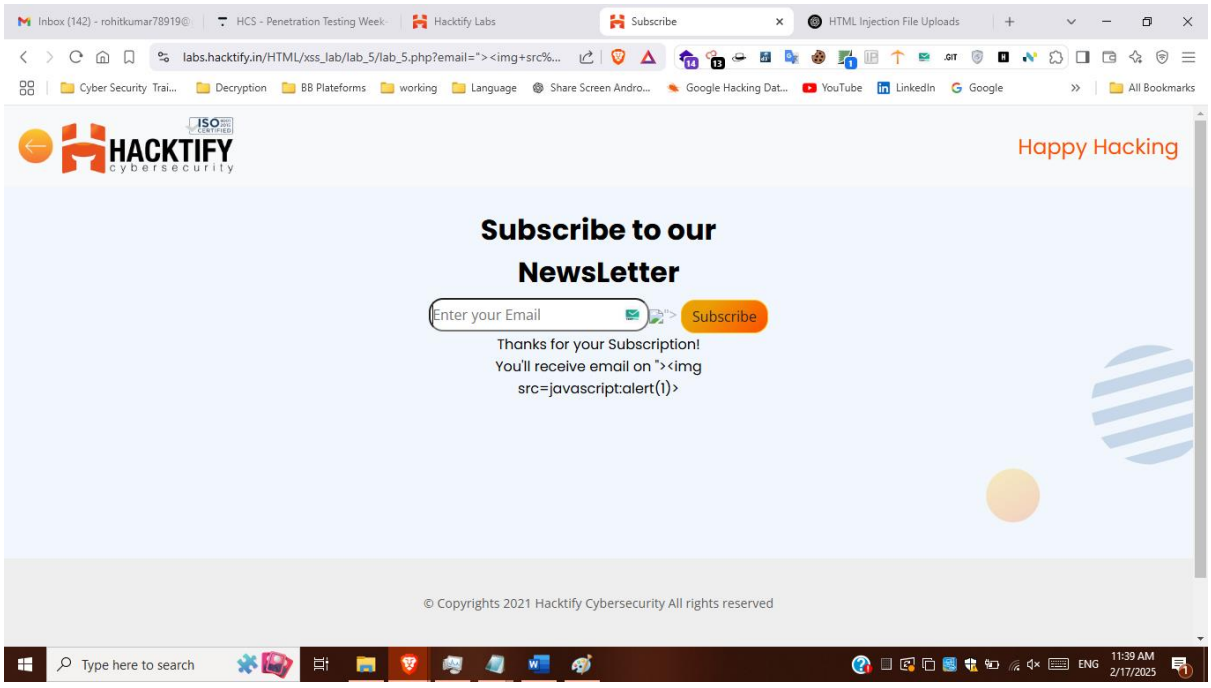
2.5. Developer hates scripts!

Reference	Risk Rating
Developer hates scripts!	**Hard**
Tools Used	
Web Browser, XSS Payload { "><img src=javascript:alert(1)> }	
Vulnerability Description	
Cross-Site Scripting (XSS) is a type of vulnerability that allows attackers to inject malicious scripts into webpages viewed by other users	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_5/lab_5.php	
Consequences of not Fixing the Issue	
If this vulnerability is not patched user will be victim of XSS attack	
Suggested Countermeasures	
Ensure user input is properly encoded before rendering it in HTML	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/lessons/62405060-3-stored-xss	

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab



2.6. XSS with File Upload (File Content)

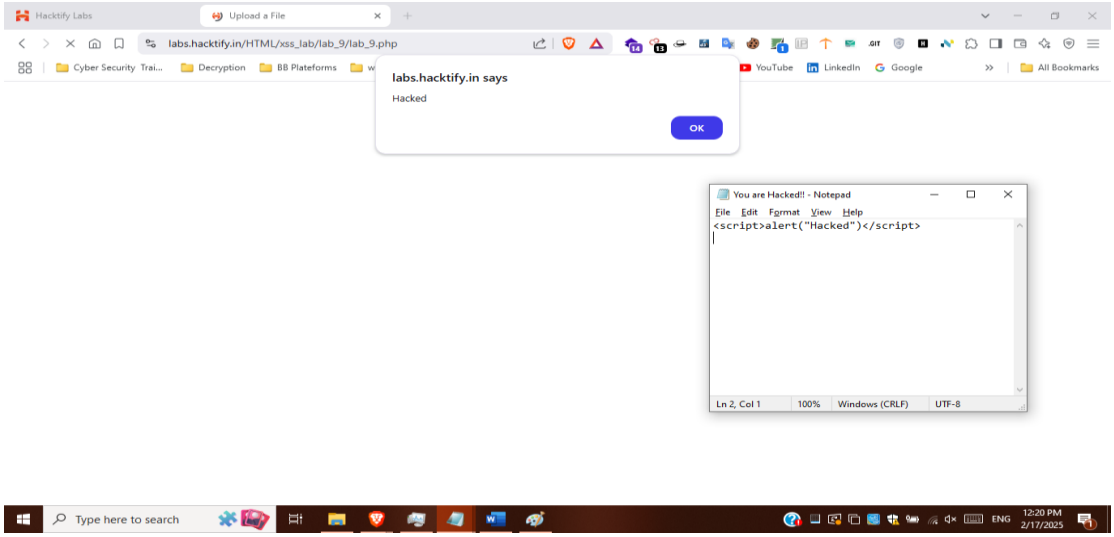
Reference	Risk Rating
XSS with File Upload (File Content)	**Hard**
Tools Used	
Web Browser, XSS Payload { <script>alert("Hacked")</script>}	
Vulnerability Description	
Cross-Site Scripting (XSS) is a type of vulnerability that allows attackers to inject malicious scripts into webpages viewed by other users	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_9/lab_9.php	
Consequences of not Fixing the Issue	
If this vulnerability is not patched user will be victim of XSS attack	
Suggested Countermeasures	
Ensure user input is properly encoded before rendering it in HTML	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/lessons/62405060-3-stored-xss	

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab.

Step 1: Create a file and write a code and save it as shown in below screenshot.
Step 2: Upload and show script running in the browser.



2.7. Stored Everywhere!

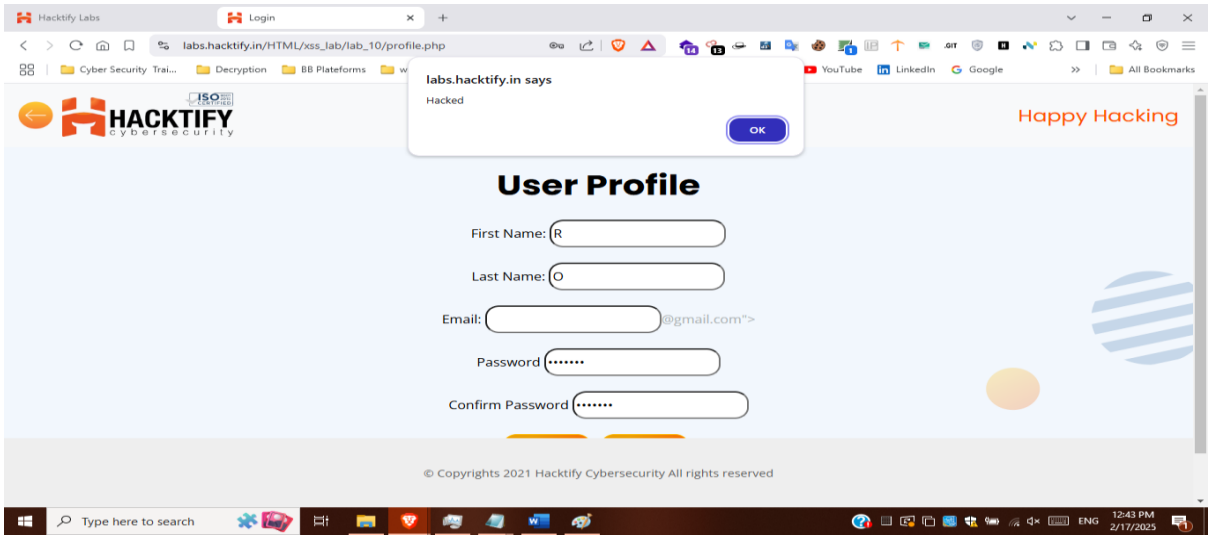
Reference	Risk Rating
Stored Everywhere!	**Low**
Tools Used	
Web Browser, XSS Payload { "><script>alert("Hacked")</script>@gmail.com }	
Vulnerability Description	
Cross-Site Scripting (XSS) is a type of vulnerability that allows attackers to inject malicious scripts into webpages viewed by other users	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_10/profile.php	
Consequences of not Fixing the Issue	
If this vulnerability is not patched user will be victim of XSS attack	
Suggested Countermeasures	
Ensure user input is properly encoded before rendering it in HTML	
References	
https://hacktify.thinkific.com/courses/take/ai-placeholder-3/lessons/62405080-4-dom-xss	

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

Step 1: Click Register fille details and in the email field type {"><script>alert("Hacked")</script>@gmail.com} and password 1234567. As shown in the below screenshot
Step 2: login with this above username as mail and password



NOTES:

- **Everything mentioned inside {} has to be changed based on your week, labs and sub-labs.**
- **If you have 2 labs in same week you need to mention that, if not ignore those mentions for lab 2.**
- **Here it is given with 2 Sub-labs vulnerability, you need to add all the sub-labs based on your labs.**
- **Don't forget to add the screenshot of the vulnerability in the proof of concept.**
- **Add only 1 screenshot in the Proof of Concept section.**
- **This NOTE session is only for your reference, don't forget to delete this in the report you submit.**