**WAPH-Web Application Programming and Hacking**

**Instructor: Dr. Phu Phung**

**Student**

**Name**: Sai Keerthi Vadnala

**Email**: vadnalsi@ucmail.uc.edu

**Short bio**: Sai Keerthi Vadnala has great interest in learning web development.

and wants to explore more about it by doing hands-on projects.

Figure 1: Sai Keerthi vadnala headshot

**Repository Information**

**Repository's URL:** https://github.com/Saikeerthi72/waph-vadnalsi.git

**Lab4 URL:** https://github.com/Saikeerthi72/waph-vadnalsi/tree/main/labs/lab4

This is a private repository for Sai Keerthi Vadnala to store all code from the course.

## Lab 4 Overview: A Secure Login System with Session Authentication

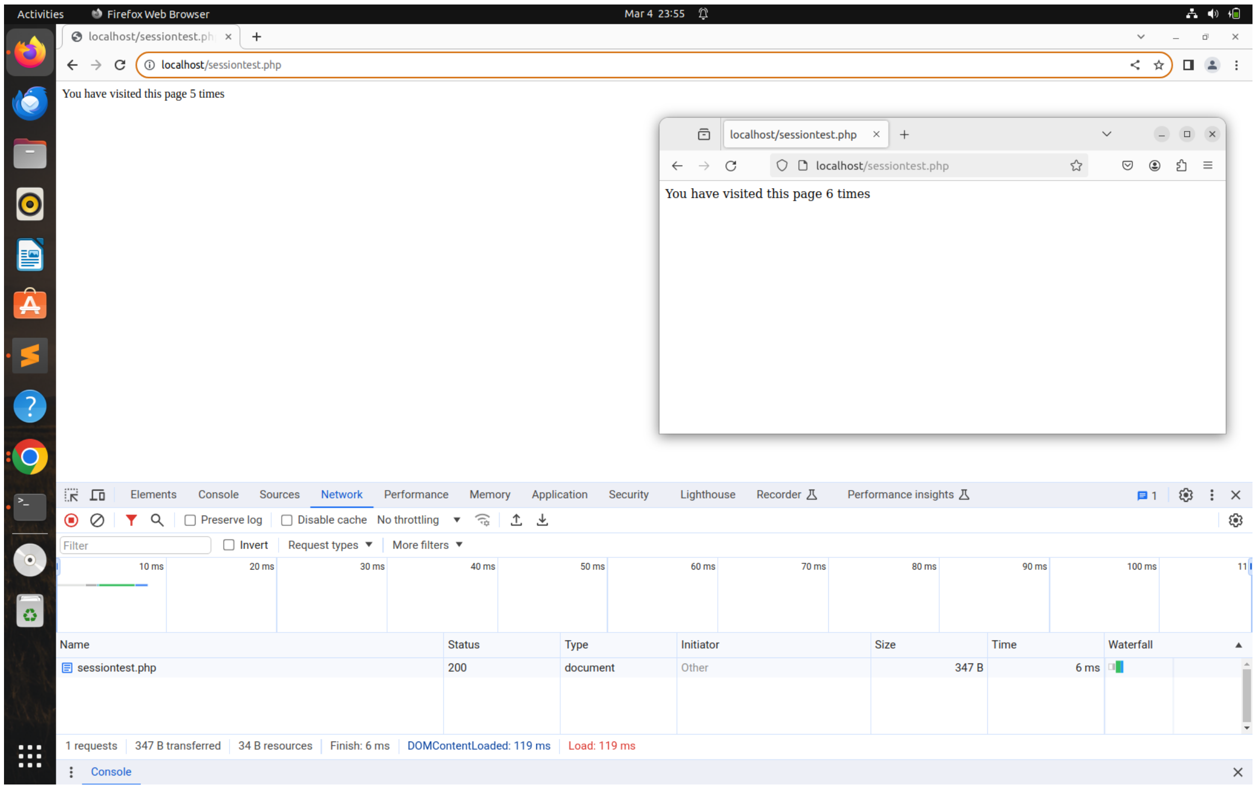
- This lab works on session management and security in PHP web applications.
- The lab contains three 3 tasks.
- Understanding session management through PHP script deployment and testing, managing session control through Wireshark, detecting and mitigating session hijacking attacks.
- Implementation of secure session authentication, configuring HTTPS for web applications, and using countermeasures against session hijacking.
- These tasks aim to enhance practical competencies in implementing and deploying secure session management, understanding web traffic related to session handling, and safeguarding web applications from common security threats.

### Task 1: Understanding Session Management in a PHP Web Application

#### 1.a. Deploy and test sessiontest.php

- In Task 1.a, I have firstly cloned the repository as given in the instructions and then copied the Lab 3 content (index.php, form.php) to my private repository (waph-vadnalsi) and then deployed the 'sessiontest.php' to Apache root folder.
- Now, I accessed this file using a web browser such as Firefox in the VM, and refresh the page frequently to test session functionality.

- Test the session.php file in another browser (Chrome) to see that session values change each time you refresh.
- This task demonstrates implementation and testing of session management in PHP, and focuses on how these sessions work across different browsers.



You have visited this page 5 times

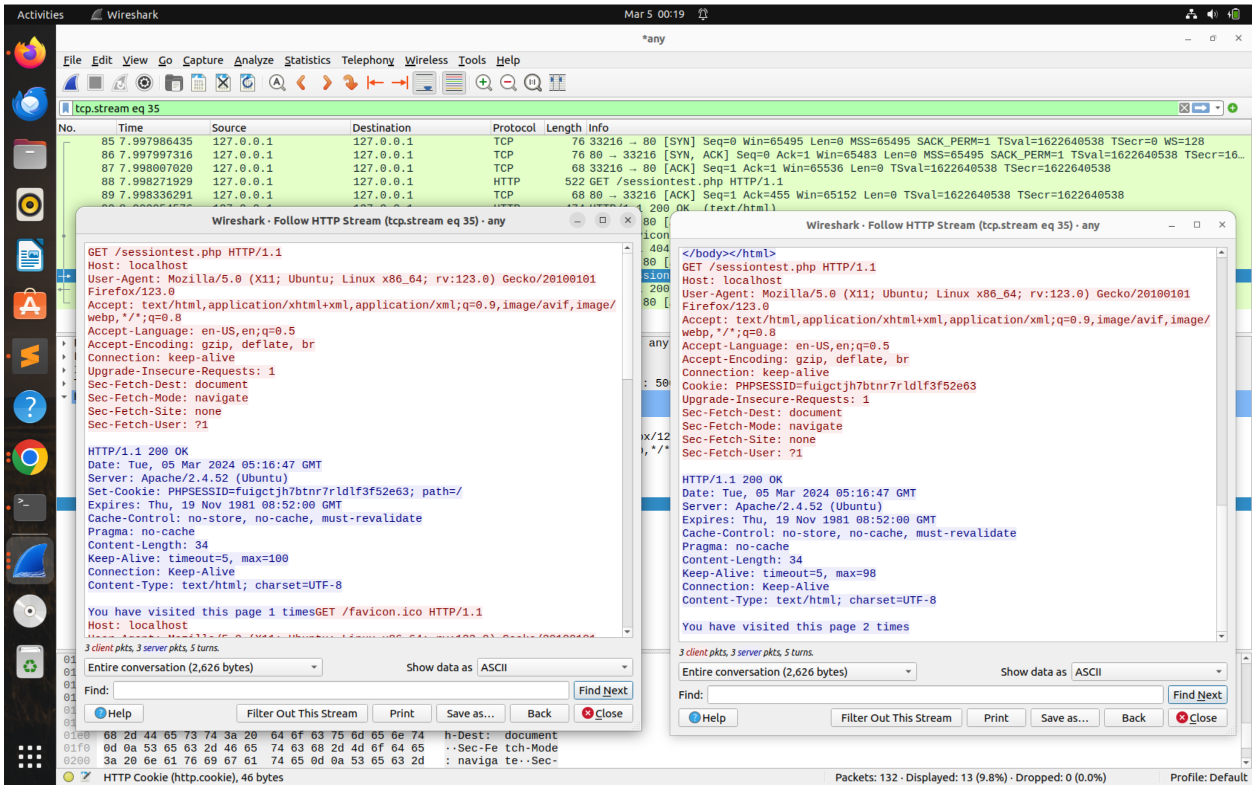
You have visited this page 6 times

| Name | Status | Type | Initiator | Size | Time | Waterfall |
|---|---|---|---|---|---|---|
| sessiontest.php | 200 | document | Other | 347 B | 6 ms | |

1 requests | 347 B transferred | 34 B resources | Finish: 6 ms | DOMContentLoaded: 119 ms | Load: 119 ms

## 1.b. Observe the Session-Handshaking using Wireshark

- Task 1.b uses Wireshark to observe the session handshaking process in a PHP web application by accessing `sessiontest.php`.
- Deleting browser cookies at work to ensure that no session information is stored. By using Wireshark, we refresh the web page twice to capture the first and last HTTP requests and responses.
- Here we observe the change by setting a session cookie on the first response, making sure that there is no cookie on the first request.
- This session cookie is included in the next request by the browser but is not reset by the server in a subsequent response.
- This task helps in understanding how sessions are set up and managed in web applications, security risks of session hijacking.



```
tcp.stream eq 35

No.  Time         Source      Destination  Protocol  Length  Info
  85 7.997986435  127.0.0.1   127.0.0.1    TCP        76 33216 → 80 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=1622640538 TSecr=0 WS=128
  86 7.997997316  127.0.0.1   127.0.0.1    TCP        76 80 → 33216 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=1622640538 TSecr=16…
  87 7.998007020  127.0.0.1   127.0.0.1    TCP        68 33216 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1622640538 TSecr=1622640538
  88 7.998271929  127.0.0.1   127.0.0.1    HTTP      522 GET /sessiontest.php HTTP/1.1
  89 7.998336291  127.0.0.1   127.0.0.1    TCP        68 80 → 33216 [ACK] Seq=1 Ack=455 Win=65152 Len=0 TSval=1622640538 TSecr=1622640538
```

Wireshark · Follow HTTP Stream (tcp.stream eq 35) · any

```
GET /sessiontest.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:123.0) Gecko/20100101
Firefox/123.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1

HTTP/1.1 200 OK
Date: Tue, 05 Mar 2024 05:16:47 GMT
Server: Apache/2.4.52 (Ubuntu)
Set-Cookie: PHPSESSID=fuigctjh7btnr7rldlf3f52e63; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 34
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

You have visited this page 1 timesGET /favicon.ico HTTP/1.1
Host: localhost
```

3 client pkts, 3 server pkts, 5 turns.

Entire conversation (2,626 bytes) · Show data as ASCII

Wireshark · Follow HTTP Stream (tcp.stream eq 35) · any

```
</body></html>
GET /sessiontest.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:123.0) Gecko/20100101
Firefox/123.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Cookie: PHPSESSID=fuigctjh7btnr7rldlf3f52e63
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1

HTTP/1.1 200 OK
Date: Tue, 05 Mar 2024 05:16:47 GMT
Server: Apache/2.4.52 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 34
Keep-Alive: timeout=5, max=98
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

You have visited this page 2 times
```

3 client pkts, 3 server pkts, 5 turns.

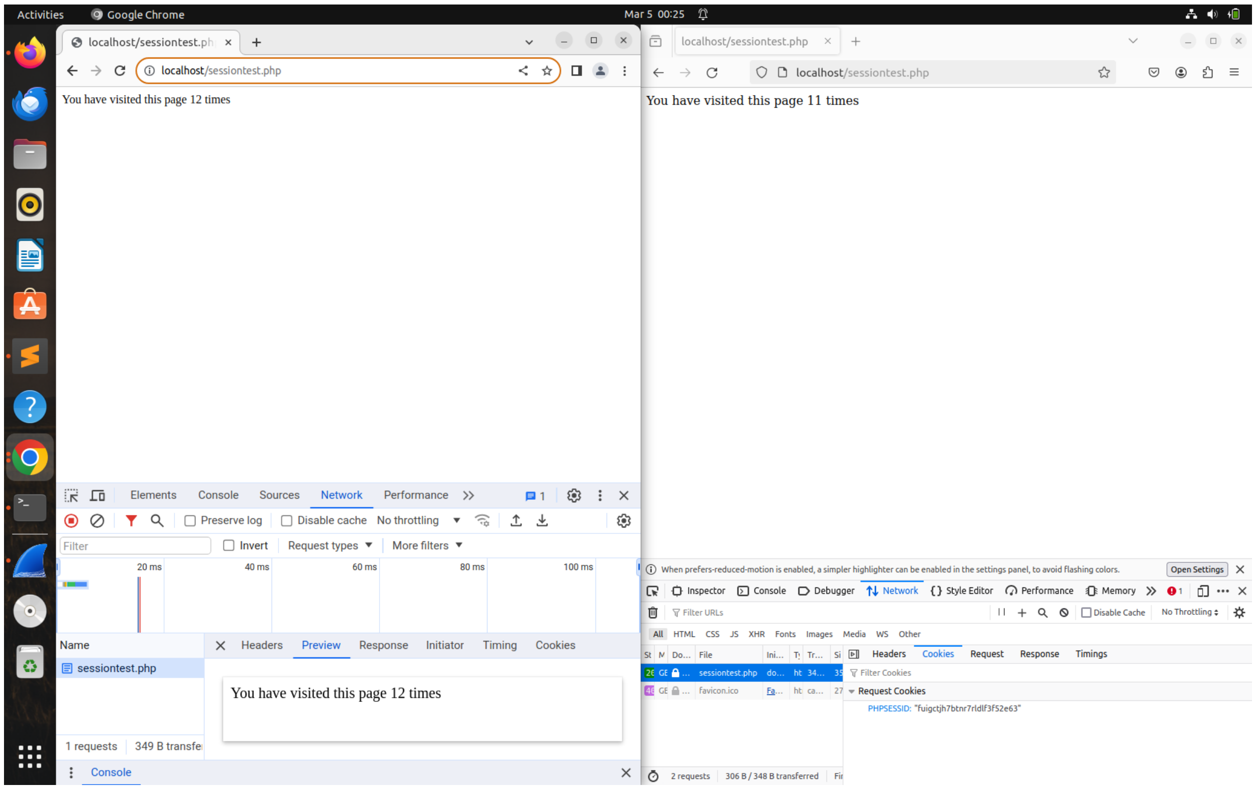
Entire conversation (2,626 bytes) · Show data as ASCII

```
01e0  68 2d 44 65 73 74 3a 20  64 6f 63 75 6d 65 6e 74   h-Dest:  document
01f0  0d 0a 53 65 63 2d 46 65  74 63 68 2d 4d 6f 64 65   ··Sec-Fe tch-Mode
0200  3a 20 6e 61 76 69 67 61  74 65 0d 0a 53 65 63 2d   : naviga te··Sec-
```

HTTP Cookie (http.cookie), 46 bytes · Packets: 132 · Displayed: 13 (9.8%) · Dropped: 0 (0.0%) · Profile: Default

### 1.c. Understanding Session Hijacking

- This task is about session hijacking attack.
- The process involves refreshing the web browser a number of times in the browser, then using the console we try to retrieve and copy the session cookie value using `document.cookie`.
- This step depends on how attackers can obtain session IDs, possibly through usage XSS and other vulnerabilities.
- Then, the session ID is manually inserted into another browser through the console, simulating the process of session hijacking by forcing the browser to use the copied session ID.
- This change allows the attacker to access the session.
- This task shows the importance of protection against session hijacking in web applications.


localhost/sessiontest.php

You have visited this page 12 times

You have visited this page 11 times

When prefers-reduced-motion is enabled, a simpler highlighter can be enabled in the settings panel, to avoid flashing colors.

Request Cookies

PHPSESSID: "fuigctjh7btnr7rldlf3f52e63"

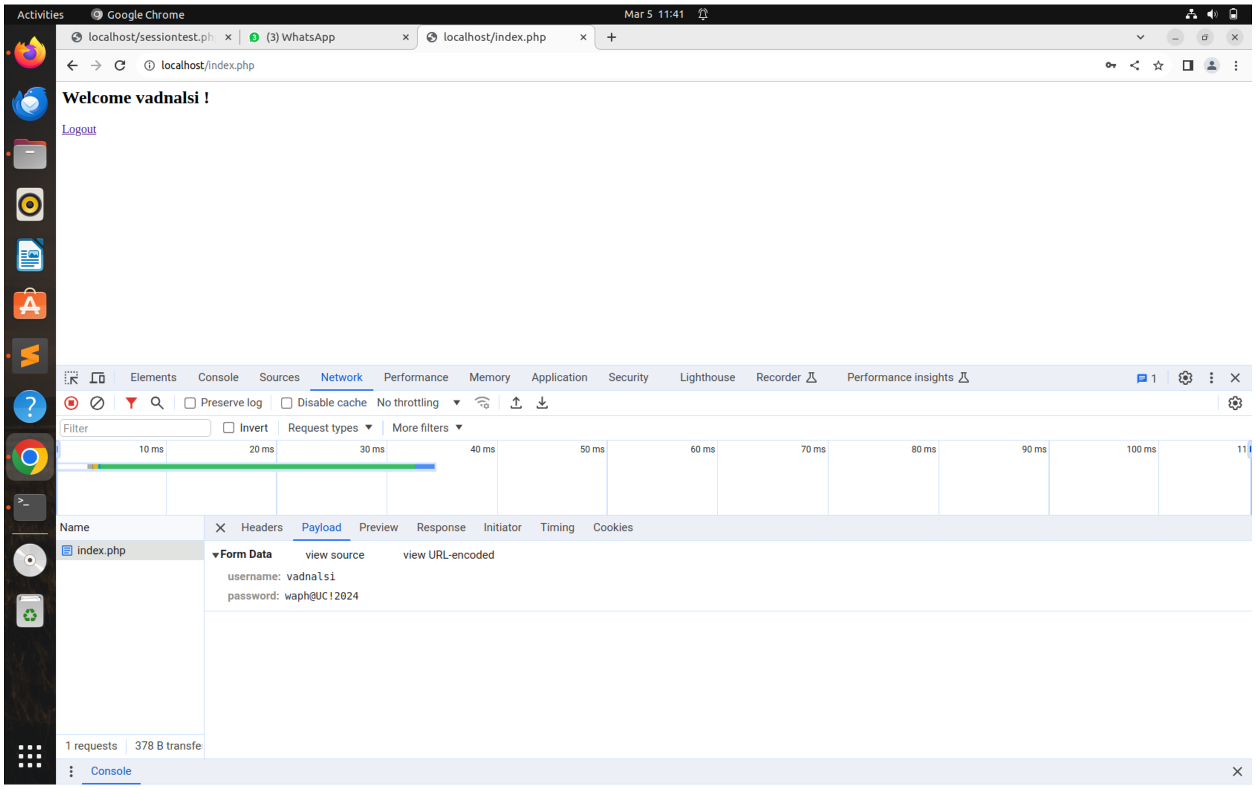
1 requests | 349 B transfe

2 requests | 306 B / 348 B transferred

## Task 2: Insecure Session Authentication

### 2.a. Revised Login System with Session Management

- In this task, we revise the login system with session management.
- For that we use the index.php file from lab3 for session authentication.
- Here we update the code in this file to perform session management for login system. If the login is successful the session is set as "authenticated" and stores the username, enabling access to authenticated pages.
- Otherwise, if a user tried to authenticate without logging in, the user is alerted and it asks to login, for that it redirects form.php and asks for credentials to login.
- Logout.php file starts and destroys the session. It is used to manage the session start and termination.
- The following screenshot shows that only users with valid credentials can successfully log into the system.



localhost/index.php

Welcome vadnalsi !

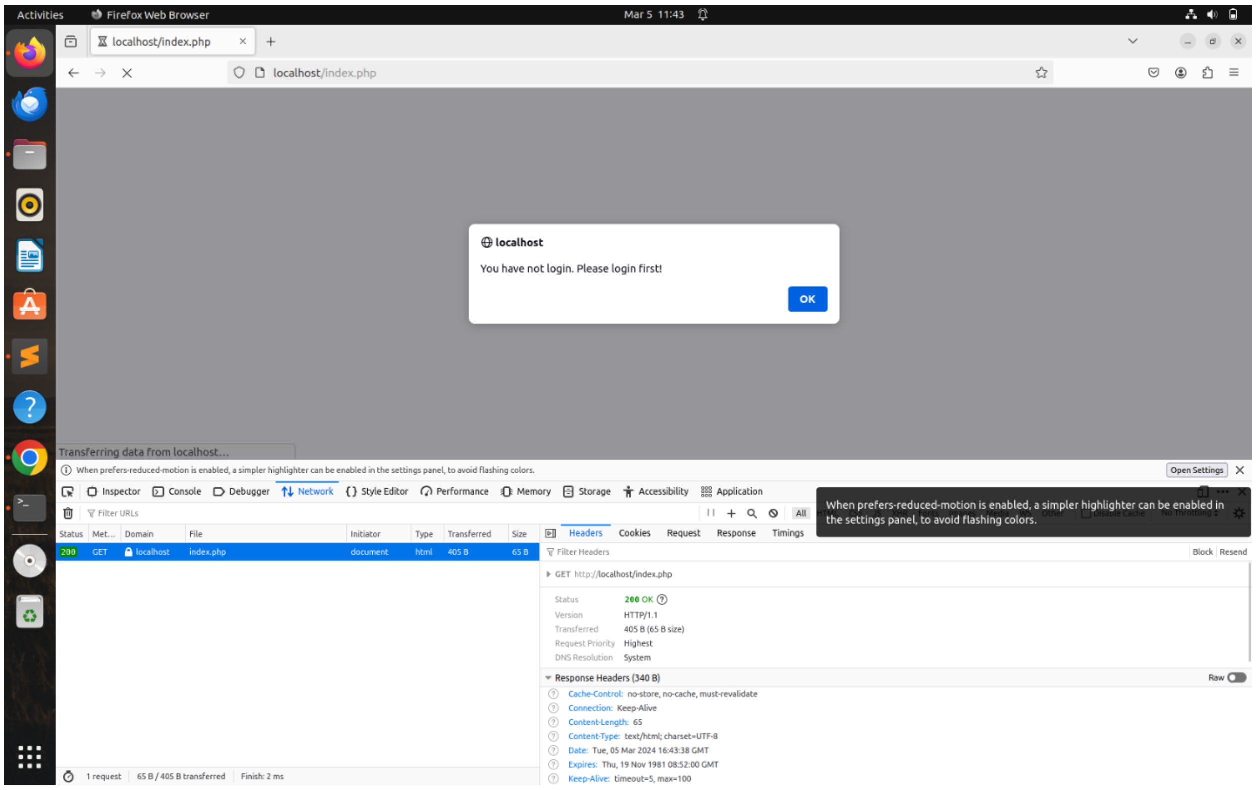
Logout

Form Data

```
username: vadnalsi
password: waph@UC!2024
```

1 requests | 378 B transfe

The following screenshot prompts the user to log in before accessing the system through the URL.



localhost/index.php

localhost

You have not login. Please login first!

OK

Transferring data from localhost...

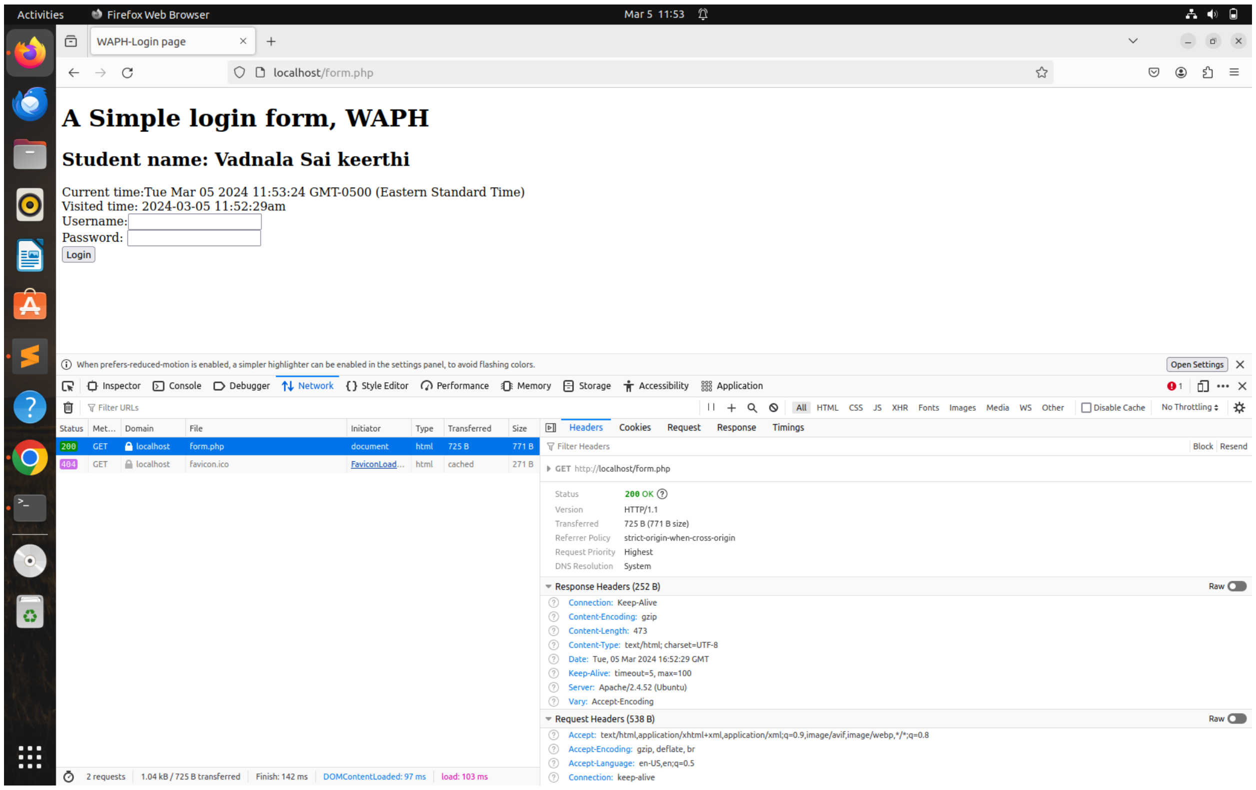
When prefers-reduced-motion is enabled, a simpler highlighter can be enabled in the settings panel, to avoid flashing colors.

| Status | Met... | Domain | File | Initiator | Type | Transferred | Size |
|---|---|---|---|---|---|---|---|
| 200 | GET | localhost | index.php | document | html | 405 B | 65 B |

```
GET http://localhost/index.php

Status            200 OK
Version           HTTP/1.1
Transferred       405 B (65 B size)
Request Priority  Highest
DNS Resolution    System

Response Headers (340 B)
Cache-Control: no-store, no-cache, must-revalidate
Connection: Keep-Alive
Content-Length: 65
Content-Type: text/html; charset=UTF-8
Date: Tue, 05 Mar 2024 16:43:38 GMT
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Keep-Alive: timeout=5, max=100
```

1 request | 65 B / 405 B transferred | Finish: 2 ms

After clicking on "OK," the user is directed to the form.php page, as shown in the following screenshot.



A Simple login form, WAPH

Student name: Vadnala Sai keerthi

Current time:Tue Mar 05 2024 11:53:24 GMT-0500 (Eastern Standard Time)
Visited time: 2024-03-05 11:52:29am
Username:
Password:

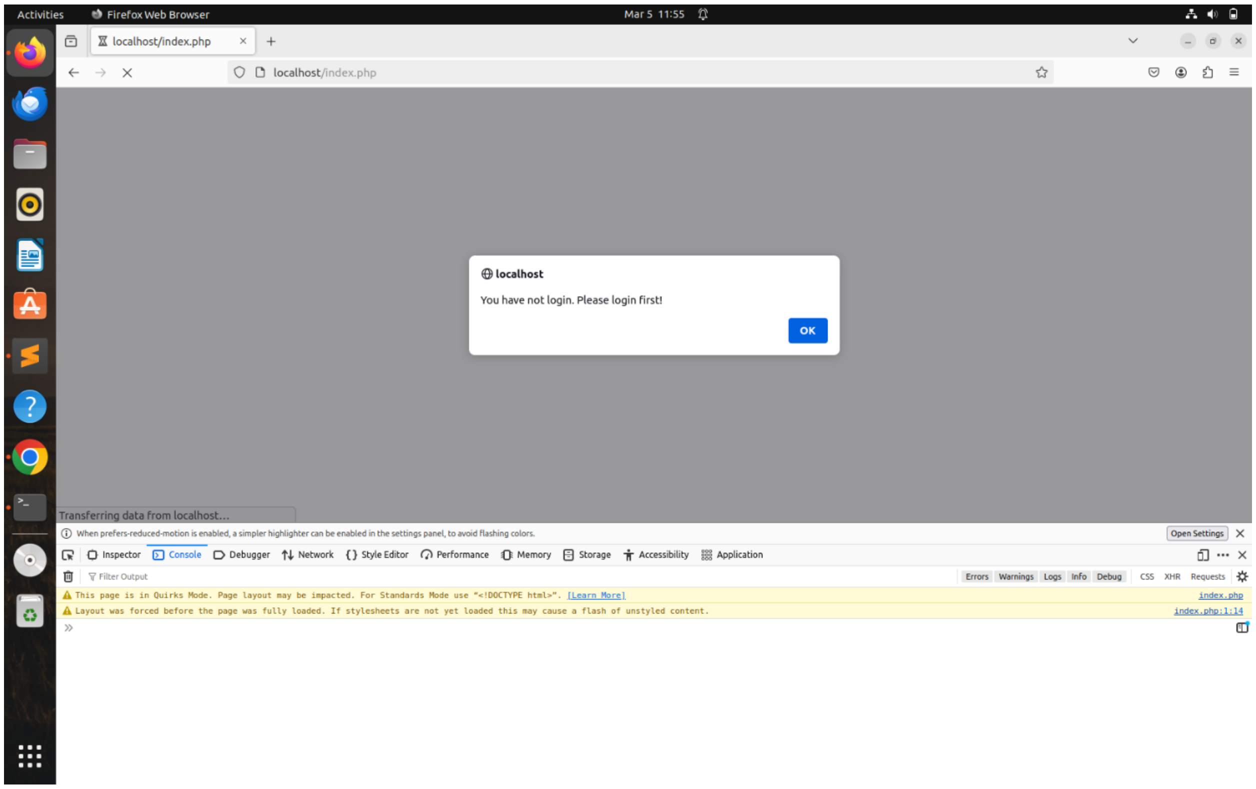
| Status | Met... | Domain | File | Initiator | Type | Transferred | Size |
|---|---|---|---|---|---|---|---|
| 200 | GET | localhost | form.php | document | html | 725 B | 771 B |
| 404 | GET | localhost | favicon.ico | FaviconLoad... | html | cached | 271 B |

```
GET http://localhost/form.php

Status            200 OK
Version           HTTP/1.1
Transferred       725 B (771 B size)
Referrer Policy   strict-origin-when-cross-origin
Request Priority  Highest
DNS Resolution    System

Response Headers (252 B)
Connection: Keep-Alive
Content-Encoding: gzip
Content-Length: 473
Content-Type: text/html; charset=UTF-8
Date: Tue, 05 Mar 2024 16:52:29 GMT
Keep-Alive: timeout=5, max=100
Server: Apache/2.4.52 (Ubuntu)
Vary: Accept-Encoding

Request Headers (538 B)
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
```

```
2 requests   1.04 kB / 725 B transferred   Finish: 142 ms   DOMContentLoaded: 97 ms   load: 103 ms
```

## 2.b. Session Hijacking Attacks

- This task is about session hijacking attack.
- The attack is that when one user logs into a system using a browser like firefox, the attacker copies the session ID and uses that session ID in different browser like chrome into the cookie and accesses the site.
- Here in this task will demonstrate the session hijacking attack. I logged in with the correct credentials in firefox browser and copied the session ID and later pasted the ID in chrome browser and accessed the page. Where I got an alert message saying to login first and this redirects to form.php.
- I set the cookie with ‘document.cookie=”SESSION_ID” ‘ in the Chrome console, now I have refreshed the page and returned to the index.php. Using session ID from the Firefox browser, I logged into the system without providing any user credentials.
- The following screenshot demonstrates a user logged into Firefox, where the session ID has been copied from the console tab and user attempts to access the "index.php" page using Chrome.



localhost

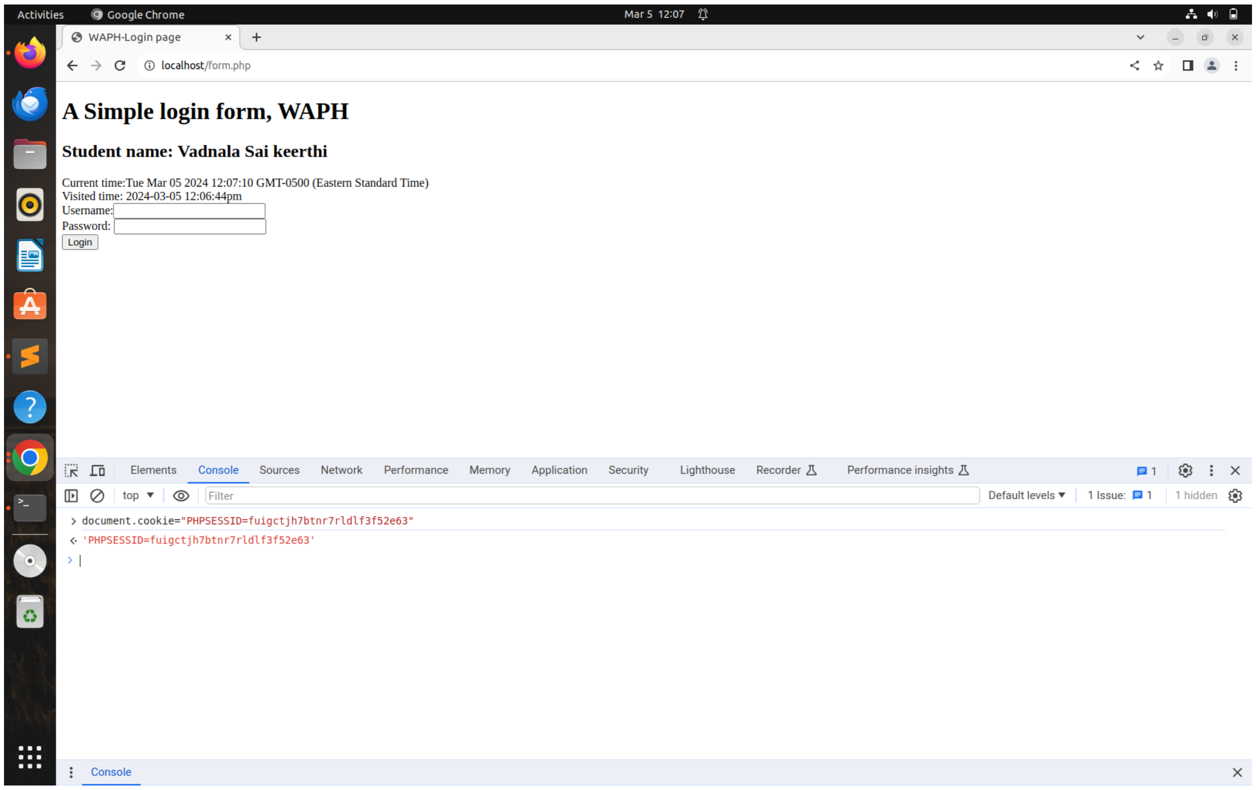
You have not login. Please login first!

Transferring data from localhost...

When prefers-reduced-motion is enabled, a simpler highlighter can be enabled in the settings panel, to avoid flashing colors.

```
This page is in Quirks Mode. Page layout may be impacted. For Standards Mode use "<!DOCTYPE html>". [Learn More]    index.php
Layout was forced before the page was fully loaded. If stylesheets are not yet loaded this may cause a flash of unstyled content.    index.php:1:14
```

- The following screenshot shows upon clicking on OK, the user is redirected to form.php



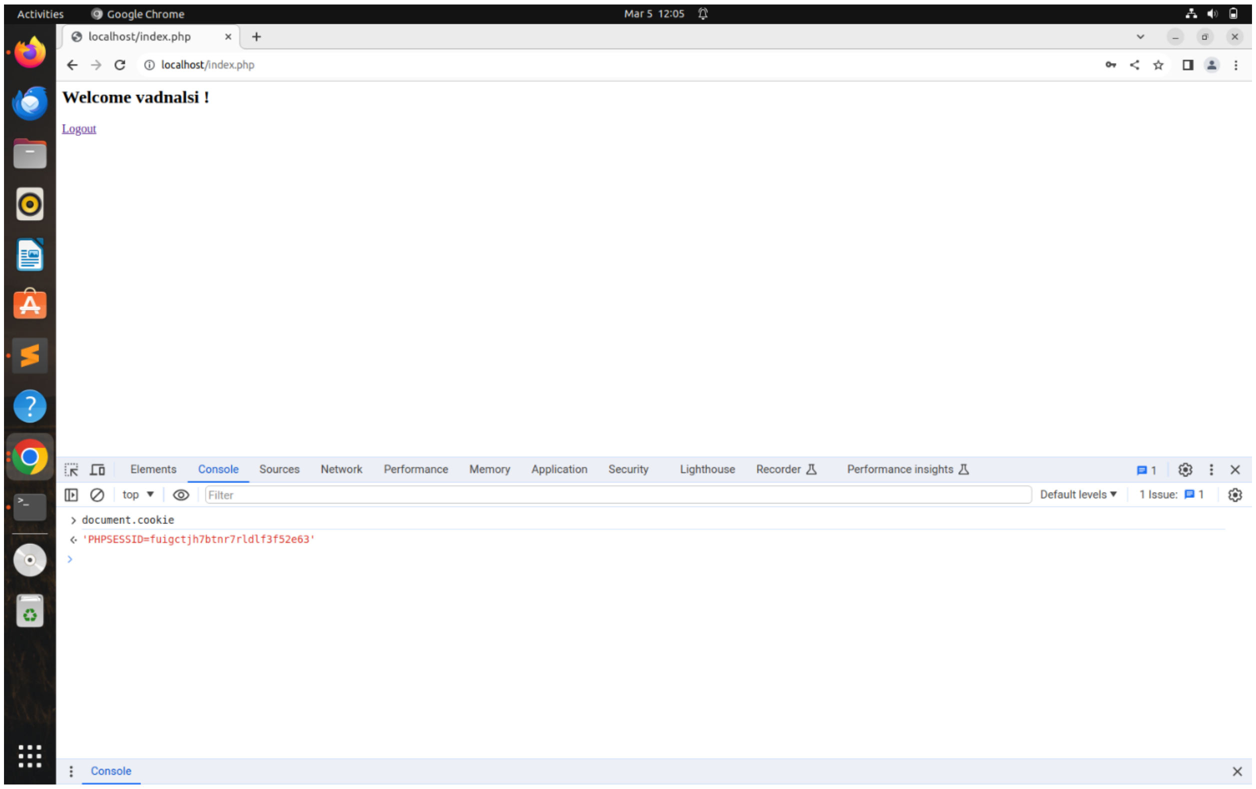
A Simple login form, WAPH

Student name: Vadnala Sai keerthi

Current time:Tue Mar 05 2024 12:07:10 GMT-0500 (Eastern Standard Time)
Visited time: 2024-03-05 12:06:44pm
Username:
Password:

```
> document.cookie="PHPSESSID=fuigctjh7btnr7rldlf3f52e63"
<· 'PHPSESSID=fuigctjh7btnr7rldlf3f52e63'
> |
```

- The following screenshot shows that the user is able to login to the system using the copied session ID from firefox browser.



**Welcome vadnalsi !**

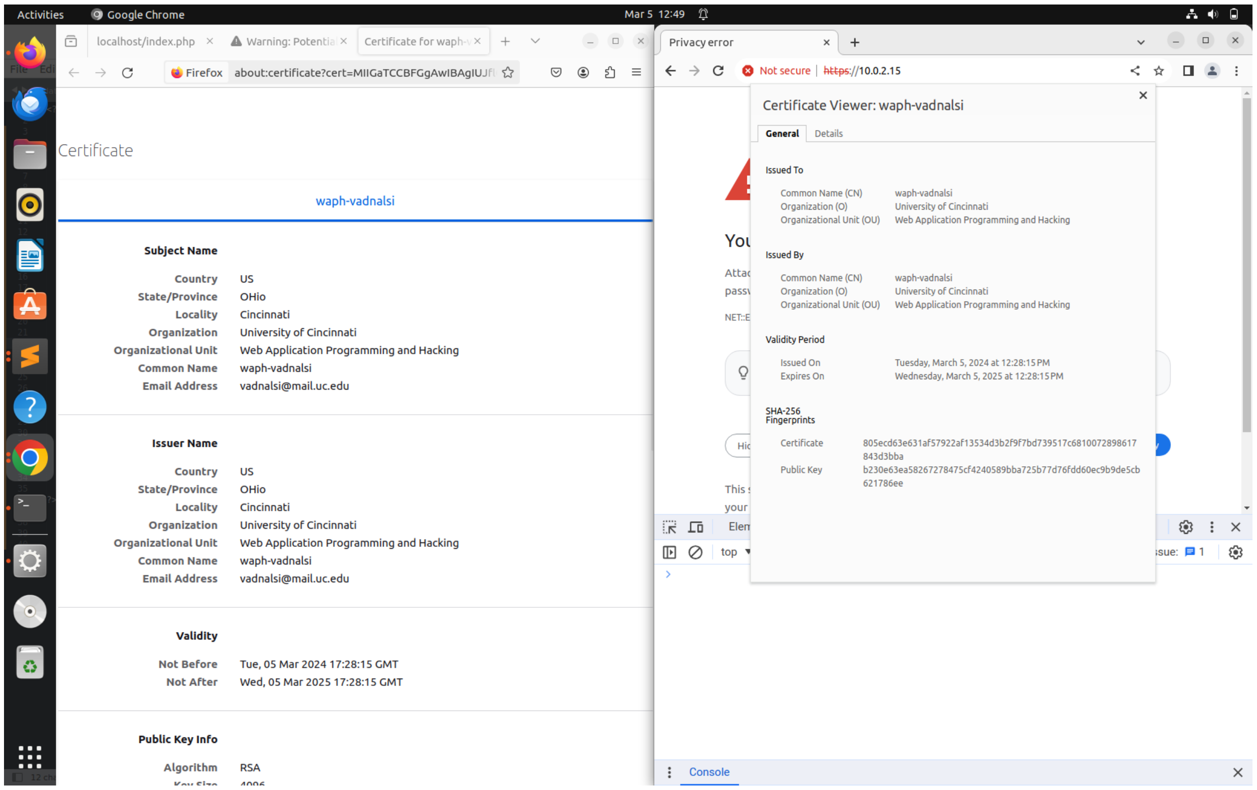
Logout

```
> document.cookie
<· 'PHPSESSID=fuigctjh7btnr7rldlf3f52e63'
```

## Task 3: Securing Session and Session Authentication

### 3.a. Data Protection and HTTPS Setup

- In this task, I have created a folder named sslkey and then created pair of key and certificate.
- Once key pair creation is done, it prompted me to enter details like country name, state, locality name, organization name, unit name, email address.
- Upon longlisting two files were created in sslkey: waph.crt, waph.key.
- default-ssl.conf is a https configuration file and we route this configuration file to waph.crt and waph.key files.
- After enabling the SSL module and restarting the apache we are able to view the certificate.
- I accessed local host from both the browsers which has given an alert message saying it is not secure.
- Later, I modified the host's file by adding the IP address (10.0.2.15) of the machine along with the corresponding domain name. Then, I attempted to access the "form.php" page using the IP address in two different browsers. The screenshot below displays the SSL certificate.



Certificate

waph-vadnalsi

**Subject Name**

| | |
|---|---|
| Country | US |
| State/Province | OHio |
| Locality | Cincinnati |
| Organization | University of Cincinnati |
| Organizational Unit | Web Application Programming and Hacking |
| Common Name | waph-vadnalsi |
| Email Address | vadnalsi@mail.uc.edu |

**Issuer Name**

| | |
|---|---|
| Country | US |
| State/Province | OHio |
| Locality | Cincinnati |
| Organization | University of Cincinnati |
| Organizational Unit | Web Application Programming and Hacking |
| Common Name | waph-vadnalsi |
| Email Address | vadnalsi@mail.uc.edu |

**Validity**

| | |
|---|---|
| Not Before | Tue, 05 Mar 2024 17:28:15 GMT |
| Not After | Wed, 05 Mar 2025 17:28:15 GMT |

**Public Key Info**

| | |
|---|---|
| Algorithm | RSA |

Certificate Viewer: waph-vadnalsi

Issued To

| | |
|---|---|
| Common Name (CN) | waph-vadnalsi |
| Organization (O) | University of Cincinnati |
| Organizational Unit (OU) | Web Application Programming and Hacking |

Issued By

| | |
|---|---|
| Common Name (CN) | waph-vadnalsi |
| Organization (O) | University of Cincinnati |
| Organizational Unit (OU) | Web Application Programming and Hacking |

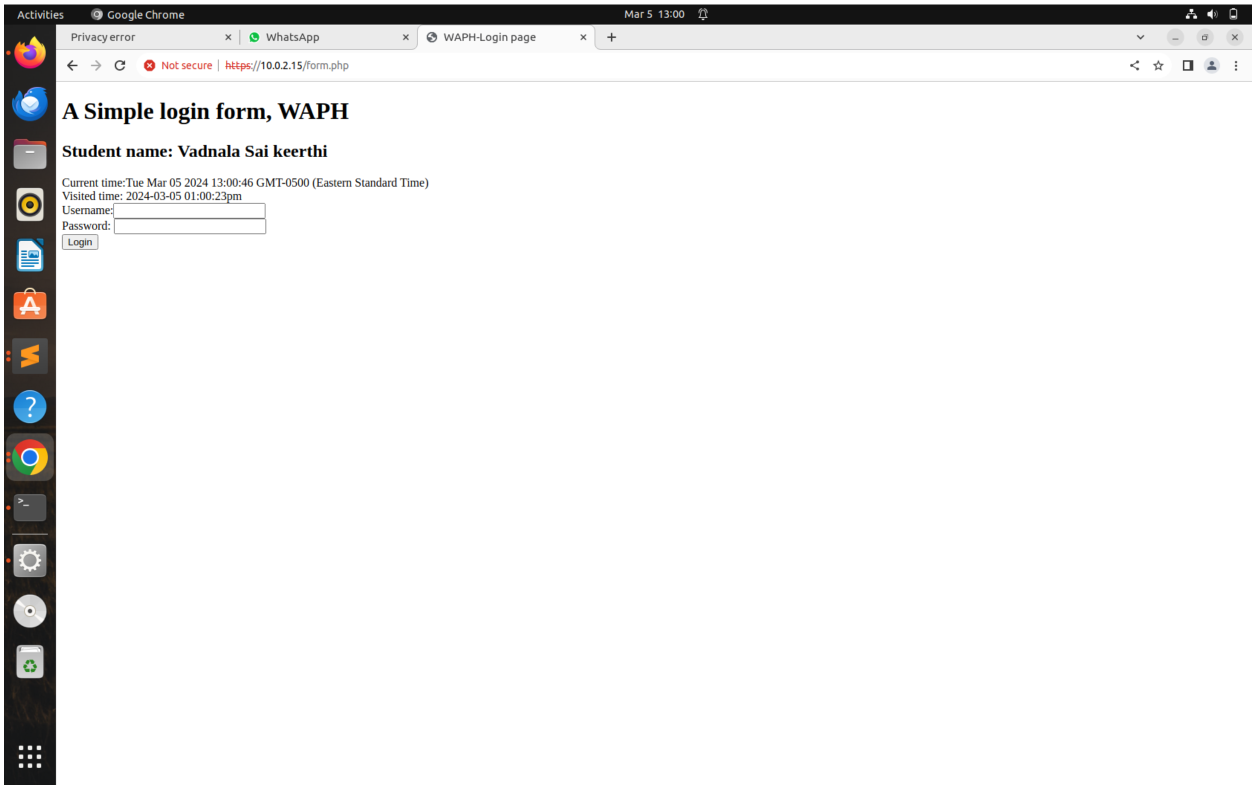
Validity Period

| | |
|---|---|
| Issued On | Tuesday, March 5, 2024 at 12:28:15 PM |
| Expires On | Wednesday, March 5, 2025 at 12:28:15 PM |

SHA-256 Fingerprints

| | |
|---|---|
| Certificate | 805ecd63e631af57922af13534d3b2f9f7bd739517c6810072898617 843d3bba |
| Public Key | b230e63ea58267278475cf4240589bba725b77d76fdd60ec9b9de5cb 621786ee |

- The following screenshot displays accessing the form.php using IP address.



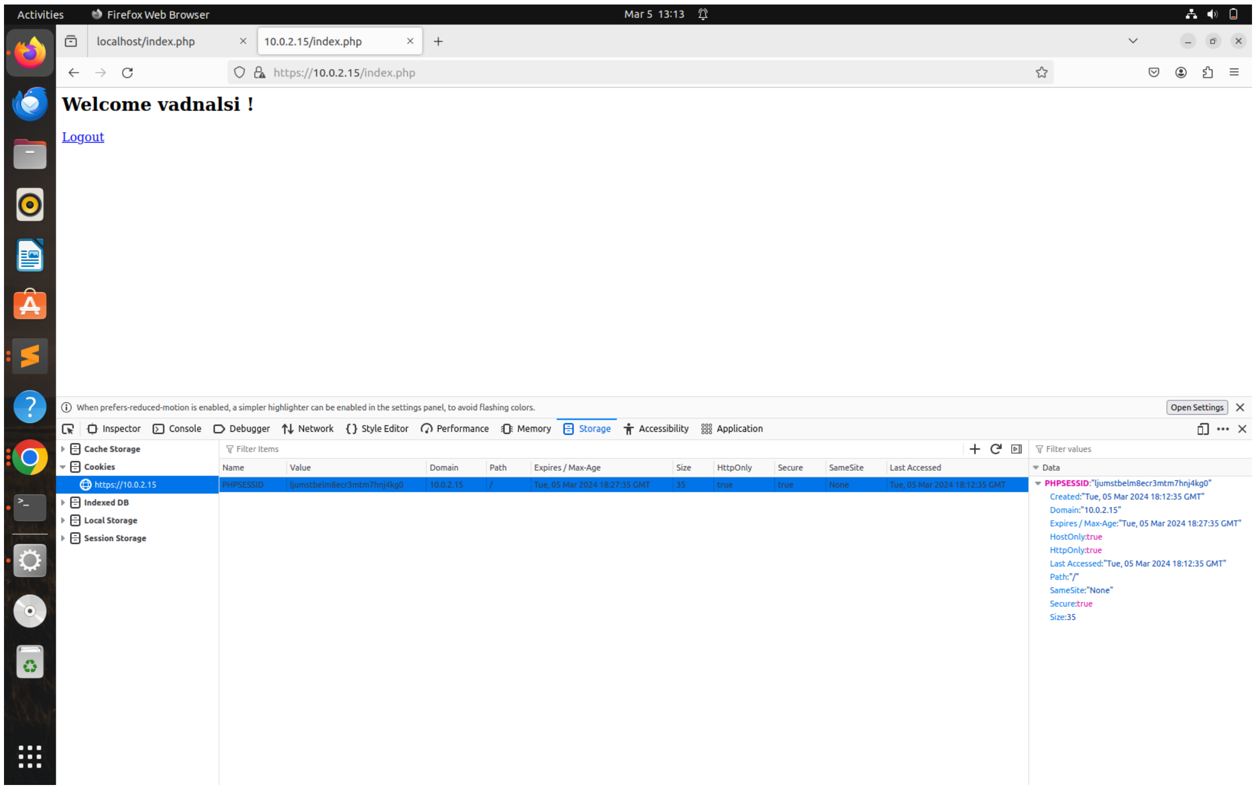
A Simple login form, WAPH

Student name: Vadnala Sai keerthi

Current time:Tue Mar 05 2024 13:00:46 GMT-0500 (Eastern Standard Time)
Visited time: 2024-03-05 01:00:23pm
Username:
Password:
Login

### 3.b. Securing Session Against Session Hijacking Attacks - setting HttpOnly and Secure flags for cookies

- In this task, the index.php file is modified to better session security. Then, I opened the form.php page and logged into it with valid user credentials. However, no changes to the cookie information were found when logged in, which means the site is secure. To maintain the security of the site, I deleted the cookies I had previously set and logged back in with valid user credentials. Now, I checked the storage tab in the inspect of the generated cookie in the browser, and the outcome is shown in the screenshot below.



**Welcome vadnalsi !**

Logout

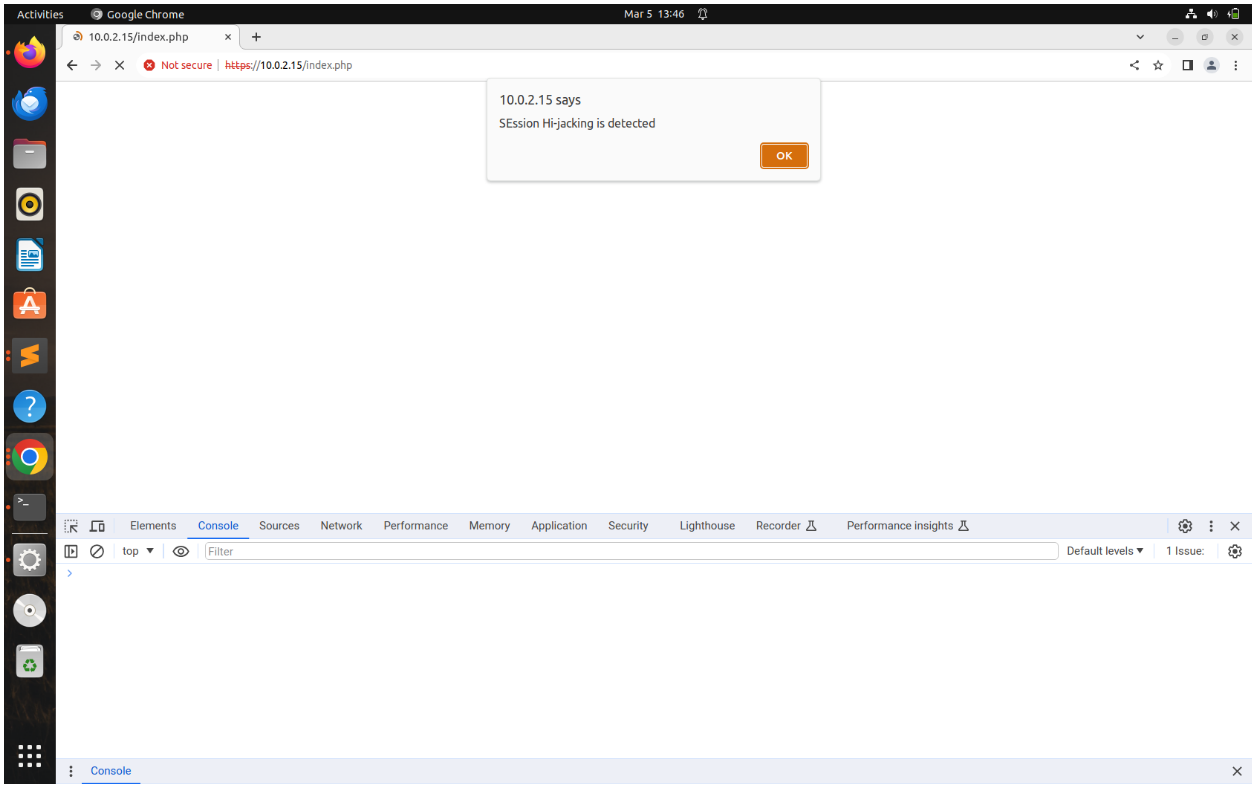
When prefers-reduced-motion is enabled, a simpler highlighter can be enabled in the settings panel, to avoid flashing colors.

| Name | Value | Domain | Path | Expires / Max-Age | Size | HttpOnly | Secure | SameSite | Last Accessed |
|---|---|---|---|---|---|---|---|---|---|
| PHPSESSID | ljumstbelm8ecr3mtm7hnj4kg0 | 10.0.2.15 | / | Tue, 05 Mar 2024 18:27:35 GMT | 35 | true | true | None | Tue, 05 Mar 2024 18:12:35 GMT |

```
Data
PHPSESSID:"ljumstbelm8ecr3mtm7hnj4kg0"
  Created:"Tue, 05 Mar 2024 18:12:35 GMT"
  Domain:"10.0.2.15"
  Expires / Max-Age:"Tue, 05 Mar 2024 18:27:35 GMT"
  HostOnly:true
  HttpOnly:true
  Last Accessed:"Tue, 05 Mar 2024 18:12:35 GMT"
  Path:"/"
  SameSite:"None"
  Secure:true
  Size:35
```

### 3.c. Securing Session Against Session Hijacking Attacks - Defense In-Depth

- This task is about preventing session hijacking, So I increased the security of the index.php file by adding more protection. I changed the code to include a check for the HTTP_USER_AGENT attribute stored in the session browser. If the HTTP_USER_AGENT of the new session does not match the session browser, indicating that it might be hijacked, the user is warned with a message. Upon receiving the warning, users are redirected to the form.php page to re-enter valid credentials.



10.0.2.15 says

SEssion Hi-jacking is detected

OK