# WAPH-Web Application Programming and Hack- ing

**Instructor: Dr. Phu Phung**

# Mini Facebook - Project Team: 31

# Team members

1. Rithu Reddy Vundavelli – vundavry@mail.uc.edu
2. Srujana Reddy Vadagandla – vadagasy@mail.uc.edu
3. Sneha Jillela – jillelsa@mail.uc.edu
4. Soummith Challa – challast@mail.uc.edu

# Project Management Information

Source code repository (private access):

Project homepage (public):

**Revision History**

Date Version Description 23/03/2024 0.1 Init draft

# Overview

The WAPH Team Project aims to create a scaled-down version of Facebook, dubbed "miniFacebook," with a strong emphasis on full-stack web development and secure coding practices. The project adopts an agile methodology, specifically utilizing Scrum techniques.

Sprint 0 kickstarts the collaborative efforts, laying the groundwork for the project's scope and requirements. All team members actively participate by contributing code and documentation to a shared team repository.

To facilitate effective communication and coordination, teams are encouraged to establish dedicated channels for discussion and hold bi-weekly meetings. These meetings serve as opportunities for efficient planning, task allocation, and monitoring the project's progress.

Throughout the project, the team works collectively, leveraging their collective expertise to deliver a functional and secure web application that emulates key features of Facebook while adhering to industry-standard development practices.

# System Analysis

# **Database**

# Security analysis

Here's a paraphrased version with the same side headings:

Session Management:

The project ensures secure session management by implementing session cookie expiration after 15 minutes, securing cookies, and enabling site-wide accessibility.

Authentication:

User credentials are verified through the checklogin\_mysql() function, which securely queries the database to validate the provided username and password combination.

Session Hijacking Prevention:

To mitigate session hijacking attempts, the code compares the user agent of the current request with the one stored in the session.

Security Programming Principles Applied:

Least Privilege: Access to sensitive content is granted only after user authentication.

Input Validation: User inputs are sanitized and validated using htmlentities() to prevent cross-site scripting (XSS) attacks.

Secure Password Storage: Passwords are securely stored using MD5 hashing.

Database Security Principles:

Prepared Statements: SQL injection attacks are prevented through the use of prepared statements.

Minimal Privileges: The database user (waph\_team31) has limited privileges (SELECT) to mitigate risks.

Code Robustness and Defense:

Error Handling: Database connection errors are checked and handled appropriately.

Session Management: Sessions are destroyed, and users are redirected to the login page if authentication fails. Cookie parameters are set to be secure, HttpOnly, and limited to the specified path.

Defenses Against Known Attacks:

XSS: User inputs are escaped using htmlentities() to prevent XSS attacks.

SQL Injection: Prepared statements (stmt->bind\_param()) are utilized to prevent SQL injection attacks.

CSRF: While not explicitly addressed, implementing CSRF tokens would enhance security against cross-site request forgery attacks.

Session Hijacking: User agent comparison ($\_SERVER['HTTP\_USER\_AGENT']) is performed to detect session hijacking attempts.

**Team Screenshots:**

**Team Leader: Rithu Reddy Vundavelli**

1. A screenshot demo for the login system working on the HTTPS team’s local domain

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

# Software Process Management

Here is the passage mentioning that the team follows the Scrum methodology:

Our team followed the Scrum methodology, which emphasizes collaboration, adaptability, and transparency for effective project management. Sprint 0 served as the foundational phase where the groundwork was laid for the entire project. Each team member had clearly defined responsibilities to maximize efficiency.

The team leader initiated the process by establishing the GitHub organization and adding members, creating a centralized hub for collaboration. Team members then contributed by creating repositories, setting up README.md templates, and implementing index.html files.

Collaboration was crucial as team members worked both individually and collectively to address all aspects of the project. Regular meetings were held to discuss progress, address challenges, and plan future steps, promoting effective communication and coordination.

By adhering to Scrum principles and fostering a collaborative environment, our team navigated the complexities of software development with agility and cohesion, ultimately delivering high-quality outcomes.**Scrum process**

**Sprint 0**

Duration: 19/03/2024-DD/MM/YYYY

**Overall team Completed Tasks:**

1. An organization named waph-team31 was created on GitHub, and other members, including phung-waph,

were added as organization members.

2. Member contributions to the team 31 organization:

- A private repository called waph-teamproject was created for the project.

- A public repository named waph-team31.github.io was created to host the team project website.

- The waph-teamproject repository was checked out, and a README.md template was added.

- The waph-team31.github.io repository was checked out, and an index.html template was added.

- The waph-teamproject and waph-team31.github.io repositories were checked out, and code, README.md, and

index.html files were added/revised accordingly. The changes were committed and pushed.

3. A team SSL key/certificate was created, and HTTPS and a team local domain name were set up.

4. The team database was designed and set up.

5. The code skeleton from Lab 3 was copied to the team repository waph-teamproject, and the code was

revised based on Lab 4 requirements. The code, along with README.md and index.html files, was tested and

committed for Sprint 0 submission.**Contributions:**

Member 1, x commits, y hours, contributed in xxx

Member 2, x commits, y hours, contributed in xxx

Member 3, x commits, y hours, contributed in xxx

Member 4, x commits, y hours, contributed in xxx

# Appendix

**We are incorporating the contents of the SQL files and the entire source code of our PHP files within this submission.**

**form.php:**

<!DOCTYPE html>

<html lang='en'>

<head>

<meta charset="utf-8">

<title>WAPH-TEAM31 Login page</title>

<script type="text/javascript">

function displayTime() {

document.getElementById('digit-clock').innerHTML = "Current time:" + new Date();

}

setInterval(displayTime,500);

</script>

</head>

<body>

<h1>A simple login form, WAPH-TEAM31</h1>

<h2>I'm rithu reddy vundaveli, Team Project-31</h2>

<div id="digit-clock"></div>

<?php

//some code here

echo "Visited time: " . date("Y-m-d h:i:sa")

?>

<form action="index.php" method="POST" class="form login">

Username:<input type="text" class="text\_field" name="username" /> <br>

Password: <input type="password" class="text\_field" name="password" /> <br>

<button class="button" type="submit">Login</button>

</form>

</body>

</html>

**index.php:**

<?php

session\_start();

if (checklogin\_mysql($\_POST["username"],$\_POST["password"])) {

?>

<h2> Welcome <?php echo htmlentities($\_POST['username']); ?> !</h2>

<?php

}else{

echo "<script>alert('Invalid username/password');window.location='form.php';</script>";

die();

}

//function checklogin($username, $password) {

//$account = array("admin","1234");

//if (($username== $account[0]) and ($password == $account[1]))

//return TRUE;

//else

//return FALSE;

//}

function checklogin\_mysql($username,$password){

$mysqli = new mysqli('localhost',

'waph-team31',

'Pa$$w0rd'

,'waph\_team');

if($mysqli->connect\_errno){

printf("Database connection failed: %s\n", $mysqli->connect\_error);

exit();

}

$prepared\_sql = "SELECT \* FROM users WHERE username= ?" . " AND password = md5(?);";

$stmt = $mysqli -> prepare($prepared\_sql);

$stmt ->bind\_param("ss", $username, $password);

$stmt ->execute();

$result= $stmt->get\_result();

//echo "DEBUG>sql= $sql"; //return TRUE;

if($result->num\_rows == 1)

return TRUE;

return FALSE;

}

?>

<a href="logout.php">logout</a>