WAPH-Web Application Programming and Hacking

Instructor: Dr. Phu Phung

Project Topic/Title: Mini Facebook - Team 18

Team members

1. Gali Sai Divakar Reddy, galisy@mail.uc.edu

2. Sruthi Sridhar Bopparthi, bopparsr@mail.uc.edu

3. Vikhyath Reddy Bheemreddy, bheemrvy@mail.uc.edu

4. Varun Reddy Patel, patel2vy@mail.uc.edu

Project Management Information

Source code repository (private access): https://github.com/waph-team18/waph-teamproject

Project homepage (public): https://waph-team18.github.io

Revision History

Date	Version	Description
04/02/2024	0.0	Database Setup
04/04/2024	0.0	Project Setup
04/20/2024	0.1	Added PHP
04/21/2024	1.0	Sprint 0 Done
04/21/2024	2.0	Sprint 1 Done
04/22/2024	3.0	Sprint 2 Done
04/25/2024	4.0	Sprint 3 Done
04/25/2024	4.0	project completed
04/25/2024	4.0	project report completed

Overview

The WAPH Team Project centres on the development of a "mini Facebook" web application, with a strong emphasis on employing full-stack web development technologies and secure programming/hacking principles. The project is structured around the Agile methodology, with Sprint 0 serving as the initial phase to facilitate collaboration and prepare the team for the project ahead. In Sprint 0, the team leader takes the lead in setting up a GitHub organization named "waph-team##" and adds all team members as organization members. Each team member is assigned specific tasks: one creates a private repository ("waph-teamproject"), while another establishes a public repository ("waph-team18.github.io") to host the team project website. Additionally, individual members contribute by adding README.md and index.html templates to their respective repositories. Collaboratively, the team proceeds to create a team SSL key/certificate and sets up HTTPS, ensuring secure communication channels. Furthermore, members individually design and set up the team database, laying the groundwork for data storage and retrieval. At the end of Sprint 0, the whole team works together to update the code from Lab 3 to meet the criteria of Lab 4 by transferring it to the "waph-teamproject" repository. Everyone on the team commits the code, README.md, and index.html files in preparation for Sprint 0 submission, after which the code is tested extensively and deployed. During these tasks, the team is told to keep in touch regularly to talk about progress and plan for future tasks.

System Analysis

From the beginning of the project's system design in Sprint 1, we prioritized laying a solid groundwork with a scalable architecture that would guarantee security. We started by building a relational database to hold information about users, posts, and comments. We made sure that the data was very secure by using foreign key restrictions and cascade deletion rules. We also used PHP to build a modular backend architecture, which gave us more control over the system and made it easier to add new features. User-assigned roles and real-time communication capabilities were added to the system as we proceeded to polish its architecture during succeeding sprints of the project. We kept an eye on security throughout the process to make sure the system could handle more sophisticated and heavy loads while still being able to resist common threats. By iteratively improving the design, we were able to meet changing needs while simultaneously enhancing performance and security.

Database design

As part of our project, we have meticulously structured the database to guarantee the seamless operation of our web application. To get things started, we will clean out any comments, posts, users, or superusers data tables that may already

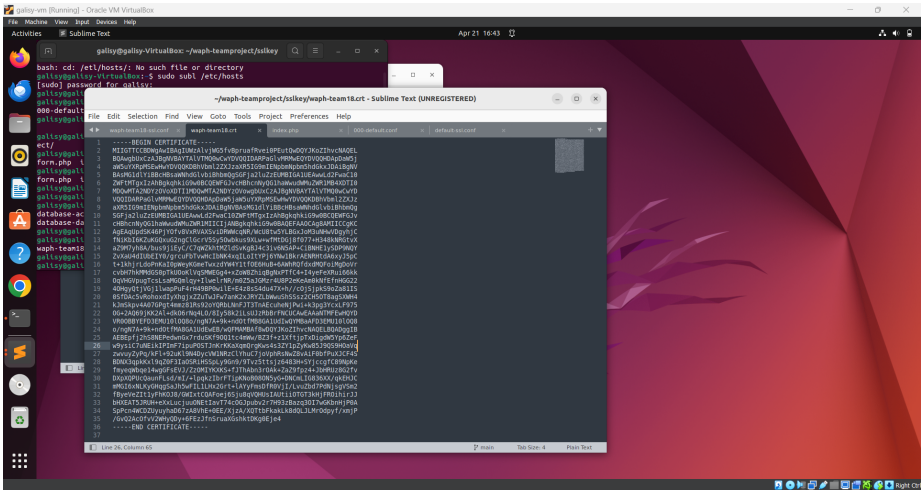
be there. After that, we rearranged these tables. For each user, we keep their name, email address, phone number, password, and active status in the Users Table. Our system makes it easy to enable or disable accounts based on their state, and each user has a unique username.

To facilitate user-to-user communication within the program, the Posts Table and the Comments Table have been designed. The posts table records information such as post titles, content, and authors. Every post is associated with a certain user; therefore, when a user is removed, all of their posts are also removed. Just like that, the comments table has fields for the comment's details and the poster's identity. They are linked to the Users Table, so when a post or person is removed, their comments are also removed. We test the functionality of these tables by adding some sample data. When it comes to our app's database, everything is in good shape.

User Interface

Using HTML, CSS, and an integrated freely available CSS template, our project's user interface was developed to be responsive, easy to use, and aesthetically pleasing. We prioritized usability features such as responsive design for various screen sizes, simple navigation, and interactive components like buttons and forms. The upgraded user interface was immediately apparent in the simplified design of the application's login, registration, and post-interaction pages, guaranteeing a pleasant and intuitive experience for all users.

The setup was successfully configured and tested by every team member on their PC. Below are the testing screenshots that were also shared.



Implementation

Sprint 0: Getting Things Started and Collaborating as a Team

Sprint 0 was all about getting our web app off the ground. With the help of Scrum principles, we set up the development environment and got the team working together. Configuring branches for various contexts, enlisting team members as collaborators, and setting up the project repository on GitHub were all part of this process. By creating an SSL certificate for the project domain and setting up HTTPS on our web server, we also made sure that data was transmitted securely. A local development database instance was created and initial data was loaded into it. Scripts and database setup files were organized within the repository. Also, the login system code migration and testing were finished, so everything is working and reliable now. We held review sessions and kept detailed documentation throughout the sprint to make sure everyone was on the same page and that any problems were resolved quickly.

Sprint 1: Database Integration and User Functionality

In Sprint 1, we improved our app by adding features like `registrationform.php` and `form.php`, which allow users to register and log in. Furthermore, we prioritized user security by incorporating bcrypt password hashing into the registration process. The creation of `editprofileform.php`, which enables users to safely change their information, established the groundwork for user profile management. In addition, we securely authenticated users and maintained their sessions by integrating session management through `session_auth.php`. We also started building the database structure, making tables for users and posts to efficiently store important data. In order to make future development and collaboration easier, every work was carefully recorded. All things considered, Sprint 1 was a huge step forward in defining critical user functions and setting the stage for future feature development.

Sprint 2 focuses on enhancing user interaction and implementing security measures. During Sprint 2, we focused on making the user experience better and adding more security features. The `newpost.php`, `editpost.php`, and `comment.php` functionalities were developed to enable users to create, edit, and comment on posts through these modules. Our team implemented role-based access control in `session_auth.php` to prevent unauthorized users from editing any user-created posts and prevent data corruption. To further strengthen security, we included security features such as CSRF protection for submissions of forms and thoroughly sanitized inputs to mitigate cross-site scripting threats. The goal of these improvements was to make the program more dynamic and user-friendly while also making user interactions safer.

Throughout Sprint 3, administrative capabilities were enhanced. The ability to activate or deactivate user accounts via superuserprofile.php has been granted to superusers, ensuring that account status can only be modified by authorized individuals. This role-specific functionality is critical for administering user

access securely.

Security Analysis

1. We used strong security programming concepts in our project, To keep our web application safe. Attacks such as the injection of SQL and XSS scripts were prevented by strictly enforcing input validation. The form.php and addnewpost.php scripts show how this was accomplished by sanitizing all user inputs using PHP methods like htmlspecialchars and mysqli_real_escape_string. Our registrationform.php scipt and changepassword.php scripts demonstrate that we implemented Secure Authentication by storing passwords using powerful hashing algorithms like bcrypt. This enhances the security of our user credentials like login information and protects them from possible data breaches and threats. Important parts of our security plan also included handling errors and managing sessions. We employed a few secure session management methods like HTTPS and specified cookie characteristics like Private and HttpOnly in the sessionauth php file to avoid session hijackings. We installed comprehensive logging to identify suspicious activity and ensure that warning signals were common to prevent the disclosure of sensitive system information. This will aid in forensic investigation following security occurrences. Database connectivity was securely handled and configured to withstand multiple attack vectors in the database.php file, which played a significant role in this. By taking these precautions, we could build a web application that is both trustworthy and durable, one that follows industry standards and mitigates threats caused by online defects.

2. Database security principles that are used in the project:

For the sake of our project's data privacy and integrity, we implemented multiple database security standards. Protecting data while it was in transit from our application to the database server was our top priority, so we implemented Connection Encryption in the database.php file. Data transmission over the network is thus protected against snooping. The implementation of role-based restrictions within the database was another way that we utilized Access Controls. Clients and amenities could only access the data they needed to perform their tasks, thus reducing the likelihood of illegal access. All user interactions were carefully managed according to these access regulations. This included adding new posts in newpost.php and changing profile information in editprofileform.php.

3. Is your code robust and defensive? How?

Yes, we made sure that security and resilience were top priorities in our code. All across our program, we have integrated comprehensive error-handling techniques to guarantee resilience. Files such as `editpost.php` and `newpost.php` make use of try-catch structures and conditional tests to handle errors gently, guaranteeing that the application can continue to function even when unexpected problems

arise.

4. How did you defend your code against known attacks such as XSS, SQL Injection, CSRF, and Session Hijacking?

We used a variety of security techniques within the project files to protect them from common web threats including XSS, SQL Injection, CSRF, and Session Hijacking. A lot of input validation and sanitization was done to prevent SQL Injection and XSS issues. Files such as `form.php` and `newpost.php` use PHP's htmlspecialchars function to remove malicious characters from user inputs. This helps to prevent cross-site scripting attacks by making sure that inputs are shown safely. When communicating with the database, these files and others like `editpost.php` use prepared statements with query parameters to prevent SQL Injection. By separating the data from the command, this method prevents attackers from manipulating the SQL statements themselves.

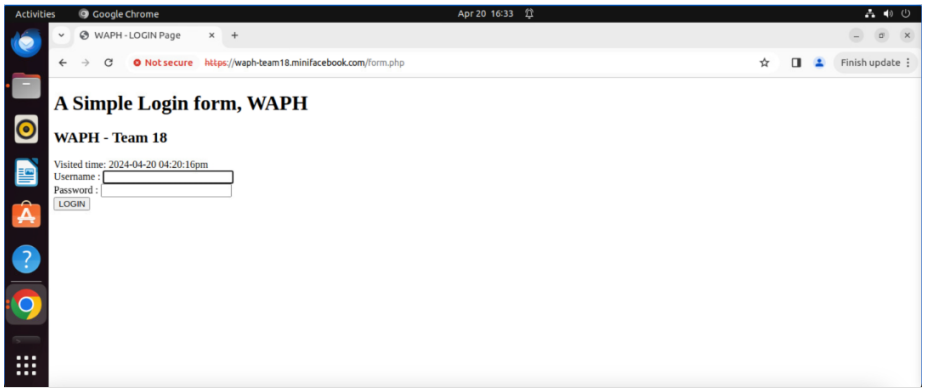
Our program uses secure session handling procedures and CSRF tokens to protect users against Session Hijacking and Cross-Site Request Forgery attacks. An essential part of session security is the `session_auth.php` file, which uses attributes like Secure and HttpOnly to implement secure session cookies. Because of these features, it is difficult for malicious actors to execute scripts on the client side and take over user sessions. Additionally, as can be seen in files such as `editprofileform.php` and `changepasswordform.php`, to handle CSRF concerns, distinct tokens are created and validated upon submission of each form. This system is in place to make sure that requests are coming from legitimate users and not malicious third-party websites. Tokens that are closely associated with the user's session are validated for each important action, which strengthens protections against cross-site request forgery (CSRF) attacks.

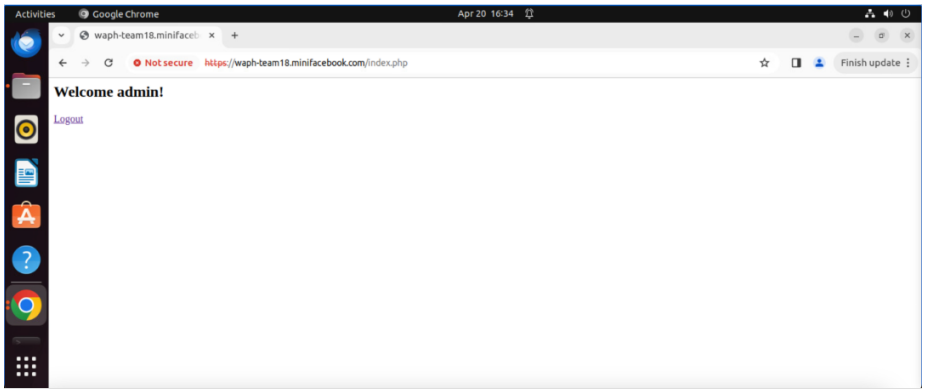
5. How do you separate the roles of super users and regular users?

Demo (screenshots)

In Sprint-0, we effectively conducted testing on the login page of our small-scale Facebook application.

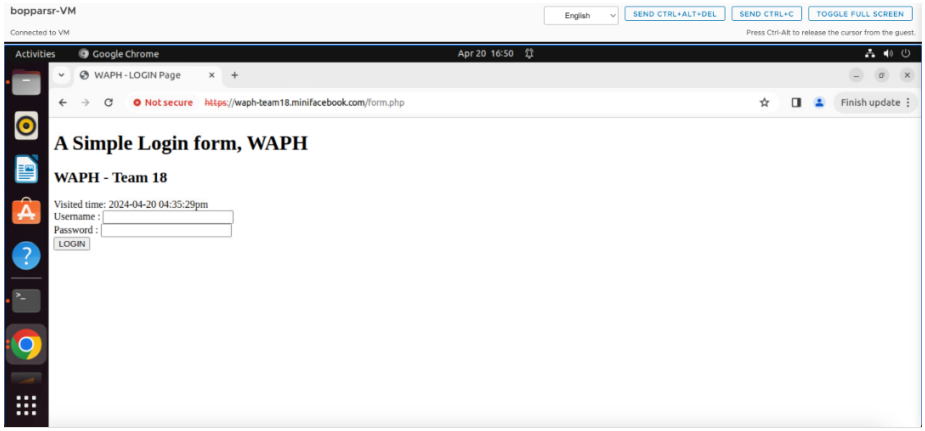
Group Screenshots-

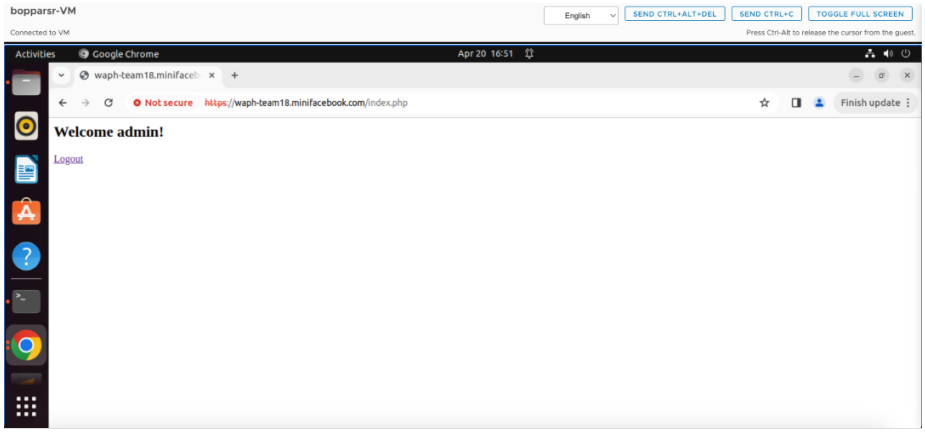




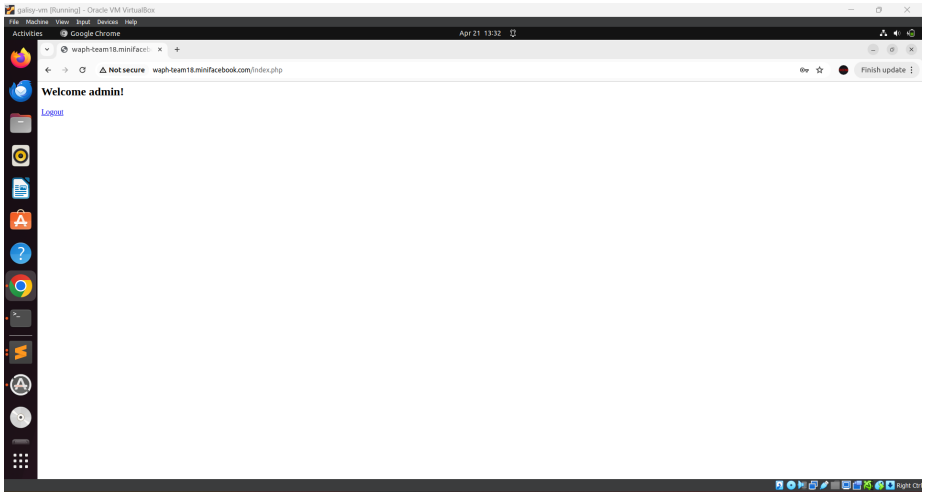
Individual Screenshots-

Sruthi Sridhar Screenshots-

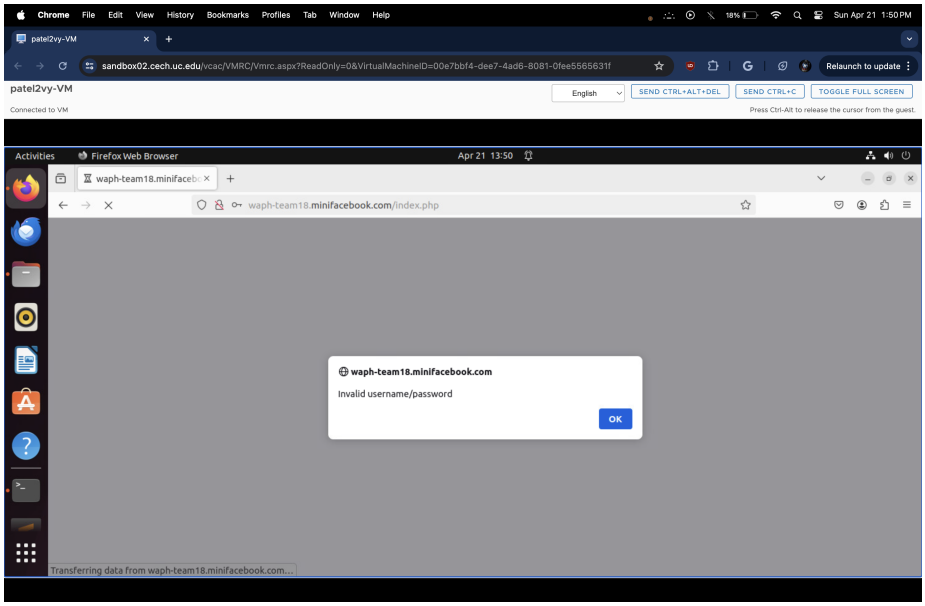


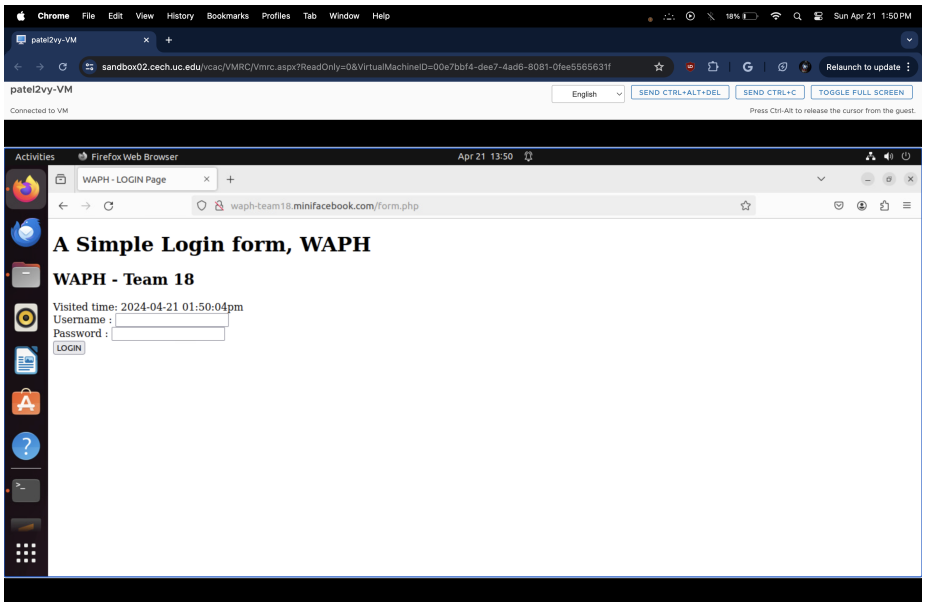


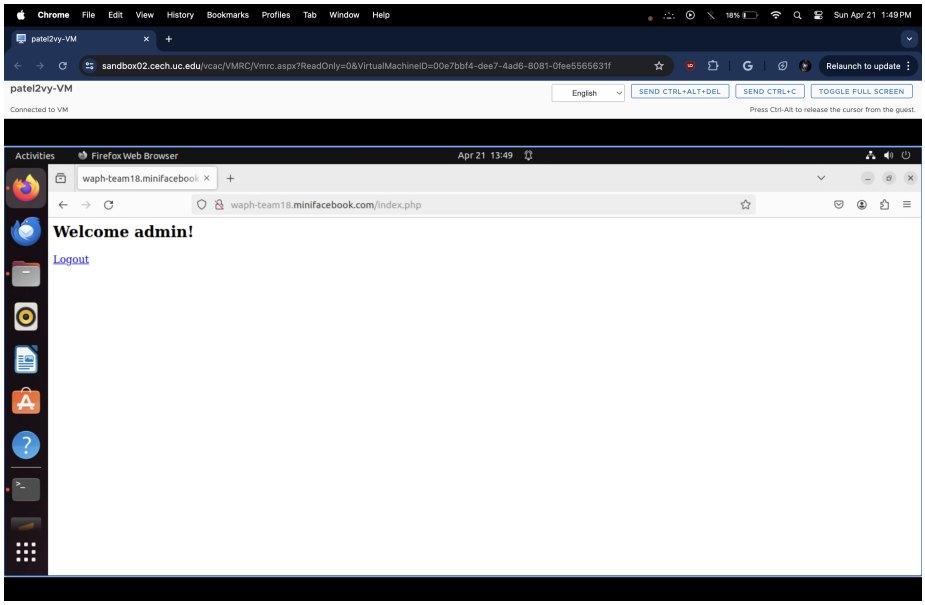
Gali Sai Divakar Reddy Screenshots-



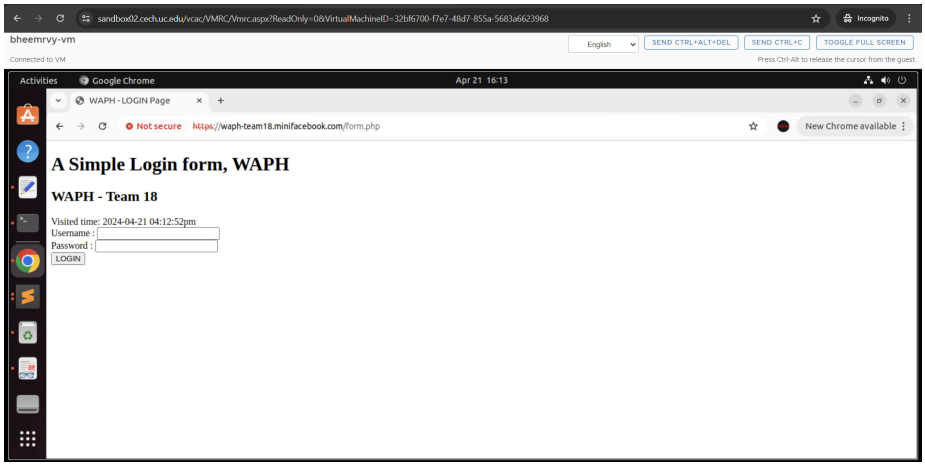
Varun Reddy Patel Screenshots-

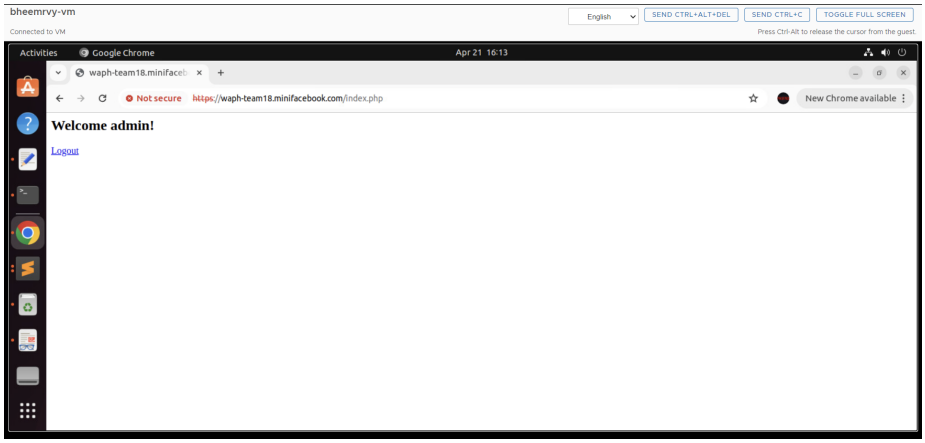



Chrome File Edit View History Bookmarks Profiles Tab Window Help
Sun Apr 21 1:50 PM
patel2vy-VM
sandbox02.cech.uc.edu/vcac/VMRC/Vmrc.aspx?ReadOnly=0&VirtualMachineID=00e7bbf4-dee7-4ad6-8081-0fee5565631f
Relaunch to update
patel2vy-VM
English
SEND CTRL+ALT+DEL
SEND CTRL+C
TOGGLE FULL SCREEN
Connected to VM
Press Ctrl-Alt to release the cursor from the guest.
Activities
Firefox Web Browser
Apr 21 13:50
WAPH - LOGIN Page
waph-team18.minifacebook.com/form.php
A Simple Login form, WAPH
WAPH - Team 18
Visited time: 2024-04-21 01:50:04pm
Username :
Password :
LOGIN

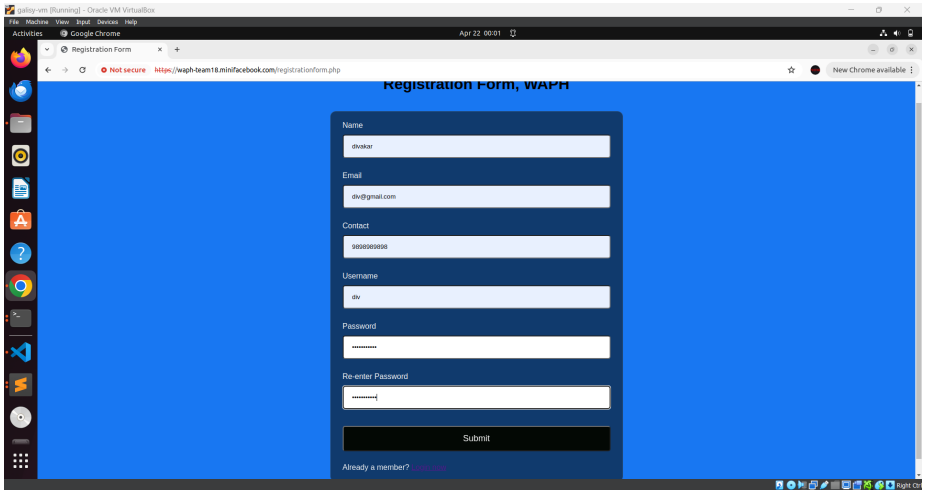

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Sun Apr 21 1:49 PM
patel2vy-VM
sandbox02.cech.uc.edu/vcac/VMRC/Vmrc.aspx?ReadOnly=0&VirtualMachineID=00e7bbf4-dee7-4ad6-8081-0fee5565631f
Relaunch to update
patel2vy-VM
English
SEND CTRL+ALT+DEL
SEND CTRL+C
TOGGLE FULL SCREEN
Connected to VM
Press Ctrl-Alt to release the cursor from the guest.
Activities
Firefox Web Browser
Apr 21 13:49
waph-team18.minifacebook
waph-team18.minifacebook.com/index.php
Welcome admin!
Logout

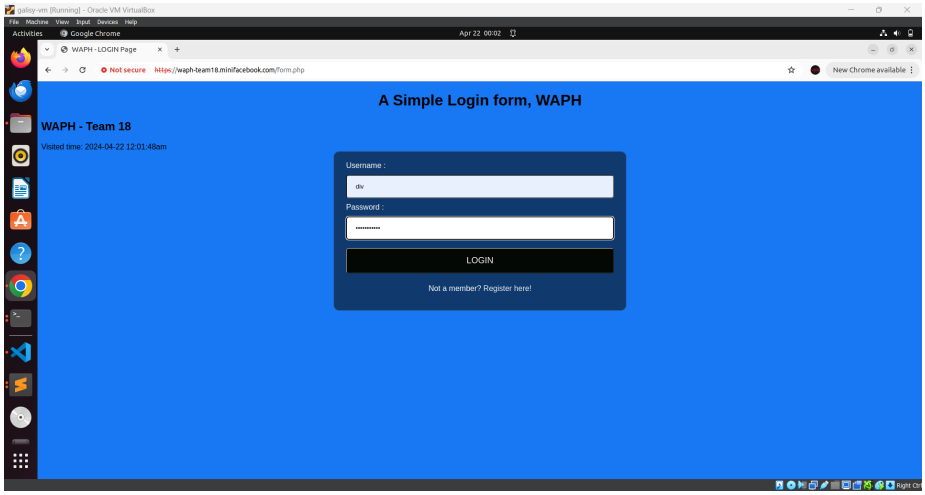
Vikhyath Reddy Bheemreddy Screenshots-

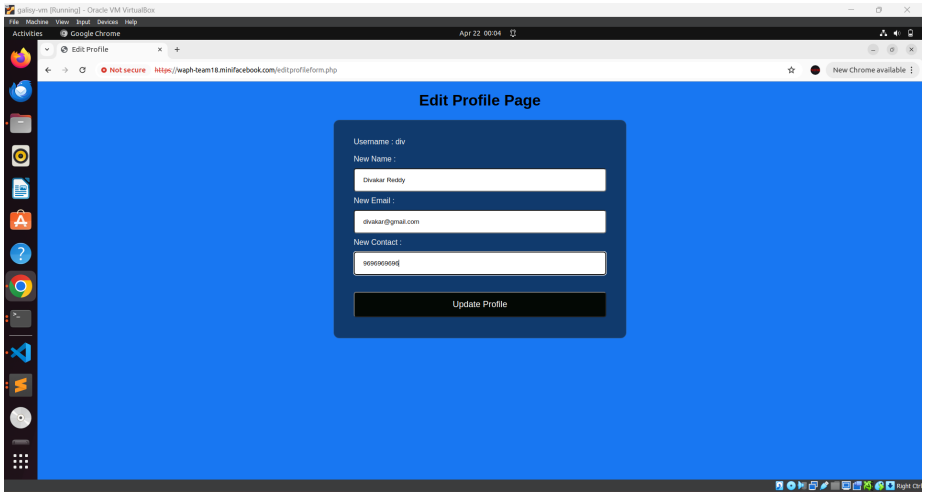


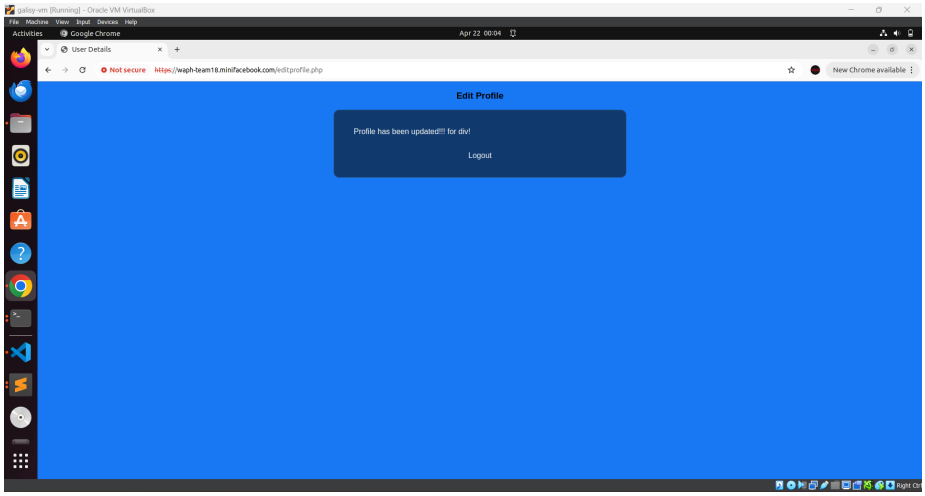


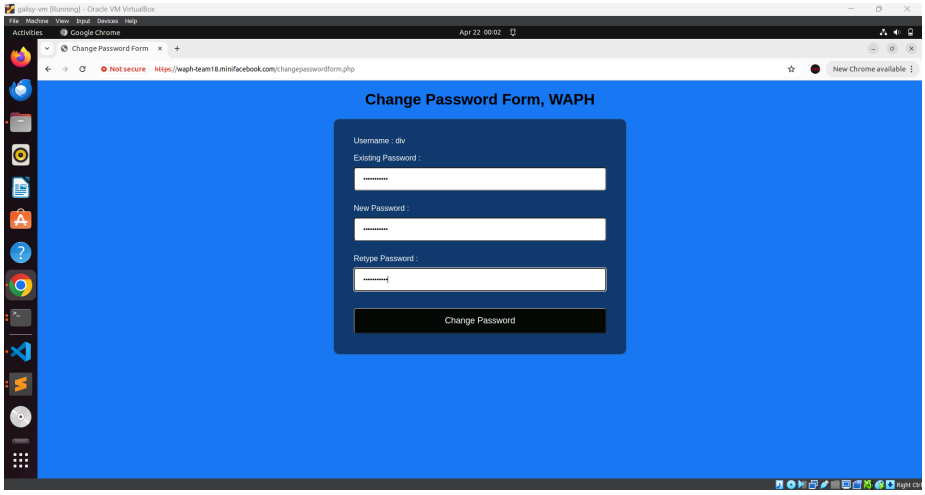
Demo Screenshots of sprint-1

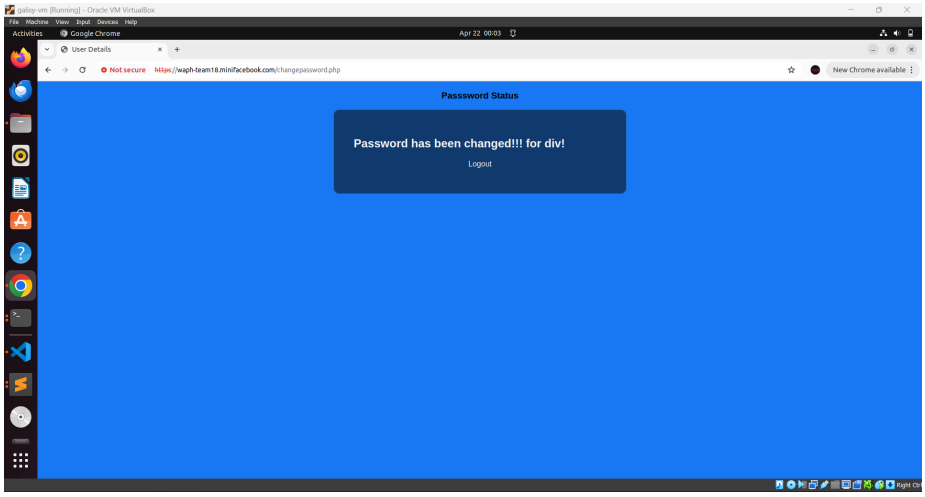


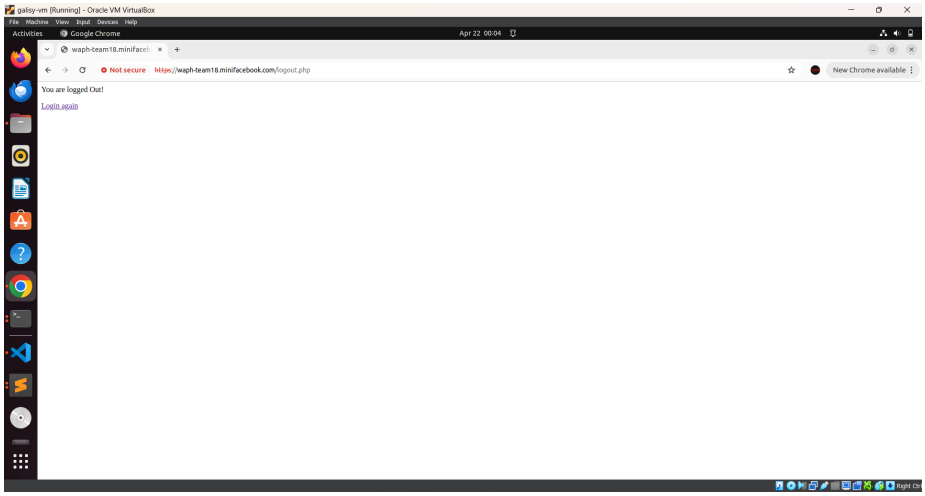



Edit Profile
Not secure https://waph-team18.minifacebook.com/editprofileform.php
New Chrome available
Edit Profile Page
Username : div
New Name :
Divakar Reddy
New Email :
divakar@gmail.com
New Contact :
9696969696
Update Profile

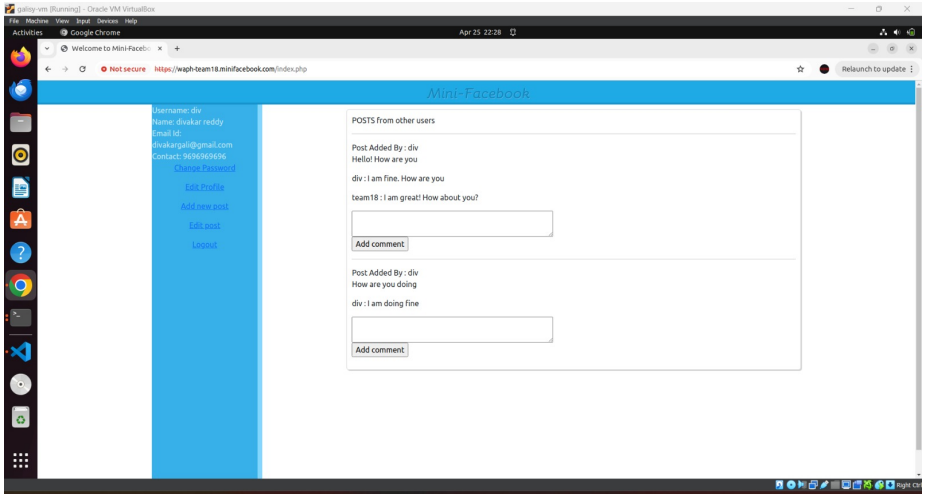

User Details
Not secure https://waph-team18.minifacebook.com/editprofile.php
New Chrome available
Edit Profile
Profile has been updated!!! for div!
Logout

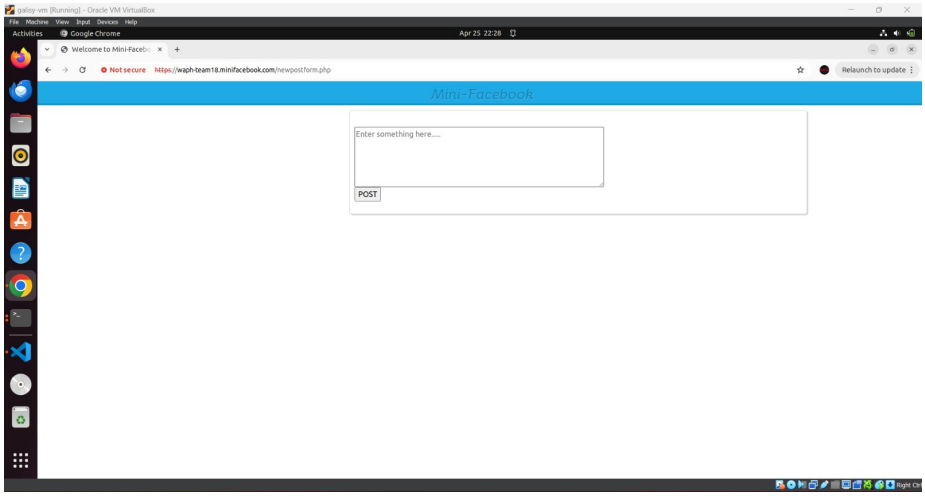

Change Password Form
Not secure
https://waph-team18.minifacebook.com/changepasswordform.php
New Chrome available
Change Password Form, WAPH
Username : div
Existing Password :
New Password :
Retype Password :
Change Password

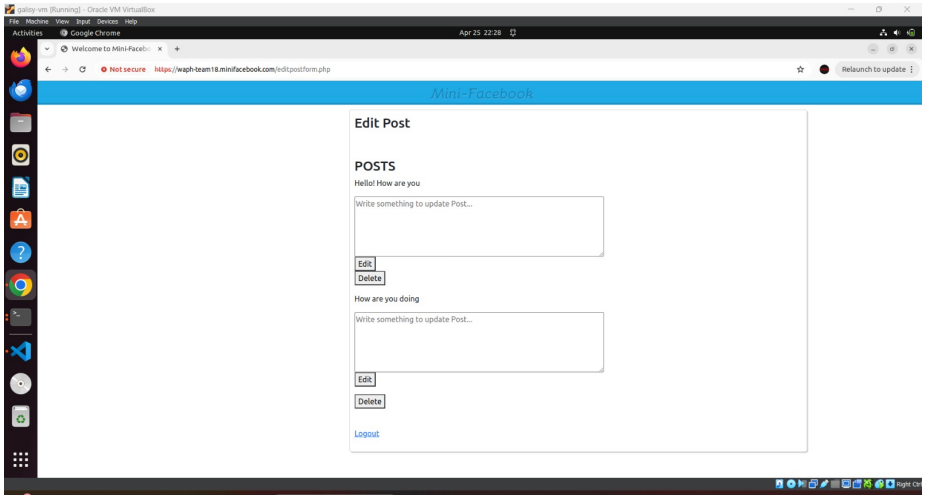

User Details
Not secure
https://waph-team18.minifacebook.com/changepassword.php
New Chrome available
Password Status
Password has been changed!!! for div!
Logout

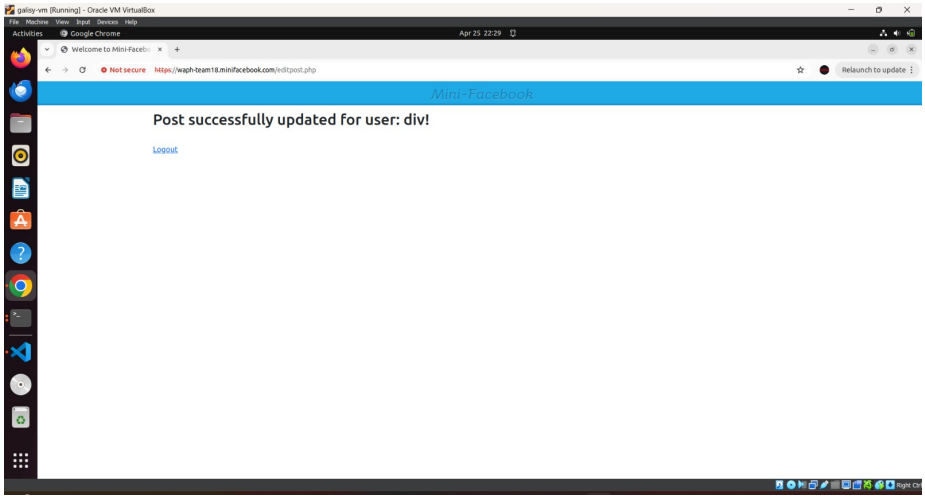


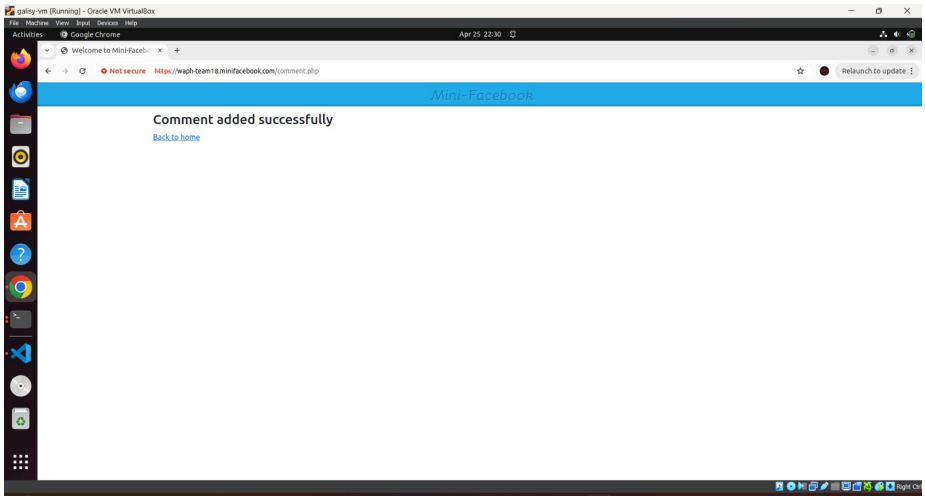
Demo Screenshots of sprint-2



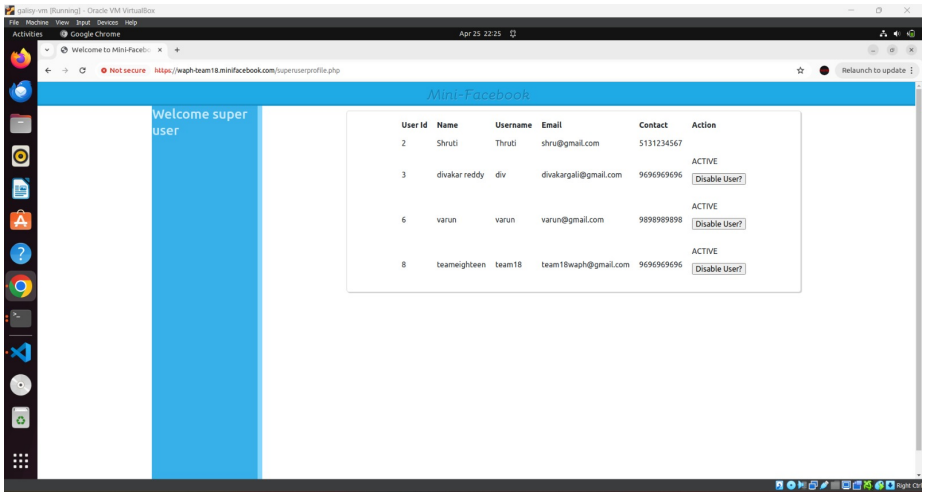

Mini-Facebook
Enter something here.....
POST

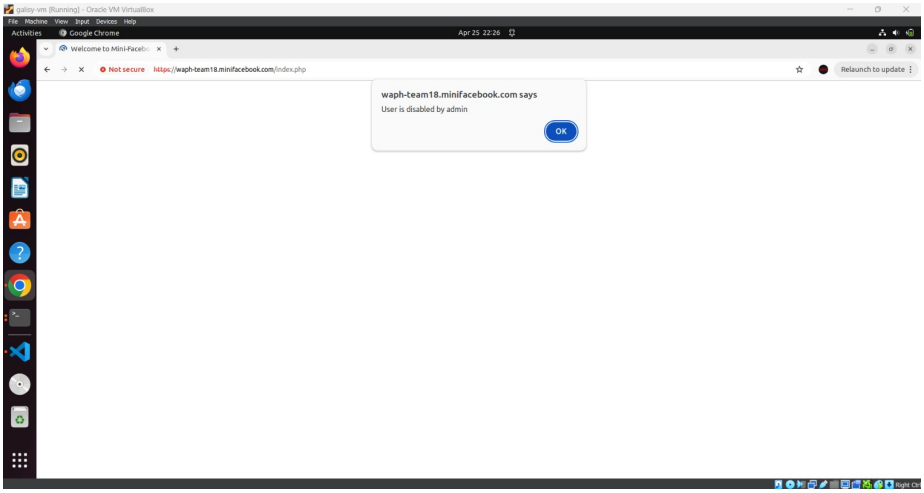

Mini-Facebook
Edit Post
POSTS
Hello! How are you
Write something to update Post...
Edit
Delete
How are you doing
Write something to update Post...
Edit
Delete
Logout


galisy-vm [Running] - Oracle VM VirtualBox
Activities
Google Chrome
Apr 25 22:29
Not secure
https://waph-team18.minifacebook.com/editpost.php
Relaunch to update
Mini-Facebook
Post successfully updated for user: div!
Logout


galisy-vm [Running] - Oracle VM VirtualBox
Activities
Google Chrome
Apr 25 22:30
Not secure
https://waph-team18.minifacebook.com/comment.php
Relaunch to update
Mini-Facebook
Comment added successfully
Back to home

Demo Screenshots of sprint-3 (Superuser disable/enable account)





Managing Software Processes

Our team used Scrum, a disciplined approach to software workflow management, to make sure the project moved along smoothly from sprint to sprint. We built a specialized webpage for consolidated information access, established communication procedures using Microsoft Teams, and generated extensive documentation via an extensive README file in the first phase, i.e in Sprint 0. In addition, we established the framework for the project's infrastructure, which allowed for precise job allocation and encouraged open lines of communication, paving the way for later phases of growth.

Expanding upon this groundwork, our attention was directed towards the integration of critical functionalities during further sprints i.e (Sprints-1 and sprint-2) of the project. Key user capabilities like enrollment, login, management of passwords, and profile editing were integrated during sprint 1's database setup. In this stage, we completed the backend architecture for safe and efficient user data management. In Sprint 2, we moved our attention to improving the system's functionality and interaction by adding tools like commenting and post management, which made users more engaged with the platform.

Sprint 3: Superuser Capabilities During Sprint 3, we made a substantial improvement to the application's administrative control features, giving superusers access to advanced capabilities such as the ability to enable or disable user accounts. Furthermore, we integrated a real-time chat platform to enable users to engage in dynamic conversations. The main goals of this sprint were to improve administrative and user functionalities and to strengthen security measures. We were able to bring the project in on schedule by strictly following the Scrum approach and keeping in constant contact through Microsoft Teams. Collaboration and work management were made easier with this technique throughout the sprints.

The Scrum workflow

Sprint-0

Completed work:

1. Setting Up Github Repositories
2. Creating a team SSL key and Setting Up HTTPS
3. Management of scripts and team database setup
4. Migration and testing of the login system code

team members contribution:

1. Gali Sai Divakar Reddy's contribution in Database setup was 3 hours and 2 commits.
2. Sruthi Sridhar Bopparthi contribution 3 commits, 3 hours, contributed in creating login system in waph-teamproject.
3. Vikhyath Reddy Bheemreddy's Contributed in index.html file for the waph-team18.github.io website with 3 commits and 2 hours
4. Varun Reddy Patel's contribution 12 commits,2 hours, contributed to creating README.md file and developing with-team project.

Sprint-1

Completed Tasks:

1. Task-1: Updated Database by creating new tables - posts, messages, comment
2. Task-2: Register Page design for the users to be able to Register to the web application
3. Task-3: Login page design for the users to be able to Login to the web application
4. Task-4: Users can able to edit the profile like name, email and contact information
5. Task-5: Registerd users can be able to edit/change thier password
6. Task-6: Added CSS style sheet to the web application
7. Task-7: Updated README file
8. Task-8: Logout functionality

team members contribution:

1. Gali Sai Divakar Reddy's contribution in completing Task-4, Task-5, Task-6, Task-7 was 5 hours and 3 commits.
2. Sruthi Sridhar Bopparthi contribution 7 commits, 5 hours, contributed in Task-2, Task-4 frontend, and created `addnewuser.php`, `registrationform.php` and `editprofileform.php` file.
3. Vikhyath Reddy Bheemreddy's Contributed in Task-5 frontend, Task-1, and created `session_auth.php`,`changepasswordform.php`,

`database-data.sql` with 3 commits and 4 hours.

4. Varun Reddy Patel's contribution 3 commits,4 hours, contributed in Task-3, Task-8, and connecting the database to the frontend `database.php`.

Sprint Retrospection:

Good	Could have been better	How to improve?
Database Design and Development	Implementation and design of well-structured databases	Keep improving and updating the database structure as development goes on to make sure it can adapt to new requirements.
Change Password	Execution and operation of the password-changing function	The password-changing procedure must undergo thorough testing to confirm its operation and security.
Sign Up and Login Process for Users	Successful implementation of registration and login functionalities	Ensure thorough testing of registration and login processes to identify and address any potential issues.

Sprint-2

Completed Tasks:

1. Task-1: Readjusted Database by creating new tables - posts, messages, comment
2. Task-2: The user can view the posts of other users after a successful login
3. Task-3: The user can Add a new post after a successful login
4. Task-4: A user can only update his post i.e edit or delete a post
5. Task-5: Users can comment on other users' posts
6. Task-6: Updated README file

team members contribution:

1. Gali Sai Divakar Reddy's contribution in completing Task-3, Task-6, and updated `Index.php` file, 6 hours and 5 commits.
2. Sruthi Sridhar Bopparthi's contributed 3 commits, 6 hours, contributed in Task-4 frontend and backend, and also made changes to the `database.php` file.
3. Vikhyath Reddy Bheemreddy's Contributed in Task-5, and Task-4 deleted post with 2 commits and 4 hours.

4. Varun Reddy Patel's contribution 2 commits,4 hours, contributed in Task-3 and contributed in solving the bugs and documentation

Sprint Retrospection:

Good	Could have been better	How to improve?
Added features focused on the experience of the user	Improved feature robustness testing procedures	Find and fix problems by conducting comprehensive testing.
Efficient implementation of core features	The front end and back end should work together more efficiently.	Promote efficient collaboration for holistic growth.
Prioritized safety and reliability of information	Streamline the explanation of new features in the documentation.	Please create thorough documentation that can be used for future reference.

Sprint-3

Completed Tasks:

1. Task-1: Added superuser functionality to allow administrators to activate or deactivate user accounts, granting them the authority to control user access.
2. Task-2: involved implementing security mechanisms to prevent a disabled account from logging in, hence improving system security and safeguarding user data.

team members contribution:

1. Gali Sai Divakar Reddy's contribution of 6 commits, 4 hours, in developing superuser functionality and created `admin_session_.php` helped in

preparing the README.md file.

2. Sruthi Sridhar Bopparthi contribution of 5 commits, 5 hours, in creating the superuser functionality by creating `adminlogin.php`, `superuserprofile.php`, `activeuser.php`, `disableuser.php`.
3. Vikhyath Reddy Bheemreddy's Contributed 7 commits, 5 hours in creating the assets folder and updating the index and logout forms.
4. Varun Reddy Patel's contribution of 5 commits,4 hours, contributed to implementing security mechanisms to prevent a disabled account from logging in and helped in preparing the README.md file.

Sprint Retrospection:

Good	Could have been better	How to improve?
Administration of user accounts by a superuser	Enhanced ability to precisely manage account features	Improve the functionality of superuser features by implementing comprehensive activity logs and sophisticated control parameters for accounts used by users.
Put safeguards in place to stop an inactive account from logging in	Stronger protections for inactive accounts	Improve the security measures for inactive accounts even further to avoid security breaches and guarantee the complete safety of user information.

Video Demo

Link to the demo video of our team project:https://www.youtube.com/watch?v=kQMsokm1lQc

Appendix

addnewuser.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Welcome to Mini-Facebook</title>
    <link rel="stylesheet" type="text/css" href="assets/css/register_style.css">
<!--    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scr
```

```
    <script src="assets/js/jquery.min.js"></script>
    <script src="assets/js/register.js"></script>
    <style>
        .error
        {
            color:red;
            display:none;
        }
    </style>
</head>

<body>
<body>
<div class="wrapper">
    <div class="login_box">

            <div class="login_header">
<?php
require "database.php";
    $username = $_POST["username"];
    $password = $_POST["password"];
    $email = $_POST["email"];
    $name = $_POST["name"];
    $contact=$_POST['contact'];
    if(isset($username) and isset($password))
    {
        $username=sanitize_input($_POST['username']);
        $password=sanitize_input($_POST['password']);
        $email=sanitize_input($_POST['email']);
        $name = sanitize_input($_POST['name']);
        $contact = sanitize_input($_POST['contact']);

        #input length check
        if(strlen($username) < 1)
        {
            ?>
            <div class="title">
            Invalid Length for username : <?php echo htmlentities($username);?>!
             </div>
            <?php
        }
        else if(strlen($password) < 8)
        {
            ?>
            <div class="title">
            Invalid Length in password for Username: <?php echo htmlentities($username);?>!
```

```
     </div>
    <?php
}
else if(strlen($email)< 3 || strlen($name)<1)
{
    ?>
    <div class="title">
    Invalid Length in email for username : <?php echo htmlentities($username);?>!
     </div>
    <?php
}
else if(strlen($name)<1)
{
    ?>
    <div class="title">
    Invalid Length in name for username : <?php echo htmlentities($username);?>!
     </div>
    <?php
}
else if(strlen($contact)<10)
{
    ?>
    <div class="title">
    Invalid Length in contact for username : <?php echo htmlentities($username);?>!
     </div>
    <?php
}

//Reg exp check
else if (!preg_match("/^[a-zA-Z][a-zA-Z0-9_]*$/", $username))
{
    ?>
    <div class="title">
    Invalid pattern matching for username input <?php echo htmlentities($username);?
     </div>
    <?php
}

else if (!preg_match("/[0-9]{10}/", $contact))
{
    ?>
    <div class="title">
    Invalid pattern matching for contact input <?php echo htmlentities($username);?>
     </div>
    <?php
}
```

```
else if (!preg_match("/^[a-zA-Z]+$/", $name))
{
    ?>
    <div class="title">
    Invalid pattern matching for name input <?php echo htmlentities($username);?>!
     </div>
    <?php
}

else if(!preg_match('/^(?=.*[0-9])(?=.*[!@#$%^&*])[a-zA-Z0-9!@#$%^&*]{8,16}$/',$pass
{

    ?>
    <div class="title">
    Invalid pattern matching for password input <?php echo htmlentities($username);?
     </div>
    <?php

}

else if(!preg_match('/^[^\s@]+@[^\s@]+\.[^\s@]+$/', $email ))
{
    ?>
    <div class="title">
    Invalid pattern matching for email input <?php echo htmlentities($username);?>!
     </div>
    <?php
}

else if((addnewuser($name, $email,$contact,$username,$password)) and (insertaction($
{
    ?>
    <div class="title">
    Registration successfull!! <?php echo htmlentities($username);?>!
    <br><h5><a href="form.php" style="color: black;">Sign in</a></p></h5>
     </div>
    <?php
}
else
{
    ?>
    <div class="title">
    Registration FAILED!! <?php echo htmlentities($username);?>!
     </div>
```

```
            <?php

        }
    }
    else
    {
        ?>
            <div class="title">
            No username/password provided!
             </div>
        <?php
    }

    function sanitize_input($input)
    {
        $input=trim($input);
        $input=stripslashes($input);
        $input=htmlspecialchars($input);
        return $input;
    }


?>
</div>
</div>
</form>
</div>
   </body>
</html>
```

admin_session_auth.php

```
<?php
    session_set_cookie_params(15*60,"/","waph-team18.minifacebook.com",TRUE,TRUE);
    session_start();

    if(isset($_POST["username"]) and isset($_POST['password']))
    {

        if(checksuperuser($_POST["username"],$_POST["password"]) )
        {
            $_SESSION["authenticated"] = TRUE;
            $_SESSION["username"] = $_POST["username"];
            $_SESSION["browser"] = $_SERVER["HTTPS_USER_AGENT"];
        }
```

```
    }

    if(!$_SESSION["authenticated"] or $_SESSION["authenticated"]!= TRUE)
    {
        session_destroy();
        echo "<script>alert('You have not login. Please login first');</script>";
        header("Refresh:0;url=form.php");
        die();
    }

    if($_SESSION["browser"]!=$_SERVER["HTTPS_USER_AGENT"])
    {
        session_destroy();
        echo "<script>alert('Session hijacking is detected!');</script>";
        header("Refresh:0; url=form.php");
        die();
    }
?>
```

adminlogin.php

```
<!DOCTYPE html>
<html lang="en">

<head>
    <title>Welcome to Mini-Facebook</title>
    <link rel="stylesheet" type="text/css" href="assets/css/register_style.css">
<!--    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scr
    <script src="assets/js/jquery.min.js"></script>
    <script src="assets/js/register.js"></script>
</head>

<body>
<body>
<div class="wrapper">
        <div class="login_box">
<!--            header text-->
            <div class="login_header">
                <h1>Mini-Facebook</h1>
                Admin Login
            </div>
            <div id="first">
                <!--    Login form-->
                <form action="superuserprofile.php" method="post">
                    <input type="text" name="username" placeholder="Enter Username" required
```

```
                    <br>
                    <input type="password" name="password" placeholder="Enter Password">
                    <br>
                    <input type="submit" name="login_button" value="Login">
                    <br>
                </form>
            </div>
        </div>
    </div>
</body>
</html>
```

changepassword.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>User Details</title>
    <link rel="stylesheet" href="design.css">
</head>
<body>
<div class="headingsContainer">
            <h3>Passsword Status</h3>
        </div>
    <form>
<?php
    require "session_auth.php";
    require "database.php";
    $username = $_SESSION['username'];
    $oldpassword = $_REQUEST["oldpassword"];
    $newpassword = $_REQUEST["newpassword"];
    $token=$_POST["nocsrftoken"];
    if(!isset($token) or ($token!==$_SESSION["nocsrftoken"]))
    {
        echo "CSRF Attack is detected";
        die();
    }
    if(isset($username) and isset($oldpassword) and isset($newpassword))
    {

        if(changepassword($username,$oldpassword,$newpassword))
        {
            ?>
            <div class="mainContainer">
            <h2>Password has been changed!!! for <?php echo htmlentities($username);?>!</h2>
```

```
            <p class="register"> <a href="logout.php">Logout</a></p>
             </div>
            <?php
        }
        else
        {
            ?>
            <div class="mainContainer">
            <h2>Change password failed!!!! for <?php echo htmlentities($username);?>!</h2>

    <p class="register"> <a href="logout.php">Logout</a></p>
             </div>
            <?php
        }
    }
    else
    {
        ?>
            <div class="mainContainer">
            <?php echo "No username/password provided!";?>!
             </div>
            <?php

    }



    $token.die();
    ?>
    </form>
    </div>
   </body>
</html>
```

changepasswordform.php

```
<?php
    require "session_auth.php";
    $rand = bin2hex(openssl_random_pseudo_bytes(16));
    $_SESSION["nocsrftoken"]=$rand;
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Change Password Form</title>
    <style type="text/css">
```

```
        .error
        {
            color: red;
            display: none;
        }
    </style>
    <link rel="stylesheet" href="design.css">
</head>
<body>
    <h1>Change Password Form, WAPH</h1>
         <form action="changepassword.php" method="POST"  onsubmit="return validateForm();">
            <div class="mainContainer">
               <label for="username">Username : </label> <?php echo htmlentities($_SESSION['
               <input type="hidden" name="nocsrftoken" value="<?php echo $rand; ?>"/>
               <br><br>
               <label for="oldpassword">Existing Password : </label>
                <input type="password" name="oldpassword" required>
                <div class="error" id="oldpassword_error">Please enter password</div>
                <br><br>
               <label for="newpassword">New Password : </label>
                <input type="password" name="newpassword" placeholder="Enter new Password" p
                <div class="error" id="newpassword_error">Password requirements not met</div

                <br><br>

                <label for="password">Retype Password : </label>
                <input type="password" placeholder="Re-enter Password"  title="Password does
                <div class="error" id="confirmpassword_error">Password does not match</div>
                <br><br>


                <button type="submit" formnovalidate>Change Password</button>
               </div>
         </form>
        </div>
        <script>
         function validateForm() {
             var oldpassword = document.getElementsByName("oldpassword")[0].value.trim();
             var newpassword = document.getElementsByName("newpassword")[0].value.trim();
             var confirmPassword = document.getElementsByName("confirmpassword")[0].value.tr
             var isValid = true;

             // Reset error messages
             document.getElementById("oldpassword_error").style.display = "none";
             document.getElementById("newpassword_error").style.display = "none";
             document.getElementById("confirmpassword_error").style.display = "none";
```

```
            // Validate password
            if (oldpassword === "") {
                document.getElementById("oldpassword_error").style.display = "block";
                isValid = false;
            }

            // Validate password
            if (newpassword === "" || !isValidPassword(newpassword)) {
                document.getElementById("newpassword_error").style.display = "block";
                isValid = false;
            }

            // Validate confirm password
            if (confirmPassword === "" || confirmPassword !== newpassword) {
                document.getElementById("confirmpassword_error").style.display = "block";
                isValid = false;
            }

            return isValid;
        }

        function isValidPassword(newpassword) {
            // Password must have at least 8 characters with at least one uppercase letter
            var passwordRegex = new RegExp(/^(?=.*[0-9])(?=.*[!@#$%^&*])[a-zA-Z0-9!@#$%^&*
            return passwordRegex.test(newpassword);
        }

        </script>
    </body>
</html>
```

comment.php

```
<?php
    require "database.php";
    require "header.php";
    $username = $_SESSION['username'];
    $comment = $_REQUEST['comment'];
    $post_id = $_REQUEST['post_id'];
    if(isset($username) and isset($comment) and isset($post_id))
    {
        if(new_comment($username,$comment,$post_id))
        {
```

```
            ?>
            <h3>Comment added successfully</h3>
                <p class="register"> <a href="form.php">Back to home</a></p>
                <?php


        }



        else
        {
            session_destroy();
            echo "<script>alert('Invalid comment');window.location='form.php';</script>";
            die();
        }
    }


?>
```

database-account.sql

```
create database waph_team;
--  CREATE USER 'waph_team18'@'localhost' IDENTIFIED BY  'Pa$$w0rd';
 GRANT ALL ON waph_team.* TO 'waph_team18'@'localhost';
```

database-data.sql

```
drop table if exists users;
-- -- Create the users table
 create table users(
  user_id INT AUTO_INCREMENT PRIMARY KEY,
  name varchar(50),
  username varchar(50) UNIQUE,
  password varchar(100) NOT NULL,
  email varchar(100),
  contact varchar(10)
);

INSERT INTO users VALUES (1,'Team18','Team18',md5('Pa$$w0rd@123'),'team18@gmail.com','513123
INSERT INTO users VALUES (2,'Shruti','Thruti',md5('Shru@123'),'shru@gmail.com','5131234567')

drop table if exists superusers;
create table superusers(
    superuser_id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    name varchar(50) NOT NULL,
    username varchar(255) UNIQUE ,
    password varchar(50) NOT NULL,
    email varchar(50)
);

create table action_info(
    action_id INT AUTO_INCREMENT PRIMARY KEY,
    username varchar(255) UNIQUE ,
    action varchar(50) DEFAULT 'ACTIVE',
    FOREIGN KEY(username) REFERENCES users(username) ON DELETE CASCADE
);

drop table if exists messages;
-- Create the messages table
CREATE TABLE messages (
    message_ID INT PRIMARY KEY AUTO_INCREMENT,
    content TEXT,
    type VARCHAR(20),
    time_stamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    sender_username VARCHAR(50),
    FOREIGN KEY (sender_username) REFERENCES users(username) ON DELETE CASCADE
);

drop table if exists sends;
-- Create the sends relationship table
CREATE TABLE sends (
    message_ID INT,
    sender_username VARCHAR(50),
    PRIMARY KEY (message_ID),
    FOREIGN KEY (message_ID) REFERENCES messages(message_ID) ON DELETE CASCADE,
    FOREIGN KEY (sender_username) REFERENCES users(username) ON DELETE CASCADE
);

drop table if exists receives;
-- Create the receives relationship table
CREATE TABLE receives (
    message_ID INT,
    receiver_username VARCHAR(50),
    PRIMARY KEY (message_ID, receiver_username) ,
    FOREIGN KEY (message_ID) REFERENCES messages(message_ID) ON DELETE CASCADE,
    FOREIGN KEY (receiver_username) REFERENCES users(username) ON DELETE CASCADE
);

drop table if exists posts;
-- Create the posts table
```

```
CREATE TABLE posts(
    post_id INT PRIMARY KEY AUTO_INCREMENT,
    post_type VARCHAR(50),
    post_content TEXT,
    user_id INT,
    posttime varchar(50),
    time_stamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY(user_id) REFERENCES users(user_id) ON DELETE CASCADE
);

drop table if exists comments;
--CREATE TABLE comments
CREATE TABLE comments(
    comments_id INT PRIMARY KEY AUTO_INCREMENT,
    post_id INT,
    user_id INT,
    comment TEXT,
    time_stamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY(user_id) REFERENCES users(user_id) ON DELETE CASCADE,
    FOREIGN KEY(post_id) REFERENCES posts(post_id) ON DELETE CASCADE
);
```

database.php

```
<?php
$mysqli = new mysqli('localhost','waph_team18','Pa$$w0rd','waph_team');
if($mysqli->connect_errno)
{
    printf("Database connection failed: %s\n", $mysqli->connect_error);
    return FALSE;
}

function checklogin_mysql($username,$password)
    {
        global $mysqli;

        $sql = "SELECT * FROM users WHERE username=? AND password=md5(?)";
        $stmt=$mysqli->prepare($sql);
        $stmt->bind_param("ss",$username,$password);
        $stmt->execute();
        $result=$stmt->get_result();
        if($result -> num_rows==1)
            return TRUE;
        return FALSE;
    }
```

```
function superuser($username,$password)

{
    global $mysqli;

    $sql = "SELECT * FROM superusers WHERE username=? AND password=md5(?)";
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_param("ss",$username,$password);
    $stmt->execute();
    $result=$stmt->get_result();

    if($result -> num_rows==1)
        return TRUE;
    return FALSE;
}


function users()
{
    global $mysqli;

    $sql = "SELECT * FROM users";
    $stmt=$mysqli->prepare($sql);
    $stmt->execute();
    $result=$stmt->get_result();
    if($result -> num_rows>=0)
    {
        $posts_row=$result;
        return $posts_row;
    }
    else
    {
        return null;
    }

}

function user_data($user_id)
{
    global $mysqli;

    $sql = "SELECT username FROM users where user_id=?";
    $stmt=$mysqli->prepare($sql);

    $stmt->bind_param("s",$user_id);
```

```
    $stmt->execute();
    $result=$stmt->get_result();
    if($result -> num_rows>=0)
    {
        $posts_row=$result->fetch_assoc();
        return $posts_row;
    }
    else
    {
        return null;
    }

}



function posts_info()
{
    global $mysqli;
    $usersql = "SELECT * FROM users;";
    $userstmt=$mysqli->prepare($usersql);
    #$userstmt->bind_param("s",$username);
    $userstmt->execute();
    $result=$userstmt->get_result();
    if($row=$result->fetch_assoc())
    {
        #$user_id=$row['user_id'];
        $sql = "SELECT users.user_id, users.username, posts.post_id, posts.post_content
            FROM users
            JOIN posts on users.user_id=posts.user_id";
        $stmt=$mysqli->prepare($sql);
        //$stmt->bind_param("s",$user_id);
        $stmt->execute();
        $result=$stmt->get_result();

        if($result -> num_rows>=0)
        {
            $posts_row=$result;
            return $posts_row;
        }
        else
        {
            return null;
        }
    }
}
```

```php
function current_posts_info($username)
{
    global $mysqli;
    $usersql = "SELECT * FROM users WHERE username=?;";
    $userstmt=$mysqli->prepare($usersql);
    $userstmt->bind_param("s",$username);
    $userstmt->execute();
    $result=$userstmt->get_result();
    if($row=$result->fetch_assoc())
    {
        $user_id=$row['user_id'];
        $sql = "SELECT users.user_id, users.username, posts.post_id, posts.post_content
            FROM users
            JOIN posts on users.user_id=posts.user_id WHERE users.user_id=?";
        $stmt=$mysqli->prepare($sql);
        $stmt->bind_param("s",$user_id);
        $stmt->execute();
        $result=$stmt->get_result();

        if($result -> num_rows>=0)
        {
            $posts_row=$result;
            return $posts_row;
        }
        else
        {
            return null;
        }
    }
}


function new_comment($username,$comment,$post_id)
{
    global $mysqli;
    $usersql = "SELECT * FROM users WHERE username=?;";
    $userstmt=$mysqli->prepare($usersql);
    $userstmt->bind_param("s",$username);
    $userstmt->execute();
    $result=$userstmt->get_result();
    if($row=$result->fetch_assoc())
    {
        $user_id=$row['user_id'];
            $sql_comment="INSERT into comments(post_id,user_id,comment) VALUES(?,?,?);";
            $stmt_comment=$mysqli->prepare($sql_comment);
```

```
            $stmt_comment->bind_param("sss",$post_id,$user_id,$comment);
            if($stmt_comment->execute())
            {
                return TRUE;
            }
            else{
                return FALSE;

            }

    }
    else
    {
        return FALSE;
    }

}

function comment_info($post_id)
{
    global $mysqli;
        $sql_comment="SELECT * FROM comments WHERE post_id=?";
        $stmt_comment=$mysqli->prepare($sql_comment);
        $stmt_comment->bind_param("s",$post_id);
        $stmt_comment->execute();
        $result=$stmt_comment->get_result();
        if($result -> num_rows>=0)
        {
            $row=$result;
            return $row;
        }
        else
        {
            return null;
        }

}


function addnewuser($name, $email,$contact,$username,$password)
{
    global $mysqli;
    $sql = "INSERT INTO users(name, email,contact, username,password) VALUES (?,?,?,?, m
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_param("sssss",$name, $email,$contact, $username,$password);
    if($stmt->execute())
```

```
        return TRUE;
    return FALSE;
}

function insertaction($username)
{
    global $mysqli;
    $sql = "INSERT INTO action_info(username) VALUES (?); ";
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_param("s",$username);
    if($stmt->execute())
        return TRUE;
    return FALSE;
}

function disableuser($username)
{
    global $mysqli;
    $sql = "UPDATE action_info set action='DISABLE' where username=?; ";
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_param("s",$username);
    if($stmt->execute())
        return TRUE;
    return FALSE;
}

function activateuser($username)
{
    global $mysqli;
    $sql = "UPDATE action_info set action='ACTIVE' where username=?; ";
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_param("s",$username);
    if($stmt->execute())
        return TRUE;
    return FALSE;
}

function checkuserstatus($username)
{
    global $mysqli;
    $sql = "SELECT action FROM action_info where username=?; ";
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_param("s",$username);
    $stmt->execute();
    $stmt->bind_result($status);
    $stmt->fetch();
```

```
    if($status !== null )
    {
        if($status == 'ACTIVE')
        {
            return TRUE;
        }
        else
        {
            return FALSE;
        }
    }
    else
    {
        return FALSE;
    }
}

function action_data($username)
{
    global $mysqli;

    $sql = "SELECT * FROM action_info where username=?";
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_param("s",$username);
    $stmt->execute();
    $result=$stmt->get_result();
    if($result -> num_rows>=0)
    {
        $user_row=$result;
        return $user_row;
    }
    else
    {
        return null;
    }
}

function addnewpost($username,$post)
{
    global $mysqli;
    $usersql = "SELECT * FROM users WHERE username=?;";
    $userstmt=$mysqli->prepare($usersql);
    $userstmt->bind_param("s",$username);
    $userstmt->execute();
    $result=$userstmt->get_result();
    if($row=$result->fetch_assoc())
```

```
    {
        $user_id=$row['user_id'];
        $sql = "INSERT INTO posts(post_type,post_content,user_id,posttime) VALUES ('DEFA
        $stmt=$mysqli->prepare($sql);
        $stmt->bind_param("ss",$post, $user_id);
        if($stmt->execute())
            return TRUE;
        return FALSE;
    }
    else{


    return FALSE;
    }
}


function changepassword($username,$oldpassword,$newpassword)
{
    global $mysqli;
    $sql = "UPDATE users SET password=md5(?) WHERE username=? and password=md5(?);";
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_param("sss",$newpassword,$username,$oldpassword);
    if($stmt->execute())
        return TRUE;
    return FALSE;
}

function changeprofile($username,$name, $email, $contact)
{
    global $mysqli;
    $sql = "UPDATE users SET name=?, email=?, contact=? WHERE username=?;";
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_param("ssss",$name,$email,$contact,$username);
    if($stmt->execute())
        return TRUE;
    return FALSE;
}

function updatepost($username,$newpost,$post_id)
{
    global $mysqli;
    $usersql = "SELECT * FROM users WHERE username=?;";
    $userstmt=$mysqli->prepare($usersql);
    $userstmt->bind_param("s",$username);
    $userstmt->execute();
```

```
    $result=$userstmt->get_result();
    if($row=$result->fetch_assoc())
    {
        $user_id=$row['user_id'];
        $sql = "UPDATE posts SET post_content=? where user_id=? and post_id=?;";
        $stmt=$mysqli->prepare($sql);
        $stmt->bind_param("sss",$newpost, $user_id,$post_id);
        if($stmt->execute())
            return TRUE;
        return FALSE;
    }
}

function deletepost($username,$post_id)
{
    global $mysqli;
    $usersql = "SELECT * FROM users WHERE username=?;";
    $userstmt=$mysqli->prepare($usersql);
    $userstmt->bind_param("s",$username);
    $userstmt->execute();
    $result=$userstmt->get_result();
    if($row=$result->fetch_assoc())
    {
        $user_id=$row['user_id'];
        $sql = "DELETE from posts where user_id=? and post_id=?;";
        $stmt=$mysqli->prepare($sql);
        $stmt->bind_param("ss", $user_id,$post_id);
        if($stmt->execute())
            return TRUE;
        return FALSE;
    }
}

function userinfo($username,$password)
{
    global $mysqli;

    $sql = "SELECT * FROM users WHERE username=? AND password=md5(?)";
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_param("ss",$username,$password);
    $stmt->execute();
    $result=$stmt->get_result();
    if($result -> num_rows==1)
    {
        $row=$result->fetch_assoc();
        return $row;
```

```
        }
        else
        {
            return null;
        }

    }

?>
```

deletepost.php

```
<?php
    require "session_auth.php";
    require "database.php";
    require "header.php";
    $username = $_SESSION['username'];
    $post_id= $_REQUEST['post_id'];
    if(isset($username) and isset($post_id))
    {
        if(deletepost($username,$post_id))
        {
            $_SESSION["authenticated"] = TRUE;
            $_SESSION["username"] = $_POST["username"];
            $_SESSION["browser"] = $_SERVER["HTTPS_USER_AGENT"];
            ?>
            <h2> Post successfully deleted for user: <?php echo htmlentities($_SESSION["user
            <?php

                ?>
                <br>
                <p class="register"> <a href="logout.php">Logout</a></p>

                <?php
            }



        else
        {
            session_destroy();
            echo "<script>alert('Invalid post');window.location='form.php';</script>";
            die();
        }
    }
```

```
?>
```

design.css

```
body
{
  font-family:sans-serif;
  /* background: -webkit-linear-gradient(to right, #006653, white);   */
  /* background: linear-gradient(to right, #006653, white);  */
  background: #1877F2;

  color:black;
}


h1{
    text-align: center;
}


form{
    width:35rem;
    margin: auto;
    color:whitesmoke;
    background: white;
    /* -webkit-backdrop-filter: blur(16px) saturate(180%); */
    background-color: rgba(11, 15, 13, 0.582);
    border-radius: 12px;
    border: 1px solid rgba(255, 255, 255, 0.125);
    padding: 20px 25px;
}

input[type=text], input[type=password], input[type=email]{
    width: 100%;
    margin: 10px 0;
    border-radius: 5px;
    padding: 15px 18px;
    box-sizing: border-box;
  }

button {
    background-color: #030804;
    color: white;
```

```
    padding: 14px 20px;
    border-radius: 5px;
    margin: 7px 0;
    width: 100%;
    font-size: 18px;
  }

button:hover {
    opacity: 0.6;
    cursor: pointer;
}

.headingsContainer{
    text-align: center;
}

.error
{
  color:red;
  display: none;
}

.headingsContainer p{
    color: gray;
}
.mainContainer{
    padding: 16px;
}


.subcontainer{
    display: flex;
    flex-direction: row;
    align-items: center;
    justify-content: space-between;
}

.subcontainer a{
    font-size: 18px;
    margin-bottom: 12px;
}

.details
{
  margin:auto;
  font-size: 20px;
```

```
  color: white;
}

span.forgotpsd a {
    float: right;
    color: whitesmoke;
    padding-top: 16px;
  }

.forgotpsd a{
    color: rgb(241, 244, 247);
  }

.forgotpsd a:link{
    text-decoration: none;
  }

  .register{
    color: white;
    text-align: center;
  }

  .register a{
    color: rgb(233, 237, 243);
  }

  .register a:link{
    text-decoration: none;
    color: rgb(233, 237, 243);
  }

  /* Media queries for the responsiveness of the page */
  @media screen and (max-width: 600px) {
    form{
      width: 25rem;
    }
  }

  @media screen and (max-width: 400px) {
    form{
      width: 20rem;
    }
  }
```

disableuser.php

```
<?php
    require "admin_session_auth.php";
    require "database.php";
    require "header.php";
    $user = $_REQUEST['user'];
    $username = $_SESSION['superuser'];
    if(isset($username) )
    {
        if(disableuser($user))
        {
            $_SESSION["authenticated"] = TRUE;
            $_SESSION["superuser"] = $_POST["username"];
            $_SESSION["browser"] = $_SERVER["HTTPS_USER_AGENT"];
            ?>
            <h2> User successfully Disabled <?php echo htmlentities($username);?></h2>
            <?php

                ?>
                <br>
                <p class="register"> <a href="adminlogin.php">Logout</a></p>

                <?php
            }



        else
        {
            session_destroy();
            echo "<script>alert('Invalid post');window.location='form.php';</script>";
            die();
        }
    }



?>
```

editpost.php

```
<?php
```

```
    require "session_auth.php";
    require "database.php";
    require "header.php";
    $username = $_SESSION['username'];
    $newpost = $_REQUEST['newpost'];
    $post_id= $_REQUEST['post_id'];
    if(isset($username) and isset($newpost) and isset($post_id))
    {
        if(updatepost($username,$newpost,$post_id))
        {
            $_SESSION["authenticated"] = TRUE;
            $_SESSION["username"] = $_POST["username"];
            $_SESSION["browser"] = $_SERVER["HTTPS_USER_AGENT"];
            ?>
            <h2> Post successfully updated for user: <?php echo htmlentities($_SESSION["user
            <?php

                ?>
                <br>
                <p class="register"> <a href="logout.php">Logout</a></p>

                <?php
            }



        else
        {
            session_destroy();
            echo "<script>alert('Invalid post');window.location='form.php';</script>";
            die();
        }
    }



?>
```

editpostform.php

```
<?php
   require "session_auth.php";
   require "header.php";
   $rand = bin2hex(openssl_random_pseudo_bytes(16));
   $_SESSION["nocsrftoken"]=$rand;
?>
```

```
<div class="main_column column" id="main_column">
        <div class="posts_area">
    <h3>Edit Post</h3>
         <form action="editpost.php" method="POST">
         <div class="mainContainer">
         <input type="hidden" name="username" value="<?php echo htmlentities($_SESSION['user
               <input type="hidden" name="nocsrftoken" value="<?php echo $rand; ?>"/>
               <br><br>
               <?php
               require "database.php";

    session_set_cookie_params(20*60,"/","waph-team18.minifacebook.com",TRUE,TRUE);
    session_start();
    $username = $_SESSION['username'];
    if(isset($username))
    {
            $_SESSION["authenticated"] = TRUE;
            $_SESSION["browser"] = $_SERVER["HTTPS_USER_AGENT"];
            ?>
                <h3> POSTS </h3>
                <?php

                $posts_info = current_posts_info($username);
                if($posts_info)
                {
                    while($row = $posts_info->fetch_assoc())
                    {
                        ?>
                        <p><?php echo $row['post_content']?></p>
                        <form action="editpost.php" method="POST">
                            <textarea rows="5" cols="50" placeholder="Write something to upd
                            <input type="hidden" name="username" value="<?php echo $username
                            <input type="hidden" name="post_id" value="<?php echo $row['post
                            <button type="submit">Edit</button>
                        </form>
                        <form action="deletepost.php" method="POST">

                            <input type="hidden" name="username" value="<?php echo $username
                            <input type="hidden" name="post_id" value="<?php echo $row['post
                            <button type="submit">Delete</button>
                        </form>


                        <?php
                    }
```

```
                }
                ?>
                <br>
                <p class="register"> <a href="logout.php">Logout</a></p>

                <?php
            }

            else{
                echo "<h3>User not found.</h3>";
            }
```

```
    if(!$_SESSION["authenticated"] or $_SESSION["authenticated"]!= TRUE)
    {
        session_destroy();
        echo "<script>alert('You have not login. Please login first');</script>";
        header("Refresh:0;url=form.php");
        die();
    }

    if($_SESSION["browser"]!=$_SERVER["HTTPS_USER_AGENT"])
    {
        session_destroy();
        echo "<script>alert('Session hijacking is detected!');</script>";
        header("Refresh:0; url=form.php");
        die();
    }

            ?>
         </form>
        </div>
    </body>
</html>
```

editprofile.php

```
<?php
    require "database.php";
    require "header.php";

    require "session_auth.php";
    $username = $_SESSION['username'];
    $name = $_REQUEST["newname"];
```

```
$email = $_REQUEST["newemail"];
 $contact = $_REQUEST["newcontact"];
 $token=$_POST["nocsrftoken"];
 if(!isset($token) or ($token!==$_SESSION["nocsrftoken"]))
 {
     ?>
         <div class="mainContainer">
         <h2>CSRF Attack is detected!!! for <?php echo htmlentities($username);?>!</h2>
          </div>
         <?php
     die();
 }
 if(isset($username) and isset($name) and isset($email) and isset($contact))
 {

     if(changeprofile($username,$name,$email,$contact))
     {
         ?>
         <div class="mainContainer">
         Profile has been updated!!! for <?php echo htmlentities($username);?>!
          </div>
          <p class="register"> <a href="logout.php">Logout</a></p>
         <?php
     }
     else
     {
         ?>
         <div class="mainContainer">
         Profile Update failed!!!! for <?php echo htmlentities($username);?>!
          </div>
          <p class="register"> <a href="logout.php">Logout</a></p>
         <?php
     }
 }
 else
 {
     ?>
         <div class="mainContainer">
         <?php echo "No name/email provided!";?>!
          </div>
          <p class="register"> <a href="logout.php">Logout</a></p>
         <?php

 }
```

```
    $token.die();
    ?>

    </form>
    </div>
   </body>
</html>
```

editprofileform.php

```
<?php
   require "session_auth.php";
   $rand = bin2hex(openssl_random_pseudo_bytes(16));
   $_SESSION["nocsrftoken"]=$rand;

    require "database.php";
    require "header.php";
    ?>
    <h1>Edit Profile Page</h1>
         <form action="editprofile.php" method="POST">
         <div class="mainContainer">
         <label for="username">Username : </label> <?php echo htmlentities($_SESSION['userna
               <input type="hidden" name="nocsrftoken" value="<?php echo $rand; ?>"/>
               <br><br>
               <label for="newname">New Name : </label>
                <input type="text" name="newname" required>
             <br>
               <label for="newemail">New Email : </label>
                <input type="email" name="newemail" required>
                <br>
               <label for="newcontact">New Contact : </label>
                <input type="text" name="newcontact" required>
             <br><br>
             <button type="submit">Update Profile</button>
             </div>
         </form>
        </div>
    </body>
</html>
```

form.php

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <title>Welcome to Mini-Facebook</title>
    <link rel="stylesheet" type="text/css" href="assets/css/register_style.css">
<!--    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scr
    <script src="assets/js/jquery.min.js"></script>
    <script src="assets/js/register.js"></script>
</head>

<body>
<body>
<div class="wrapper">
        <div class="login_box">
<!--            header text-->
            <div class="login_header">
                <h1>Mini-Facebook</h1>
                Login
            </div>
            <div id="first">
                <!--    Login form-->
                <form action="index.php" method="post">
                    <input type="text" name="username" placeholder="Enter Username" required
                    <br>
                    <input type="password" name="password" placeholder="Enter Password">
                    <br>
                    <input type="submit" name="login_button" value="Login">
                    <br>
                    <a href="registrationform.php" id="signup" class="signup">NOt a Member?
                </form>
                <form action="adminlogin.php" method="post">
                    <input type="submit" name="admin_login_button" value="Admin Login">
                    <br>
                </form>
            </div>
        </div>
    </div>
</body>
</html>
```

header.php

```
<?php

include("session_auth.php");
```

```
?>

<html lang="en">
<head>
    <title>Welcome to Mini-Facebook</title>


    <!-- Javascript -->
<!--    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scr
<!--    <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></scr
    <script src="assets/js/bootstrap.min.js"></script>
    <script src="assets/js/jquery.min.js"></script>
    <script src="assets/js/bootbox.min.js"></script>
    <script src="assets/js/jquery.Jcrop.js"></script>
    <script src="assets/js/jcrop_bits.js"></script>
    <script src="assets/js/zychat.js"></script>


    <!-- CSS -->
    <link rel="stylesheet" type="text/css" href="assets/css/bootstrap.min.css">
    <link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/font-awesome/4.3.0/css/font-awesc
    <link rel="stylesheet" type="text/css" href="assets/css/style.css">
    <link rel="stylesheet" href="assets/css/jquery.Jcrop.css">
    <link rel="stylesheet" href="assets/css/responsive.css">

</head>
<body>

    <div class="top_bar">
        <div class="logo">
            <a href="index.php">Mini-Facebook</a>
        </div>



        <!-- Menu -->
        <div class="menu" onclick="showHideMenu()">
            <div class="line line-1"></div>
            <div class="line line-2"></div>
            <div class="line line-3"></div>
        </div>
        <!-- End of Menu -->


    </div>
```

```
    <div class="wrapper">
```

index.php

```
<?php
require "database.php";
require "session_auth.php";
?>

<?php

include("header.php");

            //session_set_cookie_params(15*60,"/","waph-team18.minifacebook.com",TRUE,TRUE);
            //session_start();
            if(isset($_POST["username"]) and isset($_POST['password']))
            {
                if(checklogin_mysql($_POST["username"],$_POST["password"]))
                {
                    if(checkuserstatus($_POST["username"]))
                    {
                    $_SESSION["authenticated"] = TRUE;
                    $_SESSION["username"] = $_POST["username"];
                    $_SESSION["browser"] = $_SERVER["HTTPS_USER_AGENT"];


?>
                        <div class="profile_left">

                    <?php
                    $user_info= userinfo($_POST["username"],$_POST["password"]);
                    if($user_info)
                    {
                        echo "Username: ".$user_info['username'];
                        echo "<br>Name: ".$user_info['name'];
                        echo "<br>Email Id: ".$user_info['email'];
                        echo "<br>Contact: ".$user_info['contact'];
                        ?>
                        <p class="register"><a href="changepasswordform.php">Change Passwor
                        <p class="register"><a href="editprofileform.php">Edit Profile</a></
                        <p class="register"><a href="newpostform.php">Add new post</a></p>
                        <p class="register"><a href="editpostform.php">Edit post</a></p>
```

```
    <p class="register"> <a href="logout.php">Logout</a></p>
</div>


        <div class="main_column column" id="main_column">
        <div class="posts_area">

            <p> POSTS from other users</p><?php
        $posts_info = posts_info();
        if($posts_info)
        {
            while($row = $posts_info->fetch_assoc())
            {
                ?>
                <hr>
                Post Added By : <?php echo $row['username']?>
                <div style="border-color: blue;"><p><?php echo $row['pos
                <p style="display:none"><?php echo $row['post_id']?></p>
                <?php
                ?>
                <div id="comments" class="comments">
                    <?php
                    $post_id = $row['post_id'];
                    $username = $row['username'];
                        $comments_info=comment_info($post_id);
                        if($comments_info)
                        {
                            while($comment = $comments_info->fetch_assoc
                            {
                                $user_data=user_data($comment['user_id']
                                ?>
                                <p><?php echo $user_data['username']?> :
                                <?php
                            }
                        }
                    ?>
                </div>
                <form method="POST" action="comment.php">
                    <input type="hidden" name="post_id" value="<?php ech
                    <textarea name="comment" rows="2" cols="40"></textar
                    <input type="submit" value="Add comment">
                </form>
            </hr>
                <?php
            }
```

```
                }
            ?>
            </div></div>
            <br>
            </div>
            <?php
        }

        else{
            echo "<h3>User not found</h3>";
        }


    }
    else
    {
        session_destroy();
        echo "<script>alert('User is disabled by admin');window.location='form.p
        die();
    }
}

    else
    {
        session_destroy();
        echo "<script>alert('Invalid username/password ');window.location='form.
        die();
    }
}
else
    {
        session_destroy();
        echo "<script>alert('Invalid usernamed ');window.location='form.php';</s
        die();
    }


/*if(!$_SESSION["authenticated"] or $_SESSION["authenticated"]!= TRUE)
{
    session_destroy();
    echo "<script>alert('You have not login. Please login first');</script>";
    header("Refresh:0;url=form.php");
    die();
}

if($_SESSION["browser"]!=$_SERVER["HTTPS_USER_AGENT"])
```

```
            {
                session_destroy();
                echo "<script>alert('Session hijacking is detected!');</script>";
                header("Refresh:0; url=form.php");
                die();
            }*/



        ?>
    </form>
</body>
</html>
```

logout.php

```
<?php
    session_start();
    require "header.php";
?>

<p> You are logged Out!</p>
<a href="form.php"> Login again </a>

<?php

    session_destroy();?>
```

newpost.php

```
<?php
    require "session_auth.php";
    require "database.php";
    require "header.php";
    $username = $_SESSION['username'];
    $post = $_REQUEST['post'];

    $token=$_POST["nocsrftoken"];

    if(!isset($token) or ($token!==$_SESSION["nocsrftoken"]))
    {
        ?>
            <div class="mainContainer">
            <h2>CSRF Attack is detected!!! for <?php echo htmlentities($username);?>!</h2>
             </div>
```

```
        <?php
    die();
}
if(isset($username) and isset($post))
{
    if(addnewpost($username,$post))
    {
        $_SESSION["authenticated"] = TRUE;
        $_SESSION["username"] = $_POST["username"];
        $_SESSION["browser"] = $_SERVER["HTTPS_USER_AGENT"];
        ?>
        <h2> Post successfully added for user: <?php echo htmlentities($_SESSION["userna
        <?php

            ?>
            <br>
            <p class="register"> <a href="logout.php">Logout</a></p>

            <?php
        }



    else
    {
        session_destroy();
        echo "<script>alert('Invalid post');window.location='form.php';</script>";
        die();
    }
}
if(!$_SESSION["authenticated"] or $_SESSION["authenticated"]!= TRUE)
{
    session_destroy();
    echo "<script>alert('You have not login. Please login first');</script>";
    header("Refresh:0;url=form.php");
    die();
}

if($_SESSION["browser"]!=$_SERVER["HTTPS_USER_AGENT"])
{
    session_destroy();
    echo "<script>alert('Session hijacking is detected!');</script>";
    header("Refresh:0; url=form.php");
    die();
}
```

```
?>
```

newpostform.php

```
<?php
    require "session_auth.php";
    include "header.php";
    $rand = bin2hex(openssl_random_pseudo_bytes(16));
    $_SESSION["nocsrftoken"]=$rand;
?>
    <style type="text/css">
        .error
        {
            color: red;
            display: none;
        }
    </style>
    <script>
             function validateForm() {

             var post = document.getElementsByName("post")[0].value.trim();
             var isValid = true;

             // Reset error messages
             document.getElementById("post_error").style.display = "none";

             // Validate name
             if (post === "") {
                 document.getElementById("post_error").style.display="block";
                 console.log("Inside post error");
                 isValid = false;
             }

            return isValid;
            console.log(isValid.value());
         }


         </script>

<div class="main_column column" id="main_column">
        <div class="posts_area">
    <form action="newpost.php" method="POST" onsubmit="return validateForm();">
        <!-- Headings for the form -->
        <div class="headingsContainer">
```

```
        </div>

        <!-- Main container for all inputs -->
        <div class="mainContainer">


         <input type="hidden" name="username" value="<?php echo htmlentities($_SESSION['user
               <input type="hidden" name="nocsrftoken" value="<?php echo $rand; ?>"/>
               <br>

                <textarea name="post" rows="5" cols="50" placeholder="Enter something here.
                <div class="error" id="post_error">Post is required</div>


            <!-- Submit button -->
            <input type="submit" formnovalidate value="POST">

        </div>

    </form>
        </div>
        </div>
</body>
</html>
```

registrationform.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Welcome to ZyChat</title>
    <link rel="stylesheet" type="text/css" href="assets/css/register_style.css">
<!--    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scr
    <script src="assets/js/jquery.min.js"></script>
    <script src="assets/js/register.js"></script>
    <style>
        .error
        {
            color:red;
            display:none;
        }
    </style>
</head>

<body>
<body>
```

```
<div class="wrapper">
        <div class="login_box">
            <div class="login_header">
                <h1>Mini-Facebook</h1>
                SIGN UP
            </div>
            <div id="first">
                <!--    Login form-->
         <form action="addnewuser.php" method="POST" onsubmit="return validateForm();">
            <input type="text" placeholder="Enter Name" name="name" pattern="^[a-zA-Z]+$" re
            <div class="error" id="name_error">Name is required</div>
            <br>

            <input type="email" placeholder="Enter Email" name="email" required>
            <div class="error" id="email_error">Valid email is required</div>
            <br>

            <input type="text" placeholder="Enter 10 digit Phone no." name="contact" patterr
            <div class="error" id="contact_error">Valid contact details are required</div>
            <br>

            <input type="text" placeholder="Enter Username" name="username" pattern="^[a-zA-
            <div class="error" id="username_error">Username is required</div>
            <br>

            <!-- Password -->
            <input type="password" placeholder="Enter Password" pattern="^(?=.*[0-9])(?=.*[!
            <div class="error" id="password_error">Password requirements not met</div>
            <br>

            <input type="password" placeholder="Re-enter Password"  title="Password does not
            <div class="error" id="confirmpassword_error">Passwords do not match</div>
            <br>
            <input type="submit" name="register_buttin" placeholder="Register" required form
            <br><br>

            Already a member? <a href="form.php" id="signin" class="signin">Login now</a>
         </form>
      </div>
      <script>
         function validateForm() {

             var name = document.getElementsByName("name")[0].value.trim();
             var email = document.getElementsByName("email")[0].value.trim();
             var username = document.getElementsByName("username")[0].value.trim();
             var contact = document.getElementsByName("contact")[0].value.trim();
```

```
var password = document.getElementsByName("password")[0].value.trim();
var confirmPassword = document.getElementsByName("confirmpassword")[0].value.tr
var isValid = true;

// Reset error messages
document.getElementById("name_error").style.display = "none";
document.getElementById("email_error").style.display = "none";
document.getElementById("contact_error").style.display = "none";
document.getElementById("username_error").style.display = "none";
document.getElementById("password_error").style.display = "none";
document.getElementById("confirmpassword_error").style.display = "none";

// Validate name
if (name === "" || !isValidName(name)) {
    document.getElementById("name_error").style.display="block";
    console.log("Inside name error");
    isValid = false;
}

// Validate email
if (email === "" || !isValidEmail(email)) {
  console.log("Inside email error");
    document.getElementById("email_error").style.display = "block";
    isValid = false;
}

// Validate contact
if (contact === "" || !isValidContact(contact)) {
  console.log("Inside contact error");
    document.getElementById("contact_error").style.display = "block";
    isValid = false;
}

// Validate username
if (username === "" || !isValidUsername(username)) {
    document.getElementById("username_error").style.display = "block";
    isValid = false;
}

// Validate password
if (password === "" || !isValidPassword(password)) {
    document.getElementById("password_error").style.display = "block";
    isValid = false;
}

// Validate confirm password
```

```
    if (confirmPassword === "" || confirmPassword !== password) {
        document.getElementById("confirmpassword_error").style.display = "block";
        isValid = false;
    }

    return isValid;
}

function isValidName(name) {
    // var emailRegex = new RegExp(/^\S+@\S+\.\S+$/);
   var nameRegex = new RegExp(/^[a-zA-Z]+$/);
    console.log(nameRegex);
    console.log(nameRegex.test(name));
    return nameRegex.test(name);
}

function isValidUsername(username) {
    // var emailRegex = new RegExp(/^\S+@\S+\.\S+$/);
   var usernameRegex = new RegExp(/^[a-zA-Z][a-zA-Z0-9_]*$/);
    console.log(usernameRegex);
    console.log(usernameRegex.test(username));
    return usernameRegex.test(username);
}

function isValidEmail(email) {
    // var emailRegex = new RegExp(/^\S+@\S+\.\S+$/);
   var emailRegex = new RegExp(/^[^\s@]+@[^\s@]+\.[^\s@]+$/);
    console.log(emailRegex);
    console.log(emailRegex.test(email));
    return emailRegex.test(email);
}

function isValidContact(contact) {
   var contactRegex = new RegExp(/^\+?[0-9]{10}$/);
    console.log(contactRegex);
    console.log(contactRegex.test(contact));
    return contactRegex.test(contact);
}

function isValidPassword(password) {
    // Password must have at least 8 characters with at least one uppercase letter
    var passwordRegex = new RegExp(/^(?=.*[0-9])(?=.*[!@#$%^&*])[a-zA-Z0-9!@#$%^&*
    return passwordRegex.test(password);
}

</script>
```

```
   </body>
</html>
```

session_auth.php

```
<?php
    session_set_cookie_params(15*60,"/","waph-team18.minifacebook.com",TRUE,TRUE);
    session_start();

    if(isset($_POST["username"]) and isset($_POST['password']))
    {

            $_SESSION["authenticated"] = TRUE;
            $_SESSION["username"] = $_POST["username"];
            $_SESSION["browser"] = $_SERVER["HTTPS_USER_AGENT"];



    }

    if(!$_SESSION["authenticated"] or $_SESSION["authenticated"]!= TRUE)
    {
        session_destroy();
        echo "<script>alert('You have not login. Please login first');</script>";
        header("Refresh:0;url=form.php");
        die();
    }

    if($_SESSION["browser"]!=$_SERVER["HTTPS_USER_AGENT"])
    {
        session_destroy();
        echo "<script>alert('Session hijacking is detected!');</script>";
        header("Refresh:0; url=form.php");
        die();
    }
?>
```

superuserprofile.php

```
<?php
require "database.php";
require "admin_session_auth.php";
?>

<style>
```

```
    table, th, td
    {
        padding: 10px;
        border: black;
        text-align: auto;
        margin: auto;
    }
    th
    {
        text-align: center;

    }


</style>

<?php

include("header.php");

            if(isset($_POST["username"]) and isset($_POST['password']))
            {
                if(superuser($_POST["username"],$_POST["password"]))
                {
                    $_SESSION["authenticated"] = TRUE;
                    $_SESSION["superuser"] = $_POST["username"];
                    $_SESSION["browser"] = $_SERVER["HTTPS_USER_AGENT"];


?>
                        <div class="profile_left">
                                    <h3>Welcome super user</h3>

                        </div>
                        <div class="main_column column" id="main_column">
                            <div class="posts_area">
                        <div class="table_design">
                            <table>
                                <tr>
                                    <th>User Id</th>
                                    <th>Name</th>
                                    <th>Username</th>
                                    <th>Email</th>
                                    <th>Contact</th>
                                    <th>Action</th>
                                </tr>
```

```
<?php
$user = users();
 while($row=$user->fetch_assoc())
 {
    ?>
    <tr>

    <td><?php echo $row['user_id']?></td>
    <td><?php echo $row['name']?></td>
    <td><?php echo $row['username']?></td>
    <td><?php echo $row['email']?></td>
    <td><?php echo $row['contact']?></td>
    <td><?php

    $action_info = action_data($row['username']);
    if($action_info)
    {

        while($actions= $action_info->fetch_assoc())
        {
            ?>
            <p><?php
            if($actions['action']=='ACTIVE')
            {

                echo "ACTIVE" ?><form action="disableuser.ph
                    <input type="submit" name="admin_login_b
                    <input type="hidden" name="user" value='
                    <input type="hidden" name="username" val
                </form><?php
            }
            elseif($actions['action']=='DISABLE')
            {
                echo "DISABLED" ?><form action="activateuser
                    <input type="submit" name="admin_login_b
                    <input type="hidden" name="user" value='
                    <input type="hidden" name="username" val
                </form>
            </p>
            <?php
                    }
        }
    }
    ?></td>

</tr>
```

```
                    <?php
                    }
                    ?>
                </table>
            </div>
        </div>
    </div>
            <br>
            </div>
            <?php
        }

        else{
            session_destroy();
            echo "<script>alert('You are not a super user');window.location='for
            die();
        }


    }


    else
    {
        session_destroy();
        echo "<script>alert('Invalid username');window.location='form.php';</scr
        die();
    }

if(!$_SESSION["authenticated"] or $_SESSION["authenticated"]!= TRUE)
{
    session_destroy();
    echo "<script>alert('You have not login. Please login first');</script>";
    header("Refresh:0;url=form.php");
    die();
}

if($_SESSION["browser"]!=$_SERVER["HTTPS_USER_AGENT"])
{
    session_destroy();
    echo "<script>alert('Session hijacking is detected!');</script>";
    header("Refresh:0; url=form.php");
    die();
}
```

```
        ?>
    </form>
</body>
</html>
```

activateuser.php

```
<?php
    require "database.php";
    require "header.php";
    $user = $_REQUEST['user'];
    $username = $_SESSION['superuser'];
    if(isset($username) )
    {
        if(activateuser($user))
        {
            $_SESSION["authenticated"] = TRUE;
            $_SESSION["superuser"] = $_POST["username"];
            $_SESSION["browser"] = $_SERVER["HTTPS_USER_AGENT"];
            ?>
            <h2> User successfully Activated <?php echo htmlentities($_SESSION["username"]);
            <?php

                ?>
                <br>
                <p class="register"> <a href="adminlogin.php">Logout</a></p>

                <?php
            }



        else
        {
            session_destroy();
            echo "<script>alert('Invalid post');window.location='form.php';</script>";
            die();
        }
    }



?>
```