**RAJALAKSHMI ENGINEERING COLLEGE**



**A MINI PROJECT REPORT**

**ON**

**"PRIVATEVERSE – PRIVATE NOTES POSTING APPLICATION"**

**FOR THE COURSE INTERNET PROGRAMMING**

## SUBMITTED BY:

**VIDHIYA S B (210701306)**

## COMPUTER SCIENCE ENGINEERING : 2021-2025

**BONAFIDE CERTIFICATE**

Certified that this project “**PRIVATEVERSE – PRIVATE NOTES POSTING APPLICATION**” is the bonafide work of “**VIDHIYA S B(210701306)**” who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form any part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

### SIGNATURE

**Mr.S.SURESH KUMAR M.E, ASSOCIATE PROFESSOR,**

### Dept. of Computer Science and Engg, Rajalakshmi Engineering College, Chennai

This mini project report is submitted for the viva voce examination to be held on

**INTERNAL EXAMINER EXTERNAL EXAMINER**

# ACKNOWLEDGEMENT

I express my sincere thanks to my beloved and honourable chairman **MR.S.MEGANATHAN** and the honorable chairperson **DR.M.THANGAM MEGANATHAN** for their timely support and encouragement.

I am greatly indebted to my respected and honourable Principal **Dr.**

**S.N. MURUGESAN** for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by my head of the department **Dr. P. KUMAR,** and my Academic Head **Dr.N.DURAIMURUGAN,** for being ever supporting force during my project work.

I also extend my sincere and hearty thanks to my internal guide **Mr.S.SURESH KUMAR** for his valuable guidance and motivation during the completion of this project.

My sincere thanks to my family members, friends and other staff members of Computer Science and Engineering.

**VIDHIYA S B**

**210701306**

# ABSTRACT

The "Privateverse" project is a web application designed to provide users with a secure and personalized space for social interaction and self-expression. The platform incorporates user authentication, allowing individuals to register, log in, and securely access their personal dashboard. The dashboard serves as a central hub for users, displaying personalized information, such as their username and email. Beyond basic user functionalities, Privateverse introduces a social component by enabling users to create and share posts within the platform. These posts are stored securely in a backend database, ensuring the privacy and integrity of user-generated content. The application offers a seamless and visually appealing user interface, encouraging individuals to connect with themselves and share their thoughts in a safe online environment. Overall, Privateverse aims to provide a user-friendly and secure space for individuals to express themselves and engage in a community that values privacy and personal connection.

The Privateverse project leverages a comprehensive technology stack to ensure a robust and secure user experience. The frontend is built using HTML, CSS, and JavaScript, providing an interactive and visually appealing interface for users to navigate seamlessly. jQuery, a fast and lightweight JavaScript library, enhances the user experience by simplifying DOM manipulation and event handling. The incorporation of these technologies enables a responsive and dynamic frontend, ensuring a smooth interaction for users as they register, log in, and engage with the platform.

On the server side, PHP is employed for backend development, facilitating database connectivity and server-side logic. MySQL is used as the database management system to store and manage user data, posts, and authentication information securely. The combination of PHP and MySQL ensures efficient data handling and retrieval, contributing to the overall reliability and performance of the Privateverse application. Additionally, the project adopts a user authentication mechanism to safeguard user information and restrict unauthorized access, emphasizing the importance of privacy and security in the online environment. By integrating these technologies, Privateverse delivers a well-rounded solution that prioritizes both functionality and user.

# Table of Contents

**CHAPTER 1** – Introduction

**CHAPTER 2** – Project Overview and Scope

* 1. Project Overview
  2. Scope

**CHAPTER 3** - Functional and Specific requirements

* 1. External Interface
     1. Hardware Interface
     2. Software Interface
  2. Non Functional Requirements

**CHAPTER 4** - Analysis & Design

* 1. Use Cases
     1. Context Level Diagram
     2. Login Activity Diagram

**CHAPTER 5 -** Sample code and Output

**CHAPTER 6 –** Conclusion

**CHAPTER 7 –** References

### CHAPTER 1 INTRODUCTION

The Privateverse project introduces a user-centric web application designed to provide individuals with a secure, private, and visually engaging online space. In the era of digital interconnectedness, this platform emphasizes the importance of personal connection and self-expression, offering users a controlled environment to share thoughts, create posts, and interact with a community of like-minded individuals. The project's foundation lies in robust web development technologies, employing HTML, CSS, and JavaScript for a dynamic and responsive frontend. User authentication, powered by PHP and MySQL on the backend, ensures data security, allowing individuals to register, log in, and access their personalized dashboards with confidence. Privateverse aims to foster a sense of community where users can connect to themselves and others while prioritizing privacy and a visually appealing user experience.

In a digital landscape where social interaction often comes at the cost of privacy, Privateverse stands as a beacon for users seeking a more intimate and secure online experience. With its seamless integration of frontend technologies such as HTML, CSS, and JavaScript, the platform provides an intuitive and aesthetically pleasing interface. Beyond the surface, the robust backend, powered by PHP and MySQL, ensures the protection of user data, creating a trustworthy foundation for engagement. Privateverse is not just a web application; it's a digital sanctuary where individuals can freely express themselves, share their stories, and build connections within a community that values the essence of personal connection. In a world inundated with information, Privateverse offers a refreshing alternative that prioritizes the individual's right to privacy and self-expression.

Top of Form

### CHAPTER 2

* 1. **Project Overview:**

ThePrivateverse is a comprehensive web application that aims to create a secure and engaging online space for users to connect with themselves and others. Built on a technology stack including HTML, CSS, JavaScript, PHP, and MySQL, the platform ensures a visually appealing and responsive interface while prioritizing the security of user data. Users can seamlessly register, log in, and access their personalized dashboards, where they can create and share posts within a community that values privacy and personal connection. With an emphasis on both functionality and user protection, Privateverse offers a digital sanctuary for individuals seeking a more intimate and secure online experience in the age of ubiquitous social connectivity.

* 1. **Scope:**

The scope of the Privateverse project extends to providing a versatile and user-friendly online platform that not only prioritizes user privacy but also fosters a sense of community and self-expression. With the integration of key technologies, the project aims to accommodate a growing user base seeking a secure space for personal connection and content sharing. The scalability of the platform allows for potential future enhancements, such as additional features, improved user interactions, and expanded community engagement. The scope also encompasses ongoing updates to maintain the security and relevance of the platform, ensuring that Privateverse continues to meet the evolving needs of its users while staying at the forefront of web development. .

## CHAPTER 3

**FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS**

.

Functional requirements for the Privateverse project encompass the essential features and capabilities that contribute to the platform's intended functionality. These requirements include:

1. **User Authentication:**
   * Users should be able to register for an account with a unique username, valid email, and secure password.
   * A login mechanism should allow users to securely access their accounts.
2. **User Dashboard:**
   * Upon login, users should be redirected to a personalized dashboard displaying their username and email.
   * The dashboard should include a navigation system for seamless interaction with different sections of the application.
3. **Post Creation and Sharing:**
   * Users should have the ability to create and share posts.
   * Each post should include a content field for tracking post creation.
4. **Post Display:**
   * The platform should display a user's posts on their dashboard.
   * Posts should be organized chronologically, with the latest posts appearing first

.

1. **Data Storage:**
   * User data, including account information and posts, should be securely stored in a backend database (e.g., MySQL).
   * Database connectivity should be established for efficient data retrieval and management.
2. **User Data Retrieval:**
   * Users should be able to retrieve and view their account information.
   * The application should support the retrieval of a user's posts for display on their dashboard.
3. **Security Measures:**
   * Passwords should be securely hashed before storage.
   * User authentication mechanisms should prevent unauthorized access to user accounts.
4. **Error Handling:**
   * The application should provide meaningful error messages to users in case of unsuccessful login attempts, registration issues, or other errors.
   * Input validation should be implemented to ensure data integrity and security.
5. **Responsive Design:**
   * The user interface should be responsive, providing a consistent and user-friendly experience across various devices and screen sizes.
6. **Logout Functionality:**
   * Users should be able to log out securely, terminating their session.

## External Interface Requirements

* + 1. **External Hardware Interfaces:**

In Privateverse project, external hardware interfaces play a crucial role in facilitating communication between the software system and various hardware devices.

1. **Web Server:**
   * Privateverse requires a web server for hosting the application. This can be achieved using popular web server software such as Apache, Nginx, or Microsoft Internet Information Services (IIS).
2. **Database Server:**
   * The platform relies on a database server to store and manage user data. MySQL, PostgreSQL, or other relational database management systems (RDBMS) can serve this purpose.
3. **Hosting Environment:**
   * Privateverse should be hosted on a server that meets standard web hosting specifications. This includes sufficient processing power, memory, and storage capacity to handle user requests and data storage efficiently.
4. **Internet Connection:**
   * Users accessing Privateverse need an internet connection to interact with the platform. The platform itself requires a reliable internet connection for data exchange between the server and users.
5. **SSL Certificate:**
   * To ensure secure data transmission between users and the server, Privateverse should implement HTTPS using an SSL (Secure Sockets Layer) certificate. This is crucial for protecting sensitive user information during login and data transfer.
6. **Client Devices:**
   * Users can access Privateverse from various client devices, including desktop computers, laptops, tablets, and smartphones. The platform should be designed to provide a responsive and user-friendly experience.

## Software Interface

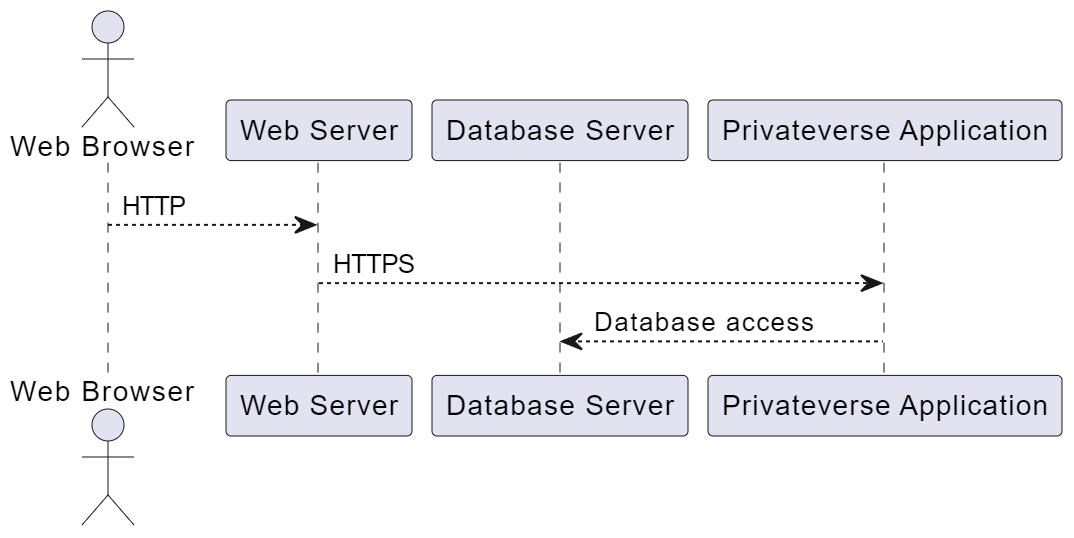
## The Software interfaces include :

1. **Web Browser:**
   * Privateverse is designed to be accessed through standard web browsers such as Google Chrome, Mozilla Firefox, Safari, or Microsoft Edge. The platform should be compatible with the latest versions of these browsers to ensure optimal performance.
2. **Web Server Software:**
   * The web server hosting Privateverse should run web server software like Apache, Nginx, or Microsoft IIS. The chosen server software should support PHP for server-side scripting and serve static files efficiently.
3. **Database Management System (DBMS):**
   * Privateverse relies on a relational database management system (DBMS) for data storage. MySQL or PostgreSQL can be used to manage user accounts, posts, and other related data.
4. **Server-Side Scripting Language:**
   * PHP is the primary server-side scripting language used in Privateverse. The web server should be configured to process PHP scripts, allowing dynamic content generation and interaction with the database.
5. **JavaScript Framework (jQuery):**
   * The frontend interaction and asynchronous communication with the server are facilitated by JavaScript. The project uses the jQuery library to simplify JavaScript development, enabling smooth AJAX requests and DOM manipulation.
6. **AJAX (Asynchronous JavaScript and XML):**
   * Asynchronous communication between the client and server is achieved using AJAX. This technology allows the platform to dynamically update content without requiring a full page reload, enhancing the user experience.
7. **JSON (JavaScript Object Notation):**
   * Privateverse uses JSON for data interchange between the server and client. JSON is a lightweight and human-readable format that facilitates efficient communication and parsing of data.
8. **Secure Sockets Layer (SSL):**
   * Privateverse implements SSL to secure data transmission between users and the server. The SSL interface ensures encrypted communication, protecting sensitive user information during login and data transfer.

## CHAPTER 4 ANALYSIS AND DESIGN

### Use Cases

* + 1. **Context Diagram**

****

* + 1. **Activity Diagram**

****

## Related Technologies:

### Frontend

1. **HTML5 and CSS3 :** Standard web technologies for structuring content and styling the user interface, ensuring a modern and visually appealing design. Cascading Style Sheets (CSS) is a style sheet language used to control the presentation and layout of HTML documents.
2. **JavaScript:** JavaScript is an extensive technology for tasks like updating specific portions of page content without modifying the whole page or validating submission forms. JavaScript is primarily used as a client-side scripting language, meaning it runs in the user's web browser rather than on the server. It enables dynamic content updates, interactivity, and enhanced user interfaces on websites.

### Backend

**1.PHP:** PHP is a server side scripting language that is classified as a backend language. MySQL is one of the most popular relational database management systems, and it is often used in combination with PHP. PHP has specific functions for interacting with MySQL databases, allowing developers to perform tasks like connecting to the database, querying data, and updating records.

### Authentication and Security

* **Username and Password**:Users should have unique usernames and strong,

encrypted passwords.

**CHAPTER 5 CODE AND OUTPUT**

### index.html

<!-- index.html -->

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>User Authentication App</title>

    <link rel="stylesheet" href="style.css">

    <script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>

    <script src="script.js"></script>

</head>

<body>

    <!-- Add this inside the <body> tag -->

<div class="header">

    <h1 class="title">Privateverse</h1>

    <p class="caption">By Vidhiya S B</p>

</div>

    <div class="container">

        <div id="registrationContainer">

            <h2>Register</h2>

            <form id="registrationForm" action="register.php" method="post">

                <label for="username">Username:</label>

                <input type="text" id="username" name="username" required>

                <label for="email">Email:</label>

                <input type="email" id="email" name="email" required>

                <label for="password">Password:</label>

                <input type="password" id="password" name="password" required>

                <input type="submit" value="Register">

            </form>

        </div>

        <div id="loginContainer">

            <h2>Login</h2>

            <form id="loginForm" action="login.php" method="post">

                <label for="loginEmail">Email:</label>

                <input type="email" id="loginEmail" name="email" required>

                <label for="loginPassword">Password:</label>

                <input type="password" id="loginPassword" name="password" required>

                <input type="submit" value="Login">

            </form>

        </div>

        <div id="dashboardContainer" style="display:none;">

            <h2>Welcome, <span id="welcomeUsername"></span>!</h2>

            <p>Email: <span id="userEmail"></span></p>

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

            <div id="postFormContainer">

                <h3>Create a Post</h3>

                <form id="postForm">

                    <textarea id="postContent" placeholder="Write your post..." required></textarea>

                    <input type="submit" value="Post">

                </form>

            </div>

            <div id="postsContainer">

                <h3>Your Posts</h3>

                <ul id="userPosts"></ul>

            </div>

        </div>

    </div>

</body>

</html>

**style.css**

body {

    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

    background-image: url('bg.png'); /\* Replace 'background.jpg' with your background image file path \*/

    background-size: cover;

    color: #006ea1;

}

.container {

    width: 500px;

    margin: 50px auto;

    background-color: rgba(255, 255, 255, 0.9); /\* Semi-transparent white background \*/

    padding: 20px;

    border-radius: 10px;

    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

.logo {

    text-align: center;

    margin-bottom: 20px;

}

.logo img {

    width: 5%; /\* Adjust the width of the logo \*/

    height: auto;

}

.title {

    text-align: center;

    font-size: 24px;

    font-weight: bold;

    margin-bottom: 10px;

}

.caption {

    text-align: center;

    font-size: 14px;

    color: #888;

    margin-bottom: 20px;

}

form {

    display: flex;

    flex-direction: column;

    margin-bottom: 10px;

}

label {

    margin-bottom: 8px;

    color: #555;

}

input, textarea {

    margin-bottom: 16px;

    padding: 10px;

    border: 1px solid #ccc;

    border-radius: 5px;

}

#postFormContainer textarea {

    width: 500px;

    height: 100px;

    margin-bottom: 30px;

}

#userPosts {

    list-style-type: none;

    padding: 0;

}

.post {

    background-color: #fff;

    border: 1px solid #ddd;

    border-radius: 5px;

    margin-bottom: 10px;

    padding: 15px;

    box-shadow: 0 0 5px rgba(0, 0, 0, 0.1);

}

/\* Add or modify these styles \*/

.header {

    text-align: center;

    margin-bottom: 30px;

}

.caption {

    font-style: bold;

    font-size: 16px;

    color: #ffffff;

}

**script.js**

$(document).ready(function () {

    $('#registrationForm').submit(function (e) {

        if ($('#username').val() === '' || $('#email').val() === '' || $('#password').val() === '') {

            alert('Please fill in all fields.');

            e.preventDefault();

        }

    });

    $('#loginForm').submit(function (e) {

        e.preventDefault();

        $.ajax({

            type: 'POST',

            url: 'login.php',

            data: $(this).serialize(),

            success: function (response) {

                if (response === 'success') {

                    $('#registrationContainer, #loginContainer').hide();

                    $('#dashboardContainer').show();

                    loadUserData();

                    loadUserPosts();

                } else {

                    alert('Invalid email or password.');

                }

            }

        });

    });

// Function to handle post submission

    $('#postForm').submit(function (e) {

        e.preventDefault();

        // Get post content

        var postContent = $('#postContent').val();

        // Perform AJAX request

        $.ajax({

            type: 'POST',

            url: 'create\_post.php',

            data: { postContent: postContent },

            success: function (response) {

                if (response === 'success') {

                    $('#postContent').val('');

                    loadUserPosts();

                } else {

                    alert('Failed to create post.');

                }

            }

        });

    });

    function loadUserData() {

        $.ajax({

            type: 'GET',

            url: 'get\_user\_data.php',

            success: function (data) {

                var userData = JSON.parse(data);

                $('#welcomeUsername').text(userData.username);

                $('#userEmail').text(userData.email);

            }

        });

    }

function loadUserPosts() {

        $.ajax({

            type: 'GET',

            url: 'get\_user\_posts.php',

            success: function (data) {

                var posts = JSON.parse(data);

                displayUserPosts(posts);

            }

        });

    }

    function displayUserPosts(posts) {

        var postsContainer = $('#userPosts');

        postsContainer.empty();

        posts.forEach(function (post) {

            var postItem = $('<li class="post"></li>').text(post.content);

            postsContainer.append(postItem);

        });

    }

});

**create\_post.php**

<?php

session\_start();

// Check if the user is authenticated

if (!isset($\_SESSION['user\_id'])) {

    header("HTTP/1.1 401 Unauthorized");

    exit();

}

// Ensure the request is a POST request

if ($\_SERVER['REQUEST\_METHOD'] !== 'POST') {

    header("HTTP/1.1 405 Method Not Allowed");

    exit();

}

// Validate and sanitize input

$user\_id = $\_SESSION['user\_id'];

$content = isset($\_POST['postContent']) ? trim($\_POST['postContent']) : '';

if (empty($content)) {

    header("HTTP/1.1 400 Bad Request");

    echo "Error: Post content cannot be empty.";

    exit();

}

// Database configuration

$servername = "localhost";

$username = "root";

$password = "";

$dbname = "user\_authentication";

try {

    // Create a new PDO instance (safer and more feature-rich than mysqli)

    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);

    $conn->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

    // Prepare the SQL statement to prevent SQL injection

    $sql = "INSERT INTO posts (user\_id, content) VALUES (:user\_id, :content)";

    $stmt = $conn->prepare($sql);

    $stmt->bindParam(':user\_id', $user\_id, PDO::PARAM\_INT);

    $stmt->bindParam(':content', $content, PDO::PARAM\_STR);

    $stmt->execute();

    echo "success"; // Add quotes for consistency with script.js

} catch (PDOException $e) {

    header("HTTP/1.1 500 Internal Server Error");

    echo "Error: " . $e->getMessage();

    error\_log("PDOException in create\_post.php: " . $e->getMessage()); // Log the error

} finally {

    $conn = null; // Close the database connection

}

?>

**register.php**

<?php

$servername = "localhost";

$username = "root";

$password = "";

$dbname = "user\_authentication";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect\_error) {

    die("Connection failed: " . $conn->connect\_error);

}

$username = $\_POST['username'];

$email = $\_POST['email'];

$password = password\_hash($\_POST['password'], PASSWORD\_DEFAULT);

$sql = "INSERT INTO users (username, email, password) VALUES ('$username', '$email', '$password')";

if ($conn->query($sql) === TRUE) {

    echo "success Now go back and login";

} else {

    echo "Error: " . $sql . "<br>" . $conn->error;

}

$conn->close();

?>

**login.php**

<?php

session\_start();

$servername = "localhost";

$username = "root";

$password = "";

$dbname = "user\_authentication";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect\_error) {

    die("Connection failed: " . $conn->connect\_error);

}

$email = $\_POST['email'];

$password = $\_POST['password'];

$sql = "SELECT \* FROM users WHERE email='$email'";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

    $row = $result->fetch\_assoc();

    if (password\_verify($password, $row['password'])) {

        $\_SESSION['user\_id'] = $row['id'];

        echo "success";

    } else {

        echo "Invalid password";

    }

} else {

    echo "User not found";

}

$conn->close();

?>

**get\_user\_data.php**

<?php

session\_start();

if (!isset($\_SESSION['user\_id'])) {

    header("HTTP/1.1 401 Unauthorized");

    exit();

}

$servername = "localhost";

$username = "root";

$password = "";

$dbname = "user\_authentication";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect\_error) {

    die("Connection failed: " . $conn->connect\_error);

}

$user\_id = $\_SESSION['user\_id'];

$sql = "SELECT \* FROM users WHERE id='$user\_id'";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

    $row = $result->fetch\_assoc();

    echo json\_encode(['username' => $row['username'], 'email' => $row['email']]);

} else {

    echo "User not found";

}

$conn->close();

?>

**get\_user\_posts.php**

<?php

session\_start();

if (!isset($\_SESSION['user\_id'])) {

    header("HTTP/1.1 401 Unauthorized");

    exit();

}

$servername = "localhost";

$username = "root";

$password = "";

$dbname = "user\_authentication";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect\_error) {

    die("Connection failed: " . $conn->connect\_error);

}

$user\_id = $\_SESSION['user\_id'];

$sql = "SELECT \* FROM posts WHERE user\_id='$user\_id' ORDER BY created\_at DESC";

$result = $conn->query($sql);

$posts = array();

if ($result->num\_rows > 0) {

    while ($row = $result->fetch\_assoc()) {

        $posts[] = array('content' => $row['content']);

    }

}

echo json\_encode($posts);

$conn->close();

?>

**logout.php**

<?php

session\_start();

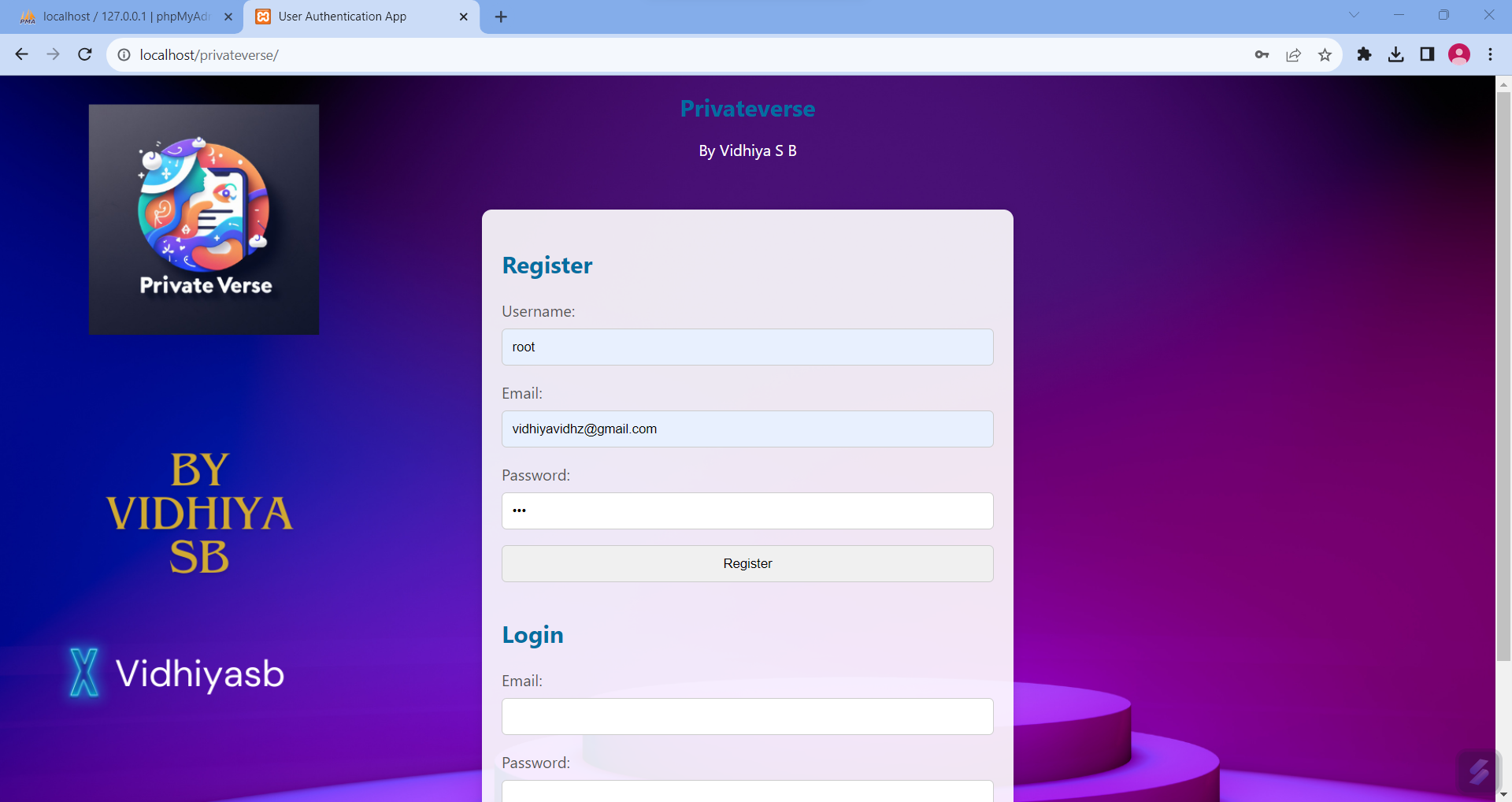
session\_destroy();

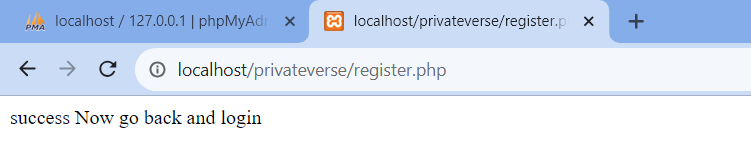
header("Location: index.html");

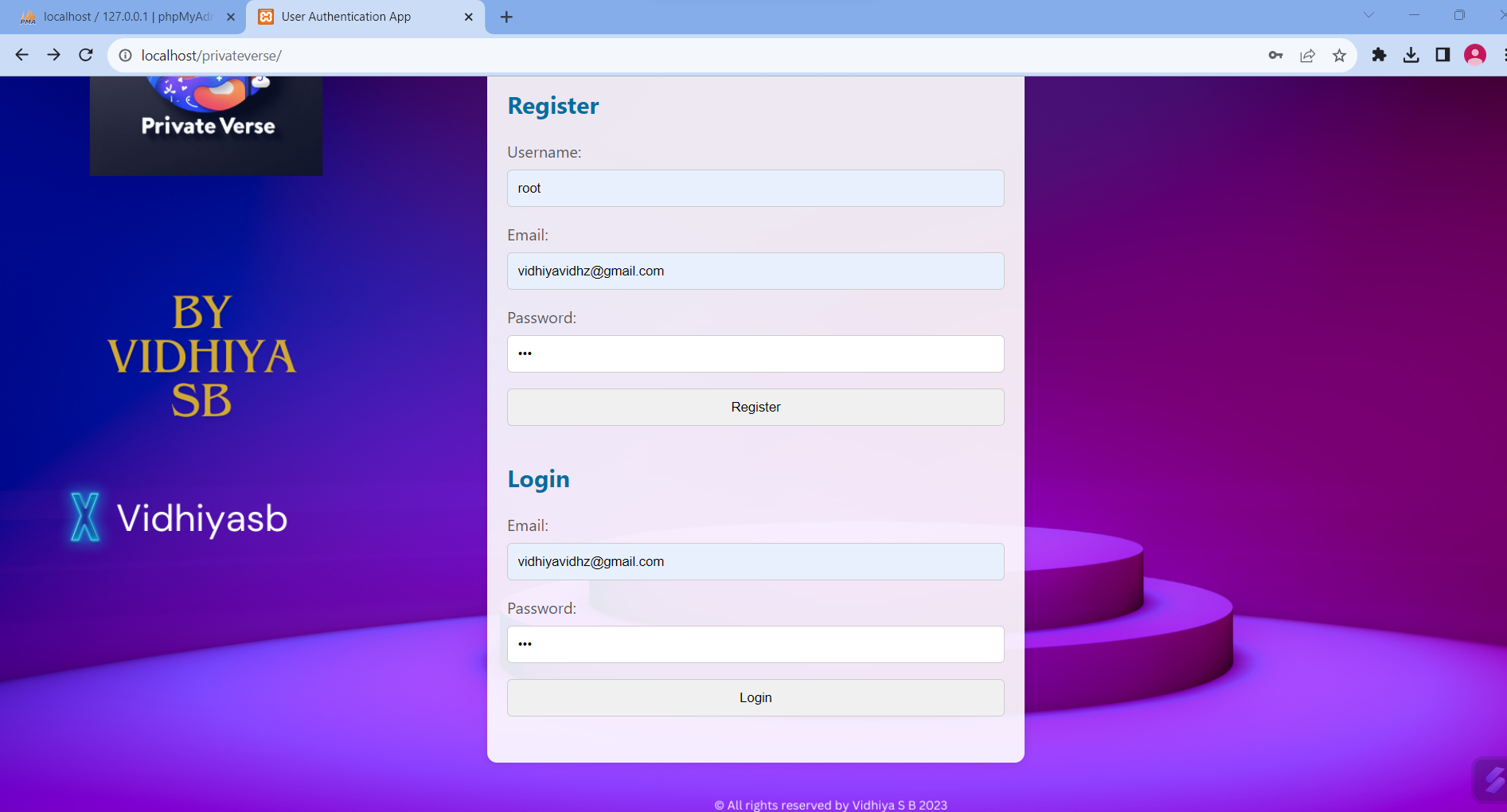
exit();

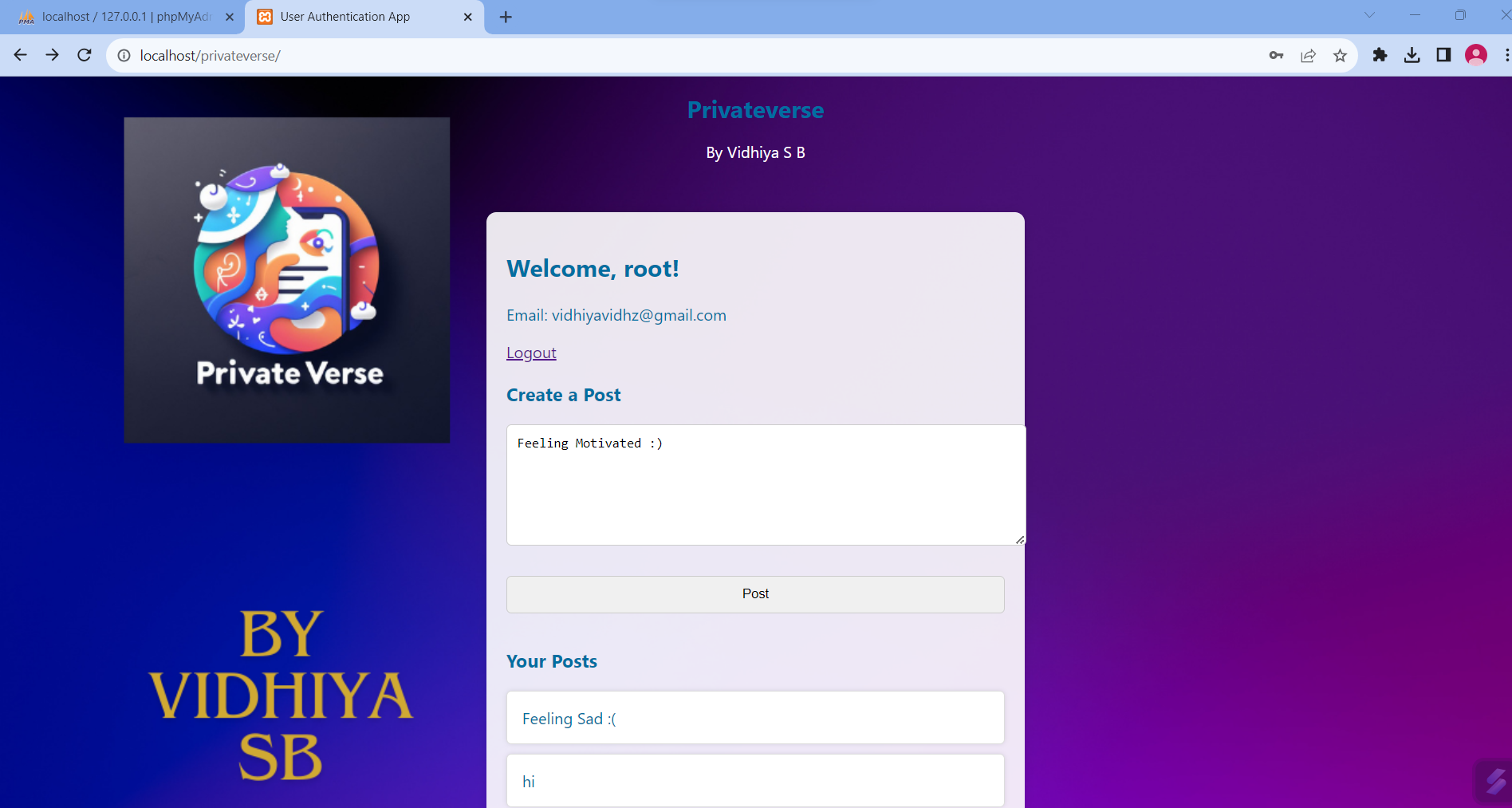
?>

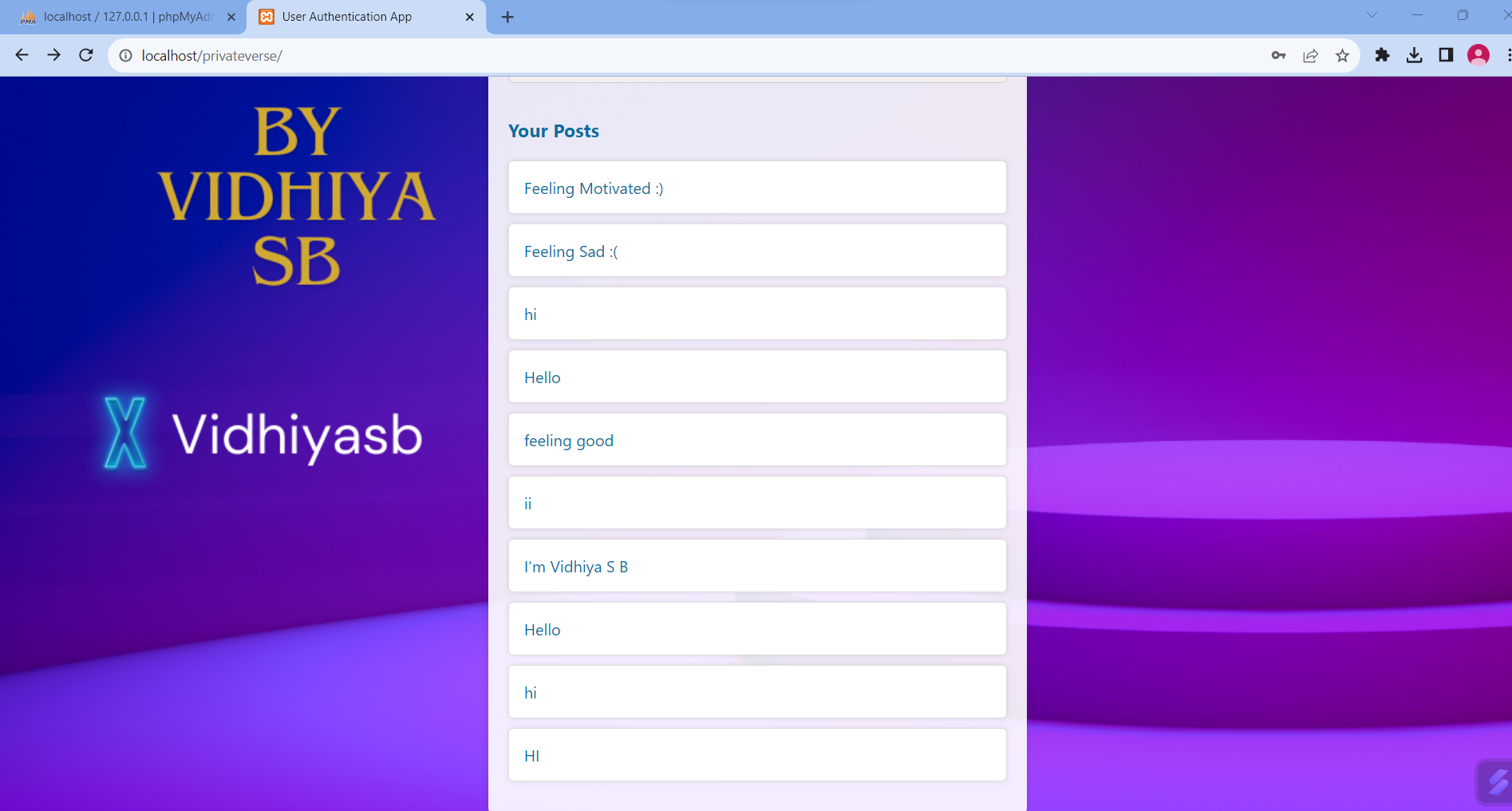
**OUTPUT**

****

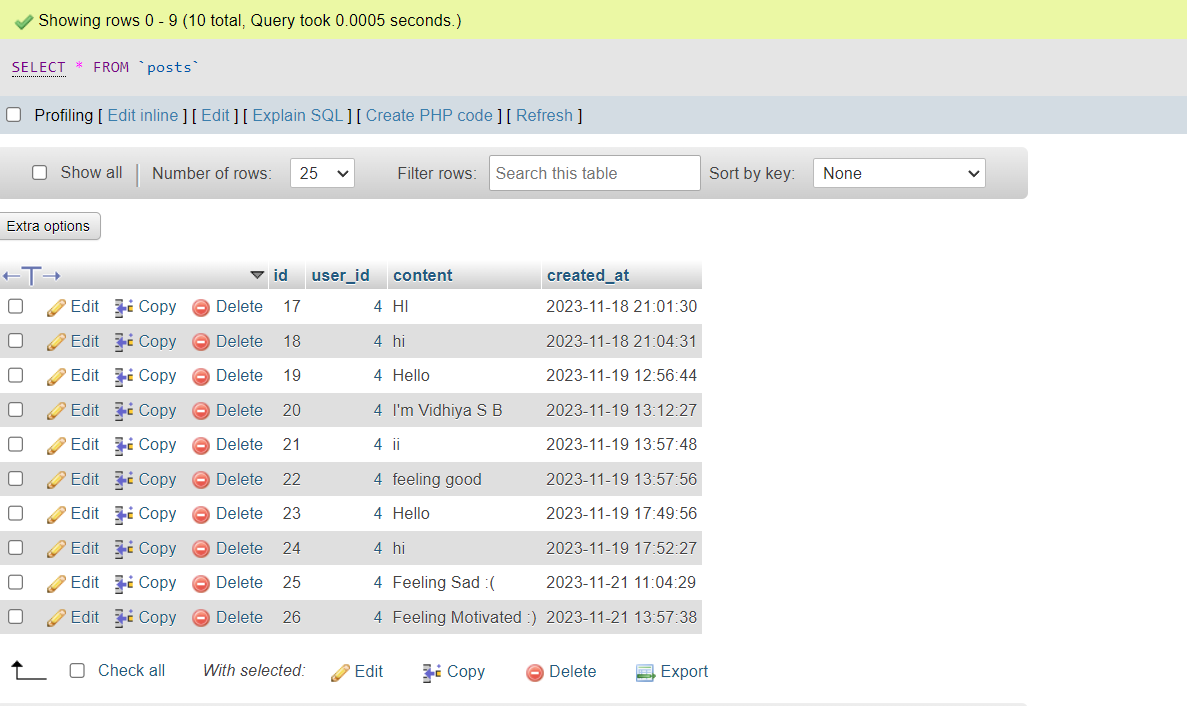
****

****

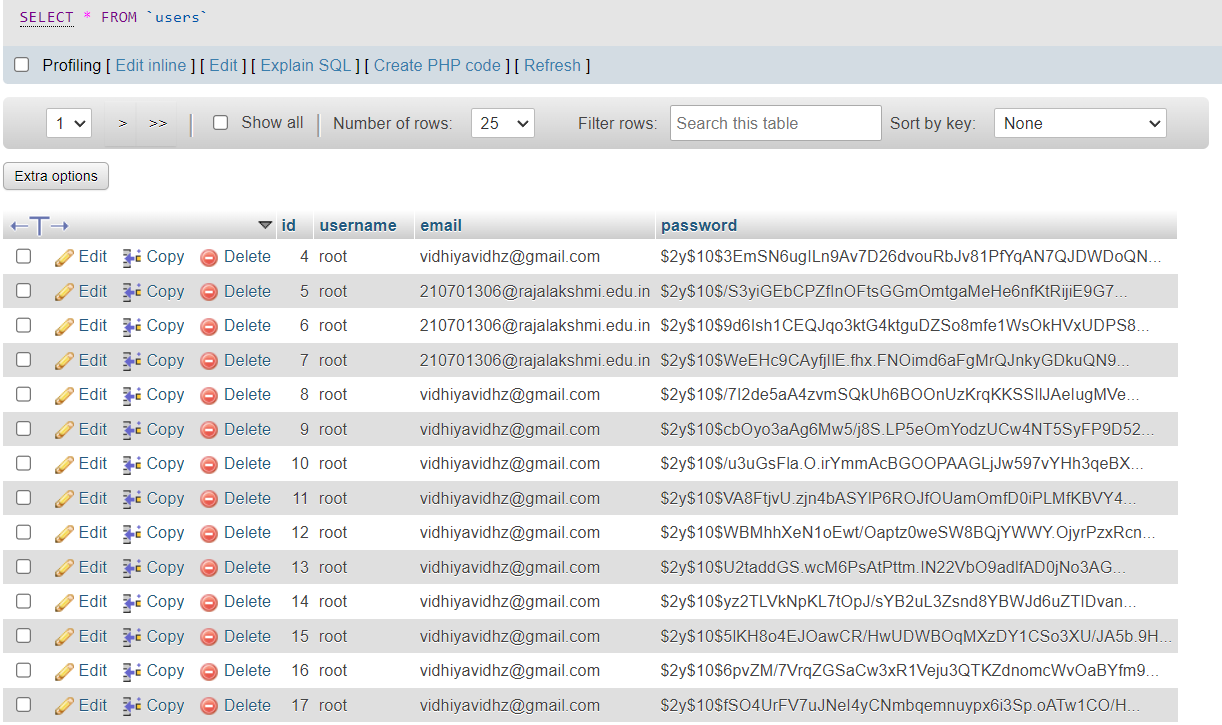
****

****

**DATABASE**

****

**PASSWORD ENCRYPTION IN THE DATABASE**



**CHAPTER 6**

### CONCLUSION

In conclusion, Privateverse is a user authentication and social media platform designed to provide a secure and personalized online space for users to connect and share their thoughts. The project incorporates robust authentication mechanisms, ensuring that user data remains confidential and protected. With features like user registration, login, and post creation, Privateverse facilitates seamless communication while prioritizing user privacy.

The platform leverages a stack of web technologies, including HTML, CSS, JavaScript (jQuery), PHP, and MySQL, to deliver a dynamic and interactive user experience. The use of AJAX enables real-time updates and enhances the overall responsiveness of the application. The integration of SSL ensures secure data transmission, adding an extra layer of protection to user interactions.

Privateverse, with its clean and intuitive interface, empowers users to create posts, share content, and connect with others in a meaningful way. The project's architecture, combining frontend and backend technologies, provides a scalable foundation for future feature enhancements and optimizations.

Through this project, the aim is to offer users a private and versatile online environment, fostering connections while maintaining the utmost security. Privateverse stands as a testament to the seamless integration of various technologies to create a cohesive and functional web application

### REFERENCES:

* <https://scholar.google.com/>
* https://chat.openai.com/