USER MANAGEMENT SYSTEM USING ADMIN PANEL

Mr. Dhanwate Akshay D, Mr. Nale Siddhant M, Mr. Kale Ganesh A, Mr. Godge Akshay R

Guide: Prof. Mundhe.B.B

Department of Computer Engineering, Bachelor of Engineering
Sahyadri Valley College Of Engineering & Technology At – Rajuri , Tal - Junner, Dist – Pune(412 411)

ABSTRACT

This project aims at creating a simple user management system that is required by every website where multiple users can login. This let new user registration , login & logout for every user.

A User Management System implemented with JavaScript and PHP is a vital component of web applications that facilitates user registration, authentication, authorization, and profile management. This abstract outlines the key features and components of such a system, focusing on the synergy between JavaScript and PHP to deliver a secure, interactive, and efficient user experience. The system allows users to create accounts, log in securely, and manage their profiles while administrators can oversee and administer user accounts.Robust security measures, including data validation and protection against common web vulnerabilities, ensure the integrity of user data and the application's overall security. The system empowers web applications to efficiently manage and serve a diverse user base, ensuring a seamless and secure user experience.

CHAPTER 1

INTRODUCTION

1.1 Introduction

A User Management System is a crucial component in many web applications and websites. It allows administrators and users to manage and interact with user accounts, profiles, and permissions. In this introduction, I'll provide an overview of what a User Management System is and how it can be implemented using JavaScript and PHP.

Introduction to User Management System

A User Management System is a software component designed to handle the creation, authentication, authorization, and management of user accounts within a web application. It plays a pivotal role in ensuring the security, accessibility, and personalized experience for users. A typical User Management System offers the following core functionalities:

1. **User Registration:** Users can create new accounts by providing essential information such as username, email, and password.

2. **User Authentication:** Registered users can log in using their credentials to access their accounts and perform various actions on the platform.

3. **Password Management:** Users can reset or change their passwords in case they forget them or want to update their security.

4. **User Profiles:** Users can update their profile information, including personal details,

5. **User Roles and Permissions:** Administrators can assign different roles to users, granting specific permissions or restrictions based on their roles. This helps control who can perform certain actions within the application.

6. **User Search and Listing:** The system should allow administrators to search for users and list them for easy management.

7. **Account Deactivation and Deletion:** Users or administrators can deactivate or delete user accounts when necessary.

Implementation Using JavaScript and PHP -

JavaScript is typically used for the frontend (client-side) of web applications, while PHP is commonly used for the backend (server-side). Here's how the User Management System can be implemented using this technology stack:

1. **Frontend (JavaScript):** JavaScript is used to create the user interface, handle user interactions, and make AJAX requests to the backend for user-related operations. Libraries and frameworks like React, Angular, or Vue.js can be employed for building responsive and interactive user interfaces.

2. **Backend (PHP):** PHP is used for the server-side logic, database interactions, and user authentication. PHP frameworks like Laravel, Symfony or CodeIgniter can expedite the development process. Here's how PHP is used:

3. **User Registration:** PHP scripts handle user registration by accepting user input, validating it, and then saving user details to a database.

4. **User Authentication:** PHP handles user login sessions, verifying user credentials, and maintaining user state throughout their interaction with the application.

5. **User Profiles:** PHP manages the retrieval and update of user profile information from the database.

6. **User Roles and Permissions:** PHP defines and enforces user roles and permissions by checking the user's role before allowing or denying access to certain functionalities.

1.2 Problem Statement:

The user management system is a vital part of the every website which can have any number of users, For example, in Facebook there are currently 2.6 billion users currently.

The data of these users needs to be stored in the database to let them access Facebook based on their credential details.

Similarly, there are many other websites with large number of users. But even for websites with small number of users , User management system is needed.

For example financial websites, Online banking websites, government sites, military sites and more. The data of users in such sites can be a hectic thing to account for.

The most important factor that comes is the security & authentication of this user data. User data is the most commonly sold thing over dark- internet. User credentials are easily tempered with and used for ill purposes.

Safeguarding this data is also an important part of User management system. For this database needs to be well protected.

Every website have data which should be accessed with proper authorization. So, management of user is much needed. Most of the website allows users to access it only on login or when a user creates an account in that website.

1.3 Objective

The objectives of a User Management System are to provide effective management, control, and security over user accounts within a system or application. These objectives are crucial for maintaining a secure and organized environment and ensuring a positive user experience. Here are the primary objectives of a User Management System:

User Registration: Allow users to create new accounts within the system. This includes collecting essential information such as usernames, email addresses, and passwords.

User Authentication: Verify the identity of users when they log in to the system, ensuring that only authorized users gain access.

User Authorization: Assign and manage user roles and permissions to control what actions and features users can access within the system. Differentiate between administrators, regular users, and other roles as needed.

Password Management: Enable users to change or reset their passwords when necessary. Ensure that password policies are in place to enhance security.

User Profile Management: Allow users to update and manage their profile information, including personal details, avatars, contact information, and other customizable data.

User Search and Listing: Provide the ability to search for and list users within the system, making it easier for administrators to manage and view user accounts.

1.4 Methodology

Developing a User Management System in JavaScript and PHP involves a structured methodology to ensure a secure, scalable, and user-friendly system. Below is a step-by- step methodology for creating such a system:

1. Project Planning and Requirements Gathering:

Define the scope of your User Management System.

Gather user requirements, including user registration, authentication, authorization, and other features.

Decide on the technology stack, including JavaScript for the frontend and PHP for the b

2. Database Design:

Design the database schema to store user information, roles, permissions, and other relevant data.

Choose a suitable database management system (e.g., MySQL, PostgreSQL) and create the necessary tables.

3. Frontend Development (JavaScript):

Create user interfaces for registration, login, user profile management, and other user- related actions.

Use JavaScript frameworks or libraries (e.g., React, Angular, Vue.js) to build a responsive and interactive frontend.

1.5 Organisation of Report

- Chapter 1 gives the basic introduction about project and basic fundamentals that will be used in implementing the project.

- Chapter 2 gives us the literature survey of "User Management System". In this we will have brief of journals, research papers, internet source of application.

- Chapter 3 is most important chapter in which we will discuss about the implementation of project including explanation and source code of the project.

- Chapter 4 is the model used for analyzing the performance of User Management System. Outputs of the project at various stages and comparisons between different outputs.

- Chapter 5 will cover the end conclusion of the project and what further can be implemented in future to this project so that it becomes more efficient and reliable.

CHAPTER 2

LITERATURE SURVEY

2.1 Setting up XAMPP for PHP and MySQL

Step 1: Download XAMPP

- Visit the official XAMPP website (https://www.apachefriends.org/index.html) and click on the "Download" button
- Select the version of XAMPP that is compatible with your operating system and proceed with the installation

Step 2: Install XAMPP

- Follow the on-screen instructions to install XAMPP on your computer
- During the installation process, you will be prompted to select the components you want to install. Make sure to select "Apache" and "MySQL"
- Once the installation is complete, launch the XAMPP control panel to start the Apache and MySQL services

Step 3: Test XAMPP

- Open your web browser and navigate to http://localhost
- You should see the XAMPP dashboard, which confirms that the Apache web server and MariaDB database are up and running
- To access PHPMyAdmin, the web-based database management tool, navigate to http://localhost/phpmyadmin

2.2 Role Of Javasscript In User Management System

JavaScript plays a crucial role in user management systems by providing dynamic and interactive functionality to web applications. Here are some of the ways JavaScript is used in user management systems:

1. **User Registration and Authentication**: JavaScript is used to create and validate user registration and login forms. It can perform client-side validation to ensure that users enter the correct data format for email addresses, passwords, and other user information.

2. **User Interface (UI) Enhancements:** JavaScript is essential for improving the user experience by adding interactive elements to the user interface. This can include dropdown menus, modals for password recovery, and other user-friendly features.

3. **Real-time Notifications**: JavaScript can be used to implement real-time notifications, such as displaying alerts for successful logins or informing users of account activity, like password changes or profile updates.

4. **Data Validation**: It can perform data validation on the client-side to ensure that the information entered by the user is consistent with predefined rules, like username availability checks during registration.

5. **Client-Side Routing:** For single-page applications (SPAs), JavaScript frameworks like React, Angular, or Vue are often used to manage client-side routing. This allows users to navigate between different views of the application without requiring a full page reload.

6. **Password Hashing:** While the actual hashing of passwords is typically done on the server-side, JavaScript can be used to help securely transmit passwords to the server through encryption and to perform client-side hashing before sending data to the server.

7. **User Profile Management**: JavaScript can be used to create user profile pages, allowing users to edit their information, change their passwords, upload profile pictures, and more.

8. **Session Management**: JavaScript can handle user sessions, such as determining whether a user is logged in or logged out and managing session timeouts.

CHAPTER 3

SYSTEM DEVELOPMENT

3.1 Description of user management system:

A user management system is a crucial component of many web applications, responsible for user registration, authentication, authorization, and user profile management. It typically involves both the front-end, implemented using JavaScript, and the back-end, implemented using PHP or any other server-side language. Here's a description of how a user management system can be implemented using JavaScript and PHP:

Front-End (JavaScript):

1. User Registration:

•Create a user registration form with fields like username, email, and password.

•Use JavaScript for client-side validation to ensure that the user provides valid input.

•Validate email format, password strength, and confirm password matching.

2. Authentication:

•Implement a login form with fields for username/email and password.

3. User Interface (UI) Enhancements:

•Enhance the user interface with interactive elements, such as modals for password recovery, success/error notifications, and dynamic content loading

3.1.1 Functional Requirement : Functional requirement for user management system are :

1. All users who access the website must be able to visit homepage regardless they are logged in or not. Option for login and new user registration must be given to every user that visit the website.

2. Users must be able to change their password and delete the account whenever they want to. User can also change their details provided it stays unique in the database.

3. User must be logged out completely when they want to.

4. User data must be protected at any cost consisting of login credentials and other user data.

3.1.2 Non-Functional Requirement :

1. User data must be protected at any cost.

2. The following application can be used on different operating systems, thus It offers portability.

3. The website is to be run on local server.

3.1.3 Software Requirements :

- Web Brower like Internet Explorer , Google Chrome , Microsoft Edge , Mozilla Firefox.

POST AND GET METHOD :

GET and POST are the two HTTP methods which are used when we deal with forms.

POST method is used to return Javascript's login form, in which the browser collects the form data, encodes it for sending, sends it to the server, and then receives back its response.

Any request that will be used to change the state of the system - for example, a request that makes changes in the database, provides data into database - should always use POST method. GET is used

only for requests that do not bring changes in the website.

GET should not be used for a password form, because the password will then appear in the URL, and in browser history and server logs, all in plain text. Neither it is suitable for large amount of data, or for binary data, such as an image, videos. A Web application that uses GET requests for admin forms is also a security risk:
it can be easy for an attacker to mimic a form's request to gain access to sensitive parts of the system.

On the other hand, GET is more suitable for things like a web search form, because the URLs that represent a GET request can easily help to redirect and search.

3.5 AGILE METHODOLODY

Agile Methodology is basically a process in which we do constant iterations of production and testing phase whole time during the software development life cycle and contains more advantages over waterfall model. If we talk about waterfall model then it initially required to be fully designed then it is made forward for doing testing but in agile model we could produce a few of product commit it and then do parallel testing and finally verify it. It is pretty simple to make alters in the program and deployment of product is also quick. I am following agile method in this user management system.

Fig - Agile Model

CHAPTER 4

PERFORMANCE ANALYSIS

4.1 ANALYTICS

- In this project, we are to monitor what is going on in the backend of server. How data is being processed and how GET & POST commands are working. For this we use CMD in terminal to print out the details of flow of data for better understanding

- Also in Google Chrome we use Inspect element and console features to tweak the scripts and template. This give us better understand of how the program is working.

- Refresh command on browsers let user see quick changes in their templates.

CHAPTER 5

CONCLUSIONS

3.1 CONCLUSION

So, this is the last chapter of report where there is whole conclusion of the report. We herby conclude the importance of user management system in every website. How necessary they are for better interaction with the users.

We learnt the importance of confidentiality of the user data. How website access should be restricted to the members who have registered.

The marketing value of adding more users to the website and better interaction of user with the website.

FUTURE SCOPE

If throwing some light on the future of this program, so this project has great scope in future. In future we will include many features to this program.

We have countless amendments to make on this project from security & encryption of data to the frontend work

- We can provide encryption algorithms to save user data.
- Password combinations can be made more secure by using combination of alpha numeric & symbols.
- The front end interface of user to login and create new account can be made more appealing and as the same time more secure and robust
- Email validation can be the most important step we need to make on this project.

OUTPUT AND RESULT

5.7 OUTPUT

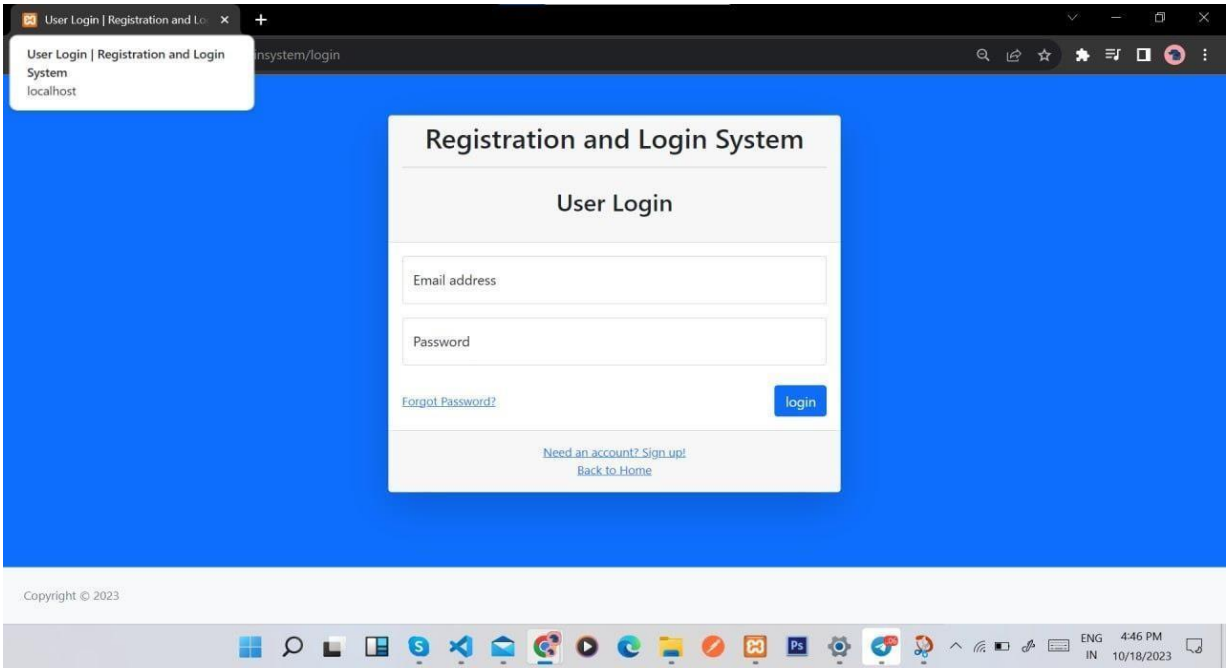


Fig - Register the user and Log In.

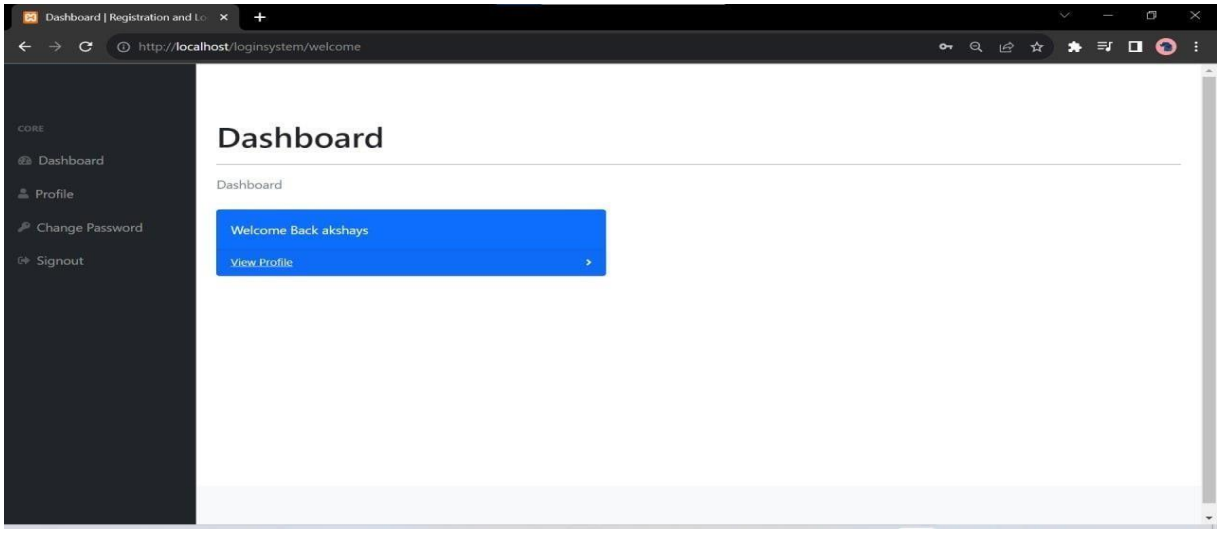


Fig – Dashboard After Log In by the

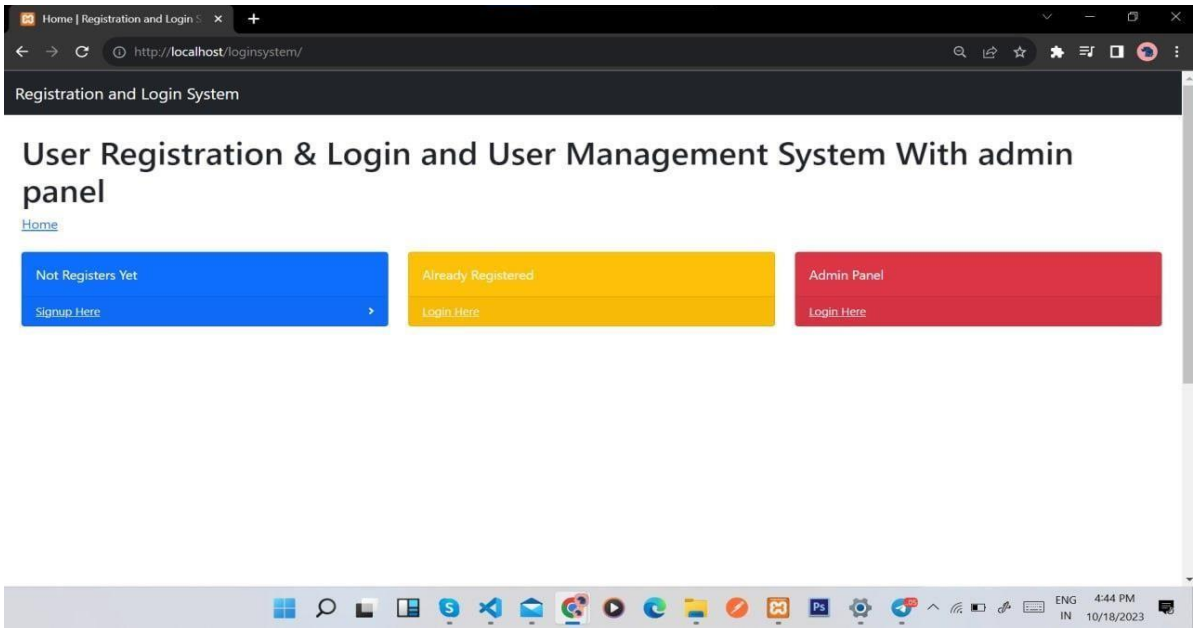
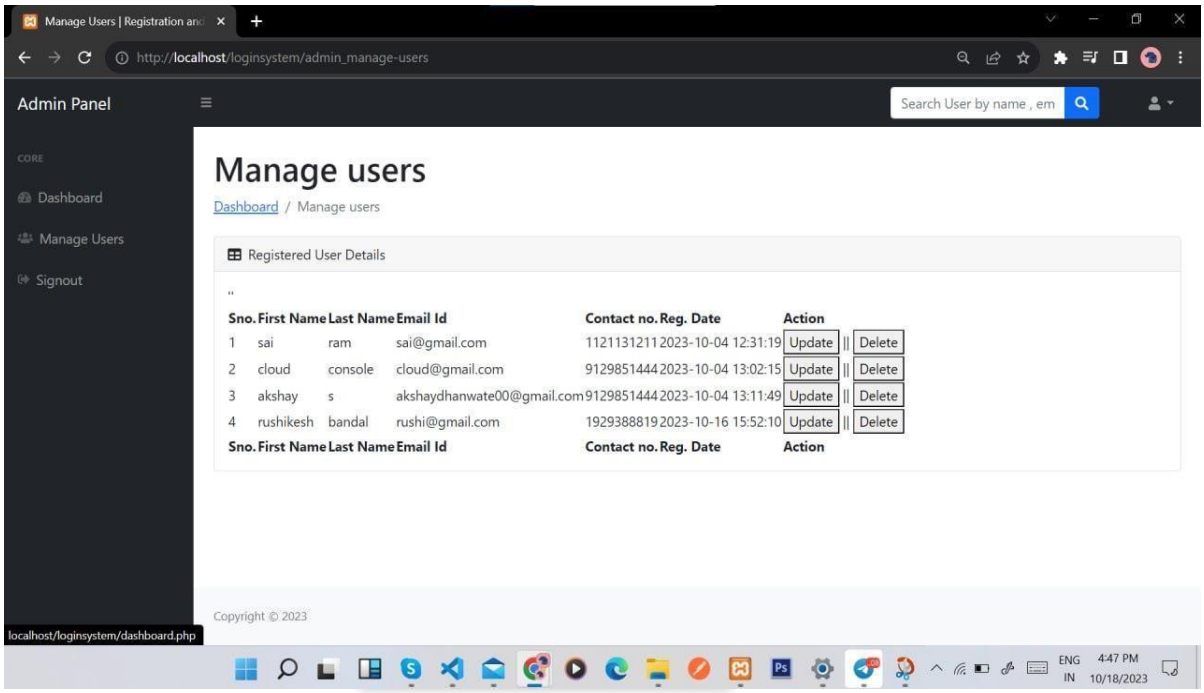


Fig – Dashboard After Log In by th



REFERENCES

***CHAPTER* 6**

- https://www.w3schools.com/php/
- https://stackoverflow.com/questions/tagged/php
- https://www.php.net/docs.php
- https://www.w3schools.com/php/php_mysql_connect.asp
- https://www.youtube.com/results?search_query=php+
- https://www.php.net/manual/en/features.commandline.webserver.php
- https://learn.microsoft.com/en-us/azure/app-service/quickstart- php?tabs=cli&pivots=platform-linux