CODE EXPLANATION FOR MY SECURE LOGIN WEBSITE.

1)CONFIG.PHP.

This PHP script, config.php, sets up database connectivity and manages session initiation:

- session_start(); initiates a new session or resumes an existing session.
- define('DB_HOST', '127.0.0.1'); through define('DB_NAME', 'fildberg_db'); define constants for the database host, user, password, and database name.
- The try...catch block attempts to create a new PDO (PHP Data Objects) instance for database connection using the provided host, database name, user, and password. It also sets the error mode to exception handling. If the connection fails, it catches the PDOException and terminates the script, displaying the error message.

2)FORGOT.PHP.

This PHP script forgot.php handles the "forgot password" functionality:

- It includes the config.php script for database connection.
- The content type is set to JSON using header('Content-Type: application/json');.
- It retrieves and sanitizes the email address from the POST data.
- If the email is empty, it responds with a JSON message indicating that the email is required and exits the script.
- The script then tries to fetch the user with the provided email from the database.
- If the user exists, it generates a random reset token and sets an expiry time for the token.

- The reset token and expiry time are updated in the database for the user.
- The script responds with a JSON message indicating success and that password reset instructions have been sent.
- If no user is found with the provided email, it responds with a JSON message indicating no account found.
- If any error occurs during the process, it catches the exception and responds with an appropriate error message in JSON format.

3)INDEX.PHP**.**

PHP Section

- session_start();: Initiates a PHP session to track user login state across pages.

HTML Structure

- A responsive webpage with a modern design for "Fildberg Institution."
- Contains two main views:
    1. **Authentication Forms**: Login, Sign Up, and Forgot Password forms.
    2. **Main Content**: Displays institution info (Departments, Contact, Courses, About) for logged-in users.

CSS (Embedded in <style>**)**

- **Global Styling**: Resets margins/padding, uses Arial font, and applies a gradient background.

- **3D Background**: A fixed, semi-transparent image with a parallax effect (transform: translateZ(-1px) scale(2)).
- **Form Wrapper**: A centered, glassmorphism-style container (blurred background, shadows) holding the forms.
- **Forms**: Three sliding panels (Login, Sign Up, Forgot Password) with smooth transitions.
- **Overlay Panels**: Decorative sliding panels with gradients for aesthetic transitions.
- **Main Content**: Hidden by default, shown after login with a header and sections.
- **Interactivity**: Hover effects on buttons/links, error message styling.

PHP Logic

- Checks $_SESSION['user_id']: If set (user logged in), hides auth forms and shows main content via inline CSS.

JavaScript (Embedded in <script>**)**

- **Event Listeners**: Runs when the DOM is loaded.
- **Form Toggling**:
    - toggleForms(): Switches between Login and Sign Up forms with the .active class.
    - showForgotPassword() / showLogin(): Handles Forgot Password transitions.
- **Form Submission**:
    - **Sign Up/Login/Forgot Password**: Uses fetch to send form data to signup.php, login.php, or forgot.php. Shows main content on success or displays errors.

    - **Logout**: Sends a request to logout.php and switches back to auth forms.

- **Error Handling**: Displays temporary error messages below inputs.

Functionality

- A user authentication system with a sleek UI.
- Unauthenticated users see login/signup forms; authenticated users see institutional content.
- Forms submit asynchronously, and navigation is handled via sliding transitions and dynamic content display.

This code combines PHP for session management, HTML/CSS for structure and style, and JavaScript for interactivity, creating a functional and visually appealing web interface.

4)LOGIN.PHP.

This PHP script login.php handles user login functionality:

- It includes the config.php script for database connection.
- The content type is set to JSON using header('Content-Type: application/json');.
- It retrieves POST data and logs it to a login_debug.txt file for debugging purposes.
- It trims and validates the email and password from the POST data.
- If either email or password is empty, it responds with a JSON message indicating all fields are required and exits the script.
- The email is sanitized using filter_var.
- The script then tries to fetch the user with the provided email from the database.

- If a user is found and the password matches using password_verify, it sets the user ID in the session and creates a secure cookie for user authentication.
- The script responds with a JSON message indicating a successful login.
- If the credentials are invalid, it responds with a JSON message indicating invalid credentials.
- If any database error occurs, it catches the PDOException and responds with an appropriate error message in JSON format.

5)LOGOUT.PHP.

This PHP script logout.php handles user logout functionality:

- It starts a session with session_start();.
- The content type is set to JSON using header('Content-Type: application/json');.
- It destroys the session using session_destroy(); which logs the user out.
- The script invalidates the authentication cookie by setting its expiration time to the past (time() - 3600).
- It responds with a JSON message indicating that the logout was successful.

6)SCRIPT.JS.

This script.js code manages the behavior and interactions for a series of authentication forms:

- It waits for the DOM content to be fully loaded using document.addEventListener('DOMContentLoaded', () => { ... });.

- It selects various elements related to authentication forms and buttons.
- The toggleForms function toggles the active state of the form wrapper and ensures the forgot password form is hidden.
- The showForgotPassword and showLogin functions manage visibility of the forgot password and login forms respectively.
- The showMainContent function hides the form wrapper and displays the main content.
- The showError function displays error messages for a specific form and hides them after 3 seconds.
- Event listeners are added to handle the submit events for the signup, login, and forgot password forms. These listeners prevent the default form submission, gather form data, and send asynchronous requests to their respective PHP scripts (signup.php, login.php, forgot.php). Based on the response, they either show the main content or display error messages.
- The logoutBtn click event listener sends an asynchronous request to logout.php to log the user out and updates the display accordingly.

7)SIGNUP.PHP.

This PHP script S handles user registration functionality:

- It includes the config.php script for database connection.
- The content type is set to JSON using header('Content-Type: application/json');.
- It retrieves POST data and logs it to a debug.txt file for debugging purposes.

- It trims and validates the fullname, email, password, and confirm password from the POST data.
- If any required field is empty, it responds with a JSON message indicating all fields are required and exits the script.
- It checks if the password and confirm password match. If not, it responds with a JSON message indicating passwords do not match and exits the script.
- The email is sanitized and validated using filter_var.
- If the email format is invalid, it responds with a JSON message indicating invalid email format and exits the script.
- The fullname is sanitized using htmlspecialchars.
- The password is hashed using password_hash.
- The script checks if the email already exists in the database. If it does, it responds with a JSON message indicating the email already exists and exits the script.
- If the email does not exist, it inserts the new user into the database with the provided fullname, email, and hashed password.
- The user ID is stored in the session and a secure cookie is created for user authentication.
- It responds with a JSON message indicating successful registration.
- If any database error occurs, it catches the PDOException and responds with an appropriate error message in JSON format.