Responsive Weekly Schedule Web Application

**Introduction**

This project is a responsive web application designed for managing weekly schedules. The application includes user registration and login functionalities, ensuring each user has their own schedule saved in local storage. The application is developed using HTML, CSS, and JavaScript, and follows principles of responsive design to ensure usability across various devices.

**Project Structure**

The project consists of the following files:  
- login.html: The login and registration page.  
- index.html: The main application page for the weekly schedule.  
- styles.css: The CSS file for styling the application.  
- auth.js: JavaScript file for handling user registration and login.  
- scripts.js: JavaScript file for managing the weekly schedule.

**HTML Files**

The HTML files form the structure of the web pages.   
1. login.html: Contains the form for user registration and login. This page includes input fields for username and password, and buttons for registering and logging in. The form data is validated and handled by the auth.js script.   
2. index.html: Displays the weekly schedule and provides options to add, edit, and delete tasks. The page includes a table that represents the schedule, with rows for hours of the day and columns for days of the week. A modal dialog is used for adding and editing tasks.

**CSS File**

The styles.css file is responsible for the visual styling of the web application. It ensures the application is visually appealing and responsive across different devices such as desktops, tablets, and phones. Media queries are used to adjust the layout and font sizes based on the screen width. For example, the layout of the login/register forms changes from side-by-side to stacked on smaller screens. The schedule table also adjusts its font size and padding to ensure readability on smaller screens.

**JavaScript Files**

The JavaScript files handle the functionality of the application.   
1. auth.js: Manages user registration and login, storing user data in local storage. This script includes functions to register new users by saving their credentials and schedule data in local storage, and to authenticate existing users by verifying their credentials. Upon successful login, the user is redirected to the main application page (index.html).   
2. scripts.js: Manages the weekly schedule, allowing users to add, edit, complete, and delete tasks. Each user's schedule is stored in local storage under their username. The script uses a class-based approach to encapsulate the functionality related to managing the schedule, including rendering the schedule table, handling task operations, and managing the modal dialog for task details.

**Main Principles**

1. \*\*Local Storage:\*\* Used to store user credentials and schedules, ensuring data persistence across sessions. This approach allows the application to function without a backend server, making it lightweight and easy to deploy.  
2. \*\*Responsive Design:\*\* Implemented using CSS media queries to ensure the application is accessible and functional on various devices, including desktops, tablets, and phones. The design principles ensure that the layout and components adjust gracefully to different screen sizes, enhancing user experience.  
3. \*\*User-specific Data:\*\* Each user has a unique schedule, managed through user-specific data in local storage. This ensures personalized schedules for each registered user. The application's logic checks the logged-in user and loads their specific data, providing a customized experience.  
4. \*\*JavaScript Classes:\*\* Utilized to organize code and manage application state, enhancing code readability and maintainability. The class-based structure allows for better separation of concerns and makes the code easier to extend and debug.  
5. \*\*Security Considerations:\*\* While the application uses local storage for simplicity, it is important to note that local storage should not be used for sensitive data in production environments. Proper authentication and data encryption methods should be implemented for real-world applications.  
6. \*\*User Experience:\*\* The application includes features such as task completion indicators (changing colors), modal dialogs for task details, and responsive design to ensure a smooth and intuitive user experience. The use of visual cues and interactive elements helps users to manage their schedules effectively.

**Conclusion**

This web application demonstrates the use of HTML, CSS, and JavaScript to create a functional and responsive user interface for managing weekly schedules. By leveraging local storage, the application maintains user-specific data across sessions, providing a personalized experience. The principles of responsive design ensure that the application is accessible and user-friendly on a variety of devices. This project serves as an example of how client-side web technologies can be used to build practical and interactive applications without the need for a backend server.