JavaScript assignment .

Task 1: User Authentication

Create a login and logout system using localStorage.

1. **Login Form**:
    - Create an HTML login form with fields for email, password, name, and a checkbox to determine if the user is an admin (true/false).
    - Add a "Login" button to submit the form.
2. **Authentication**:
    - When the "Login" button is clicked, validate the email and password.
    - If the admin checkbox is checked, set a flag to indicate the user's admin status.
    - Store the user's information in localStorage.
3. **Logout**:
    - Create a "Logout" button that, when clicked, clears the user's information from localStorage.

Task 2: User Roles and Features

Determine user roles and define available features accordingly.

1. **Admin Features**:
    - If the user is an admin, provide options to add flights, search, and sort flights(price).
2. **Regular User Features**:
    - If the user is not an admin, allow access to search and sort flights (price) only.

Task 3: Flight Data

Define the flight data and structure it as an array of objects.

1. Create a JavaScript array called `flights` containing flight objects with details like "from," "to," "price," and "dates."

Task 4: Booking System

Implement a booking system with features to select flights, manage the cart, and complete the booking process.

1. **Selecting Flights**:
    - Allow users to select flights, which can be added to a cart.
2. **Cart Management**:
    - In the cart, allow users to adjust the number of travelers, remove flights from the cart, and edit booking details.

3. **Booking Process**:
    - Provide a "Book" button in the cart.
    - Upon clicking "Book," calculate the total price based on the selected flights and the number of travelers.
    - Display a popup with a confirmation message and the total amount to pay.

Task 5: SetLocalStorage - My Flights

After the booking is confirmed, store the user's booked flights in localStorage under a key like "myFlights."

Task 6: Additional Instructions

1. Create a user-friendly user interface (HTML and CSS) for your web application.
2. Implement JavaScript functions to handle user interactions, validation, and data manipulation.
3. Use pop-up dialogs or modals for user feedback and to display booking information.
4. structure the code in a modular and organized manner, separating different functionalities into functions and modules.
5. Provide thorough comments and documentation for the code.
6. Test the application to ensure it works as expected and is free of errors.
7. Consider discussing potential challenges and strategies for solving them with your students.

By following these instructions and breaking the project down into smaller tasks, you should be able to create a comprehensive travel planner website with user authentication, flight management, and booking features. This project can serve as a valuable hands-on experience.