Documentation for Webstack Lab 6 Exercise

Overview

The exercise involves a registration form with validation for a Library management system. The application utilizes various technologies including HTML, CSS, JavaScript, and potentially frameworks like Bootstrap and Tailwind CSS.

Directory Structure

- index.html: Main page of the library management system.
- style.css: Custom styles for the application.
- formscript.js: JavaScript for handling form validation and interaction.
- Success.html : To inform the user that the registration has been successfully completed

Code Documentation

index.html

Purpose: This file defines the structure and layout of the library management system's main page.

Key Sections:

- **Header**: Contains the navigation bar with links to other pages.
- **Main Content**: Includes the registration form with input fields for user details.
- **Footer**: Contains footer information and contact details.

Design Decisions:

1. **HTML5 Semantics**: Used HTML5 semantic elements (<header>, <nav>, <main>, <footer>) for better structure and accessibility.
2. **Responsive Design**: Applied responsive classes from Tailwind CSS and Bootstrap to ensure the layout adjusts well on different screen sizes.

style.css

Purpose: This file includes custom styles to override or extend the default styles provided by Tailwind CSS and Bootstrap.

Key Sections:

- **Body**: Adds padding and a backdrop filter effect.
- **Valid/Invalid Icons**: Defines custom styles for validation icons.

Design Decisions:

1. **Custom Styles**: Added specific styles for validation feedback icons to match the application's design requirements.
2. **Utility Classes**: Minimized custom styles by leveraging Tailwind and Bootstrap utility classes whenever possible.

formscript.js

Purpose: Contains JavaScript functions to handle form validation and submission.

Key Functions:

- validateName(): Validates the name input field.
- validateEmail(): Validates the email input field.
- validatePassword(): Validates the password input field.
- validateConfirmPassword(): Validates the confirmation password input field.
- validateDOB(): Validates the date of birth input field.
- calculateAge(): Calculates the age based on the provided date of birth.
- showLoadingAnimation() and hideLoadingAnimation(): Manage the display of a loading spinner during form submission.
- validateForm(): Handles form submission and overall validation.

Design Decisions:

1. **Modular Functions**: Each validation function is modular to ensure separation of concerns and easier maintenance.
2. **Real-Time Feedback**: Validation is performed in real-time as the user types, enhancing the user experience.
3. **Loading Animation**: Provides visual feedback to users while form processing is ongoing.

Rationale Behind Design and Implementation Decisions

1. **Use of Tailwind CSS and Bootstrap**:
    - **Tailwind CSS**: Provides utility-first classes that allow for rapid styling and responsive design.
    - **Bootstrap**: Used for additional UI components and responsiveness.
    - **Rationale**: Combining Tailwind with Bootstrap helps leverage the strengths of both frameworks, ensuring a visually appealing and functional design.
2. **Form Validation**:
    - **Real-Time Validation**: Improves user experience by providing immediate feedback.
    - **Modular Approach**: Each field's validation is handled separately to ensure clear and manageable code.
3. **Responsive Layout**:
    - **Tailwind and Bootstrap Classes**: Ensure that the layout adapts to various screen sizes, enhancing accessibility and usability.
4. **Custom Styles**:
    - **Minimal Overrides**: Custom styles are kept minimal to avoid conflicts with Tailwind and Bootstrap, ensuring consistency and maintainability.