Modernizing MVC C# Applications with Tailwind CSS 4.0 and JavaScript

1. Introduction

The landscape of web development is continuously evolving, prompting developers to seek out modern tools and techniques to enhance application performance, maintainability, and user experience. This report addresses the common scenario of modernizing a .NET MVC C# application by transitioning away from legacy front-end libraries like Bootstrap and jQuery towards more contemporary solutions. Specifically, the focus is on the adoption of Tailwind CSS 4.0 for styling and the identification of suitable modern JavaScript libraries to replace jQuery's DOM manipulation and event handling capabilities, as well as Bootstrap's interactive components such as modals and menus. This transition aims to leverage the performance benefits and flexibility offered by Tailwind CSS 4.0 and to align with current best practices in JavaScript development, all while prioritizing the use of free and open-source solutions that are easy to implement within the existing MVC C# environment.

2. Tailwind CSS 4.0: Key Features and Considerations for JavaScript Integration

Tailwind CSS 4.0 represents a significant evolution of the utility-first CSS framework, introducing a completely rewritten core known as the Oxide engine [1]. This new architecture delivers substantial performance improvements, with full builds reported to be up to 5 times faster and incremental builds achieving speeds over 100 times faster, often measured in mere microseconds [1]. This emphasis on speed suggests that any accompanying JavaScript libraries should also be chosen with performance in mind to ensure the overall application benefits from these advancements.

Beyond performance, Tailwind CSS 4.0 embraces modern web platform features, including native cascade layers, registered custom properties with @property, and the color-mix() function [1]. The integration of these advanced CSS capabilities may reduce the reliance on JavaScript for functionalities that were previously handled through polyfills or workarounds in older CSS frameworks. For instance, the ability to animate gradients using registered custom properties natively can potentially eliminate the need for JavaScript-based animation libraries for such tasks.

The setup process for Tailwind CSS 4.0 has been streamlined, featuring a simplified installation with fewer dependencies and a CSS-first configuration approach [1]. This shift means that much of the customization that previously required JavaScript configuration files can now be achieved directly within CSS using theme variables.

This trend towards CSS-driven configuration might also be reflected in modern JavaScript component libraries, potentially simplifying their integration.

Tailwind CSS 4.0 also includes a first-party Vite plugin for optimized performance in projects using this build tool, along with automatic content detection, which eliminates the need for manual configuration of template file paths [1]. While the user's environment is Visual Studio, understanding the capabilities of modern JavaScript build tools like Vite provides valuable context about the broader front-end development ecosystem. Furthermore, the new version simplifies CSS imports by replacing the @tailwind directives with a single @import "tailwindcss" statement [1]. This more straightforward import mechanism could influence how component libraries are integrated into the project's CSS pipeline.

In essence, Tailwind CSS 4.0's core principles of performance optimization and leveraging modern CSS features should guide the selection of JavaScript libraries. Opting for performant and potentially CSS-centric JavaScript solutions will likely lead to a more cohesive and efficient application.

3. Exploring Modern JavaScript UI Component Libraries for Tailwind CSS

When replacing Bootstrap and jQuery in a modern web application with Tailwind CSS 4.0, developers often turn to contemporary JavaScript UI component libraries to provide interactive elements and enhanced functionality. These libraries can be broadly categorized into headless UI libraries and pre-styled component libraries.

3.1. Headless UI Libraries

Headless UI libraries provide the structural logic and accessibility features for UI components without imposing any specific visual styling [3]. This approach offers maximum flexibility, as developers can fully control the appearance of the components using Tailwind CSS 4.0 utility classes. This aligns well with the user's desire to customize the look and feel of their application.

Headless UI (Tailwind Labs): Developed by the creators of Tailwind CSS, Headless UI offers a set of completely unstyled and fully accessible UI components designed to integrate seamlessly with Tailwind CSS [5]. While primarily available as React and Vue implementations, the core concepts and accessibility features are valuable to understand even in a plain JavaScript context [5]. The existence of framework-specific versions highlights the trend towards component-based architecture in modern front-end development. Headless UI prioritizes accessibility by handling keyboard navigation, focus management, and ARIA attributes [3]. Relevant components for the

user's needs include Dialog for creating modal pop-ups [6], Menu for building navigation menus and dropdowns [6], Disclosure for creating expandable/collapsible sections, and Input as a basic form control [6]. Styling these components with Tailwind CSS 4.0 is facilitated through data attributes that reflect the component's state (e.g., data-open, data-focus) and render props that provide access to the component's state within the rendering logic [11]. This allows for conditional application of Tailwind utility classes based on the component's current state. Headless UI provides a robust foundation for building accessible and highly customized UI elements that are styled using Tailwind CSS 4.0, catering directly to the user's need for control and adherence to modern best practices.

Radix UI: Another prominent headless UI library, Radix UI, focuses on providing a set of primitives for building accessible and performant user interfaces [3]. While primarily centered around React, the underlying principles of separating functionality from styling and its emphasis on accessibility make it a noteworthy option [3]. Although direct integration into a plain JavaScript MVC C# application might require more effort compared to libraries with vanilla JavaScript support, the concepts and potential for adapting its unstyled components with Tailwind CSS 4.0 remain relevant.

Other Headless Options: Libraries like TanStack, while often focused on specific areas like tables (formerly React Table), exemplify the headless approach by providing powerful functionality without pre-defined styles [3]. Depending on the future needs of the application, exploring such specialized headless libraries might be beneficial.

3.2. Pre-styled Tailwind CSS Component Libraries

Pre-styled Tailwind CSS component libraries offer ready-made UI components with built-in styles using Tailwind CSS utility classes [17]. These libraries can significantly accelerate the development process by providing pre-built solutions for common UI elements, reducing the need to style every component from scratch.

Flowbite: Flowbite is an open-source library boasting an extensive collection of UI components (over 56 types, with more than 400 components) built on top of Tailwind CSS and incorporating interactive JavaScript functionality [18]. This library directly addresses the user's need for replacements for Bootstrap's JavaScript components like dropdowns, modals, and potentially form elements with built-in validation capabilities [18]. Flowbite also offers features like dark mode support, a Figma design system, and pre-designed templates, which can aid in maintaining design consistency and speeding up the initial design phase [18]. It provides various methods for JavaScript integration, including NPM installation, CDN inclusion, utilizing data attributes directly

in HTML, initialization functions, and a programmatic JavaScript API [18]. This flexibility allows developers to choose the integration method that best suits their project setup and desired level of JavaScript interaction. Importantly, Flowbite is reported to be compatible with Tailwind CSS v4 [23]. Given its wide range of components and built-in JavaScript functionality relevant to the user's requirements, along with its compatibility with Tailwind CSS 4.0, Flowbite appears to be a strong candidate.

Preline UI: Preline UI is another free and open-source library that offers a vast collection of Tailwind CSS components and examples, with over 840 free components and examples available [17]. It includes support for dark mode and claims compatibility with various frameworks, although its core is built with Tailwind CSS [17]. Preline UI emphasizes ease of use with a "no install, no config, no setup" approach for many of its basic components [25]. It also provides JavaScript plugins for enabling interactive elements [26], indicating solutions for dynamic menus and modals are available. Like Flowbite, Preline UI is also listed as compatible with Tailwind CSS v4 [23]. The combination of its extensive component library and focus on ease of implementation makes Preline UI a promising option for the user.

HyperUI: HyperUI is presented as a collection of free Tailwind CSS components specifically categorized for marketing websites, application UIs, and e-commerce stores [17]. Similar to Preline UI, HyperUI promotes a "no install, no config, no setup" approach, making it straightforward to use by simply copying and pasting HTML snippets [25]. Its compatibility with Tailwind CSS v4 is also noted [23]. The categorized nature of its components could help the user quickly locate elements relevant to different parts of their application, and its simplicity makes it a good choice for rapid prototyping and building basic UI elements with Tailwind CSS 4.0.

DaisyUI: DaisyUI takes a unique approach by functioning as a plugin for Tailwind CSS that introduces higher-level semantic class names [19]. This allows developers to write cleaner HTML with fewer Tailwind utility classes directly applied to elements [31]. DaisyUI offers an extensive library of components and customizable themes [23]. While its core is based on pure CSS without JavaScript dependencies [30], it does provide interactive components like modals that utilize JavaScript [35]. DaisyUI is also reported to be compatible with Tailwind CSS v4 [23]. This library offers an alternative way of working with Tailwind, potentially simplifying HTML structure and providing a wide array of pre-styled, customizable components.

FlyonUI: FlyonUI is described as a free and open-source Tailwind CSS component library that combines semantic classes with powerful JavaScript plugins [26]. This hybrid approach aims to offer both ease of use through semantic naming and enhanced

interactivity via its JavaScript plugins [26]. FlyonUI claims compatibility with various frameworks like React and Vue [26], suggesting a modern and flexible design. Its JavaScript plugins are described as unstyled and accessible, focusing on providing functionality without dictating visual appearance [26]. While explicit mention of Tailwind CSS 4.0 compatibility was not found, as a Tailwind CSS library, it is highly likely to be compatible. FlyonUI's combination of semantic classes and accessible JavaScript plugins makes it an interesting option for the user.

Library Name	Free/Paid	Number of Components	JavaScript Features	Tailwind CSS 4.0 Compatibility	Ease of Implementation	Key Features
Headless UI	Free	Varies (primitives)	Focus on accessibility, requires custom styling and JavaScript for advanced behavior	Yes	Medium (styling and JS logic required)	Modals, Menus, Forms (unstyled)
Flowbite	Free	400+	Interactive components (modals, dropdowns, etc.), form elements	Yes	Easy	Validation (potential), Forms, Menus, Modals, UI Components
Preline UI	Free	840+	Interactive plugins for components	Yes	Easy	Validation (mentions), Forms, Menus, Modals, UI

						Components
HyperUI	Free	226+	Primarily static components	Yes	Easy	Forms, Menus, Modals, UI Components
DaisyUI	Free	50+	JavaScript modals, core is pure CSS	Yes	Easy	Validation (Validator component), Forms, Menus, Modals, UI Components
FlyonUI	Free	78+	JavaScript plugins for interactivity	Likely	Easy	Validation (components), Forms, Menus, Modals, UI Components

4. Form Validation Strategies with Tailwind CSS 4.0

Implementing robust form validation is crucial for any web application. The identified UI component libraries offer varying approaches to handling this aspect. Flowbite provides form components [44], and while specific advanced validation features are not explicitly detailed in the provided snippets, its JavaScript capabilities allow for custom validation logic to be implemented. Preline UI also offers form components [48] and mentions "Validation states New," suggesting some level of built-in validation features might be available. DaisyUI stands out with its dedicated "Validator" component [49], which appears to integrate with standard HTML form validation, applying error and success styling based on input validity. Headless UI, being unstyled, provides the basic Input component [15], but form validation would typically be handled using external validation libraries or custom JavaScript implementations [51]. This offers maximum flexibility but requires more manual effort. FlyonUI has specific documentation on form validation [55], indicating that it provides components like success-message and error-message for displaying validation feedback, likely in

conjunction with JavaScript validation logic.

Beyond the features offered by these component libraries, standalone JavaScript form validation libraries can be effectively used with forms styled using Tailwind CSS 4.0 [58]. Libraries like Formik and Yup provide comprehensive form management and validation capabilities that can be readily integrated with any HTML form structure. Even simpler custom validation functions written in plain JavaScript can be employed to check input values against specific criteria. Integrating these validation mechanisms with Tailwind CSS involves applying Tailwind's utility classes to style the form elements and any associated error or success messages based on the validation status [55]. For instance, Tailwind classes can be dynamically added or removed to change the border color of an input field to indicate an error or success state.

In summary, the user has a range of options for form validation. They can leverage the potential built-in features or components offered by pre-styled UI libraries like Flowbite, Preline UI, DaisyUI, and FlyonUI, or opt for the greater flexibility of using a headless UI library like Headless UI in conjunction with a standalone JavaScript validation library, taking advantage of Tailwind CSS 4.0 for styling the validation feedback.

5. Implementing Dynamic Menus and Modals with Tailwind CSS 4.0

Creating dynamic menus and modal pop-ups is a fundamental requirement for modern web applications. The recommended UI component libraries offer various solutions for achieving this with Tailwind CSS 4.0. Flowbite provides dedicated Menu and Modal components [18] that come with built-in JavaScript functionality. These components can be controlled dynamically using data attributes for simple toggling and a JavaScript API for more advanced programmatic control, such as showing, hiding, and toggling visibility, as well as setting options and handling events. Preline UI also offers Navbar, Navs, Dropdown, and Modal components [24] that include JavaScript functionalities to manage their interactive behavior, allowing for dynamic control of their visibility and state. DaisyUI provides Navbar, Menu, Dropdown, and Modal components [35]. Its Modal component, in particular, can be implemented using the native HTML <dialog> element, which offers built-in accessibility and can be controlled using JavaScript methods like showModal() and close(). DaisyUI also utilizes CSS classes like menu-dropdown-show that can be toggled using JavaScript to create dynamic submenus. Headless UI's Menu and Dialog components [7], while unstyled, are designed to manage their open/close state through state management in frameworks like React or Vue. In a plain JavaScript context, achieving dynamic behavior with Headless UI would involve manually managing the component's state and applying

Tailwind CSS 4.0 classes accordingly based on that state. FlyonUI offers Modal and potentially Menu/Dropdown components [26] that rely on its included JavaScript plugins to handle dynamic behavior, providing ready-to-use interactive elements.

Each of these libraries provides specific built-in JavaScript functionality or recommends patterns for integrating JavaScript to achieve dynamic behavior in menus and modals. For users who prefer more control over the implementation and are comfortable with managing component state and DOM manipulation in plain JavaScript, Headless UI offers a highly accessible and customizable foundation. However, for ease of use and quicker implementation, pre-styled libraries like Flowbite, Preline UI, DaisyUI, and FlyonUI offer ready-to-use components with built-in JavaScript functionalities for creating dynamic menus and modals.

6. Free and Open-Source UI Component Kits

In addition to full-fledged UI component libraries, several free and open-source UI component kits provide collections of basic UI elements built with Tailwind CSS. These kits can be particularly useful for quickly obtaining foundational components like buttons and cards that can then be easily modified and extended using Tailwind CSS 4.0's utility classes to match the specific design requirements of the application. HyperUI [25] is one such example, offering a variety of components categorized for different use cases. Other options like Tailwind Starter Kit and Meraki UI (as listed in [19]) also provide a range of pre-built components that can serve as a starting point for the user's front-end development. These kits typically include common UI elements such as buttons, cards, forms, and navigation components, all styled with Tailwind CSS, making customization through Tailwind's utility classes straightforward. By leveraging these resources, the user can significantly reduce the time spent building basic UI elements from scratch and focus on implementing the application's unique features and more complex components.

7. Integration into MVC C# Application

Integrating front-end assets like CSS and JavaScript from npm packages or CDNs into an MVC C# application within Visual Studio involves standard web development practices. For libraries installed via npm, the CSS and JavaScript files are typically located within the node_modules directory. These files can be included in the MVC project by either copying them into the project's static asset folders (e.g., wwwroot/css, wwwroot/js) or by configuring a build process to handle asset management. NuGet Package Manager can also be used to install libraries that might

provide pre-packaged CSS and JavaScript files suitable for direct inclusion.

CSS files are generally linked in the <head> section of the application's layout file (_Layout.cshtml) or within specific views using the <link> tag, referencing the path to the CSS file. JavaScript files are typically included at the end of the <body> section using the <script> tag, again referencing the file path. For more complex JavaScript logic or when using libraries that benefit from module bundling (like those used with Headless UI or standalone validation libraries), employing a task runner such as Gulp or Grunt, or a more sophisticated build tool like Webpack or Parcel, might be beneficial. While these tools operate outside the standard .NET build process, they can be configured to process JavaScript files, bundle modules, and output static assets that can then be included in the MVC application.

Managing JavaScript dependencies within an MVC C# project can be done using package.json and npm (or yarn/pnpm). While Visual Studio has some built-in support for npm, developers might also choose to manage dependencies and run build scripts from the command line. The key is to ensure that the resulting CSS and JavaScript assets are correctly placed within the project's static asset structure and referenced appropriately in the views or layout files. By following these standard practices, the user can effectively integrate the chosen JavaScript UI component libraries and any standalone validation libraries into their existing MVC C# application.

8. Recommendations and Best Practices

Based on the user's requirements and the analysis of available options, several JavaScript libraries and component kits stand out as suitable replacements for jQuery and Bootstrap in their MVC C# application using Tailwind CSS 4.0. For users prioritizing ease of use and a wide selection of pre-styled components with built-in JavaScript functionality, **Flowbite** and **Preline UI** are highly recommended. Both offer extensive component libraries, are compatible with Tailwind CSS 4.0, and provide relatively straightforward integration methods. For those who desire maximum control over styling and are comfortable with implementing more of the interactive behavior themselves, **Headless UI** is an excellent choice. While it requires more effort in a plain JavaScript context, it offers unparalleled customization and accessibility. **DaisyUI** presents a unique approach with its semantic class names and large component library, making it a strong contender for users who prefer cleaner HTML. **HyperUI** offers a simple and free collection of basic UI components that can serve as a good starting point for many common elements. Finally, **FlyonUI** provides an interesting blend of semantic classes and accessible JavaScript plugins.

To maintain a modern and efficient front-end architecture within the MVC C# context, adopting component-based thinking is advisable, even without a full JavaScript framework. Keeping JavaScript logic separate from the HTML markup will improve maintainability. Leveraging Tailwind CSS 4.0's utility-first approach will ensure consistent styling across the application. Considering progressive enhancement will make the application more robust by ensuring core functionality is accessible even if JavaScript fails. Lastly, optimizing front-end assets through minification and compression will enhance the application's performance.

9. Conclusion

Modernizing the front-end of an MVC C# application by replacing Bootstrap and jQuery with Tailwind CSS 4.0 and contemporary JavaScript libraries is a viable and beneficial undertaking. The analysis indicates that several free and open-source JavaScript UI component libraries are well-suited for this purpose. Flowbite and Preline UI offer extensive sets of pre-styled components with built-in JavaScript, providing ease of use and rapid development. Headless UI allows for maximum customization and control over styling using Tailwind CSS 4.0. DaisyUI offers a unique semantic approach, and HyperUI provides a simple collection of basic components. FlyonUI combines semantic classes with JavaScript plugins. The best choice for the user will depend on their specific priorities regarding ease of implementation, level of customization, and the desired balance between pre-built functionality and manual control. By exploring the recommended libraries and experimenting with their integration into the MVC C# project, the user can achieve a modern, efficient, and highly customizable front-end for their application.

Citerade verk

1. Tailwind CSS v4.0, hämtad mars 20, 2025, https://tailwindcss.com/blog/tailwindcss-v4
2. What is New in Tailwind CSS 4.0 Update? - Elightwalk Technology, hämtad mars 20, 2025, https://www.elightwalk.com/blog/tailwind-css-v4
3. Exploring the shift from CSS-in-JS to headless UI libraries - LogRocket Blog, hämtad mars 20, 2025, https://blog.logrocket.com/exploring-shift-css-in-js-headless-ui-libraries/
4. Tailwind as a Headless UI Primitive | by Hichem Fantar - Medium, hämtad mars 20, 2025, https://medium.com/@hichemfantar/tailwind-is-a-headless-ui-primitive-8b034eba2a43
5. tailwindlabs/headlessui: Completely unstyled, fully accessible UI components, designed to integrate beautifully with Tailwind CSS. - GitHub, hämtad mars 20, 2025, https://github.com/tailwindlabs/headlessui

6. Headless UI - Unstyled, fully accessible UI components, hämtad mars 20, 2025, https://headlessui.com/
7. Dialog (Modal) - Headless UI, hämtad mars 20, 2025, https://headlessui.com/v1/vue/dialog
8. Dialog - Headless UI, hämtad mars 20, 2025, https://headlessui.com/react/dialog
9. Dialog (Modal) - Headless UI, hämtad mars 20, 2025, https://headlessui.com/v1/react/dialog
10. Displaying a stateful Modal using Headless UI | by Shaikh Rezwan | Medium, hämtad mars 20, 2025, https://medium.com/@shaikhrezwan66/displaying-a-stateful-modal-using-headless-ui-1b6c68eeaf3a
11. Dropdown Menu - Headless UI, hämtad mars 20, 2025, https://headlessui.com/react/menu
12. Menu (Dropdown) - Headless UI, hämtad mars 20, 2025, https://headlessui.com/v1/react/menu
13. Menu (Dropdown) - Headless UI, hämtad mars 20, 2025, https://headlessui.com/v1/vue/menu
14. Creating a Dropdown Menu With Headless UI in React - Locofy.ai, hämtad mars 20, 2025, https://www.locofy.ai/blog/create-a-dropdown-menu-with-headless-ui
15. Input - Headless UI, hämtad mars 20, 2025, https://headlessui.com/react/input
16. Button - Headless UI, hämtad mars 20, 2025, https://headlessui.com/react/button
17. 13 best Tailwind CSS component libraries - LogRocket Blog, hämtad mars 20, 2025, https://blog.logrocket.com/13-best-tailwind-css-component-libraries/
18. Tailwind CSS component library - Flowbite, hämtad mars 20, 2025, https://flowbite.com/docs/getting-started/introduction/
19. 10 Free Tailwind CSS UI Kits & Component Libraries 2024 - Daily.dev, hämtad mars 20, 2025, https://daily.dev/blog/10-free-tailwind-css-ui-kits-and-component-libraries-2024
20. TailwindCSS libraries you should know and use. | by Enechukwu Chibuike - Medium, hämtad mars 20, 2025, https://decodementor.medium.com/tailwindcss-library-you-should-know-and-use-3608b902b7a3
21. Tailwind CSS JavaScript - Flowbite, hämtad mars 20, 2025, https://flowbite.com/docs/getting-started/javascript/
22. Quickstart - Flowbite, hämtad mars 20, 2025, https://flowbite.com/docs/getting-started/quickstart/
23. 8 Best Tailwind v4-ready Component Libraries You Should Know | Tailkits, hämtad mars 20, 2025, https://tailkits.com/blog/tailwind-v4-component-libraries/
24. Preline UI - Tailwind CSS components library, hämtad mars 20, 2025, https://preline.co/
25. HyperUI: Free Open Source Tailwind CSS v4 Components, hämtad mars 20, 2025, https://www.hyperui.dev/
26. FlyonUI - Free Tailwind CSS Components Library, hämtad mars 20, 2025, https://flyonui.com/
27. Preline JavaScript | Preline UI, crafted with Tailwind CSS, hämtad mars 20, 2025,

https://preline.co/docs/preline-javascript.html
28. 10 Best Free Tailwind-based UI Component Libraries and UI Kits | Blog - GreatFrontEnd, hämtad mars 20, 2025, https://www.greatfrontend.com/blog/10-best-free-tailwind-based-component-libraries-and-ui-kits
29. HyperUI: 226 Free Tailwind CSS Components - Tailkits, hämtad mars 20, 2025, https://tailkits.com/components/hyperui/
30. daisyUI — Tailwind CSS Components (version 5 update is here), hämtad mars 20, 2025, https://daisyui.com/
31. DaisyUI: CSS Components for Tailwind - CodeParrot, hämtad mars 20, 2025, https://codeparrot.ai/blogs/daisyui-css-components-for-tailwind
32. Introduction — Tailwind CSS Components (version 5 update is here) - daisyUI, hämtad mars 20, 2025, https://daisyui.com/docs/intro/
33. daisyUI — Tailwind CSS Components (version 4 update is here), hämtad mars 20, 2025, https://v5.daisyui.com/
34. daisyUI — Tailwind CSS Components (version 4 update is here), hämtad mars 20, 2025, https://v4.daisyui.com/
35. Tailwind Modal Component — Tailwind CSS Components (version ..., hämtad mars 20, 2025, https://daisyui.com/components/modal/
36. Actions / Modal - Default · Storybook - React Daisy UI, hämtad mars 20, 2025, https://react.daisyui.com/?path=/story/actions-modal
37. DaisyUIComponents.Modal - HexDocs, hämtad mars 20, 2025, https://hexdocs.pm/daisy_ui_components/DaisyUIComponents.Modal.html
38. daisyUI modal • Playground - Svelte, hämtad mars 20, 2025, https://svelte.dev/playground/f61fe53d3d0544b98892ce629c5f6ccb
39. Tailwind Modal Component — Tailwind CSS Components (version 4 update is here) - daisyUI, hämtad mars 20, 2025, https://v4.daisyui.com/components/modal/
40. React Project Setup with Tailwind V4 & DaisyUI V5 install | Step-by-Step Guide with React Router, hämtad mars 20, 2025, https://daisyui.com/resources/videos/react-project-setup-with-tailwind-v4-daisyui-v5-install-step-by-step-guide-with-react-router-4qexfv6gffk
41. Open Source & Fully Free Components Library For Tailwind CSS - FlyonUI - Reddit, hämtad mars 20, 2025, https://www.reddit.com/r/tailwindcss/comments/1g0dz95/open_source_fully_free_components_library_for/
42. FlyonUI: Semantic Tailwind CSS with JS Plugins - NextGen JavaScript, hämtad mars 20, 2025, https://next.jqueryscript.net/tailwind-css/flyonui-semantic-tailwind/
43. Exploring FlyonUI with Vue.js: A Fresh and Evolving Library - dbi services, hämtad mars 20, 2025, https://www.dbi-services.com/blog/exploring-flyonui-with-vue-js-a-fresh-and-evolving-library/
44. Tailwind CSS Forms - Flowbite, hämtad mars 20, 2025, https://flowbite.com/docs/components/forms/

45. React Forms - Flowbite, hämtad mars 20, 2025, https://flowbite-react.com/docs/components/forms
46. Tailwind CSS Number Input - Flowbite, hämtad mars 20, 2025, https://flowbite.com/docs/forms/number-input/
47. hämtad januari 1, 1970, https://flowbite.com/docs/forms/
48. Tailwind CSS Checkbox - Preline UI, hämtad mars 20, 2025, https://preline.co/docs/checkbox.html
49. Tailwind Validator Component - daisyUI, hämtad mars 20, 2025, https://v5.daisyui.com/components/validator/
50. Tailwind Validator Component - daisyUI, hämtad mars 20, 2025, https://daisyui.com/components/validator/
51. How to implement react hook form with headlessui - Codemancers, hämtad mars 20, 2025, https://www.codemancers.com/blog/2024-02-22-react-hook-foem-with-headless-ui
52. Headless Forms - GitHub Gist, hämtad mars 20, 2025, https://gist.github.com/simonihmig/ff39facb915d41f4b27717a10e2196e9
53. Headless Forms. Forms and validation of forms can be a... | by Lior Amsalem - Medium, hämtad mars 20, 2025, https://medium.com/@lior_amsalem/headless-forms-7933ff55d146
54. vee-validate Headless UI example - Codesandbox, hämtad mars 20, 2025, https://codesandbox.io/s/vee-validate-headless-ui-example-9jz9h
55. Tailwind CSS Forms Validation - FlyonUI, hämtad mars 20, 2025, https://flyonui.com/docs/third-party-plugins/form-validation/
56. Tailwind CSS Form wizard - FlyonUI, hämtad mars 20, 2025, https://flyonui.com/docs/advanced-forms/form-wizard/
57. hämtad januari 1, 1970, https://flyonui.com/docs/advanced-forms/form-validation/
58. How to Add Form Validation to Your Tailwind CSS Contact Form - Web3Forms, hämtad mars 20, 2025, https://web3forms.com/blog/tailwindcss-form-validation
59. Client-side form validations with Tailwind CSS - GitHub, hämtad mars 20, 2025, https://github.com/realstoman/tailwind-form-validations
60. Forms - Tailwind CSS, hämtad mars 20, 2025, https://v1.tailwindcss.com/components/forms
61. flowbite/content/components/modal.md at main - GitHub, hämtad mars 20, 2025, https://github.com/themesberg/flowbite/blob/main/content/components/modal.md
62. hämtad januari 1, 1970, https://flowbite.com/docs/navigation/
63. Tailwind CSS Modal - Flowbite, hämtad mars 20, 2025, https://flowbite.com/docs/components/modal/
64. Tailwind Menu Component - daisyUI, hämtad mars 20, 2025, https://daisyui.com/components/menu/
65. Tailwind Dropdown Component - daisyUI, hämtad mars 20, 2025, https://daisyui.com/components/dropdown/
66. Tailwind Navbar Component - daisyUI, hämtad mars 20, 2025, https://daisyui.com/components/navbar/

67. Creating dropdown menus - Build Modern UI Components with DaisyUI and Tailwind CSS, hämtad mars 20, 2025, https://app.studyraid.com/en/read/12415/400835/creating-dropdown-menus
68. Headless UI v2.1: Simplified transition API and improved multi-dialog support, hämtad mars 20, 2025, https://tailwindcss.com/blog/2024-06-21-headless-ui-v2-1
69. Tailwind CSS Modal - FlyonUI, hämtad mars 20, 2025, https://flyonui.com/docs/overlays/modal/
70. hämtad januari 1, 1970, https://flyonui.com/docs/navigation/