**UI Design Standards Documentation**

1. **Introduction**

This document defines the standards and practices for front-end development within ASP.NET Core projects. It ensures consistency, maintainability, and scalability across all user interfaces while leveraging Bootstrap 5 as the foundational framework.

2. **Objective**

* To establish a unified approach to UI development using Bootstrap 5.
* To enforce modular organization of CSS and JavaScript assets.
* To ensure future design changes are traceable and non-destructive.
* To maintain separation between core libraries and custom modifications.

3. **Purpose**

This standard serves as a reference for developers, designers, and maintainers to:

* Build responsive, accessible, and visually consistent interfaces.
* Avoid direct modification of third-party libraries.
* Promote clean, maintainable code through structured folder organization.
* Facilitate onboarding and collaboration across teams.

4. **Scope**

Applies to:

* All ASP.NET Core web applications developed by the team.
* All UI components, pages, layouts, and reusable elements.
* All CSS and JavaScript files used in the front-end layer.

5. **Design Principles**

* **Consistency**: All UI elements must follow a unified design language.
* **Modularity**: Components and styles should be reusable and isolated.
* **Accessibility**: Interfaces must meet WCAG 2.1 AA standards.
* **Responsiveness**: Layouts must adapt to various screen sizes.
* **Separation of Concerns**: Core libraries and custom code must be clearly separated.

6. **Folder Structure**

/wwwroot/

│

├── /css/

│ ├── bootstrap.min.css ← Do not modify

│ ├── custom.css ← All design overrides go here

│ └── components/

│ ├── buttons.css

│ ├── forms.css

│ └── layout.css

│

├── /js/

│ ├── bootstrap.bundle.min.js ← Do not modify

│ ├── custom.js ← All custom scripts go here

│ └── modules/

│ ├── modal.js

│ ├── validation.js

│ └── interactivity.js

│

├── /lib/

│ ├── bootstrap/ ← Downloaded Bootstrap files

│ └── jquery/ ← Optional, legacy support

7. **Bootstrap Integration Guidelines**

* Use **Bootstrap 5** as the base framework.
* Do **not** modify bootstrap.min.css or bootstrap.bundle.min.js.
* All overrides must be placed in custom.css and custom.js.
* Use Bootstrap utility classes where possible before writing custom styles.
* Extend Bootstrap components using BEM or utility-first naming conventions.

8. **Custom Styling Rules**

* Define all variables and tokens in custom.css using CSS custom properties.
* Use semantic class names (.btn-primary, .form-error) for clarity.
* Avoid inline styles—use classes and external stylesheets.
* Document every custom rule with comments:

/\* v1.2 - Added dark theme override for navbar \*/

.navbar {

background-color: var(--color-dark);

}

9. **JavaScript Standards**

* Use vanilla JS or ES Modules for new features.
* Avoid jQuery unless required for legacy support.
* Encapsulate functionality in modules (/js/modules/)
* Document changes with version tags:

// v1.3 - Added modal open/close logic

export function initModal() {

// ...

}

10. **Versioning & Change Management**

* Maintain a changelog for custom.css and custom.js.
* Use semantic versioning (e.g., v1.0.0 → v1.1.0 for minor updates).
* All changes must be backward-compatible unless explicitly deprecated.
* Deprecated styles/scripts must be marked and isolated.

11. **Governance & Review**

* All UI changes must be reviewed by the design lead.
* Pull requests must include screenshots and changelog entries.
* Monthly audits will ensure compliance with this standard.