Buriza Wallet: UI/UX Design and Front-End Development

Clark Alesna[1] Stan Kiefer Gallego[2] Caitlin Lindsay[3]

2025-09-15

clark@saib.dev
stan.gallego@saib.dev
caitlin.lindsay@saib.dev
SAIB Inc.

Table of Contents

I. Introduction . . . 3
II. Buriza Front-End . . . 4
ii.1 Design . . . 4
ii.1.1 Material Design 3 . . . 4
ii.2 Development . . . 4
ii.2.1 Project Architecture . . . 4
ii.2.2 Platform Implementations . . . 4
ii.2.3 Front-End Implementation . . . 5
III. User Interface . . . 7
IV. Conclusion . . . 8
V. References . . . 9
VI. Links . . . 10
VII. References . . . 11

I. Introduction

II. Buriza Front-End

ii.1 Design

ii.1.1 Material Design 3

ii.2 Development

This section covers the technical implementation of Buriza's wallet suite, including architecture decisions, platform implementations, and front-end technologies.

ii.2.1 Project Architecture

Buriza's architecture is built around component sharing and platform independence, enabling a single codebase to serve multiple deployment targets.

ii.2.1.1 Solution Structure

Buriza uses a modular architecture with 5 .NET 9 projects designed for scalability and maintainability:

- **`Buriza.UI`** - Shared Blazor component library containing all user interface elements
- **`Buriza.Data`** - Core data models and services for wallet functionality
- **`Buriza.Extension`** - Browser extension for seamless dApp interaction
- **`Buriza.Web`** - Progressive web app for universal access
- **`Buriza.App`** - Cross-platform MAUI app for mobile and desktop

This architecture eliminates the need to implement separate UI layers for each platform. Instead of building distinct interfaces for the browser extension, web app, and mobile app, all platforms consume the same **`Buriza.UI`** components. This approach ensures consistent user experience while dramatically reducing development effort and maintenance overhead.

ii.2.1.2 Shared Component Library (`Buriza.UI`)

The component library provides a comprehensive set of reusable Blazor components organized into a hierarchical structure:

- **Common** - Foundational UI primitives like buttons, text fields, and navigation tabs
- **Controls** - Advanced interactive elements including asset cards and search functionality
- **Layout** - Application structure components for sidebars, headers, and main content areas
- **Pages** - Complete page/screen implementations for wallet operations like assets, transaction history, dapp access, send, and receive

The library leverages MudBlazor's Material Design 3 implementation alongside Tailwind CSS for utility-first styling and responsive design patterns. This combination ensures both visual consistency and flexible customization across all platform implementations.

ii.2.2 Platform Implementations

Each platform implementation leverages the shared **`Buriza.UI`** components while providing platform-specific features and deployment mechanisms.

ii.2.2.1 Browser Extension

The browser extension represents Buriza's most integrated approach to decentralized web interaction, embedding wallet functionality directly into the user's browsing experience. Built with Blazor WebAssembly and the Blazor.BrowserExtension framework, the extension follows Manifest V3 standards to ensure compatibility with modern browser security requirements and enhanced performance through service workers.

ii.2.2.2 Progressive Web App

The progressive web application serves as Buriza's most accessible deployment target, requiring only a modern web browser to provide full wallet functionality. Built with Blazor WebAssembly, the PWA delivers near-native performance by compiling C# code to WebAssembly bytecode that executes directly in the browser's runtime environment.

The PWA architecture enables installation directly from the browser without requiring app store distribution, creating a native-like experience while maintaining web-based deployment advantages.

ii.2.2.3 Cross-Platform App

The MAUI application targets iOS, Android, macOS, and Windows through a hybrid architecture that combines native platform capabilities with web-based UI components. At the core of this implementation is BlazorWebView, a native control that hosts Blazor content within a webview container while providing seamless integration with platform-specific APIs.

BlazorWebView acts as a bridge between the native MAUI shell and the Blazor UI layer, allowing the same **`Buriza.UI`** components to render within a native application context. This approach eliminates the need to rebuild the entire user interface using platform-specific controls like XAML, while still providing access to native device features such as secure storage, biometric authentication, and push notifications.

The hybrid model significantly reduces development complexity by leveraging existing web technologies and shared component libraries. Rather than maintaining separate native codebases for each platform, the application shares a single UI implementation while the MAUI framework handles platform-specific compilation and native API integration automatically.

ii.2.3 Front-End Implementation

The front-end implementation combines modern web technologies with component-driven architecture to deliver consistent user experiences across all platform deployments.

ii.2.3.1 Blazor Components

Blazor serves as the foundational UI framework, enabling C# development for web interfaces through WebAssembly compilation. This approach allows the entire Buriza application stack to use a single programming language, eliminating context switching between backend and frontend development.

The component architecture follows a hierarchical structure where complex UI elements are composed of smaller, reusable primitives. Blazor's two-way data binding simplifies form interactions and real-time updates, particularly important for wallet operations that require immediate visual feedback on transaction states and balance changes.

ii.2.3.2 MudBlazor Design System

MudBlazor v8.11.0 provides the Material Design 3 foundation for Buriza's visual language, delivering pre-built components with accessibility standards and smooth animations. The theming system integrates seamlessly with CSS custom properties, allowing Buriza to maintain brand identity while leveraging Material Design's proven usability patterns.

Buriza extends this foundation with custom wallet-specific components:

Common Components:

- **BurizaButton** - Standardized button with consistent styling and interaction states
- **BurizaTextField** - Custom text input with validation and wallet-specific formatting
- **BurizaTabs** - Navigation tabs with custom styling for asset type switching
- **BurizaSelect** - Dropdown selection with wallet account and asset filtering
- **BurizaHeader** - Consistent header layout across all pages

Control Components:

- **BurizaAssetCard** - Asset display with balance, price changes, and interactive actions
- **BurizaSearchBar** - Universal search functionality for assets and transactions
- **BurizaDappCard** - dApp connection interface with authorization controls

ii.2.3.3 Tailwind CSS Integration

Tailwind CSS v4 complements MudBlazor by providing utility-first styling capabilities for custom layouts and responsive behavior. The integration uses a Bun build pipeline for rapid CSS compilation and automatic purging of unused styles, ensuring optimal bundle sizes.

The utility-first approach enables precise control over spacing, positioning, and responsive breakpoints, particularly valuable for wallet interfaces that require exact alignment for transaction details and balance displays. Custom CSS variables bridge the gap between Tailwind utilities and MudBlazor's theming system.

ii.2.3.4 Responsive Design Patterns

Buriza implements a mobile-first responsive design strategy that adapts to various screen sizes and interaction methods. The design system accommodates everything from mobile browser extensions to desktop applications while maintaining usability and visual hierarchy.

The responsive approach extends beyond screen size to consider platform-specific interaction patterns. Touch targets are appropriately sized for mobile interfaces, while desktop versions support keyboard navigation and hover states, ensuring consistent functionality regardless of how users access Buriza.

III. User Interface

IV. Conclusion

V. References

Google. (2025). *Material Design 3*. Material Design. https://m3.material.io/

Google. (2023). *Manifest V3 - Chrome Extensions*. Chrome Developers. https://developer.chrome.com/docs/extensions/mv3/

Microsoft. (2024). *ASP.NET Core Blazor*. Microsoft Learn. https://learn.microsoft.com/en-us/aspnet/core/blazor/

Microsoft. (2024). *.NET Multi-platform App UI (.NET MAUI)*. Microsoft Learn. https://learn.microsoft.com/en-us/dotnet/maui/

Microsoft. (2024). *BlazorWebView for .NET MAUI*. Microsoft Learn. https://learn.microsoft.com/en-us/dotnet/maui/user-interface/controls/blazorwebview

Mozilla. (2023). *Progressive web apps (PWAs)*. MDN Web Docs. https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps

MudBlazor Team. (2024). *MudBlazor - Blazor component library*. MudBlazor. https://mudblazor.com/

Tailwind Labs. (2024). *Tailwind CSS - A utility-first CSS framework*. Tailwind CSS. https://tailwindcss.com/

WebAssembly Community Group. (2023). *WebAssembly*. WebAssembly. https://webassembly.org/

VI. Links

VII. References