Progressive SSR Framework: Ziro

Nariman Movaffaghi
Supervisor: Soroush Vedaei

November 28, 2024

Contents

1 Abstract

Ziro is a modern JavaScript framework designed to simplify web development by offering a flexible and scalable foundation. With inspiration from frameworks like Remix, Next.js, Nuxt, and Astro, Ziro prioritizes typesafety, modular design, and an improved developer experience. It provides tools to streamline workflows and address common development challenges.

Core features of Ziro include support for server-side rendering (SSR), client-side rendering, and a unique partially SSR mode that streams data to users in real time using web streams. Developers can utilize middlewares, actions, loaders, and meta tag management to build dynamic applications. These features are complemented by layouts, a modular architecture, and the ability to extend functionality using pre-created plugins, ensuring reusability across projects.

This project acts as a proof of concept, demonstrating that full-stack JavaScript applications can be integrated with plugins to accelerate development and scale more effectively.

By balancing simplicity and adaptability, Ziro aims to empower developers to build reliable and extensible web applications that grow with their needs while integrating seamlessly with existing technologies.

2 Introduction

Web development frameworks have evolved significantly in recent years, yet many still lack a cohesive approach to combining scalability, modularity, and developer-friendly abstractions. Frameworks like Next.js and Remix provide powerful tools for building web applications, but they do not expose certain critical layers, such as the router, for plugin-based customization. This limitation often forces developers to repeatedly create essential features like authentication, which involves setting up routes, middleware, actions, and components manually for every project.

Ziro addresses this gap by introducing a thoughtful approach that abstracts multiple parts of a full-stack framework while allowing them to scale independently. Ziro enables plugins to dynamically add custom pages, endpoints, middleware, and actions. This approach significantly reduces repetitive work and accelerates the development process, making it ideal for tasks like implementing reusable authentication systems or integrating predefined pages across projects.

Additionally, Ziro adheres to web standards to ensure compatibility across various runtimes, including Node.js and browsers. The framework is structured into separate packages to maintain modularity and framework-agnostic design:

- ziro/router: Contains the core routing concepts, designed to work independently of any specific framework.
- ziro/react: A React-compatible implementation for seamless integration with React-based projects.
- ziro/generator: A tool for generating TypeScript type-safe definitions to ensure robust developer workflows.

Ziro's features are grounded in modern web development principles. It offers a fully typesafe routing system, ensuring that all actions and data derived from middleware or layouts maintain type safety throughout the application. With its sequential data flow, loaders from nested pages run in order, allowing data fetched at higher levels to cascade to lower ones. The plugin system further enhances flexibility by simplifying the process of extending applications with custom functionalities.

Targeted primarily at developers, Ziro also opens the door to building tools for broader audiences, such as WordPress users who may not have a strong technical background. By introducing modularity and preconfigured components, it streamlines the creation of dynamic, scalable applications like e-commerce platforms, personal blogs, and full-stack applications.

Developing Ziro has presented several challenges, including implementing robust type safety—a feature still relatively novel in modern frameworks—and ensuring seamless integration across different rendering modes: SSR, CSR, and a partially SSR mode that streams data to users. The result is a proof of concept that not only demonstrates the feasibility of combining full-stack JavaScript apps with plugin-based scaling but also provides a modular foundation for scalable, dynamic application development.

This introduction outlines Ziro's core philosophy, features, and potential impact, setting the stage for a deeper exploration of its architecture, methodology, and applications.

3 Literature Review

4 Methodology

5 Theses

6 Technical Specifications

7 Results

8 Future Works