# Comprehensive Technical Audit of iRenown.com

**Author:** Manus AI **Date:** December 30, 2025 **Scope:** Frontend and Backend analysis of the public-facing platform, irenown.com.

## 1. Executive Summary

A technical audit of iRenown.com was conducted to identify potential flaws in the platform's frontend and backend implementation. The platform is built using Next.js, which provides a solid foundation, but several critical security and infrastructure configuration flaws were identified.

The most significant findings relate to **missing security headers** and **improper server configuration** for handling non-existent routes. These issues expose the platform to common web vulnerabilities and create confusion for external integrations and search engine optimization (SEO).

## 2. Detailed Findings and Recommendations

The audit is broken down into two primary categories: Backend/Infrastructure and Frontend/Accessibility.

### 2.1. Backend and Infrastructure Flaws

The server configuration exhibits several critical security and operational flaws, primarily related to HTTP response headers and route handling.

#### Critical Flaw 1: Missing Security Headers

The server is not configured to send several essential HTTP security headers, which leaves the application vulnerable to common attacks.

| Header | Status | Impact | Recommendation |
| --- | --- | --- | --- |
| **Content-Security-Policy (CSP)** | Missing | **High**: Allows Cross-Site Scripting (XSS) and data injection by failing to whitelist trusted content sources. | Implement a strict CSP to define trusted sources for scripts, styles, and other assets. |

| | | | |
|---|---|---|---|
| **X-Frame-Options** / **CSP: frame-ancestors** | Missing | **High**: Allows Clickjacking attacks, where the site can be embedded in a malicious iframe to trick users into clicking hidden elements. | Implement `X-Frame-Options: DENY` or `Content-Security-Policy: frame-ancestors 'none'`. |
| **X-Content-Type-Options** | Missing | **Medium**: Allows MIME sniffing, which can lead to XSS vulnerabilities in older browsers. | Implement `X-Content-Type-Options: nosniff`. |
| **Referrer-Policy** | Missing | **Low**: May leak sensitive URL information to third-party sites. | Implement a secure policy such as `Referrer-Policy: strict-origin-when-cross-origin`. |

### Critical Flaw 2: Improper 404 Handling (Catch-all Route)

The server is configured to return the main landing page HTML with an `HTTP 200 OK` status code for virtually all non-existent paths, including attempts to access sensitive files (e.g., `/.env`, `/robots.txt`) and non-existent API endpoints.

This configuration is a significant flaw because:

- **Security Masking**: It prevents security scanners from correctly identifying non-existent resources, making it harder to detect potential misconfigurations.
- **SEO**: Search engines will index non-existent pages as valid content, leading to "soft 404" errors and negatively impacting search rankings.
- **API Debugging**: It makes it impossible to distinguish between a non-existent API route and a functional one without a full request body, complicating external integration.

**Recommendation:** The Next.js server should be configured to return a proper `HTTP 404 Not Found` status code and a minimal error page for all unhandled routes.

## 2.2. Frontend and Accessibility Flaws

The frontend, built on Next.js, is generally performant with a small initial page load size (approx. 80KB on the sign-in page). However, a few accessibility issues were noted.

### Flaw 3: Accessibility Barriers (Missing ARIA/Labels)

Several interactive elements, particularly on the `/studio` page, lack proper accessibility attributes, which creates a barrier for users relying on screen readers.

- **Buttons without Labels**: Three buttons were identified that contain neither visible text nor an `aria-label` attribute. Screen readers will announce these as "button," providing no context to the user.
- **Unlabeled Input**: The file upload input on the studio page is not explicitly associated with a `<label>` element.

**Recommendation:** All interactive elements must be clearly labeled. For buttons without visible text, an appropriate `aria-label` should be added. For the file input, ensure a `<label for="input-id">` is correctly implemented.

## 3. Conclusion

The iRenown platform demonstrates a strong foundation with its use of Next.js and a focus on performance. However, the identified **Critical Flaws 1 and 2** (missing security headers and improper 404 handling) pose the most immediate risk and should be addressed as a top priority. Implementing the recommended security headers will significantly harden the application against common web attacks, while correcting the 404 handling will improve operational clarity and SEO. Addressing the accessibility flaws will ensure a better experience for all users.