**DASHBOARD LAYOUT With Code for META DATA**

✅ **SEO Audit Tool** – Let’s start by designing the **interface and backend logic for analyzing SEO performance**. ✅ **Metadata Optimization** – Implement **AI-driven updates for title tags, descriptions, and keyword structures**. ✅ **Dashboard Analytics** – Provide **interactive reports on ranking trends, keyword performance, and site insights**. ✅ **NIST Compliance Framework** – Integrate security and compliance tracking to align with **industry standards**. ✅ **User Authentication & Sessions** – Once login functionality is ready, ensure **secure access and session tracking**.

**SEO AI Dashboard Layout – Key Sections**

✅ **Header Section** – Displays **welcome message**, user status, and basic navigation. ✅ **SEO Audit Panel** – Allows users to **input a website URL** for real-time SEO analysis. ✅ **Metadata Optimization** – Provides **AI-driven recommendations** for title tags, descriptions, and keywords. ✅ **Ranking Insights & Tracking** – Shows **historical performance, keyword trends, and ranking predictions**. ✅ **SEO Automation Controls** – Configure **metadata updates, URL structuring, and site-wide optimizations**. ✅ **Interactive Reports & Export Options** – Generates downloadable nsights for **SEO performance tracking**.

**Recommended Layout Structure for GEN-AIS SEO AI Dashboard**

Here’s a **user-friendly and scalable layout** that ensures clarity and efficiency:

**✅ 1️⃣ Header Section**

* **Welcome Message** (e.g., “Welcome to GEN-AIS SEO AI Dashboard”)
  + **User Status** (Displays user tier: *Basic, Advanced, Enterprise*)
* **Navigation Links** (Quick access to SEO features, reports, and settings)
  + **How Navigation Links Work in GEN-AIS**
  + ✅ **Top-Level Menu** – Gives users direct access to the **SEO AI Dashboard, AI Services, Reports, and Settings**. ✅ **Sidebar Links (If Used)** – Allows quick switching between **SEO audit tools, metadata optimization, and ranking analysis**. ✅ **Action-Based Navigation** – Buttons like **“Run Audit” or “Apply SEO Changes”** trigger workflow transitions. ✅ **User-Based Access** – Navigation adapts based on whether a user is **new or returning**, ensuring a smooth onboarding.

**SAMPLE MENU**

* + **Primary Navigation (Top Menu)**
    - **SEO Dashboard** – /seo-ai-dashboard
    - **SEO Audit** – /seo-ai-audit
    - **Metadata Optimization** – /seo-ai-metadata
    - **Ranking Analytics** – /seo-ai-ranking
    - **Reports & Insights** – /seo-ai-reports
    - **Settings & Preferences** – /seo-ai-settings
    - **📌 User-Based Access**
    - **New User Registration** – /seo-ai-register
    - **Returning User Login** – /seo-ai-login

**✅2️⃣ SEO Audit PanelWebsite Input Field** (Users enter a URL for SEO analysis)

* **Run Audit Button** (Executes real-time ranking & SEO checks)
* **Audit Summary Display** (Shows key findings like SEO score, metadata issues)

**✅ 3️⃣ Metadata Optimization Tool**

* **Title Tag Suggestions** (AI-driven title recommendations)
* **Description & Keyword Enhancements** (Automated content optimization)
* **Generate & Apply Updates Button** (Applies changes site-wide)

**✅ 4️⃣ Ranking Insights & Keyword Tracking**

* **Historical Performance Graph** (Visualizes ranking trends)
* **Keyword Competition Analysis** (Displays top-ranking keywords)
* **Adaptive Ranking Suggestions** (AI-driven improvement recommendations)

**✅ 5️⃣ SEO Automation Controls**

* **Auto-Metadata Updates Toggle** (Configurable automation settings)
* **Canonical Tagging Options** (Ensures SEO integrity across pages)
* **Redirect & URL Structure Enhancements** (Optimizes site navigation & indexing)

**✅ 6️⃣ Interactive Reports & Export Options**

* **User Activity & SEO Performance Dashboard** (Detailed reporting interface)
* **Export Data Button** (Generates PDFs for SEO audit insights)
* **Tier-Based Feature Access Control** (Enables paid-tier restrictions)

**Key Changes Needed for External Websites**

✅ **Use** fetch **with CORS Handling** – Many sites **block direct metadata fetching** for security reasons. ✅ **Leverage a Proxy Server** – If metadata is **restricted**, use an API service to retrieve the data. ✅ **Ensure HTML Parsing Works Correctly** – External sites may have different <meta> tag structures.

**This is the code used to get the keyword metadata directly from WIX**

import { fetch } from 'wix-fetch';

export function getExternalMetadata(url) {

return fetch("https://cors-anywhere.herokuapp.com/" + url, {

method: "GET",

headers: { "Content-Type": "text/html" }

})

.then(response => response.text())

.then(html => {

let parser = new DOMParser();

let doc = parser.parseFromString(html, "text/html");

let title = doc.querySelector("title")?.innerText || "No title found";

let description = doc.querySelector('meta[name="description"]')?.content || "No description found";

let keywords = doc.querySelector('meta[name="keywords"]')?.content || "No keywords found";

return { title, description, keywords };

})

.catch(error => {

console.error("Error fetching metadata:", error);

return { title: "Error", description: "Error", keywords: "Error" };

});

}

Note: **Key Changes Needed for External Websites**

✅ **Use** fetch **with CORS Handling** – Many sites **block direct metadata fetching** for security reasons. ✅ **Leverage a Proxy Server** – If metadata is **restricted**, use an API service to retrieve the data. ✅ **Ensure HTML Parsing Works Correctly** – External sites may have different <meta> tag structures.

**Here’s the same metadata retrieval code, but using JavaScript’s built-in fetch() instead of wix-fetch which should work for most websites:**

export function getWebsiteMetadata(url) {

return fetch(url, {

method: "GET",

headers: { "Content-Type": "text/html" }

})

.then(response => response.text())

.then(html => {

let parser = new DOMParser();

let doc = parser.parseFromString(html, "text/html");

let title = doc.querySelector("title")?.innerText || "No title found";

let description = doc.querySelector('meta[name="description"]')?.content || "No description found";

let keywords = doc.querySelector('meta[name="keywords"]')?.content || "No keywords found";

return { title, description, keywords };

})

.catch(error => {

console.error("Error fetching metadata:", error);

return { title: "Error", description: "Error", keywords: "Error" };

});

}