Análise de 3 sites de preferência no mesmo ramo de atuação:

Para este trabalho, foram escolhidos três sites de academias amplamente conhecidas no mercado nacional: Smart Fit, Selfit Academias e Bluefit. A escolha visa permitir uma comparação técnica entre abordagens distintas dentro do mesmo segmento. Utilizei três métodos de análise:

1. Google LightHouse, para avaliação de desempenho, acessibilidade e SEO.
2. Wappalyzer, para identificação das tecnologias utilizadas.
3. Inspeção direta do código-fonte, para analisar a estrutura de HTML semântico, CSS e JavaScript.

Lighthouse - Avaliação de Desempenho

https://www.smartfit.com.br/

Desempenho: 45

Acessibilidade: 76

Práticas Recomendadas: 74

SEO: 100

https://www.selfitacademias.com.br

Desempenho: apresentou erro

Acessibilidade: 95

Práticas Recomendadas: apresentou erro

SEO: 92

https://www.bluefit.com.br/

Desempenho: 91

Acessibilidade: 82

Práticas Recomendadas: 59

SEO: 100

Wappalyzer - Tecnologias Utilizadas

O **Wappalyzer** é uma ferramenta de análise de sites que identifica tecnologias utilizadas em uma página web. Ele funciona como uma extensão de navegador (Chrome, Firefox) e também possui API e plataforma online.

Principais tipos de análise fornecidos pelo Wappalyzer:

1. **SmartFit:**

E-commerce
- Plataforma: Magento
- Funcionalidade de Carrinho: Ativada

CMS (Sistema de Gerenciamento de Conteúdo)
- Croct

Desenvolvimento
- **Frameworks JavaScript:**
  - React
  - styled-components (v6.1.8)

- **Bibliotecas JavaScript:**
  - lit-html (v3.2.1)
  - lit-element (v4.1.1)
  - core-js (v3.32.2)
  - Preact
  - Splide
  - Lodash (v4.17.11)

- **Gráficos em JavaScript:**
  - Chart.js

- **Ferramenta de Build:**
  - Webpack

- **Linguagens de Programação**
  - PHP

- **Banco de Dados**
  - MySQL

- **Mapas**
  - Google Maps

- **Plataforma como Serviço (PaaS)**
  - Amazon Web Services (AWS)

- **Gerenciadores de Tags**
  - Google Tag Manager

- **Segurança**

    - HSTS (HTTP Strict Transport Security)

- **Retargeting e Personalização**
    - Criteo
    - Linx Impulse

- **RUM (Real User Monitoring)**
    - New Relic

- **Fontes**
    - Google Font API
    - Font Awesome (v4.5.0)

- **CDN (Content Delivery Network)**
    - GoCache
    - jsDelivr

- **Análise e Métricas**
    - Google Analytics (GA4)
    - Facebook Pixel (v2.9.202)
    - TikTok Pixel
    - Microsoft Clarity (v0.8.8)
    - Kwai Pixel

- **Publicidade**
    - Google Ads
    - Criteo
    - DoubleClick Floodlight
    - AppNexus
    - Microsoft Advertising
    - Rubicon Project

- **Testes A/B**
    - GrowthBook (v1.4.1)

- **Outros Recursos**
    - Open Graph (para redes sociais)

2. **SelfitAcademias:**

- **Mapas**
    - Google Maps

- **Reprodutores de Vídeo**
    - YouTube

- **Servidores e Proxies**
  - Servidor Web: Nginx (v1.22.1)
  - Proxy Reverso: Nginx (v1.22.1)

- **Frameworks JavaScript**
  - Vue.js

- **Gerenciadores de Tags**
  - Google Tag Manager

- **Segurança**
  - reCAPTCHA (proteção contra bots e spam)

- **Atendimento ao Cliente**
  - WhatsApp Business Chat (atendimento direto pelo WhatsApp)

- **Bibliotecas JavaScript**
  - jQuery (v2.2.0)
  - core-js (v3.32.2)
  - Slick
  - Swiper

- **Análise e Métricas**
  - Google Analytics (GA4)
  - Facebook Pixel (v2.9.202)
  - TikTok Pixel
  - Microsoft Clarity (v0.8.9)

- **Publicidade**
  - DoubleClick Floodlight

- **Outros Recursos**
  - Open Graph (para compartilhamento social)
  - Webpack (ferramenta de build de módulos)

3. **Bluefit:**

- **Construtores de Página (Page Builders)**
  - Webflow (v1.6.0)

- **E-commerce**
  - Webflow Ecommerce
  - Funcionalidade de carrinho: Ativada

- **Gerenciadores de Tags**
  - Google Tag Manager

- **Bibliotecas JavaScript**
  - jQuery (v3.5.1)
  - core-js (v3.6.4)
  - Swiper
  - Goober

- **CDN (Content Delivery Network)**
  - Unpkg
  - jsDelivr
  - Cloudflare

- **Análise e Métricas**
  - TikTok Pixel

- **Publicidade**
  - Google Ads
  - DoubleClick Floodlight

- **Outros Recursos**
  - Webpack (empacotador de módulos)
  - Babel (transpilador JavaScript)
  - HTTP/3 (protocolo de rede moderno)

Análise Estrutural do Código

1. **Smartfit**

- **Estrutura HTML**
- Usa HTML5 com algumas marcações semânticas, mas a maior parte é gerada via React.
- Tags identificadas: <html>, <head>, <body>, <header>, <main>, <section>, <footer>, <script>, <style>, , <a>, <button>, <img>, , <meta>, <link>, <title>
  Muitos elementos dinâmicos com data-*, s e identificadores para controle via JS.
- **CSS**
- Utiliza styled-components, uma solução CSS-in-JS do React.
- O CSS não é separado por arquivos .css clássicos, mas embutido via JavaScript.
- Há uso de variáveis, alto grau de responsividade e estilos modernos.
- Utiliza também Google Fonts e Font Awesome.

- **JavaScript**
- Framework principal: React.
- Usa Chart.js para gráficos.
- Múltiplas bibliotecas: core-js, lit-html, lit-element, Preact, Splide, Lodash.
- Arquitetura SPA com forte dependência de bibliotecas e renderização client-side.
- Uso de Webpack para empacotamento.
- Alto uso de trackers e pixels (GA4, Facebook, TikTok, etc.).

- **Arquitetura**
- SPA (Single Page Application).
- Estrutura baseada em componentes com React.
- Integração com Magento e Croct como CMS/headless CMS.

- **Conclusão:**
  Arquitetura moderna e dinâmica, porém, com desempenho prejudicado por alto carregamento inicial.

2. Selfit

- **Estrutura HTML**
- HTML bem formatado e semântico, embora mais tradicional.
- Uso correto de <html>, <head>, <body>, <header>, <nav>, <main>, <section>, <article>, <footer>, <form>, <input>, <label>, <img>, <a>, <button>, , <ul>, <li>.
- Estrutura clara, com menus, seções e formulários bem identificáveis.
- Utiliza marcação para Open Graph e SEO.

- **CSS**
- CSS tradicional separado por arquivos.
- Modularizado em arquivos como style.css e responsive.css.
- Boa aplicação de responsividade, provavelmente usando media queries manuais.
- Bibliotecas como **Slick** e **Swiper** incluem seus próprios estilos.

- **JavaScript**
- Mistura de Vue.js e jQuery, sem SPA completa.
- Bibliotecas JS adicionais:
  - jQuery (v2.2.0): usado para manipulação de DOM clássica.
  - core-js (v3.32.2): polyfills para compatibilidade.
  - Slick e Swiper (carrosséis).

  Componentes dinâmicos funcionam de forma eficiente.

- **Arquitetura**
- Site híbrido (mistura partes estáticas com componentes Vue).
- Renderização predominantemente no servidor.
- Integração leve com serviços externos (chat, Google Maps, pixels).

- **Conclusão:**

  Código tradicional, bem estruturado e acessível. Menor complexidade técnica, porém eficiente.

3. Bluefit

Estrutura HTML

- Gerado automaticamente pelo Webflow, com predominância de s e classes genéricas.
- Tags identificadas: <html>, <head>, <body>, , <img>, <a>, , <script>, <link>, <meta>, <style>.
- Estrutura modular, com páginas geradas a partir de templates.

- **CSS**
  Utilização de classes utilitárias do Webflow.
  Boa responsividade, porém com menor controle semântico sobre os estilos.

- **JavaScript**
- Sem frameworks modernos.
- Uso de bibliotecas básicas e scripts para interações simples.
- Uso de bibliotecas:
  - jQuery (v3.5.1)
  - core-js (v3.6.4)
  - Swiper
  - Goober
- Webpack e Babel presentes, indicando estrutura moderna de build.

- **Arquitetura**
- Site estático com partes dinâmicas renderizadas no cliente.
- Construtor visual (Webflow) como principal ferramenta de desenvolvimento.
- Menor complexidade comparado às demais.

- **Conclusão:**

  Site estático, leve e bem estruturado para performance. Perde em semântica, mas ganha em velocidade e simplicidade.

Critério	Smart Fit	Selfit	Bluefit
Desempenho (Lighthouse)	45	N/A (erro)	91

Critério	Smart Fit	Selfit	Bluefit
Framework JS	React	Vue.js	Nenhum
Arquitetura	SPA (React)	Híbrido	Estático
CMS/E-commerce	Magento/Croct	Nginx/Vue	Webflow
CSS	styled-components	CSS tradicional	Goober/Webflow
CDN	GoCache, jsDelivr	jsDelivr	Cloudflare
HTML Semântico	Parcial	Bem aplicado	Fraco

Conclusão:

Com base na análise realizada, observa-se que o site da Smart Fit possui uma arquitetura moderna e robusta, porém com desempenho abaixo do ideal devido ao carregamento excessivo. A Selfit adota uma abordagem híbrida eficiente, mas apresenta instabilidades de análise automática. A Bluefit, por sua vez, aposta em soluções visuais via Webflow, entregando bom desempenho e simplicidade. Cada uma adota estratégias distintas de front-end conforme sua proposta e público-alvo.

Referências

DUCKETT, Jon. *HTML e CSS: design e construção de sites*. Rio de Janeiro: Alta Books, 2014. GOOGLE. *Lighthouse – Página de desenvolvedores*. Disponível em: https://developer.chrome.com/docs/lighthouse. Acesso em: 14 jun. 2025.

MOZILLA FOUNDATION. *MDN Web Docs*. Disponível em: https://developer.mozilla.org. Acesso em: 14 jun. 2025.

UNICESUMAR. Programação Front-End: HTML, CSS e JavaScript. Maringá: UniCesumar, 2023. 168 p. Material didático do curso de Engenharia de Software.

World Wide Web Consortium. Disponível em: https://www.w3.org. Acesso em: 14 jun. 2025.

WAPPALYZER. *Technology Profiler*. Disponível em: https://www.wappalyzer.com. Acesso em: 14 jun. 2025.