Análise do Sistema de Vagas - WorkUp

2. Página de Detalhamento (detalhamento.html)

2.1 Estrutura de Detalhamento

A página de detalhamento mostra informações completas sobre uma vaga específica:

```
<div class="vaga-detalhe">
    <div class="vaga-header">
        <h1 id="vaga-titulo"></h1>
    </div>

    <div class="empresa-info">
        <img id="empresa-logo" class="empresa-logo">
    </div>

    <!-- Carrossel de benefícios -->
    <div class="beneficios-carousel">
        <!-- Informações detalhadas -->
    </div>
</div>
```

2.2 Gerenciamento de Estado

O sistema utiliza o **localStorage** para manter as informações da vaga selecionada entre páginas:

Obs: O que é localStorage?

LocalStorage é uma API da Web que permite armazenar dados no navegador do usuário de forma persistente. É uma forma de armazenamento chave-valor onde tanto a chave quanto o valor são armazenados como strings.

LocalStorage é uma ferramenta poderosa para:

- Manter estado entre páginas
- Armazenar preferências do usuário
- Cache de dados pequenos

- Melhorar a experiência do usuário

No projeto WorkUp, é utilizado principalmente para manter os dados da vaga selecionada durante a navegação entre as páginas de listagem e detalhamento, proporcionando uma experiência fluida ao usuário.

```
function redirecionarDetalhamento(idVaga) {
    const vaga = vagas.find(v => v.id === idVaga);
    if (vaga) {
        localStorage.setItem('vagaDetalhada', JSON.stringify(vaga));
        window.location.href = 'detalhamento.html';
    }
}
```

2.3 Carrossel de Benefícios

Um componente interativo que exibe os benefícios da vaga:

```
function moveCarousel(direction) {
    if (isAnimating) return;

    const carouselItems = document.querySelectorAll('.carousel-item');
    const maxIndex = Math.max(0, carouselItems.length - itemsPerPage);

    isAnimating = true;

    if (direction === 'prev') {
        currentIndex = currentIndex > 0 ? currentIndex - 1 : maxIndex;
    } else {
        currentIndex = currentIndex < maxIndex ? currentIndex + 1 : 0;
    }

    updateCarouselDisplay();
}
```

Validações

```
function validarDadosVaga(vaga) {
    const requiredFields = ['id', 'titulo', 'empresa', 'localizacao'];

    for (const field of requiredFields) {
        if (!vaga[field]) {
            throw new Error(`Campo obrigatório ausente: ${field}`);
        }
    }

    if (vaga.beneficios && !Array.isArray(vaga.beneficios)) {
        throw new Error('Formato inválido para benefícios');
    }

    return true;
}
```

Otimizações de Performance

Lazy Loading

```
function initializePage() {
    // Carrega componentes essenciais primeiro
    loadCriticalContent().then(() => {
        // Carrega conteúdo secundário
        loadSecondaryContent();
        // Inicializa interatividade
        setupInteractions();
    });
}

function loadCriticalContent() {
    return Promise.all([
        loadVagaBasicInfo(),
        loadEmpresaLogo()
    ]);
}

function loadSecondaryContent() {
    // Carrega carrossel de benefícios
    import('./carousel.js').then(module => {
        module.initCarousel();
    });
}
```

Conclusão

A página de detalhamento demonstra uma implementação robusta focada em:
- Experiência do Usuário: Interface intuitiva e responsiva
- Performance: Carregamento otimizado e interações suaves
- Confiabilidade: Tratamento abrangente de erros
- Manutenibilidade: Código modular e bem estruturado
- Acessibilidade: Navegação por teclado e estrutura semântica