

src/components/AppSidebar.vue

```
1 <script setup>
2 import { ref, computed, watch } from 'vue';
3 import LocationItem from './LocationItem.vue';
4
5 const props = defineProps({
6   isOpen: Boolean,
7   allLocais: Array,
8   floors: Array,
9   currentFloorId: String,
10  selectedLocalId: [Number, String, null],
11  loading: Boolean,
12  error: String
13 });
14
15 const emit = defineEmits(['update:currentFloorId', 'select-local', 'close']);
16
17 const searchTerm = ref('');
18
19 // Filtra locais baseado na busca E opcionalmente no andar atual
20 // Decisão: Mostrar sempre locais do andar atual + resultados da busca em outros andares
21 const filteredLocais = computed(() => {
22   const term = searchTerm.value.toLowerCase().trim();
23   if (!term) {
24     // Sem busca, mostra apenas os do andar atual
25     return props.allLocais.filter(local => local.andar === props.currentFloorId);
26   } else {
27     // Com busca, mostra os que batem com a busca (em qualquer andar)
28     return props.allLocais.filter(local =>
29       local.nome.toLowerCase().includes(term)
30     );
31   }
32 });
33
34
```

```
35 | const internalCurrentFloor = ref(props.currentFloorId);
36 |
37 | // Observa mudança externa no currentFloorId
38 | watch(() => props.currentFloorId, (newVal) => {
39 |     internalCurrentFloor.value = newVal;
40 | });
41 |
42 |
43 | const changeFloor = () => {
44 |     //searchTerm.value = ''; // Limpa busca ao trocar de andar? Opcional.
45 |     emit('update:currentFloorId', internalCurrentFloor.value);
46 | };
47 |
48 | const handleSelectLocal = (local) => {
49 |     emit('select-local', local);
50 |     // Opcional: Fechar sidebar ao selecionar?
51 |     // emit('close');
52 | };
53 |
54 | </script>
55 |
56 | <template>
57 |     <div class="sidebar" :class="{ 'is-open': isOpen }">
58 |         <h1>Local</h1>
59 |
60 |         <div class="floor-selector">
61 |             <select v-model="internalCurrentFloor" @change="changeFloor">
62 |                 <option v-for="floor in floors" :key="floor.id" :value="floor.id">
63 |                     {{ floor.name }}
64 |                 </option>
65 |             </select>
66 |         </div>
67 |
68 |         <input v-model="searchTerm" class="search-box" placeholder="Buscar local..." />
69 |
70 |         <div v-if="error" class="error" style="position: static; transform: none; margin-bottom: 10px;">
71 |             {{ error }}
```

```

72     </div>
73
74     <div v-if="loading && !allLocais.length">Carregando locais...</div>
75
76     <div v-if="!loading && !filteredLocais.length && searchTerm">Nenhum local encontrado para "{{ searchTerm }}"</div>
77     </div>
78     <div v-if="!loading && !filteredLocais.length && !searchTerm">Nenhum local neste andar.</div>
79
80
81     <LocationItem v-for="local in filteredLocais" :key="local.id" :local="local"
82         :isSelected="selectedLocalId === local.id" @select="handleSelectLocal" />
83
84     </div>
85 </template>
86
87 <style scoped>
88 /* Estilos específicos da sidebar já estão em main.css */
89 /* Se precisar de algo MUITO específico, adicione aqui
90 .sidebar {
91     /* Herdado de main.css
92 }
93
94 .sidebar.is-open {
95     /* Herdado de main.css
96 }*/
97
98 /* Ajuste para erro dentro da sidebar */
99 .error {
100     position: static;
101     /* Sobrescreve o absoluto do global */
102     transform: none;
103     margin-bottom: 15px;
104     width: auto;
105     /* Garante que não fique gigante */
106     left: auto;
107     top: auto;
108     text-align: left;

```

```
109 |     padding: 10px;  
110 |     /* Menor padding */  
111 | }  
112 | </style>
```