src/components/MapDisplay.vue

```vue
<script setup>
import { ref, computed, watch, onMounted, onUnmounted } from 'vue';

const props = defineProps({
  mapImageUrl: String,
  locaisOnFloor: Array,
  userPosition: Object, // { x, y } | null
  routeSegments: Array, // Array de { x, y, length, angle }
  debugWaypoints: Array, // Array de waypoints para mostrar
  selectedLocalId: [Number, String, null],
  popupInfo: Object, // { visible, x, y, text }
  mapScale: Number,
  loading: Boolean,
});

const emit = defineEmits([
    'map-click',
    'marker-click',
    'marker-mouseenter',
    'marker-mouseleave',
    'update:mapDimensions' // Emite as dimensões do mapa para o pai
]);

const mapImageRef = ref(null); // Referência para o elemento .map-image
const mapDimensions = ref({ width: 0, height: 0 }); // Dimensões em pixels

const mapStyle = computed(() => ({
  backgroundImage: `url(${props.mapImageUrl || ''})`,
  transform: `scale(${props.mapScale || 1})`,
  // Adicionar transform-origin se não estiver no CSS global
  // transformOrigin: 'center center'
}));

const handleMapClick = (event) => {
```

```
  // Emite o evento original e a referência do elemento para cálculo no pai/composable
  emit('map-click', event, mapImageRef.value?.getBoundingClientRect());
};

const selectLocalFromMarker = (local) => {
    emit('marker-click', local);
};

const showPopupOnEnter = (local) => {
    emit('marker-mouseenter', local);
};

const hidePopupOnLeave = () => {
    emit('marker-mouseleave');
};

// Observador para atualizar dimensões quando o elemento do mapa for montado ou redimensionado
const resizeObserver = ref(null);
onMounted(() => {
  if (mapImageRef.value) {
    resizeObserver.value = new ResizeObserver(entries => {
      for (let entry of entries) {
        mapDimensions.value = {
          width: entry.contentRect.width,
          height: entry.contentRect.height,
        };
         // Emite as dimensões atualizadas para o componente pai/composable
         emit('update:mapDimensions', mapDimensions.value);
      }
    });
    resizeObserver.value.observe(mapImageRef.value);
     // Define dimensões iniciais
     const rect = mapImageRef.value.getBoundingClientRect();
     mapDimensions.value = { width: rect.width, height: rect.height };
     emit('update:mapDimensions', mapDimensions.value);

  }
```

```
});

onUnmounted(() => {
  if (resizeObserver.value && mapImageRef.value) {
    resizeObserver.value.unobserve(mapImageRef.value);
  }
});

// Recalcula dimensões se a escala mudar (embora o ResizeObserver deva pegar)
watch(() => props.mapScale, () => {
    if (mapImageRef.value) {
        const rect = mapImageRef.value.getBoundingClientRect();
        mapDimensions.value = { width: rect.width / props.mapScale, height: rect.height / props.mapScale };
         emit('update:mapDimensions', mapDimensions.value);
    }
}, { immediate: true });


</script>

<template>
  <div class="map-container">
    <div
      class="map-image"
      :style="mapStyle"
      ref="mapImageRef"
      @click="handleMapClick"
    >
      <div
        v-for="local in locaisOnFloor"
        :key="`loc-${local.id}`"
        class="location-marker"
        :style="{ left: `${local.x}%`, top: `${local.y}%` }"
        @click.stop="selectLocalFromMarker(local)"
        @mouseenter="showPopupOnEnter(local)"
        @mouseleave="hidePopupOnLeave"
        :title="local.nome"
```

```
        ></div>

        <div
            v-if="userPosition"
            class="user-marker"
            :style="{ left: `${userPosition.x}%`, top: `${userPosition.y}%` }"
            title="Sua localização estimada"
        ></div>

        <div v-if="routeSegments && routeSegments.length > 0">
            <div
                v-for="(segment, index) in routeSegments"
                :key="`route-${index}`"
                class="path-line"
                :style="{
                    left: `${segment.x}%`,
                    top: `${segment.y}%`,
                    width: `${segment.length}px`,
                    transform: `rotate(${segment.angle}deg)`
                }"
            ></div>
        </div>

         <div v-if="debugWaypoints && debugWaypoints.length > 0">
            <div
                v-for="wp in debugWaypoints"
                :key="`wp-${wp.id}`"
                class="waypoint-marker"
                :style="{ left: `${wp.x}%`, top: `${wp.y}%` }"
                :title="`Waypoint: ${wp.id}`"
            ></div>
        </div>

        <div
            v-if="popupInfo && popupInfo.visible"
            class="location-popup"
            :style="{ left: `${popupInfo.x}%`, top: `${popupInfo.y}%` }"
```

```
      >
        {{ popupInfo.text }}
      </div>

    </div>

    <div v-if="loading" class="loading">
      Carregando mapa...
    </div>

  </div>
</template>

<style scoped>
/*
/* Estilos específicos do mapa/marcadores já estão em main.css
.map-container {

}
.map-image {

}
.waypoint-marker {

}*/

/* Garante que o loading fique sobre o mapa*/
.loading {
    z-index: 40; /* Acima do mapa, abaixo dos controles/popup */
}
</style>
```