src/layouts/MapLayout.vue

```vue
<template>
  <div class="app-container">
    <button class="menu-toggle-button" @click="toggleMenu" title="Abrir/Fechar Menu">☰</button>

    <AppSidebar
      :is-open="isMenuOpen"
      :all-locais="allLocais"
      :floors="floors"
      :current-floor-id="currentFloorId"
      :selected-local-id="selectedLocal?.id"
      :loading="loading.locais"
      :error="error.locais"
      @update:currentFloorId="currentFloorId = $event"
      @select-local="handleLocalSelect"
      @close="closeMenu"
    />

    <div class="main-content" style="flex-grow: 1; position: relative;">
      <MapDisplay
        :map-image-url="currentFloorMap"
        :locais-on-floor="locaisAtualAndar"
        :user-position="userPosition"
        :route-segments="routeSegments"
        :debug-waypoints="debugWaypoints"
        :selected-local-id="selectedLocal?.id"
        :popup-info="popupInfo"
        :map-scale="mapScale"
        :loading="loading.map"
        @map-click="handleMapClick"
        @marker-click="handleLocalSelect"
        @marker-mouseenter="showPopup"
        @marker-mouseleave="hidePopup"
        @update:map-dimensions="mapDimensions = $event"
      />

      <MapControls
        @zoom-in="zoomIn"
        @zoom-out="zoomOut"
        @update-location="triggerAutoLocation"
        @set-location-manual="triggerManualLocationMode"
      />

      <div v-if="overallLoading && !displayError" class="loading">
        <span v-if="isGettingLocation">Obtendo localização...</span>
        <span v-else-if="loading.locais || loading.map">Carregando dados...</span>
      </div>

      <div v-if="displayError" class="error">{{ displayError }}</div>
      <div v-if="settingUserLocationMode" class="error" style="top: 10%; background: lightblue; color: black; border: 1px solid blue;">
        {{ error.geolocation }}
      </div>

      <!-- Aqui entram as rotas filhas -->
      <router-view />
    </div>
  </div>
</template>

<script setup>
import { ref, computed, onMounted, watch } from 'vue';
import AppSidebar from '@/components/AppSidebar.vue';
import MapDisplay from '@/components/MapDisplay.vue';
import MapControls from '@/components/MapControls.vue';
import { useGeolocation } from '@/composables/useGeolocation.js';
import { useMapInteraction } from '@/composables/useMapInteraction.js';
import { useMapRouting } from '@/composables/useMapRouting.js';
import { fetchLocais, getFloors } from '@/services/LocationService.js';

const allLocais = ref([]);
const selectedLocal = ref(null);
```

```javascript
const currentFloorId = ref('terreo');
const floors = ref([]);
const isMenuOpen = ref(false);
const loading = ref({ locais: true, map: true });
const error = ref({ locais: null, routing: null, geolocation: null });
const mapDimensions = ref({ width: 0, height: 0 });

const {
  userPosition,
  geolocationError,
  isGettingLocation,
  getUserLocationAuto,
  setUserLocationManually
} = useGeolocation();

const {
  mapScale,
  popupInfo,
  settingUserLocationMode,
  zoomIn,
  zoomOut,
  showPopup,
  hidePopup,
  handleMapClick: handleMapClickInteraction,
  enableSetUserLocationMode
} = useMapInteraction();

const {
  routeSegments,
  hasRoute,
  debugWaypoints,
  routingError: routeCalcError,
  calculateRoute
} = useMapRouting(userPosition, selectedLocal, currentFloorId, mapScale, mapDimensions);

const currentFloorData = computed(() => floors.value.find(f => f.id === currentFloorId.value));
const currentFloorMap = computed(() => currentFloorData.value?.image || '');
const locaisAtualAndar = computed(() => allLocais.value.filter(local => local.andar === currentFloorId.value));
const overallLoading = computed(() => loading.value.locais || loading.value.map || isGettingLocation.value);
const displayError = computed(() => error.value.locais || error.value.geolocation || error.value.routing);

const toggleMenu = () => { isMenuOpen.value = !isMenuOpen.value; };
const closeMenu = () => { isMenuOpen.value = false; };

const loadInitialData = async () => {
  loading.value.locais = true;
  error.value.locais = null;
  try {
    floors.value = getFloors();
    if (!floors.value.some(f => f.id === currentFloorId.value)) {
      currentFloorId.value = floors.value[0].id;
    }
    allLocais.value = await fetchLocais();
  } catch (err) {
    error.value.locais = 'Falha ao carregar dados.';
  } finally {
    loading.value.locais = false;
    loading.value.map = false;
  }
};

const handleLocalSelect = (local) => {
  selectedLocal.value = local;
  if (local.andar !== currentFloorId.value) {
    currentFloorId.value = local.andar;
  }
  closeMenu();
};

const handleMapClick = (event, mapRect) => {
  handleMapClickInteraction(
    event,
    mapRect,
    (x, y) => {
      setUserLocationManually(x, y);
      error.value.geolocation = null;
    },
    () => {
      if (isMenuOpen.value && !event.target.closest('.menu-toggle-button')) {
        closeMenu();
      }
    }
  );
};

const triggerAutoLocation = () => {
  getUserLocationAuto();
  closeMenu();
};

const triggerManualLocationMode = () => {
  enableSetUserLocationMode();
  error.value.geolocation = 'Clique no mapa para definir sua localização.';
  closeMenu();
};

watch(geolocationError, (newError) => {
  if (!settingUserLocationMode.value) error.value.geolocation = newError;
});
watch(routeCalcError, (newError) => { error.value.routing = newError; });
watch(currentFloorId, (newFloor) => { if (selectedLocal.value?.andar !== newFloor) hidePopup(); });

onMounted(() => {
  loadInitialData();
  getUserLocationAuto();
});
</script>

<!-- estilos podem ser os mesmos de App.vue -->


```