```
import { ref } from 'vue';

export function useMapInteraction(initialScale = 1, minScale = 0.5, maxScale = 3, zoomStep = 1.2) {
  const mapScale = ref(initialScale);
  const popupInfo = ref({ visible: false, x: 0, y: 0, text: "" });
  const settingUserLocationMode = ref(false); // Novo estado para modo de definição manual

  const zoomIn = () => {
    mapScale.value = Math.min(mapScale.value * zoomStep, maxScale);
  };

  const zoomOut = () => {
    mapScale.value = Math.max(mapScale.value / zoomStep, minScale);
  };

  const showPopup = (local) => {
    if (local && typeof local.x === 'number' && typeof local.y === 'number') {
      popupInfo.value = { visible: true, x: local.x, y: local.y, text: local.nome };
    } else {
      console.warn("Tentativa de mostrar popup para local inválido:", local);
      popupInfo.value = { visible: false, x: 0, y: 0, text: "" };
    }
  };

  const hidePopup = () => {
    popupInfo.value.visible = false;
  };

  /**
   * Processa um clique no mapa. Pode fechar menu, definir localização manual, etc.
   * @param {MouseEvent} event - O evento de clique.
   * @param {HTMLElement|null} mapElement - O elemento do mapa para calcular o BoundingClientRect.
   * @param {Function} callbackSetLocation - Função para chamar ao definir localização manual.
   * @param {Function} callbackCloseMenu - Função para chamar se precisar fechar o menu.
   */
  const handleMapClick = (event, mapElement, callbackSetLocation, callbackCloseMenu) => {
```

```
    // 1. Tenta fechar o menu se clicar fora dele
    if (callbackCloseMenu) {
      callbackCloseMenu(); // App.vue decide se fecha ou não
    }

    // 2. Se estiver no modo de definir localização manual
    if (settingUserLocationMode.value) {
      // Verifica se o elemento do mapa está disponível
      if (!mapElement) {
        console.error("Elemento do mapa não disponível para calcular clique.");
        settingUserLocationMode.value = false; // Sai do modo
        return;
      }

      // Obtém o BoundingClientRect diretamente do elemento
      const mapRect = mapElement.getBoundingClientRect();
      if (!mapRect) {
        console.error("Map rect não disponível para calcular clique.");
        settingUserLocationMode.value = false; // Sai do modo
        return;
      }

      // Calcula coordenadas do clique relativas ao elemento da imagem do mapa
      const clickXRelative = event.clientX - mapRect.left;
      const clickYRelative = event.clientY - mapRect.top;

      // Converte coordenadas do clique para coordenadas relativas à imagem *não escalada*
      // Precisamos considerar a origem da transformação (centro) e a escala
      const originX = mapRect.width / 2;
      const originY = mapRect.height / 2;
      const scale = mapScale.value;

      // Posição na imagem como se não houvesse escala (em pixels, relativa ao canto sup esq)
      const imageX = (clickXRelative - originX) / scale + originX;
      const imageY = (clickYRelative - originY) / scale + originY;

      // Converte para porcentagem (0-100) relativa à dimensão da imagem
```

```
      const xPercent = (imageX / mapRect.width) * 100;
      const yPercent = (imageY / mapRect.height) * 100;

      // Chama a função de callback para efetivamente definir a localização
      if (callbackSetLocation) {
        // Passa as coordenadas calculadas em %
        callbackSetLocation(xPercent, yPercent);
      }

      settingUserLocationMode.value = false; // Sai do modo após definir
    }
    // 3. Lógica adicional de clique no mapa (ex: deselecionar local?) pode ser adicionada aqui
  };

  /** Ativa o modo de definir localização manual */
  const enableSetUserLocationMode = () => {
    settingUserLocationMode.value = true;
  };

  return {
    mapScale,                // ref(number)
    popupInfo,               // ref({ visible, x, y, text })
    settingUserLocationMode, // ref(boolean) - Indica se está esperando clique para definir local

    zoomIn,                  // function
    zoomOut,                 // function
    showPopup,               // function(local)
    hidePopup,               // function()
    handleMapClick,          // function(event, mapElement, callbackSetLocation, callbackCloseMenu)
    enableSetUserLocationMode,// function()
  };
}
```