Application LizMap pour le calcul de la distance minimale entre un point sélectionné et une gare SNCF spécifiée :

1. Saisir le nom de la gare et cliquer sur **Rechercher**.
2. Cliquer un point sur la carte.
3. Cliquer sur **Calculer l'itinéraire**.
4. Attendre environ 5 minutes en raison du traitement du serveur.
5. La gare trouvée sera rapprochée (**Figure 1**) et un itinéraire sera généré (**Figure 2**).
6. Cliquer sur un autre emplacement de la carte et appuyer à nouveau sur **Calculer l'itinéraire** – l'itinéraire sera recalculé (**Figure 3**).

*Le script **Itineraire.js** contient tous les commentaires explicatifs du code.*

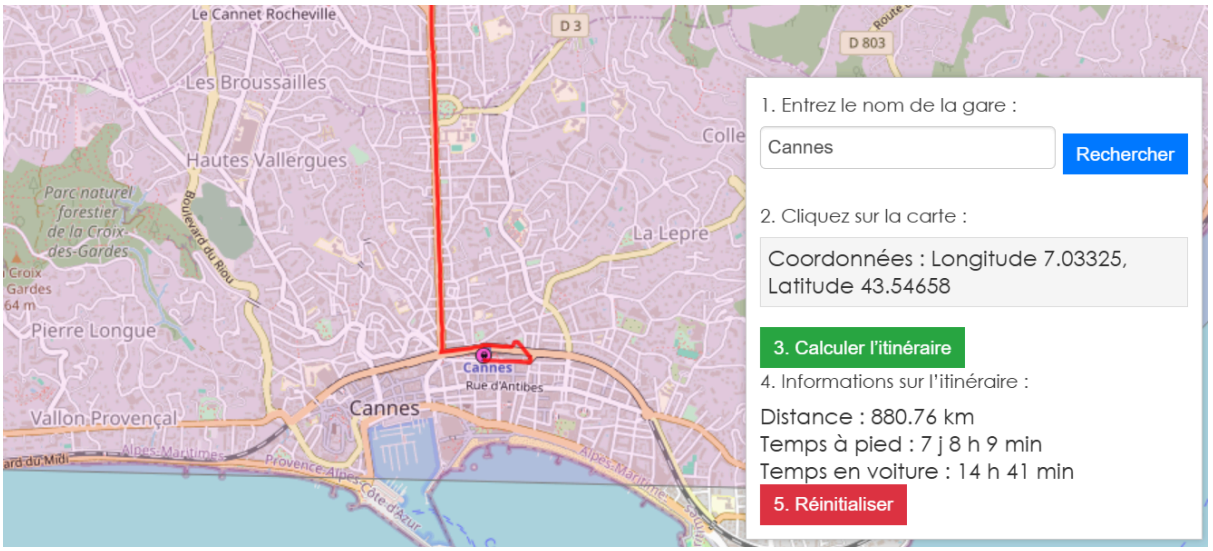

***Figure 1** – Zoom sur la gare sélectionnée*

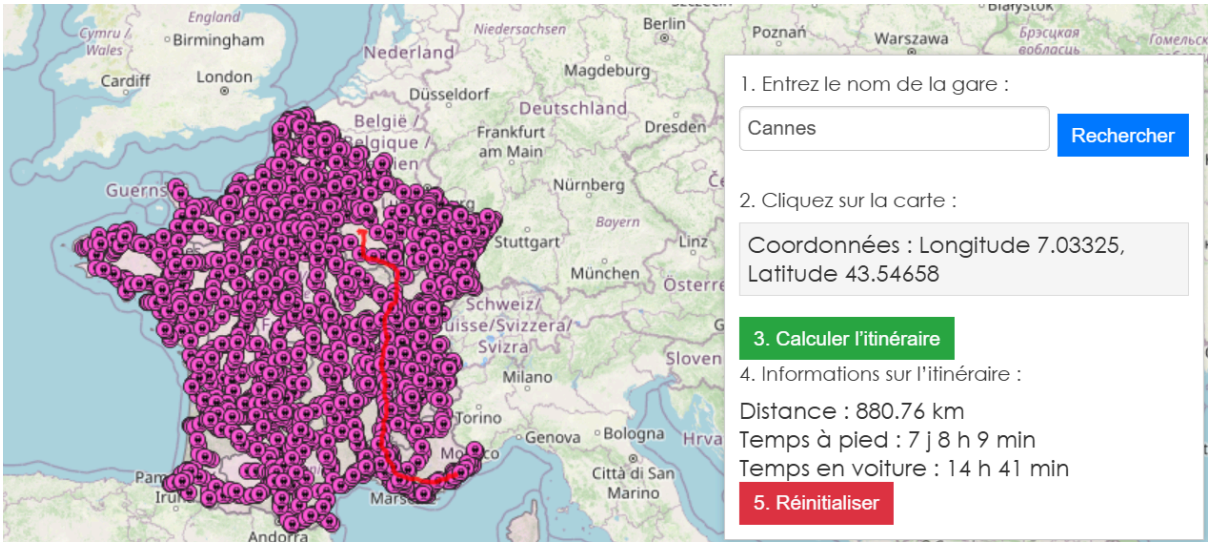

***Figure 2** – Itinéraire généré entre le point sélectionné et la gare*

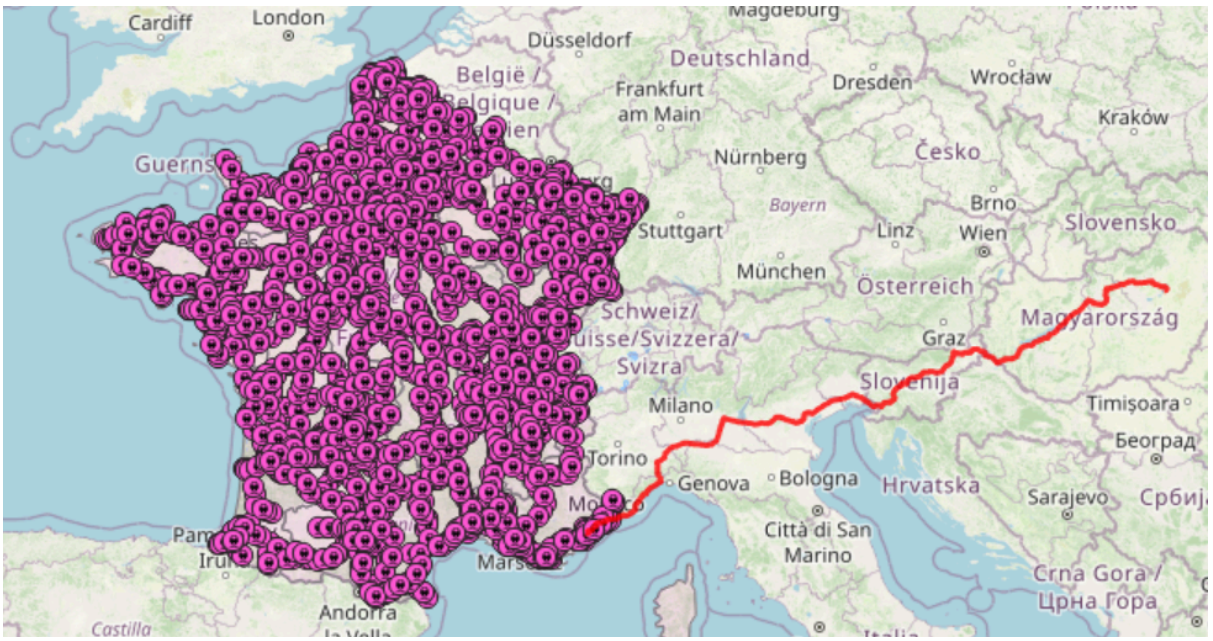

Figure 3 *– Recalcul de l'itinéraire après sélection d'un nouveau point*

Algorithme de création du widget

1. Création de l'interface au démarrage de LizMap
    - *Lors de l'événement 'uicreated' (création de l'interface), une fonction est déclenchée pour générer un widget personnalisé.*

2. Création du conteneur pour le widget utilisateur
    - *Un élément div (container) est créé et affiché au-dessus de la carte LizMap.*
    - *Son apparence est définie (position, fond, bordures, ombre et curseur).*
    - *Une fonctionnalité de déplacement du conteneur est ajoutée avec les événements mousedown, mousemove, mouseup.*

3. Ajout des champs pour l'entrée utilisateur
    - *Étape 1 : Saisie du nom de la gare*
      Un champ input est créé pour entrer le nom de la gare.
      Un bouton Rechercher est ajouté pour lancer la recherche.
    - *Étape 2 : Sélection d'un point sur la carte*
      Un texte Cliquez sur la carte et un bloc d'affichage des coordonnées sont affichés.
    - *Étape 3 : Calcul de l'itinéraire*
      Un bouton Calculer l'itinéraire est créé.
    - *Étape 4 : Affichage des informations de l'itinéraire*
      Des éléments sont créés pour afficher la distance et le temps de trajet (à pied, en voiture).

- *Étape 5 : Réinitialisation de la recherche*
  Un bouton Réinitialiser est ajouté pour réinitialiser les paramètres.

4. Ajout de toute l'interface à la page
   Le conteneur avec tous les éléments est ajouté au document.body, intégrant ainsi le widget dans LizMap.

Algorithme de gestion des boutons

1. Recherche de la gare

- Appui sur le bouton "Rechercher"
  Le texte du champ de saisie est lu.
  Les espaces sont supprimés et le texte est converti en minuscules.
  Si le champ est vide → un avertissement est affiché et la recherche est annulée.

- Envoi de la requête au serveur WFS
  fetch(wfsUrl) est exécuté, envoyant une requête HTTP à sigsurete.prod.st.sncf.fr.
  Les données reçues sont converties en JSON.

- Recherche de la gare dans les données
  Vérification si le nom saisi correspond au champ Nom dans la réponse du serveur.
  Si la gare est trouvée → ses coordonnées sont extraites.

- Centrage de la carte sur la gare trouvée
  Les coordonnées de la gare sont converties de WGS84 (EPSG:4326) vers le système de coordonnées de la carte.
  La carte est centrée sur la gare avec un zoom de 14.
  Si la gare n'est pas trouvée → un avertissement est affiché.

2. Sélection d'un point sur la carte

- L'utilisateur clique sur la carte.
- L'événement click est traité sur la carte LizMap.
- Les coordonnées du clic (longitude et latitude) sont récupérées.
- Elles sont converties en WGS84 (EPSG:4326).
- Les coordonnées sont enregistrées dans clickedPoint.
- Le texte affiché dans l'interface est mis à jour avec les coordonnées.

3. Calcul de l'itinéraire

- Appui sur le bouton "Calculer l'itinéraire"
  Vérification si une gare et un point sur la carte sont sélectionnés.
  Si non → un avertissement est affiché.
- Création de la requête vers l'API de routage
  L'URL de l'API OpenRouteService est définie.
  Un objet JSON contenant les coordonnées (selectedStation et clickedPoint) est créé.
  Une requête POST est envoyée avec fetch() pour récupérer les données.

- Traitement de la réponse de l'API
  Si le serveur ne répond pas → une erreur est affichée.
  Les itinéraires reçus sont analysés :
  - Si aucun itinéraire n'est trouvé → une erreur est affichée.
  - Si un itinéraire est trouvé → les informations suivantes sont calculées :
    Distance de l'itinéraire en kilomètres.
    Temps de marche (5 km/h).
    Temps en voiture (60 km/h).
  - Les champs de texte affichant les informations sont mis à jour.

- Affichage de l'itinéraire sur la carte
  La polyligne de l'itinéraire est décodée.
  Les coordonnées de l'itinéraire sont converties dans le système de coordonnées de la carte LizMap.
  Si un itinéraire était affiché auparavant, il est supprimé.
  Un nouvel itinéraire est créé (OpenLayers.Geometry.LineString).

4. Réinitialisation des données

- Appui sur le bouton "Réinitialiser"
  Les variables selectedStation et clickedPoint sont réinitialisées.
  Le champ de saisie de la gare est vidé.
  Les coordonnées affichées dans l'interface sont réinitialisées.
  L'itinéraire affiché sur la carte (routeLayer) est supprimé.

Fonctions auxiliaires

Fonction decodePolyline(encoded)

Décode une polyligne d'itinéraire encodée par l'API.
Convertit les données en un tableau de coordonnées [longitude, latitude].

Fonction formatTime(hours)

Convertit le temps en un format lisible (jours, heures, minutes).
Retourne une chaîne de texte comme "1 h 20 min" ou "30 min".