### **Fonctionnalités minimales (à faire absolument)**

#### **1. Page d’accueil avec liste des stations**

* Liste des stations stockées en base
* Affichage : nom, commune, coordonnées (lat/lon)
* Option de tri par département ou commune

#### **2. Carte interactive**

* Utilise Leaflet ou OpenLayers
* Affiche les **stations sur une carte**
* Clic sur un point = affiche les infos de la station + bouton “voir températures”

#### **3. Consultation dynamique des températures**

* Quand on clique sur une station :  
  + Appel de l’API temperature/chronique avec code\_station
  + Récupère les températures pour les 7 derniers jours
  + Affiche sous forme de tableau ou de **graphe (Chart.js)**

#### **4. Filtrage par date**

* Permet de choisir une période (ex: du 2024-01-01 au 2024-01-15)
* Requête personnalisée à l’API avec date\_debut\_mesure et date\_fin\_mesure

### **Fonctionnalités enrichies (optionnelles )**

#### **5. 📊 Graphique des températures**

* Courbe des températures sur 24h ou 7 jours
* Outils : Chart.js, ApexCharts...

#### **6. Recherche par commune ou code station**

* Barre de recherche intelligente
* Autocomplétion (optionnel)

#### **7. Ajout manuel ou import automatique des nouvelles stations**

* Bouton “Mettre à jour les stations” → appelle l’API referentiel/stations et met à jour la base locale

### **Fonctionnalités bonus (non obligatoires )**

#### **8. Connexion admin**

* Interface Flask avec login simple (si tu veux une section admin)
* Permet de recharger les données statiques

#### **9. Visualisation filtrée par département ou bassin**

* Permet de choisir une région, et affiche les stations concernées
* Interaction directe avec ta base PostgreSQL

#### **10. Interface utilisateur soignée (design responsive)**

* Affichage agréable sur mobile
* Un thème clair avec Bootstrap, Materialize ou Tailwind CSS