SIMPLON.CO

Cahier des charges

Affichage de la météo dans les transports en commun

MOHAMMADI ZABIULLAH

SOMMER

1. Contexte du projet

Le projet consiste à développer une interface météo destinée à être affichée sur les écrans d'information des stations de transports en commun et à bord des véhicules. Ces interfaces doivent être simples, efficaces, et garantir une mise à jour régulière des données pour les usagers des transports dans plusieurs villes de taille moyenne en France.

2. Objectifs du projet

- **Affichage en temps réel de la météo** : Les usagers des transports doivent avoir accès aux informations météorologiques actualisées.
- **Mise à jour horaire des données** : Les données météorologiques doivent être rafraîchies automatiquement toutes les heures.
- **Interface claire et simple** : L'interface doit être lisible et claire, sans surcharge d'informations.
- **Intégration dans les systèmes existants** : L'application doit pouvoir être intégrée dans la webview du système des écrans de la compagnie de transports.

3. Prérequis techniques

- **HTML/CSS** : Pour structurer et styliser l'interface de manière fonctionnelle et cohérente.
- **JavaScript** : Pour interagir avec l'API météo et actualiser les données.
- **API météo** : Choisir une API météo publique permettant d'obtenir des données fiables sur les conditions météorologiques des villes concernées.
- **Fichier de configuration** : Un fichier de configuration (JSON) devra permettre de spécifier la ville à afficher.

4. Fonctionnalités à développer

- **Récupération des données météo** : Utilisation de l'API météo pour obtenir des informations comme la température, la condition météorologique (ensoleillé, nuageux, pluie), et d'autres données pertinentes.
- **Affichage des données sur l'interface** : L'interface devra afficher de manière lisible les informations météorologiques de la ville configurée dans le fichier JSON.
- **Rafraîchissement des données** : Les données doivent être mises à jour automatiquement toutes les heures.
- **Configuration de la ville** : L'entreprise de transport définira la ville pour chaque écran à l'aide d'un fichier de configuration (conf.json).

5. Structure des fichiers

- **index.html** : Structure de la page HTML qui contient l'interface d'affichage.
- **style.css** : Fichier CSS pour la mise en forme de l'interface.
- **conf.json** : Fichier de configuration spécifiant la ville à afficher.
- **meteo.js** : Script JavaScript pour récupérer et afficher les données météorologiques.

6. Critères de qualité

- **Respect des consignes** : Toutes les consignes doivent être suivies de manière stricte.
- **Lisibilité des données** : Les informations météo doivent être clairement affichées et facilement compréhensibles par les utilisateurs.
- **Mise à jour des données** : Les données doivent se rafraîchir correctement toutes les heures, sans erreur.
- **Structure HTML fonctionnelle** : Le code HTML doit être propre, logique, et bien organisé.

7. Livrables

- Code source complet du projet (index.html, style.css, conf.json, meteo.js) déposé sur un dépôt Git en ligne (GitHub, GitLab, etc.).
- Documentation du projet détaillant les étapes de développement, les choix techniques, et la méthode de mise à jour des données météo.

8. Délais et planning

- **Durée estimée** : 1 à 2 journées.
- **Date limite de livraison** : À définir en fonction de l'avancement et des contraintes spécifiques du projet.

9. Tests et validation

- **Test de la récupération des données** : Assurer que les données météorologiques sont correctement récupérées à partir de l'API.
- **Test du rafraîchissement automatique** : Vérifier que les données se mettent à jour toutes les heures.
- **Test d'affichage** : Vérifier que l'affichage des données est clair et lisible, même sur des écrans de taille différente.