Cahier des charges - Application de Météo

1. Introduction :

L'objectif de ce projet est de développer une application en Python et SQL permettant aux utilisateurs d'obtenir des informations météorologiques actuelles, passées (d'hier) et futures (demain) pour une ville spécifiée. L'application aura une interface conviviale avec trois boutons cliquables correspondant aux options suivantes : Météo courante, Météo précédente, Météo demain.

2. Fonctionnalités principales :

Barre de recherche : Une barre de recherche permettra à l'utilisateur de saisir le nom de la ville pour laquelle il souhaite obtenir des informations météorologiques.
Boutons cliquables : Trois boutons distincts (Météo courante, Météo précédente, Météo demain) permettront à l'utilisateur de sélectionner la période pour laquelle il souhaite obtenir des données météorologiques.
Affichage des données : Les informations météorologiques, y compris la température, l'humidité, le ciel, et l'indice de protection solaire (IPA), seront affichées de manière claire et lisible à l'utilisateur.

3. Interface Utilisateur :

L'interface utilisateur doit être intuitive et conviviale.
La barre de recherche doit être bien visible et permettre une saisie facile du nom de la ville.
Les boutons Météo courante, Météo précédente et Météo demain doivent être clairement étiquetés.

4. Technologie :

Utilisation du langage de programmation Python et SQL.
Utilisation de bibliothèques tierces pour récupérer les données météorologiques (par exemple, requests pour les requêtes HTTP à une API météo).

5. Intégration API météo :

Choix d'une API météo fiable et stable.
Implémentation des appels API pour récupérer les données météorologiques en temps réel, d'hier et pour demain.

6. Affichage des données :

Les données météorologiques doivent être présentées de manière lisible.
Utilisation de formats clairs et compréhensibles pour la température (en degrés Celsius, par exemple), l'humidité (en pourcentage), le ciel (clair, nuageux, pluvieux, etc.) et l'IPA.

7. Gestion des erreurs :

Gestion des cas où la ville recherchée n'est pas trouvée.
Gestion des erreurs liées aux appels d'API.

8. Documentation :

Documentation claire du code source, expliquant le fonctionnement global de l'application et les différentes fonctions.
Instructions d'utilisation pour l'utilisateur final.

9. Tests :

Réalisation de tests unitaires pour chaque fonctionnalité de l'application.
Tests d'intégration pour garantir le bon fonctionnement global de l'application.

10. Sécurité :

Assurer la sécurité des données, en particulier lors de l'utilisation d'une API externe.

11. Livrables :

Code source complet de l'application.
Documentation technique et d'utilisation.
Rapport de tests.

12. Planning :

Définir un planning détaillé avec des étapes clés, des dates limites et des jalons pour le développement, les tests et la livraison du projet.

13. Support et Maintenance :

Prévoir une période de support après la livraison pour corriger les bugs éventuels ou faire des ajustements mineurs.

14. Validation :

Validation finale par le client avant la clôture du projet.