Procès-Verbal de test : UC Consulter les données

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Élément testé : | | Ces tests seront effectués sur les méthodes de la classe CAPI, ainsi que sa connexion jusqu’à la classe CDatabase, permettant de vérifier le cas d’utilisation « Consulter les données » Entier | | | | | | |
| Objectif du test : | | Ce test vérifiera les différentes possibilités de requête de l’API avec différentes données de test pour vérifier du bon renvoi des données ou d’un code d’erreur. | | | | | | |
| Nom du testeur : | | Théo Bourgoin | | | Date : | | 20/05/2025 | |
| Moyens mis en œuvre : | | Logiciel : API-Rest | | Matériel : Machine virtuelle hôte | | Outil de développement : Node JS & Express. | | |
| Procédure du test : | | | | | | | | |
| Id | Description du vecteur de test | | Résultat attendu | | Résultat obtenu  (voir le document de plan de test sur le github pour avoir l’échantillon exact des données) | | | Validation (O/N) |
| 1 | On envoie une requête http correcte au serveur et on reçoit les données attendues. Le fonctionnement est correct pour les différentes requêtes (soit plusieurs numéros de test).  Requête http : http://127.0.0.1:8080/SmartTerritories/dataFilter?Capteur=Roseraie&TypeDeDonnee=Temperature&Date=2025-03-10 15 | | Données correspondant aux critères ainsi que le code http 200 : « status success ». | | Code http reçu : 200 | | | O |
| 2.1 | On envoie plusieurs fois le même élément de requête (ex : TypeDeDonnee) dans une seule et même requête.  Requête http : http://127.0.0.1:8080/SmartTerritories/dataFilter?TypeDeDonnee=Temperature&Capteur=Saint-Serge&TypeDeDonnee=NO2 | | Données et code http 200 : « status success ». | | Code http reçu : 200 | | | O |
| 2.2 | Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFilter?Capteur=Roseraie&TypeDeDonnee=Temperature&Capteur=Saint-Serge | | Données et code http 200 : « status success ». | | Code http reçu : 200 | | | O |
| 2.3 | Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFilter?Date=2025-03-10 15&TypeDeDonnee=Temperature&Date=2025-03-18 12 | | Données et code http 200 : « status success ». | | Code http reçu : 200 | | | O |
| 2.4 | Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFilter?DateDebut=2025-03-10 15&TypeDeDonnee=Temperature&DateDebut=2025-04-25 10 | | Message d’erreur : « Il ne peut pas y avoir plusieurs dates de début. », et code http 400. | | Code http reçu : 400 | | | O |
| 2.5 | Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFilter?DateFin=2025-03-10 15&TypeDeDonnee=Temperature&DateFin=2025-04-25 10 | | Message d’erreur : « Il ne peut pas y avoir plusieurs dates de fin. », et code http 400. | | Code http reçu : 400 | | | O |
| 3 | On envoie une requête qui ne contient aucun critère.  Requête http : http://127.0.0.1:8080/SmartTerritories/dataFilter?  ET/OU  Requête http : http://127.0.0.1:8080/SmartTerritories/dataFilter | | Retour statut http 303 : see other, ainsi que le message : « ATTENTION : vous êtes sur le point d'afficher toutes les données et la base, utiliser la requête : "/SmartTerritories/allData" si c'est votre demande. » | | Code http reçu : 303 | | | O |
| 4 | On envoie une requête http au serveur contenant des critères (capteur, type de données, date) non présents dans la base de données. L’API ira donc chercher une donnée présentant un critère non enregistré et donc ne retournera rien.  Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFilter?Capteur=Roseraie&TypeDeDonnee=Meteo | | Retour statut http 444 : no response, ainsi que le message : « Aucune donnée trouvée avec ces critères » | | Code http reçu : 444 | | | O |
| 5 | On envoie une requête http correcte au serveur cependant le format de la date est différent de « YYYY-MM-DD hh ». Aucune des données n’est enregistrée avec ce format donc aucun retour ne sera fait, même si la date dans un autre format est la même qu’en « YYYY-MM-DD hh ».  Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFilter?Date=2025/20/03 | | Retour statut http 400 : bad request, ainsi que le message « Format de date invalide. Utiliser YYYY-MM-DD hh. » | | Code http reçu : 400 | | | O |
| 6 | On envoie une requête http avec des filtres (Capteur / TypeDeDonnee / Date) qui ne correspondent pas à ces trois-là. La requête cherchera donc une donnée avec une entité qui n’existe pas.  Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFilter?Capteur=Saint-Serge&TypeDeDonnee=Temperature&Valeur=20 | | Retour statut http 400 : bad request, ainsi que le message « Paramètres invalides détectés.» | | Code http reçu : 400 | | | O |
| 7.1 | J’ai décidé d’ajouter une limite de données si l’utilisateur souhaite limiter le nombre de données qui lui sont retournées. Il faut donc vérifier que cette limite n’est pas égale à 0 ou négative.  Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFilter?Limite=4 | | Retour statut http 200 (success) avec les données correspondant à la requête. | | Code http reçu : 200 | | | O |
| 7.2 | Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFIlter?Limite=0  OÙ  Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFIlter?Limite=-5 | | Retour statut http 400 : bad request, ainsi que le message : « Le paramètre Limite doit être positif. » | | Code http reçu : 400 | | | O |
| 8 | Dans le cas où une date (seule) est entrée avec une plage de dates, le code ignore cette date, on retourne donc un code et un message en conséquence  Requête http :  http://127.0.0.1:8080/SmartTerritories/dataFilter?Date=2025-03-10 15&DateDebut=2025-03-10 15&DateFin=2025-04-25 10 | | Retour statut http 206 : partial content, ainsi que le message : « La date (simple) sera ignorée car vous avez entré une date de début et/ou de fin » | | Code http reçu : 206 | | | O |
| Conclusion du test : | | Les tests des méthodes n’ont donné aucune erreur, ils ont tous été validés sans problème. On en conclut que la classe fonctionne sans problème, elle est donc validée. | | | | | | |