Procès-Verbal de test : CAPI

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Élément testé : | | Ces tests seront effectués sur les méthodes de la classe CAPI, sans la connexion jusqu’à la classe CDatabase, permettant ainsi une vérification seule de la classe CAPI. | | | | | | |
| 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 message d’erreur. | | | | | | |
| Nom du testeur : | | Théo Bourgoin | | | Date : | | 27/05/2025 | |
| 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 | | | Validation (O/N) |
| 1.1 | Requête renvoi de toutes les données  Ce test correspond à l’utilisation de la requête « allData » qui permets donc de retourner toutes les données de la base.  Requête :  curl "http://localhost:8080/SmartTerritories/allData" | | On attend le code retour correspondant aux données en échantillon. | | {  Capteur: "Roseraie",  TypeDeDonnee: "Temperature",  Date: "2025-05-24 15",  Valeur: 18  },  {  Capteur: "Saint-Serge",  TypeDeDonnee: "Temperature",  Date: "2025-05-28 08",  Valeur: 12  },  {  Capteur: "Saint-Serge",  TypeDeDonnee: "Humidite",  Date: "2025-05-26 09",  Valeur: 70  },  {  Capteur: "Monplaisir",  TypeDeDonnee: "Temperature",  Date: "2025-05-26 09",  Valeur: 13  } | | | O |
| 1.2 | Requête de filtre vide  Ce test correspond à l’utilisation de la requête « dataFilter » ne comprenant aucun filtre.  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter" | | On attend le code retour statut http 200 (success) avec les données correspondant à la requête | | « Requête reçue : GET /SmartTerritories/dataFilter avec paramètres : {}  Tentative de requête sans filtre. Utilisation de /allData recommandée. » | | | O |
| 2.1 | Filtre par Capteur  La suite de test suivante correspond à un filtre par critère simple : ainsi on va chercher les données en entrant un seul critère à la fois. Ici on nous retourne les données suivantes pour le capteur Saint-Serge  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?Capteur=Saint-Serge" | | On attend le retour des données qui comportent le capteur donné | | {  Capteur: "Saint-Serge",  TypeDeDonnee: "Temperature",  Date: "2025-05-28 08",  Valeur: 12  },  {  Capteur: "Saint-Serge",  TypeDeDonnee: "Humidite",  Date: "2025-05-26 09",  Valeur: 70  } | | | O |
| 2.2 | Filtre par TypeDeDonnee  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?TypeDeDonnee=Temperature" | | On attend le retour des données qui comportent le TypeDeDonnee donné. | | {  Capteur: "Roseraie",  TypeDeDonnee: "Temperature",  Date: "2025-05-24 15",  Valeur: 18  },  {  Capteur: "Saint-Serge",  TypeDeDonnee: "Temperature",  Date: "2025-05-28 08",  Valeur: 12  },  {  Capteur: "Monplaisir",  TypeDeDonnee: "Temperature",  Date: "2025-05-26 09",  Valeur: 13  } | | | O |
| 2.3 | Filtre par Date  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?Date=2025-05-26%2009" | | On attend le retour des données qui comportent la date donnée. | | {  Capteur: "Saint-Serge",  TypeDeDonnee: "Humidite",  Date: "2025-05-26 09",  Valeur: 70  },  {  Capteur: "Monplaisir",  TypeDeDonnee: "Temperature",  Date: "2025-05-26 09",  Valeur: 13  } | | | O |
| 2.4 | Filtre avec plage de dates  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?DateDebut=2025-05-25%2009&DateFin=2025-05-26%2009" | | On attend le retour des données qui sont présentes dans la plage donnée. | | {  Capteur: "Saint-Serge",  TypeDeDonnee: "Humidite",  Date: "2025-05-26 09",  Valeur: 70  },  {  Capteur: "Monplaisir",  TypeDeDonnee: "Temperature",  Date: "2025-05-26 09",  Valeur: 13  } | | | O |
| 2.5 | Filtre avec limite  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?Limite=3" | | On attend le retour des données maximum correspondant à la limite. Si limite = 3 alors 3 données. | | {  Capteur: "Roseraie",  TypeDeDonnee: "Temperature",  Date: "2025-05-24 15",  Valeur: 18  },  {  Capteur: "Saint-Serge",  TypeDeDonnee: "Temperature",  Date: "2025-05-28 08",  Valeur: 12  },  {  Capteur: "Saint-Serge",  TypeDeDonnee: "Humidite",  Date: "2025-05-26 09",  Valeur: 70  } | | | O |
| 3 | Valeur inconnue  Ce test correspond à un retour de données dit « valide » mais qui renvoie à un filtre non connu par la base de données : on filtre par une valeur qui n’existe pas.  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?Capteur=blablabla" | | Message : « Aucune donnée trouvée avec ces critères » | | « Aucune donnée trouvée avec les critères fournis » | | | O |
| 4 | Mauvais format de date  Le format de la date étant très stricte pour l’insertion dans la base, il faut être sur que l’utilisateur ne se trompe pas. Dans le cas d’une erreur on renvoit donc le format exact à suivre  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?Date=26-05-2025%2009" | | Message « Format de date invalide. Utiliser YYYY-MM-DD hh. » | | {"message":"Format de date invalide. Utiliser YYYY-MM-DD hh."} | | | O |
| 5 | Mauvais critère  Dans le cas où l’utilisateur se trompe de critère, que ce soit une faute de syntaxe ou une simple erreur, il faut renvoyer un avertissement.  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?NouveauCritère=Valeur" | | Message « Paramètres invalides détectés. » | | {"message":"Paramètres invalides détectés."}  Ainsi que du coté de l’API le message suivant :  « Paramètre(s) invalide(s) reçu(s) : NouveauCritère » | | | O |
| 6.1 | Plusieurs dates de début  On laisse le choix à l’utilisateur d’afficher les données en fonction d’une plage de dates. Cependant on ne peut pas afficher de plage comprenant deux dates de début  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?DateDebut=2025-05-24%2009&DateDebut=2025-05-26%2009" | | Message : « Il ne peut pas y avoir plusieurs dates de début ». | | « Il ne peut pas y avoir plusieurs dates de début. » | | | O |
| 6.2 | Plusieurs dates de fin  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?DateFin=2025-05-26%2009&DateFin=2025-05-27%2009" | | Message : « Il ne peut pas y avoir plusieurs dates de fin ». | | « Il ne peut pas y avoir plusieurs dates de fin. » | | | O |
| 6.3 | Date de début/fin et date simple  Malgré le fait qu’on puisse entrer une plage de date ou une date seule, les deux combinés entraine une erreur. En effet la date seule serait ignorée et on retournerait seulement la plage de dates. Il nous faut donc un avertissement pour l’utilisateur  Requête :  curl "http://localhost:8080/SmartTerritories/dataFilter?DateDebut=2025-05-24%2009&Date=2025-05-24%2011" | | Message : « La date (simple) sera ignorée car vous avez entré une date de début et/ou de fin » | | "La date (simple) sera ignorée car vous avez entré une date de début et/ou de fin" | | | O |
| 7 | Insertion des données  Ce test correspond à une insertion des données via le broker & client MQTT. En supposant que ces données seraient validées par la classe CAnalyse qui contient son propre plan de test. Ainsi on nous retourne bien la validation de l’insertion (les données ne sont pas inscrites dans la base car aucune connexion n’est mise en place).  Requête :  curl -X POST "http://localhost:8080/SmartTerritories/addData" -H "Content-Type: application/json" -d '{"Capteur": "Roseraie", "TypeDeDonnee": "Temperature", "Date": "2025-05-26 09", "Valeur": 15}' | | Message : « Insertion simulée réussie » en renvoyant les données insérées. | | "Insertion simulée réussie" | | | 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. | | | | | | |