**Plan de test Classe CDatabase : Projet Smart Territories**

**1- Identification du test**

Tests\_Classe-CDatabase

**2- Référence du module testé**

Ces tests seront effectués sur les méthodes de la classe CDatabase, principalement la méthode de connexion à la base MongoDB. Cette méthode permet de se connecter à la base à l’aide des éléments donnés dans le constructeur. Elle retournera aussi les erreurs dans le cas d’un problème de connexion.

**3- Objectif du test**

Ce test vérifiera toutes les options lors de la connexion à la base de données MongoDB. En prenant en compte les différentes erreurs possibles.

**4- Procédure du test**

Test 1 : Scénario nominal, connexion aboutie

Ce test correspond à celui qui est utilisé dans la machine de test pour les intégrations. On se connectera à la base de données de test (ici Test\_Database).

Après avoir suivi la partie 6 de ce document (« Moyens à mettre en œuvre ») ainsi que l’annexe de fiche de test, on mettra donc en place une connexion à une base de données existante pour recevoir le message de connexion adapté. Voir la fiche de test modifiée pour savoir les modifications à faire.

Avec cette mise en œuvre, on reçoit le message suivant :

« *Connexion réussie à MongoDB, base : Test\_Database* »

Test 2 : Erreur d’URI invalide

Ce test correspond à une erreur de syntaxe lors de la création de l’URI de connexion à la base, autrement dit une erreur a été faite lors de l’écriture de l’url de connexion. (Correspondant au code actuel, ce test ne devrait pas arriver car cette partie est directement gérée dans le code, cependant une erreur de recopie est vite arrivée.)

Pour ce test, une ligne du code elle-même sera modifiée (voir la fiche de test et la capture d’écran en annexe), on changera l’URI de connexion à la base, en modifiant une syntaxe importante. Ainsi, la requête de connexion est vérifiée et nous retourne une erreur :

« *Erreur URI MongoDB invalide.* »

Test 3 : Erreur de connexion à MongoDB

Pour ce test, on simulera une erreur système ou même un oubli de démarrage MongoDB, pour cela on arrête MongoDB (« sudo systemctl stop mongod ») puis on relance le programme.

Ainsi on redémarre une connexion alors que le service n’est pas démarré. On reçoit donc une erreur précisant que le service n’est pas démarré.

« *Impossible de se connecter : MongoDB est éteint ou inaccessible.* »

Il ne faudra pas oublier de redémarrer le service après ce test (« sudo systemctl start mongod »)

Test 4 : Erreur connexion à une base non existante.

Ce test vérifie l’existence ou non d’une base de données dans le système. Dans le cas où l’utilisateur a donné une base de données non répertoriée. On recevra donc un message en conséquence.   
Pour ce test, il faudra modifier la connexion à une nouvelle base de données en mettant en commentaire la connexion du test 1 puis remettre celle du test 4 (voir annexe).  
On reçoit donc le message d’erreur « *Erreur lors de la connexion : La base 'New\_base' n'existe pas sur le serveur.*»

**5- Résultats attendus**

Test 1 : Connexion établie : message « Connecté réussie à MongoDB, base : « *Nom de base* » »

Test 2 : Connexion refusée avec le message d’erreur MongoDB et : « Erreur URI MongoDB invalide. »

Test 3 : Connexion non établie avec le message : Impossible de se connecter : « Impossible de se connecter :  MongoDB est éteint ou inaccessible. »

Test 4 : Erreur : DatabaseNotFoundError et le message « Erreur lors de la connexion : La base '*Nom de base*' n'existe pas sur le serveur. »

**6- Moyens à mettre en œuvre**

Pour effectuer ces tests il faudra se servir de la machine virtuelle servant de test au code.

Il faudra lancer MongoDB avec un terminal (commande « sudo systemctl start mongod », puis mot de passe user : BTS2425)

Puis on peut se positionner dans le code database.js (Dossier Documents/API-ProgObjet/Tests) via un terminal (commande « cd Documents/API-ProgObjet/Tests » pour changer de dossier et « nano test-CDatabase.js » pour ouvrir le fichier) ou alors l’ouvrir par un explorateur de fichiers.

Pour le bon déroulement de ces tests, il faudra modifier le code en fonction de la fiche de test présente à la fin du fichier.

Il faudra mettre en commentaire ( « // ») les parties qui ne correspondent pas au code du test (voir les captures d’écran en annexe)

Pour démarrer le code, il faudra entrer, dans le terminal (connecté au répertoire de test) : « node test-CDatabase.js »

Annexe : Fiche de test et programme modifié

Test 1 :

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

Test 2 :

Une image contenant texte, Police, capture d’écran

Le contenu généré par l’IA peut être incorrect.

Test 4 :

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.