LES SCÉNARIOS DE TEST

INTRODUCTION

Ce document contient les scénarios de test de haut niveau pour l'application HTTP Curl Wrapper. Chaque scénario décrit une fonctionnalité de bout en bout à tester, se concentrant sur ce qu'il faut tester.

OBJECTIFS

- Valider les différentes fonctionnalités de l'application HTTP Curl Wrapper.
- Garantir que les requêtes HTTP (**GET**, **POST**, **PUT**, **DELETE**, etc.) fonctionnent correctement.
- Vérifier la gestion appropriée des cookies et des sessions utilisateur.
- Assurer une expérience utilisateur agréable et sans erreur.

SCÉNARIOS DE TEST

SCÉNARIO DE TEST ST001 : GESTION DES SESSIONS

Description : Vérifier que l'application gère correctement les sessions utilisateur, en particulier la création, la récupération et la validation des ID de session.
Fonctionnalités testées :

- Création d'un ID de session.
- Récupération de l'ID de session.
- Validation de l'ID de session.

Importance : Critique
Détails :

- La session doit pouvoir être créée avec un ID unique.
- L'ID de session doit être récupérable à tout moment.
- L'ID de session doit être validé pour vérifier son authenticité.

SCÉNARIO DE TEST ST002 : GESTION DES URL

Description : Vérifier que l'application gère correctement les URL, y compris l'attribution, la récupération et la modification des URL dans les requêtes HTTP.
Fonctionnalités testées :

- Attribution d'une URL.
- Récupération de l'URL actuelle.
- Modification de l'URL.

Importance : Élevée
Détails :

- L'URL doit pouvoir être définie lors de la création d'une requête.
- L'URL actuelle doit pouvoir être récupérée correctement.

- L'URL doit pouvoir être modifiée après la création de la requête.

SCÉNARIO DE TEST ST003 : GESTION DES COOKIES

Description : Vérifier que l'application gère correctement les cookies, y compris la création, la récupération et la transmission des cookies dans les requêtes HTTP.
Fonctionnalités testées :

- Création de cookies.
- Récupération des cookies.
- Transmission des cookies avec les requêtes.

Importance : Critique
Détails :

- Les cookies doivent pouvoir être créés avec un nom et une valeur.
- Les cookies stockés doivent pouvoir être récupérés.
- Les cookies doivent être transmis correctement avec chaque requête HTTP.

SCÉNARIO DE TEST ST004 : REQUÊTES HTTP GET

Description : Vérifier que l'application envoie correctement des requêtes HTTP GET et traite les réponses reçues. Ce test vérifie l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse**.
Fonctionnalités testées :

- Envoi de requêtes GET.
- Réception et traitement des réponses.

Importance : Critique
Détails :

- Une requête GET doit être envoyée à une URL spécifiée.
- La réponse doit être correctement reçue et traitée, y compris la vérification du statut de la réponse.

SCÉNARIO DE TEST ST005 : REQUÊTES HTTP POST

Description : Vérifier que l'application envoie correctement des requêtes HTTP POST avec le corps de la requête et traite les réponses reçues. Ce test vérifie l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse**.
Fonctionnalités testées :

- Envoi de requêtes POST avec un corps de requête.
- Réception et traitement des réponses.

Importance : Critique
Détails :

- Une requête POST doit être envoyée à une URL spécifiée avec un corps de requête.
- La réponse doit être correctement reçue et traitée, y compris la vérification du statut de la réponse.

SCÉNARIO DE TEST ST006 : REQUÊTES HTTP PUT

Description : Vérifier que l'application envoie correctement des requêtes HTTP PUT avec le corps de la requête et traite les réponses reçues. Ce test vérifie l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse**.
Fonctionnalités testées :

- Envoi de requêtes PUT avec un corps de requête.
- Réception et traitement des réponses.

Importance : Critique
Détails :

- Une requête PUT doit être envoyée à une URL spécifiée avec un corps de requête.
- La réponse doit être correctement reçue et traitée, y compris la vérification du statut de la réponse.

SCÉNARIO DE TEST ST007 : REQUÊTES HTTP DELETE

Description : Vérifier que l'application envoie correctement des requêtes HTTP DELETE et traite les réponses reçues. Ce test vérifie l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse**.
Fonctionnalités testées :

- Envoi de requêtes DELETE.
- Réception et traitement des réponses.

Importance : Critique
Détails :

- Une requête DELETE doit être envoyée à une URL spécifiée.
- La réponse doit être correctement reçue et traitée, y compris la vérification du statut de la réponse.

SCÉNARIO DE TEST ST008 : REQUÊTES HTTP HEAD

Description : Vérifier que l'application envoie correctement des requêtes HTTP HEAD et traite les réponses reçues, notamment les en-têtes. Ce test vérifie l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse**.
Fonctionnalités testées :

- Envoi de requêtes HEAD.
- Réception et traitement des en-têtes de réponse.

Importance : Moyenne
Détails :

- Une requête HEAD doit être envoyée à une URL spécifiée.
- Les en-têtes de réponse doivent être correctement reçus et vérifiés.

SCÉNARIO DE TEST ST009 : REQUÊTES HTTP OPTIONS

Description : Vérifier que l'application envoie correctement des requêtes HTTP OPTIONS et traite les réponses reçues, notamment les options disponibles sur le serveur. Ce test vérifie l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse**.
Fonctionnalités testées :

- Envoi de requêtes OPTIONS.

- Réception et traitement des réponses.

Importance : Moyenne
Détails :

- Une requête OPTIONS doit être envoyée à une URL spécifiée.
- La réponse doit indiquer les méthodes HTTP supportées par le serveur et être correctement traitée.

SCÉNARIO DE TEST ST010 : GESTION DES DÉLAIS D'ATTENTE

Description : Vérifier que l'application gère correctement les délais d'attente pour les requêtes HTTP. Ce test vérifie l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse**.
Fonctionnalités testées :

- Attribution de délais d'attente.
- Comportement en cas de dépassement de délai.

Importance : Moyenne
Détails :

- Un délai d'attente doit pouvoir être défini pour chaque requête.
- L'application doit gérer correctement les cas de dépassement de délai et retourner des erreurs appropriées.

SCÉNARIO DE TEST ST011 : MODIFICATION DES URL APRÈS CRÉATION DE LA REQUÊTE

Description : Vérifier que l'application permet de modifier l'URL d'une requête après sa création. Ce test vérifie l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse**.
Fonctionnalités testées :

- Modification de l'URL d'une requête existante.

Importance : Moyenne
Détails :

- L'URL d'une requête doit pouvoir être modifiée après la création de la requête.
- La nouvelle URL doit être utilisée lors de l'envoi de la requête.

SCÉNARIO DE TEST ST012 : MODIFICATION DU CORPS DE LA REQUÊTE

Description : Vérifier que l'application permet de modifier le corps de la requête après sa création. Ce test vérifie l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse**.
Fonctionnalités testées :

- Modification du corps de la requête d'une requête existante.

Importance : Moyenne
Détails :

- Le corps de la requête doit pouvoir être modifié après la création de la requête.
- Le nouveau contenu du corps doit être utilisé lors de l'envoi de la requête.

SCÉNARIO DE TEST ST013 : MODIFICATION DES DÉLAIS D'ATTENTE

Description : Vérifier que l'application permet de modifier les délais d'attente après la création de la requête. Ce test vérifie l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse**.
Fonctionnalités testées :

- Modification des délais d'attente d'une requête existante.

Importance : Moyenne
Détails :

- Les délais d'attente doivent pouvoir être modifiés après la création de la requête.
- Les nouveaux délais d'attente doivent être utilisés lors de l'envoi de la requête.

PLAN DE TEST

INTRODUCTION

Ce document décrit l'approche, la portée, les ressources, le calendrier et les livrables pour l'application HTTP Curl Wrapper. Le but est de garantir la qualité et la fonctionnalité du logiciel grâce à des tests approfondis.

OBJECTIFS

- Valider les différentes fonctionnalités de l'application HTTP Curl Wrapper.
- Garantir que les requêtes HTTP (**GET**, **POST**, **PUT**, **DELETE**, etc.) fonctionnent correctement.
- Vérifier la gestion appropriée des cookies et des sessions utilisateur.
- Assurer une expérience utilisateur agréable et sans erreur.

RÔLES ET RESPONSABILITÉS

Les rôles et responsabilités pour ce projet de test sont les suivants :

- **Développeurs SW (5 personnes) :**
    - Développent les fonctionnalités du HTTP Curl Wrapper.
    - Rédigent et exécutent les tests unitaires.
    - Participent à la révision du code pour assurer la qualité et la conformité aux normes de développement.
- **Responsable Qualité (1 personne) :**
    - Développe et documente les tests d'intégration.
    - Exécute les tests d'intégration et d'acceptation.
    - Identifie les défauts et les problèmes, et collabore avec les développeurs pour les résoudre.
    - Assure que toutes les fonctionnalités respectent les exigences de qualité définies.
- **Chef de Projet (1 personne) :**
    - Gère le planning et le budget du projet.
    - Coordonne les différentes équipes et ressources pour assurer l'avancement du projet.
    - Sert de point de contact principal avec le client pour les mises à jour et les retours.

- Supervise l'ensemble du processus de test pour s'assurer que les objectifs du projet sont atteints.

STRATÉGIE DE TEST

La stratégie de test inclut les tests unitaires et d'intégration suivants :

TESTS UNITAIRES

Les tests unitaires vérifient le bon fonctionnement des composants individuels de l'application de manière isolée.

- **Méthodes Testées** :
  - **CurlSession::setSessionId**
  - **CurlRequest::getUrl**
  - **CurlCookies::setCookie**

TESTS D'INTÉGRATION

Les tests d'intégration vérifient que les différents composants de l'application fonctionnent correctement ensemble.

- **Scénarios Testés** :
  - **CurlRequest::sendGet**
  - **CurlRequest::sendHead**
  - **CurlRequest::sendOptions** (utilisant des mocks)
  - **CurlRequest::sendPost**
  - **CurlRequest::sendPut**
  - **CurlRequest::sendDelete**
  - **CurlRequest::sendPatch**
  - **CurlRequest::setUrl**
  - **CurlRequest::setBody**
  - **CurlRequest::setTimeout**

ENVIRONNEMENT DE TEST

Les tests seront réalisés dans un environnement de test configuré pour simuler les conditions réelles d'utilisation de l'application. Les tests incluront les configurations nécessaires pour garantir la qualité du projet. L'environnement comprendra :

- **Système d'exploitation** : Linux ou Windows.
- **Bibliothèques** : cURL, gtest, gmock.
- **Réseau** : Connexion Internet stable pour tester les requêtes HTTP vers des serveurs externes comme **https://httpbin.org**.
- **Outils de développement** : Compilateur C++, IDE (par exemple, Visual Studio, CLion).

CAS DE TEST

CAS DE TEST TC001 : CURLSESSION::SETSESSIONID

Description : Vérifier que la méthode **setSessionId** attribue correctement l'ID de session.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer un objet **CurlSession**.
2. Appeler la méthode **setSessionId** avec l'ID **1234567890**.
3. Utiliser la méthode **getSessionId** pour vérifier l'ID attribué.

Données : -
Résultat Attendu : L'ID de session doit être **1234567890**.
Conditions : Il s'agit d'un test unitaire de la classe **CurlSession**.

CAS DE TEST TC002 : CURLREQUEST::GETURL

Description : Vérifier que la méthode **getUrl** retourne l'URL correcte.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer les objets nécessaires (**CurlSession**, **CurlCookies**).
2. Créer un objet **CurlRequest** avec l'URL **https://httpbin.org/get**.
3. Utiliser la méthode **getUrl** pour vérifier l'URL.

Données : -
Résultat Attendu : L'URL doit être **https://httpbin.org/get**.
Conditions : Il s'agit d'un test unitaire de la classe **CurlRequest**.

CAS DE TEST TC003 : CURLCOOKIES::SETCOOKIE

Description : Vérifier que la méthode **setCookie** attribue correctement un cookie.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer un objet **CurlCookies**.
2. Appeler la méthode **setCookie** avec le nom **freeform** et la valeur **value1**.
3. Utiliser la méthode **getCookies** pour vérifier le cookie attribué.

Données : -
Résultat Attendu : Le cookie doit être **freeform: value1**.
Conditions : Il s'agit d'un test unitaire de la classe **CurlCookies**.

CAS DE TEST TC004 : CURLREQUEST::SENDGET

Description : Vérifier l'intégration des classes **CurlRequest**, **CurlSession**, **CurlCookies** et **CurlResponse** avec une requête GET.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer un objet **CurlRequest** avec les configurations nécessaires :

- URL : **https://httpbin.org/get**
- Délai d’attente : 10 secondes
- Cookies : **freeform**: **value1**

2. Ajouter un en-tête avec la méthode **setHeader**.
3. Envoyer une requête GET avec la méthode **sendGet**.
4. Vérifier le statut de la réponse.

Données : -
Résultat Attendu : Le statut de la réponse doit être 200.
Conditions : Il s'agit d'un test d'intégration.

CAS DE TEST TC005 : CURLREQUEST::SENDHEAD

Description : Vérifier l'intégration des classes avec une requête HEAD.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer un objet **CurlRequest** avec les configurations nécessaires :
   - URL : **https://httpbin.org/get**
   - Délai d’attente : 10 secondes
   - Cookies : **freeform**: **value1**
2. Ajouter un en-tête avec la méthode **setHeader**.
3. Envoyer une requête HEAD avec la méthode **sendHead**.
4. Vérifier le statut et les en-têtes de la réponse.
5. Vérifier la validité de l'ID de session avec la méthode **isSessionIdValid**.

Données : -
Résultat Attendu : Le statut de la réponse doit être 200 et l'ID de session doit être valide.
Conditions : Il s'agit d'un test d'intégration.

CAS DE TEST TC006 : CURLREQUEST::SENDOPTIONS (MOCK)

Description : Utiliser **MockCurlRequest** pour tester la méthode **sendOptions**.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer un objet **MockCurlRequest**.
2. Définir une réponse mock avec un statut, des en-têtes et une corp spécifique.
3. Envoyer une requête OPTIONS avec la méthode **sendOptions**.
4. Vérifier le statut, les en-têtes et le corps de la réponse.

Données : -
Résultat Attendu : Le statut doit être 200, et le corps et les en-têtes doivent être corrects (**body**, **headers**).
Conditions : Il s'agit d'un test unitaire utilisant des mocks.

CAS DE TEST TC007 : CURLREQUEST::SENDPOST

Description : Vérifier l'intégration des classes avec une requête POST.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer les objets nécessaires (**CurlSession**, **CurlCookies**).
2. Configurer les cookies et l'ID de session.
3. Créer un objet **CurlRequest** avec les configurations nécessaires :
    - URL : **https://httpbin.org/post**
    - Délai d'attente : 10 secondes
4. Ajouter un en-tête avec la méthode **setHeader**.
5. Envoyer une requête POST avec la méthode **sendPost** et un contenu de corps.
6. Vérifier le statut de la réponse.

Données : -
Résultat Attendu : Le statut de la réponse doit être 200.
Conditions : Il s'agit d'un test d'intégration.

CAS DE TEST TC008 : CURLREQUEST::SENDPUT

Description : Vérifier l'intégration des classes avec une requête PUT.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer les objets nécessaires (**CurlSession**, **CurlCookies**).
2. Configurer les cookies et l'ID de session.
3. Créer un objet **CurlRequest** avec les configurations nécessaires :
    - URL : **https://httpbin.org/put**
    - Délai d'attente : 10 secondes
4. Ajouter un en-tête avec la méthode **setHeader**.
5. Envoyer une requête PUT avec la méthode **sendPut** et un contenu de corps.
6. Vérifier le statut de la réponse.

Données : -
Résultat Attendu : Le statut de la réponse doit être 200.
Conditions : Il s'agit d'un test d'intégration.

CAS DE TEST TC009 : CURLREQUEST::SENDDELETE

Description : Vérifier l'intégration des classes avec une requête DELETE.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer les objets nécessaires (**CurlSession**, **CurlCookies**).
2. Configurer les cookies et l'ID de session.
3. Créer un objet **CurlRequest** avec les configurations nécessaires :

- URL : **https://httpbin.org/delete**
- Délai d'attente : 10 secondes

4. Ajouter un en-tête avec la méthode **setHeader**.
5. Envoyer une requête DELETE avec la méthode **sendDelete**.
6. Vérifier le statut de la réponse.

Données : -
Résultat Attendu : Le statut de la réponse doit être 200.
Conditions : Il s'agit d'un test d'intégration.

CAS DE TEST TC010 : CURLREQUEST::SENDPATCH

Description : Vérifier l'intégration des classes avec une requête PATCH.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer les objets nécessaires (**CurlSession**, **CurlCookies**).
2. Configurer les cookies et l'ID de session.
3. Créer un objet **CurlRequest** avec les configurations nécessaires :
    - URL : **https://httpbin.org/patch**
    - Délai d'attente : 10 secondes
4. Ajouter un en-tête avec la méthode **setHeader**.
5. Envoyer une requête PATCH avec la méthode **sendPatch** et un contenu de corps.
6. Vérifier le statut de la réponse.

Données : -
Résultat Attendu : Le statut de la réponse doit être 200.
Conditions : Il s'agit d'un test d'intégration.

CAS DE TEST TC011 : CURLREQUEST::SETURL

Description : Vérifier que la méthode **setUrl** attribue correctement une nouvelle URL.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer les objets nécessaires (**CurlSession**, **CurlCookies**).
2. Configurer l'ID de session.
3. Créer un objet **CurlRequest** avec une URL de départ.
4. Utiliser la méthode **setUrl** pour définir une nouvelle URL.
5. Ajouter un corps et un en-tête avec **setBody** et **setHeader**.
6. Envoyer une requête POST avec la méthode **sendPost**.
7. Vérifier la nouvelle URL.

Données : -
Résultat Attendu : La nouvelle URL doit être correctement définie (**https://httpbin.org/post**).
Conditions : Il s'agit d'un test d'intégration.

CAS DE TEST TC012 : CURLREQUEST::SETBODY

Description : Vérifier que la méthode **setBody** attribue correctement un nouveau contenu de corps.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer les objets nécessaires (**CurlSession**, **CurlCookies**).
2. Configurer l'ID de session.
3. Créer un objet **CurlRequest** avec une URL de départ.
4. Utiliser la méthode **setBody** pour définir un nouveau contenu de corps et sa taille.
5. Ajouter un en-tête avec la méthode **setHeader**.
6. Envoyer une requête POST avec la méthode **sendPost**.
7. Vérifier le nouveau contenu de corps.

Données : -
Résultat Attendu : Le contenu du corps doit être correctement défini (**hello world**).
Conditions : Il s'agit d'un test d'intégration.

CAS DE TEST TC013 : CURLREQUEST::SETTIMEOUT

Description : Vérifier que la méthode **setTimeout** attribue correctement un nouveau délai d'attente.
Prérequis : Le logiciel est installé et fonctionnel.
Étapes du Test :

1. Créer les objets nécessaires (**CurlSession**, **CurlCookies**).
2. Configurer l'ID de session.
3. Créer un objet **CurlRequest** avec une URL de départ.
4. Utiliser la méthode **setTimeout** pour définir un nouveau délai d'attente.
5. Ajouter un en-tête avec la méthode **setHeader**.
6. Envoyer une requête POST avec la méthode **sendPost**.
7. Vérifier le délai d'attente.

Données : -
Résultat Attendu : Le délai d'attente doit être correctement défini (20 secondes).
Conditions : Il s'agit d'un test d'intégration.