





AP

7- Développement de l'API C# de l'application Windows

1- Rappel de l'architecture applicative globale et objectif de cette étape
2- Le diagramme des classes C#
3- Présentation de la solution fournie
4- Exécution des tests unitaires des classes Point, PointDeTrace et Trace
5- Création des tests unitaires de la classe Utilisateur
6- Test des méthodes fournies de la classe PasserelleServicesWebXML
7- Développement et test des éléments manquants de la classe PasserelleServicesWebXML

1- Rappel de l'architecture applicative globale et objectif de cette étape

La page suivante présente l'architecture générale de l'application **TraceGPS**.

Cette étape porte sur le développement et le test de **l'API C#** qui sera utilisée par l'application Windows pour accéder aux services web et permettre un accès à la base de données au moyen d'une connexion Internet.

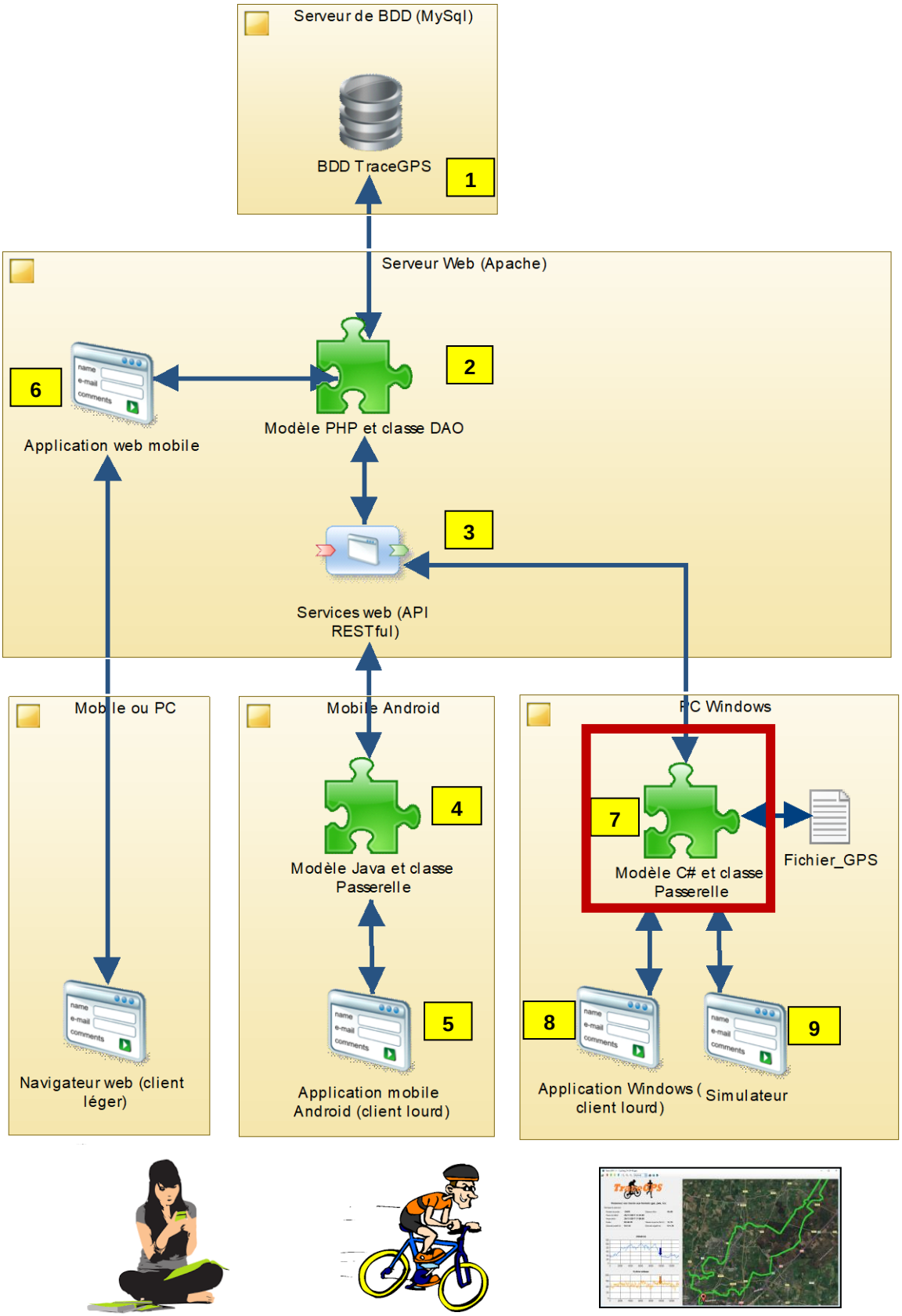
L'API Java est composée de 4 classes métiers et de 7 classes techniques.

Les 4 **classes métiers** :

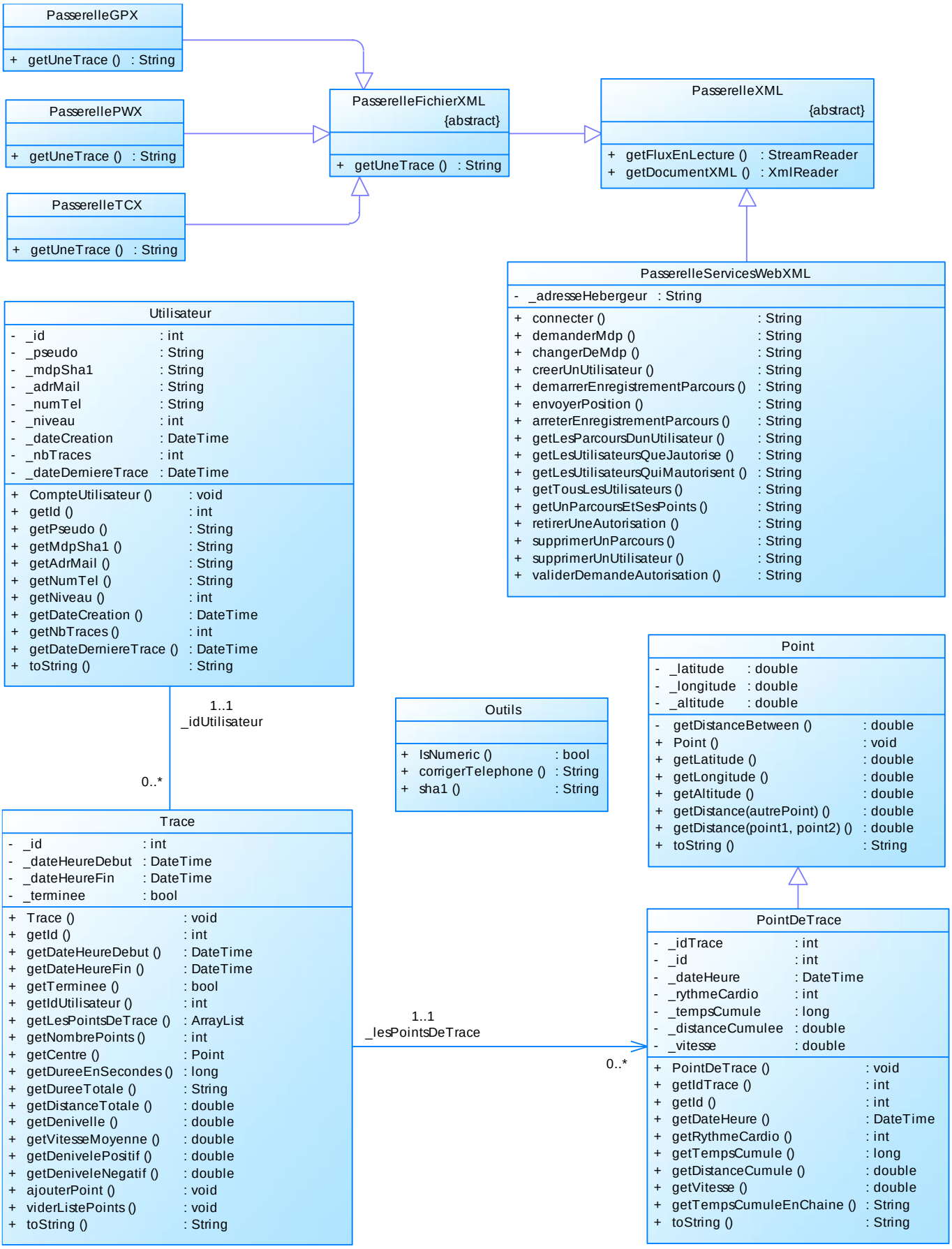
- **Point** : représente un point géographique
- **PointDeTrace** : (hérite de **Point**) ; représente un point de passage lors d'un parcours
- **Trace** : représente une trace (ou un parcours) réalisé par un utilisateur
- **Utilisateur** : représente un utilisateur

Les 7 **classes techniques** :

- **Outils** : classe contenant quelques fonctions utilitaires (comme **IsNumeric**)
- **PasserelleXML** : classe abstraite pour parser un document XML
- **PasserelleServiceWebXML** : hérite de PasserelleXML ; pour parser un document XML obtenu par l'appel d'un service web
- **PasserelleFichierXML** : hérite de **PasserelleXML** ; pour parser un document XML obtenu par lecture d'un fichier
- **PasserelleGPX** : hérite de **PasserelleFichierXML** ; pour parser un document XML obtenu par l'appel d'un fichier au format **GPX**
- **PasserellePWX** : hérite de **PasserelleFichierXML** ; pour parser un document XML obtenu par l'appel d'un fichier au format **PWX**
- **PasserelleTCX** : hérite de **PasserelleFichierXML** ; pour parser un document XML obtenu par l'appel d'un fichier au format **TCX**


Serveur de BDD (MySql)
BDD TraceGPS
1
Serveur Web (Apache)
2
Modèle PHP et classe DAO
6
Application web mobile
3
Services web (API RESTful)
Mobile ou PC
Navigateur web (client léger)
Mobile Android
4
Modèle Java et classe Passerelle
5
Application mobile Android (client lourd)
PC Windows
7
Modèle C# et classe Passerelle
Fichier_GPS
8
9
Application Windows (client lourd)
Simulateur

2- Le diagramme des classes C#

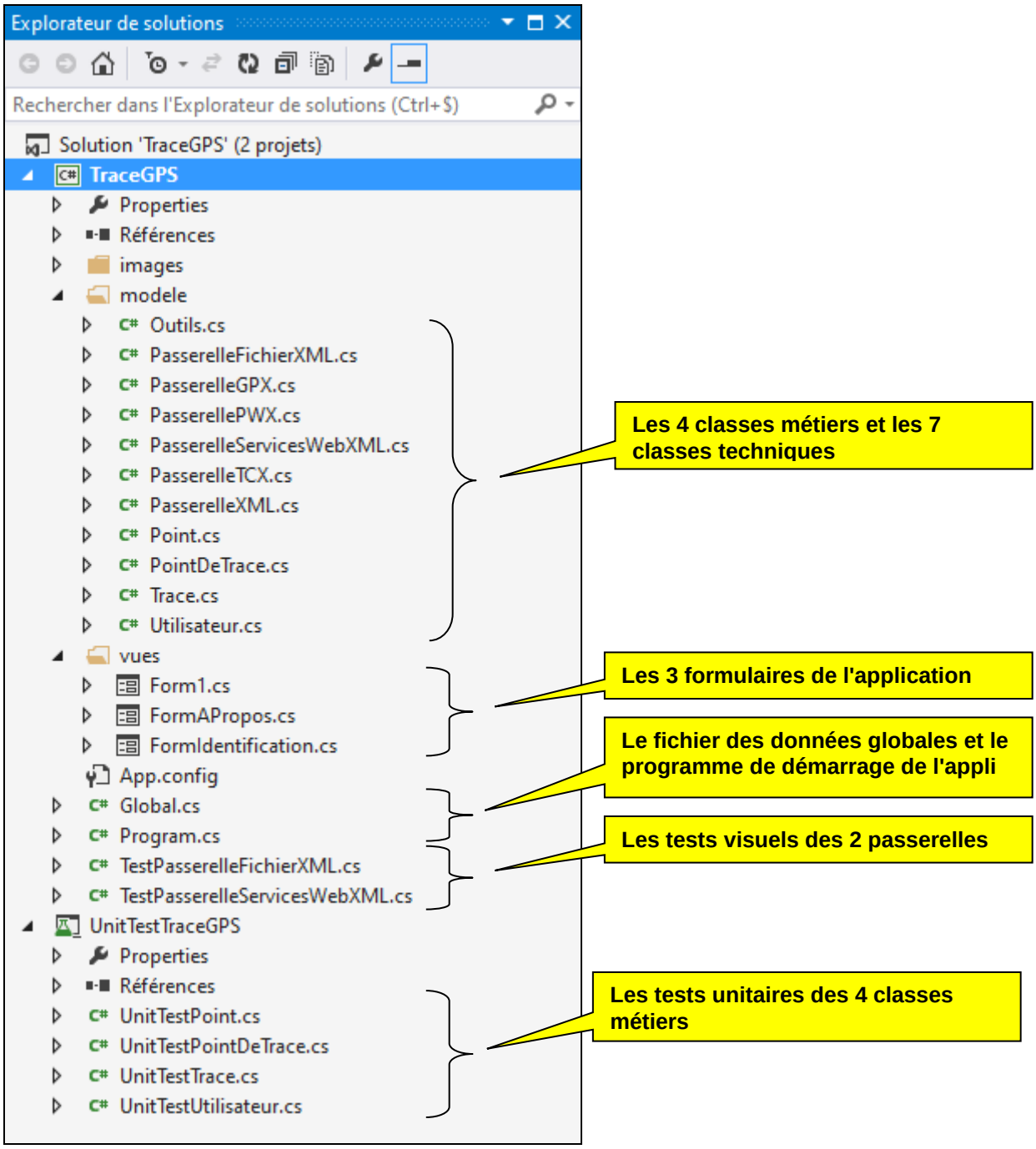


3- Présentation de la solution fournie

Le développement de l'API C# est partiellement réalisé et vous est fourni dans le fichier compressé **TraceGPS_ C#_fourni.zip** (qui contient également l'application graphique Windows) .

Téléchargez ce fichier, décompressez-le, et ouvrez la solution obtenue.

Sa structure est la suivante :



4- Exécution des tests unitaires des classes Point, PointDeTrace et Trace

Utilisez la commande **TEST / Exécuter / Tous les tests** pour exécuter les tests unitaires fournis afin de vérifier le bon fonctionnement des classes métiers **Point**, **PointDeTrace,** et **Trace** :

- **UnitTestPoint.cs**
- **UnitTestPointDeTrace.cs**
- **UnitTestTrace.cs**

En cas d'échec d'un test, trouvez la cause du problème et relancez les tests jusqu'à leur réussite.

5- Création des tests unitaires de la classe Utilisateur

Les tests unitaires de la classe **Utilisateur** ont été commencés dans le fichier **UnitTestUtilisateur.cs**.

La fonction **MyTestInitialize** est appelée avant chaque test ; elle crée 2 objets de la classe **Utilisateur** :

- **utilisateur1** : avec les données par défaut
- **utilisateur2** : avec des données indiquées au constructeur

Le test de la méthode **toString** est déjà écrit. Vous êtes chargés d'écrire les tests unitaires manquants en utilisant les 2 objets **utilisateur1** et **utilisateur2** créés par la fonction **MyTestInitialize**.
Vous pouvez bien sûr vous inspirer des tests fournis pour les autres classes, ou des tests que vous avez déjà écrit pour l'API Java.

Quand ces tests seront écrits, exécutez-les.
En cas d'échec, trouvez la cause du problème et relancez les tests jusqu'à leur réussite.

6- Test des méthodes fournies de la classe PasserelleServicesWebXML

Les 5 méthodes fournies concernent **la gestion des utilisateurs** et utilisent les services web qui vous étaient fournis à l'étape du développement des services web :

- **connecter**
- **getTousLesUtilisateurs**
- **creerUnUtilisateur**
- **supprimerUnUtilisateur**
- **changerDeMdp**

Remarque générale concernant toutes les méthodes écrites ou à écrire :

Les méthodes retournent toutes une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Remarques générales concernant les tests :

Les méthodes de la classe **PasserelleServicesWebXML** seront testées par des tests visuels regroupés dans le fichier **TestPasserelleServiceWebXML.cs**

Pensez à régler **l'objet de démarrage** du projet sur cette page de test.

Quand un test est réussi, désactivezson code en le mettant en commentaire.

6-1 la méthode connecter

public static String connecter(String pseudo, String mdpSha1)
Rôle : permet à l'utilisateur de se connecter pour accéder au menu de l'application (appelle le service **Connecter**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : son mot de passe hashé en SHA1
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode connecter
msg = PasserelleServicesWebXML.connecter("admin", "adminnnnnnnn");
// affichage de la réponse
MessageBox.Show(msg, "Test 1.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.connecter("admin", Outils.sha1("mdpadmin"));
// affichage de la réponse
MessageBox.Show(msg, "Test 1.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.connecter("europa", Outils.sha1("mdputilisateur"));
// affichage de la réponse
MessageBox.Show(msg, "Test 1.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

6-2 la méthode getTousLesUtilisateurs

public static String getTousLesUtilisateurs(String pseudo, String mdpSha1, ArrayList lesUtilisateurs)
Rôle : permet d' obtenir la liste de tous les utilisateurs de niveau 1 (appelle le service **GetTousLesUtilisateurs**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : son mot de passe hashé en SHA1 **lesUtilisateurs** : collection (vide) à remplir à partir des données fournies par le service web
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode getTousLesUtilisateurs
lesUtilisateurs = new ArrayList();
msg = PasserelleServicesWebXML.getTousLesUtilisateurs("europa", Outils.sha1("mdputilisateur"), lesUtilisateurs);
// affichage de la réponse
MessageBox.Show(msg, "Test 2.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
// affichage du nombre d'utilisateurs
msg = "Nombre d'utilisateurs : " + lesUtilisateurs.Count;
MessageBox.Show(msg, "Test 2.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
// affichage de tous les utilisateurs
foreach (Utilisateur unUtilisateur in lesUtilisateurs)
{
    msg = unUtilisateur.toString();
    MessageBox.Show(msg, "Test 2.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

Comparez l'affichage obtenu avec le contenu de votre base de données et en cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

6-3 la méthode creerUnUtilisateur

public static String creerUnUtilisateur(String pseudo, String adrMail, String numTel)
Rôle : permet de créer un compte utilisateur (appelle le service **creerUnUtilisateur**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **adrMail** : son adresse mail **numTel** : son numéro de téléphone
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode creerUnUtilisateur
msg = PasserelleServicesWebXML.creerUnUtilisateur("jim", "delasalle.sio.eleves@gmail.com", "1122334455");
// Erreur : pseudo trop court (8 car minimum) ou déjà existant.
MessageBox.Show(msg, "Test 3.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.creerUnUtilisateur("turlututu", "delasalle.sio.elevesgmail.com", "1122334455");
// Erreur : adresse mail incorrecte ou déjà existante.
MessageBox.Show(msg, "Test 3.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.creerUnUtilisateur("turlututu", "delasalle.sio.eleves@gmailcom", "1122334455");
// Erreur : adresse mail incorrecte ou déjà existante.
MessageBox.Show(msg, "Test 3.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.creerUnUtilisateur("turlututu", "delasalle.sio.eleves@gmail.com", "1122334455");
// Erreur : adresse mail incorrecte ou déjà existante.
MessageBox.Show(msg, "Test 3.4", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.creerUnUtilisateur("turlututu", "delasallesioeleves@gmail.com", "1122334455");
// Enregistrement effectué ; vous allez recevoir un courriel avec votre mot de passe.
MessageBox.Show(msg, "Test 3.5", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.creerUnUtilisateur("turlututu", "de.la.salle.sio.eleves@gmail.com", "1122334455");
// Erreur : pseudo trop court (8 car minimum) ou déjà existant.
MessageBox.Show(msg, "Test 3.6", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

6-4 la méthode supprimerUnUtilisateur

public static String supprimerUnUtilisateur(String pseudo, String mdpSha1, String pseudoAsupprimer)
Rôle : permet à un administrateur de supprimer un utilisateur (appelle le service **SupprimerUnUtilisateur**).
Paramètres à fournir : **pseudo** : le pseudo de l'administrateur qui fait appel au service web **mdpSha1** : le mot de passe hashé en sha1 **pseudoAsupprimer** : le pseudo de l'utilisateur à supprimer
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode supprimerUnUtilisateur
msg = PasserelleServicesWebXML.supprimerUnUtilisateur("europa", Outils.sha1("mdputilisateurrrrrr"), "toto");
// Erreur : authentification incorrecte.
MessageBox.Show(msg, "Test 4.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.supprimerUnUtilisateur("europa", Outils.sha1("mdputilisateur"), "toto");
// Erreur : authentification incorrecte.
MessageBox.Show(msg, "Test 4.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.supprimerUnUtilisateur("admin", Outils.sha1("mdpadminnnnn"), "toto");
// Erreur : authentification incorrecte.
MessageBox.Show(msg, "Test 4.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.supprimerUnUtilisateur("admin", Outils.sha1("mdpadmin"), "toto");
// Erreur : pseudo utilisateur inexistant.
MessageBox.Show(msg, "Test 4.4", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.supprimerUnUtilisateur("admin", Outils.sha1("mdpadmin"), "neon");
// Erreur : suppression impossible ; cet utilisateur possède encore des traces.
MessageBox.Show(msg, "Test 4.5", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.supprimerUnUtilisateur("admin", Outils.sha1("mdpadmin"), "turlututu");
// Suppression effectuée ; un courriel va être envoyé à l'utilisateur.
MessageBox.Show(msg, "Test 4.6", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

6-5 la méthode changerDeMdp

public static String changerDeMdp(String pseudo, String mdpSha1, String nouveauMdp, String confirmationMdp)
Rôle : permet de modifier son mot de passe (appelle le service **ChangerDeMdp**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : le mot de passe hashé en sha1 **nouveauMdp** : le nouveau mot de passe **confirmationMdp** : la confirmation du nouveau mot de passe
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode changerDeMdp
msg = PasserelleServicesWebXML.changerDeMdp("europa", Outils.sha1("mdputilisateur"), "passepasse",
"passepassepasse");
// Erreur : le nouveau mot de passe et sa confirmation sont différents.
MessageBox.Show(msg, "Test 5.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.changerDeMdp("europa", Outils.sha1("mdputilisateurrrr"), "passepasse",
"passepasse");
// Erreur : authentification incorrecte.
MessageBox.Show(msg, "Test 5.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.changerDeMdp("europa", Outils.sha1("mdputilisateur"), "mdputilisateurrrr",
"mdputilisateurrrr");
// Enregistrement effectué ; vous allez recevoir un courriel de confirmation.
MessageBox.Show(msg, "Test 5.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.changerDeMdp("europa", Outils.sha1("mdputilisateurrrr"), "mdputilisateur",
"mdputilisateur");
// Enregistrement effectué ; vous allez recevoir un courriel de confirmation.
MessageBox.Show(msg, "Test 5.4", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7- Développement et test des éléments manquants de la classe PasserelleServicesWebXML

Les 11 méthodes restant à développer utilisent les services web que vous avez développés à l'étape du développement des services web :

Services web du domaine **Gestion des utilisateurs** :

- demanderMdp

Services web du domaine **Gestion des autorisations** :

- getLesUtilisateursQueJautorise
- getLesUtilisateursQuiMautorisent
- demanderUneAutorisation
- retirerUneAutorisation

Services web du domaine **Gestion des points de traces** :

- envoyerPosition

Services web du domaine **Gestion des traces** :

- getUnParcoursEtSesPoints
- getLesParcoursDunUtilisateur
- supprimerUnParcours
- demarrerEnregistrementParcours
- arreterEnregistrementParcours

7-1 la méthode demanderMdp

public static String demanderMdp(String pseudo)
Rôle : permet de demander un nouveau mot de passe (appelle le service **DemanderMdp**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode demanderMdp
msg = PasserelleServicesWebXML.demanderMdp("jim");
// Erreur : pseudo inexistant.
MessageBox.Show(msg, "Test 6.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.demanderMdp("europa");
// Vous allez recevoir un courriel avec votre nouveau mot de passe.
MessageBox.Show(msg, "Test 6.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7-2 la méthode getLesUtilisateursQueJautorise

public static String getLesUtilisateursQueJautorise(String pseudo, String mdpSha1, ArrayList lesUtilisateurs)
Rôle : permet d' obtenir la liste des utilisateurs que j'autorise (appelle le service **GetLesUtilisateursQueJautorise**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : son mot de passe hashé en SHA1 **lesUtilisateurs** : collection (vide) à remplir à partir des données fournies par le service web
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode getLesUtilisateursQueJautorise
lesUtilisateurs = new ArrayList();
msg = PasserelleServicesWebXML.getLesUtilisateursQueJautorise("indigo", Outils.sha1("mdputilisateur"),
lesUtilisateurs);
// affichage de la réponse
MessageBox.Show(msg, "Test 7.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
// affichage du nombre d'utilisateurs
msg = "Nombre d'utilisateurs : " + lesUtilisateurs.Count;
MessageBox.Show(msg, "Test 7.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
// affichage de tous les utilisateurs
foreach (Utilisateur unUtilisateur in lesUtilisateurs)
{
    msg = unUtilisateur.toString();
    MessageBox.Show(msg, "Test 7.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

Comparez l'affichage obtenu avec le contenu de votre base de données et en cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7-3 la méthode getLesUtilisateursQuiMautorisent

public static String getLesUtilisateursQuiMautorisent(String pseudo, String mdpSha1, ArrayList lesUtilisateurs)
Rôle : permet d' obtenir la liste des utilisateurs qui m'autorisent (appelle le service **GetLesUtilisateursQuiMautorisent**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : son mot de passe hashé en SHA1 **lesUtilisateurs** : collection (vide) à remplir à partir des données fournies par le service web
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode getLesUtilisateursQuiMautorisent
lesUtilisateurs = new ArrayList();
msg = PasserelleServicesWebXML.getLesUtilisateursQuiMautorisent("juno", Outils.sha1("mdputilisateur"),
lesUtilisateurs);
// affichage de la réponse
MessageBox.Show(msg, "Test 8.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
// affichage du nombre d'utilisateurs
msg = "Nombre d'utilisateurs : " + lesUtilisateurs.Count;
MessageBox.Show(msg, "Test 8.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
// affichage de tous les utilisateurs
foreach (Utilisateur unUtilisateur in lesUtilisateurs)
{
    msg = unUtilisateur.toString();
    MessageBox.Show(msg, "Test 8.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

Comparez l'affichage obtenu avec le contenu de votre base de données et en cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7-4 la méthode demanderUneAutorisation

public static String demanderUneAutorisation(String pseudo, String mdpSha1, String pseudoDestinataire, String texteMessage, String nomPrenom)
Rôle : permet de demander une autorisation (appelle le service **DemanderUneAutorisation**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : le mot de passe hashé en sha1 **pseudoDestinataire** : le pseudo de l'utilisateur à qui on demande l'autorisation **texteMessage** : le texte d'un message accompagnant la demande **nomPrenom** : le nom et le prénom du demandeur
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode demanderUneAutorisation
msg = PasserelleServicesWebXML.demanderUneAutorisation("europa", Outils.sha1("mdputilisateurrrrrr"), "toto", "", "");
// Erreur : données incomplètes.
MessageBox.Show(msg, "Test 9.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.demanderUneAutorisation("europa", Outils.sha1("mdputilisateurrrrrr"), "toto",
"coucou", "charles-edouard");
// Erreur : authentification incorrecte.
MessageBox.Show(msg, "Test 9.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.demanderUneAutorisation("europa", Outils.sha1("mdputilisateur"), "toto",
"coucou", "charles-edouard");
// Erreur : pseudo utilisateur inexistant.
MessageBox.Show(msg, "Test 9.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.demanderUneAutorisation("europa", Outils.sha1("mdputilisateur"), "galileo",
"coucou", "charles-edouard");
// galileo va recevoir un courriel avec votre demande.
MessageBox.Show(msg, "Test 9.4", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7-5 la méthode retirerUneAutorisation

public static String retirerUneAutorisation(String pseudo, String mdpSha1, String pseudoARetirer, String texteMessage)
Rôle : permet de retirer une autorisation (appelle le service **RetirerUneAutorisation**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : le mot de passe hashé en sha1 **pseudoARetirer** : le pseudo de l'utilisateur à qui on veut retirer l'autorisation **texteMessage** : le texte d'un message pour un éventuel envoi de courriel
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode retirerUneAutorisation
msg = PasserelleServicesWebXML.retirerUneAutorisation("europa", Outils.sha1("mdputilisateurrrrrr"), "toto", "coucou");
// Erreur : authentification incorrecte.
MessageBox.Show(msg, "Test 10.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.retirerUneAutorisation("europa", Outils.sha1("mdputilisateur"), "toto", "coucou");
// Erreur : pseudo utilisateur inexistant.
MessageBox.Show(msg, "Test 10.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.retirerUneAutorisation("europa", Outils.sha1("mdputilisateur"), "juno", "coucou");
// Erreur : l'autorisation n'était pas accordée.
MessageBox.Show(msg, "Test 10.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.retirerUneAutorisation("neon", Outils.sha1("mdputilisateur"), "oxygen", "coucou");
// Autorisation supprimée ; oxygen va recevoir un courriel de notification.
MessageBox.Show(msg, "Test 10.4", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.retirerUneAutorisation("neon", Outils.sha1("mdputilisateur"), "photon", "");
// Autorisation supprimée.
MessageBox.Show(msg, "Test 10.5", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7-6 la méthode envoyerPosition

public static String envoyerPosition(String pseudo, String mdpSha1, PointDeTrace lePoint)
Rôle : permet d'envoyer la position de l'utilisateur (appelle le service **envoyerPosition**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : le mot de passe hashé en sha1 **lePoint** : un objet PointDeTrace (vide) qui permettra de récupérer le numéro attribué à partir des données fournies par le service web
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode envoyerPosition
DateTime laDate = Convert.ToDateTime("24/01/2018 13:42:21");
PointDeTrace lePoint = new PointDeTrace(23, 0, 48.15, -1.68, 50, laDate, 80);
msg = PasserelleServicesWebXML.envoyerPosition("europa", Outils.sha1("mdputilisateurrrrrr"), lePoint);
// Erreur : authentification incorrecte.
MessageBox.Show(msg, "Test 11.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
lePoint = new PointDeTrace(2333, 0, 48.15, -1.68, 50, laDate, 80);
msg = PasserelleServicesWebXML.envoyerPosition("europa", Outils.sha1("mdputilisateur"), lePoint);
// Erreur : le numéro de trace n'existe pas.
MessageBox.Show(msg, "Test 11.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
lePoint = new PointDeTrace(22, 0, 48.15, -1.68, 50, laDate, 80);
msg = PasserelleServicesWebXML.envoyerPosition("europa", Outils.sha1("mdputilisateur"), lePoint);
// Erreur : le numéro de trace ne correspond pas à cet utilisateur.
MessageBox.Show(msg, "Test 11.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
lePoint = new PointDeTrace(43, 0, 48.15, -1.68, 50, laDate, 80);
msg = PasserelleServicesWebXML.envoyerPosition("europa", Outils.sha1("mdputilisateur"), lePoint);
// Point créé.
MessageBox.Show(msg, "Test 11.4", MessageBoxButtons.OK, MessageBoxIcon.Information);
MessageBox.Show(lePoint.toString(), "Test 11.5", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7-7 la méthode getUnParcoursEtSesPoints

public static String getUnParcoursEtSesPoints(String pseudo, String mdpSha1, int idTrace, Trace laTrace)
Rôle : permet d' obtenir un parcours et la liste de ses points (appelle le service **GetUnParcoursEtSesPoints**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : son mot de passe hashé en SHA1 **idTrace** : l'id de la trace à consulter **laTrace** : objet Trace (vide) à remplir à partir des données fournies par le service web
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode getUnParcoursEtSesPoints
Trace laTrace = new Trace();
msg = PasserelleServicesWebXML.getUnParcoursEtSesPoints("europa", Outils.sha1("mdputilisateur"), 22, laTrace);
// affichage de la réponse
MessageBox.Show(msg, "Test 12.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
// affichage de la trace
msg = laTrace.toString();
MessageBox.Show(msg, "Test 12.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

Comparez l'affichage obtenu avec le contenu de votre base de données et en cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7-8 la méthode getLesParcoursDunUtilisateur

public static String getLesParcoursDunUtilisateur(String pseudo, String mdpSha1, String pseudoConsulte, ArrayList lesTraces)
Rôle : permet d' obtenir la liste des parcours d'un utilisateur (appelle le service **GetLesParcoursDunUtilisateur**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : son mot de passe hashé en SHA1 **idUtilisateur** : l'id de l'utilisateur dont on veut la liste des parcours **lesTraces** : collection (vide) à remplir à partir des données fournies par le service web
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode getLesParcoursDunUtilisateur
ArrayList lesTraces = new ArrayList();
msg = PasserelleServicesWebXML.getLesParcoursDunUtilisateur("juno", Outils.sha1("mdputilisateur"), "indigo",
lesTraces);
// affichage de la réponse
MessageBox.Show(msg, "Test 13.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
// affichage du nombre de traces
msg = "Nombre de traces : " + lesTraces.Count;
MessageBox.Show(msg, "Test 13.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
// affichage de toutes les traces
foreach (Trace uneTrace in lesTraces)
{
    msg = uneTrace.toString();
    MessageBox.Show(msg, "Test 13.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

Comparez l'affichage obtenu avec le contenu de votre base de données et en cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7-9 la méthode supprimerUnParcours

public static String supprimerUnParcours(String pseudo, String mdpSha1, int idTrace)
Rôle : permet de supprimer un parcours (appelle le service **SupprimerUnParcours**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : le mot de passe hashé en sha1 **idTrace** : l'id de la trace à supprimer
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode supprimerUnParcours
msg = PasserelleServicesWebXML.supprimerUnParcours("europa", Outils.sha1("mdputilisateurrrrrr"), 10);
// Erreur : authentification incorrecte.
MessageBox.Show(msg, "Test 14.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.supprimerUnParcours("europa", Outils.sha1("mdputilisateur"), 100);
// Erreur : parcours inexistant.
MessageBox.Show(msg, "Test 14.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.supprimerUnParcours("europa", Outils.sha1("mdputilisateur"), 22);
// Erreur : vous n'êtes pas le propriétaire de ce parcours.
MessageBox.Show(msg, "Test 14.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.supprimerUnParcours("europa", Outils.sha1("mdputilisateur"), 4);
// Parcours supprimé.
MessageBox.Show(msg, "Test 14.4", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7-10 la méthode demarrerEnregistrementParcours

public static String demarrerEnregistrementParcours(String pseudo, String mdpSha1, Trace laTrace)
Rôle : permet de démarrer l'enregistrement d'un parcours (appelle le service **DemarrerEnregistrementParcours**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : le mot de passe hashé en sha1 **laTrace** : un objet Trace (vide) à remplir à partir des données fournies par le service web
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode demarrerEnregistrementParcours
Trace laTrace = new Trace();
msg = PasserelleServicesWebXML.demarrerEnregistrementParcours("europa", Outils.sha1("mdputilisateurrrrrr"), laTrace);
// Erreur : authentification incorrecte.
MessageBox.Show(msg, "Test 15.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
laTrace = new Trace();
msg = PasserelleServicesWebXML.demarrerEnregistrementParcours("europa", Outils.sha1("mdputilisateur"), laTrace);
// Trace créée.
MessageBox.Show(msg, "Test 15.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.

7-11 la méthode arreterEnregistrementParcours

public static String arreterEnregistrementParcours(String pseudo, String mdpSha1, int idTrace)
Rôle : permet de terminer l'enregistrement d'un parcours (appelle le service **ArreterEnregistrementParcours**).
Paramètres à fournir : **pseudo** : le pseudo de l'utilisateur qui fait appel au service web **mdpSha1** : le mot de passe hashé en sha1 **idTrace** : l'id de la trace à terminer
Valeur de retour : une chaine contenant le message de réponse du service web (ou un message d'erreur lorsqu'une exception se produit lors de l'exécution).

Activez le code du **test** suivant dans le fichier **TestPasserelleServiceWebXML.cs** :

```
// test visuel de la méthode arreterEnregistrementParcours
msg = PasserelleServicesWebXML.arreterEnregistrementParcours("europa", Outils.sha1("mdputilisateurrrr"), 23);
// Erreur : authentification incorrecte.
MessageBox.Show(msg, "Test 16.1", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.arreterEnregistrementParcours("europa", Outils.sha1("mdputilisateur"), 230);
// Erreur : parcours inexistant.
MessageBox.Show(msg, "Test 16.2", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.arreterEnregistrementParcours("europa", Outils.sha1("mdputilisateur"), 5);
// Erreur : le numéro de trace ne correspond pas à cet utilisateur.
MessageBox.Show(msg, "Test 16.3", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.arreterEnregistrementParcours("europa", Outils.sha1("mdputilisateur"), 42);
// Erreur : cette trace est déjà terminée.
MessageBox.Show(msg, "Test 16.4", MessageBoxButtons.OK, MessageBoxIcon.Information);
msg = PasserelleServicesWebXML.arreterEnregistrementParcours("europa", Outils.sha1("mdputilisateur"), 43);
// Enregistrement terminé.
MessageBox.Show(msg, "Test 16.5", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Adaptez éventuellement le test au contenu de votre base de données, puis exécutez le test.

En cas d'échec, trouvez la cause du problème et relancez le test jusqu'à sa réussite.

Une fois le test réussi, désactivez à nouveau le code du test.