Interface à la base de données

L'interface à la base de données se compose de trois parties : une partie serveur et deux partie client, l'une en java pour android, l'autre en javaScript pour la page web.

La partie serveur se fait via des scripts PHP. Il existe deux scripts PHP, le premier pour écrire dans les tables de la BD est le plus souvent adressée par la page web, le second script sert à la lecture et est, pour l'heure, principale utilisé par le client android. Toutes les requêtes au script PHP se reçoivent une structure JSON en réponse.

Les partie clients Android se fait via une série de classes DbAccess. Ces classes supportent 2 type de bases de données : une locale et une distante. Le module d'accès distant envois les requêtes aux scripts PHP et décodent les réponses retournées pour encapsuler, s'il y a lieu, le résultat dans des classes facile à manipuler par l'interface usager. Le module d'accès local, gère, sous SQLite, une image qui est un sous ensemble de la base de donnés distante.

1. PHP

Toutes les requêtes aux scripts PHP reçoivent une réponse en format JSON du type :

{«Status» : «Success/Fail», [data]}

Si le «Status» est «Fail», data sera un message d'erreur de la forme :

«errMsg» : «message identifiant l'erreur»

autrement data variera selon le type de requête effectuée.

Il existe 4 scripts PHP :

|  |  |  |
| --- | --- | --- |
| index.pĥp | - | Sert principalement à permettre au client de valider la présence du serveur, prévient une forme superficielle de hackage. |
| connexion.php | - | Établi la connexion avec la base de donnée.  Possède les fonctions communes à tout les accès à la base de données |
| Writer.php | - | Script permettant d'écrire dans la base de données. |
| Reader.php | - | Script de lecture de la base de données. |

Les deux dernier fichier sont ceux actuellement adressé par les diverse applications client.

* 1. Writer.PHP

Le script «Writer.php» est le plus simple. Les routines ont une correspondance 1 pour 1 avec les tables de la base de données. Ses routines sont :

newLigue – Permet Permet d'inscrire une nouvelle ligue.

Paramètres :

- Nom\_Ligue – le nom de la ligue, type String

- ID\_Personne – identifiant du gestionnaire de la ligue, type Int.

Si «Status» est «Success», retourne le nouvel identifiant de la ligue sous la forme :

«Id» : «identifiant»

Set identifiant sera désigné ID\_Ligue.

newPersonne – Permet d'inscrire une nouvelle fiche Personne.

Paramètres :

- Nom – Nom de famille de la personne, type String

- prenom – Prénom de la personne, type String

- Date\_Naissance – Nom de famille de la personne, type Date (facultatif)

- telephone – numéro de téléphone, type String. (facultatif)

- courriel – adresse courriel de la personne, type String. (facultatif)

- adresse – son adresse, type String. (facultatif)

Seul les nom et prénom doivent absolument être présent pour créer l'entrée dans la table.

Si «Status» est «Success», retourne le nouvel identifiant de la personne sous la forme :

«Id» : «identifiant»

Set identifiant sera désigné ID\_Personne.

newEquipe – Enregistre une nouvelle équipe, liée à un ligue et une sous-ligue.

Paramètres :

- Nom\_Equipe – Nom de l'équipe , type String

- ID\_Ligue – identifiant de la ligue à laquelle appartient l'équipe, type Int

- ID\_SousLigue – identifiant de la sous-ligue à laquelle appartient l'équipe, type Int (facultatif)

Si «Status» est «Success», retourne le nouvel identifiant de l'équipe sous la forme :

«Id» : «identifiant»

Set identifiant sera désigné ID\_Equipe.

newAlignement – Inscrit une nouvelle entrée dans la table des Alignements

Paramètres :

- ID\_Personne – Nom de famille de la personne, type String

- ID\_Equipe – Prénom de la personne, type String

- ID\_Saison – Nom de famille de la personne, type Date.

- position – numéro de téléphone, type String.

- numero\_Chandail – adresse courriel de la personne, type String.

- temporaire – son adresse, type String. (facultatif)

Seul le champ temporaire peut être laissé vide pour créer l'entrée dans la table, la valeur «faux» sera inséré par le script si le champ est nul.

Si «Status» est «Success», retourne le nouvel identifiant de la ligue sous la forme :

«Id» : «identifiant»

Set identifiant sera désigné ID\_Alignement.

newSaison – Enregistre une nouvelle saison

Paramètres :

- ID\_Ligue – Identifiant de la ligue à laquelle appartient l'équipe, type Int

- Date\_Debut – Date de début de la saison, type Date.

- Date\_Fin – Date où la saison se termine, type Date.

- numero Saison – Numéro de saison, type int.

Si «Status» est «Success», retourne le nouvel identifiant de l'équipe sous la forme :

«Id» : «identifiant»

Set identifiant sera désigné ID\_Saison.

newPartie – Inscrit une nouvelle Partie dans la table..

Paramètres :

- lieu – Location ou la partie à lieu , type String

- duree – Durée de la partie, type time (facultatif)

- Date\_Partie – Date à laquelle la partie a lieu, type Date

- ID\_Equipe1 – Identifiant de l'équipe 1, type Int

- ID\_Equipe1 – Identifiant de l'équipe 2, type Int

- pointage1 – Pointage final de l'équipe 1, type Int (facultatif)

- pointage2 – Pointage final de l'équipe 2, type Int (facultatif)

Seul le champ duree et de pointage peuvent être laissé vide pour créer l'entrée dans la table.

Si «Status» est «Success», retourne le nouvel identifiant de l'équipe sous la forme :

«Id» : «identifiant»

Set identifiant sera désigné ID\_Partie.

newEvent – Inscrit une nouvelle Partie dans la table..

Paramètres :

- idBut – Identifiant de la personne qui marque un but , type Int (facultatif).

- idPasse – Durée de la partie, type Int (facultatif)

- idPartie – Durée de la partie, type Int

- idPenalite – Durée de la partie, type Int (facultatif)

- idLancer – Durée de la partie, type Int (facultatif)

Seul le champ idPartie est absolument nécessaire pour créer l'entrée dans la table, mais au moins une autre champ parmi idBut, idPenalite et idLancer doit être présent.

Si «Status» est «Success», retourne le nouvel identifiant de l'évènement sous la forme :

«Id» : «identifiant»

Set identifiant sera désigné ID\_Evenement.

* 1. Reader.php

Le script «Reader.php» plus complexe car il doit combiner les information de plusieurs tables. Ces routines sont :

validateLogin – Valide le nom d'usager et le mot de passe.

Paramètres :

userName – nom d'usager

passsWord – mot de passe e l'usager

retourne : un objet contenant

- le status (Success/Fail)

- un objet contenant l'id, nom et prénom de l'usager

- Un objet contenant une liste de Compétences

Si «Status» est «Success», retourne deux objets  :

un objet personne ayant la forme :

«personne» : {«Id» : «ID\_Personne»,

«nom» : «nom de la personne»,

«prenom» : «prenom de la personne»}

une liste d'objet competence ayant la forme :

«competence» : [{«Id» : «ID\_Personne»,

«competenceValue» : «type de competence»,  
«ligue» : «ID\_Ligue sur laquelle s'applique la competence»,  
«ligue» : «ID\_SousLigue sur laquelle s'applique la competence»,  
«equipe» : «ID\_Equipe sur laquelle s'applique la competence»},...]

listeEquipe – fait la liste des équipes d'une ligue

Paramètres :

idLigue – Id de la ligue à laquelle l'équipe apartient

retourne : un objet contenant

- le status (Success/Fail)

- Un objet contenant une liste d'équipes

Si «Status» est «Success», la liste d'objet équipe aura la forme :

«equipes» : [{«Id» : «ID\_Equipe»,

«nom» : «nom de l'équipe»},...]

listeJoueur – Fait la liste des joueurs d'une équipe

Paramètres :

idEquipe – Id de l'équipe de laquelle font parti les joueurs

retourne : un objet contenant

- le status (Success/Fail)

- Un objet contenant une liste de joueur

Si «Status» est «Success», la liste d'objet joueur aura la forme :

«Alignement» : [{«Id» : «ID\_Personne»,

«prenom» : «prénom de l'équipe»,  
«nom» : «nom de l'équipe»,  
«numeroChandail» : «numéro de chandail du joueur»},...]

listeJoueurLigue – Fait la liste des joueurs d'une ligue

Paramètres :

idLigue – Id de la ligue dans laquelle le joueur évolu.

retourne : un objet contenant

- le status (Success/Fail)

- Un objet contenant une liste de joueur

Si «Status» est «Success», la liste d'objet joueur aura la forme :

«Alignement» : [{«Id» : «ID\_Personne»,

«prenom» : «prénom de l'équipe»,  
«nom» : «nom de l'équipe»,  
«numeroChandail» : «numéro de chandail du joueur»},...]

listeLigue – Fait la liste des ligues d'un gestionnaire

Paramètres :

idGestionnaire – Id du gestionnaire (facultatif)

Si aucun gestionnaire n'est fourni, listeLigue retourne la liste de toutes les ligues connues

retourne : un objet contenant

- le status (Success/Fail)

- Un objet contenant une liste de ligues

Si «Status» est «Success», la liste d'objet ligue aura la forme :

«Ligues» : [{«Id» : «ID\_Ligue»,

«nom» : «nom de la ligue»},...]

listeLigueParMarqueur – Fait la liste des ligues pour lesquels un marqueur est autorisé

Paramètres :

idMarqueur – Id du marqueur

retourne : un objet contenant

- le status (Success/Fail)

- Un objet contenant une liste de ligues

Si «Status» est «Success», la liste d'objet ligue aura la forme :

«Ligues» : [{«Id» : «ID\_Ligue»,

«nom» : «nom de la ligue»},...]

listeGestionnaires

Paramètres :

aucun

retourne : un objet contenant

- le status (Success/Fail)

- Un objet contenant une liste de gestionnaires

Si «Status» est «Success», la liste d'objet ligue aura la forme :

«Gestionnaires» : [{«Id» : «ID\_Personne»,

«prenom» : «prénom du gestionnaire»,  
«nom» : «nom du gestionnaire »},...]

statJoueur

Paramètres :

ID\_Joueur – Id du joueur  
ID\_Saison – Id de l'équipe pour laquelle on veut les statistique du joueur (facultatif)  
ID\_Saison – Id de la saison d'intérêt (facultatif)

retourne : un objet contenant

- le status (Success/Fail)

- Un objet contenant une liste de gestionnaires

Si «Status» est «Success», Si un idSaison est spécifié, la réponse aura la forme :

«statJoueur» : {«but» : «nombre de but marqué»,

«passe» : «prénom du gestionnaire»,  
«tirBarrage» : «nombre de but en tir de barrage »,  
«penalite» : «nombre de pénalités »}

Si «Status» est «Success», si aucune saison ni équipe ne sont spécifiées, la réponse aura la forme :

«statJoueur» : [{«but» : «nombre de but marqué»,

«passe» : «prénom du gestionnaire»,  
«tirBarrage» : «nombre de but en tir de barrage »,  
«penalite» : «nombre de pénalités »,  
«equipe» : «nom de l'équipe»,  
«saison» : «numéro de saison»},...]

statEquipe

Paramètres :

ID\_Equipe – Id de l'équipe  
ID\_Saison – Id de la saison d'intérêt (facultatif)

retourne : un objet contenant

- le status (Success/Fail)

- Un objet contenant une liste de gestionnaires

Si «Status» est «Success», Si un idSaison est spécifié, la réponse aura la forme :

«statEquipe» : {«partie» : «nombre de partie jouées»

«victoire» : «nombre de victoires»,  
«defaite» : «nombre de défaites»,  
«butPour» : «nombre de but en faveur »,  
«butContre» : «nombre de but en défaveur »,  
«penalite» : «nombre de pénalités »}

Si «Status» est «Success», si aucune saison n'est spécifiée, la réponse aura la forme :

«statEquipe» : [{«partie» : «nombre de partie jouées»,

«victoire» : «nombre de victoires»,  
«defaite» : «nombre de défaites»,  
«butPour» : «nombre de but en faveur »,  
«butContre» : «nombre de but en défaveur »,  
«penalite» : «nombre de pénalités »,  
«saison» : «numéro de saison»},...]

1. Android