TEST SHEET

|  |  |
| --- | --- |
| Request ID | 003 |
| Request Title | Mis en place de la base de données |
| Applicant | Sarah Hautot |
| Request Date | 18/04/2024 |

Result:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TEST CONDUCTED BY** | **DATE / SPRINT** | **VERSION** | **RESULT (OK / NOK)** | | | |
| Internal | Rev. 1 | Rev. 2 | Rev. 3 |
| Tom Dunet | 18/04/2024 | V1.0 | OK |  |  |  |
|  |  |  |  |  |  |  |
|  | |  |  |  |  |  |

Summary:

|  |
| --- |
| Communication DB/PHP (Récupération, modification et suppression des données) |

Initial test conditions:

|  |
| --- |
| Base de données MySQL crée.  Informations sur la connections à la base dans le fichier DB.inc.php (ne pas exiter à modifier si besoin).  Serveur Apache et MySQL lancé.  Connecter vous à coter à la base de donnée pour verifier la présence des données.  Tous les objets php sont dans le même dossier que le fichier DB.inc.php |

Test steps:

|  | **ACTION** | **RESULT** | **COMMENT** | **TEST (OK / NOK)** | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Internal | Rev. 1 | Rev. 2 | Rev. 3 |
| 1 | Créer un fichier php ou vous incluer le fichier DB.inc.php. Crée un objet DB.inc.php avec la commande **DB ::getInstance()** | Aucune erreur s’affiche.  Si oui, assurez vous que les données de connection soit correct. |  | OK |  |  |  |
| 2 | Crée un utilisateur avec le login *loginTest* et le mail *emailTest@gmail.com*, en passant un objet Utilisateur à la méthode **insertUtilisateur().**  Verifier sa présence en affichant la liste des Utilisateur avec la méthode **getUtilisateurs()** | Vous avez les utilisateurs de la base + celui que vous avez crée.  Sur votre base de donée, vous avez les même informations. |  | OK |  |  |  |
| 3 | Prenez un utilisateur au hasard de la liste donner, et modifier son login pour qu’il est le même login qu’un des utilisateurs. Metez à jour les données avec **updateUtilisateur()** (prend un utilisateur en paramettre). | Un message apparait comme quoi le login existe déjà.  Les modification n’ont pas était apporté. |  | OK |  |  |  |
| 4 | Essayez d’inserer cet utilisateur avec la méthode **insertUtilisateur().** | Un message apparait comme quoi le login existe déjà.  Les modification n’ont pas était apporté. |  | OK |  |  |  |
| 5 | Remetez un login différent, puis modifier l’utilisateur crée précédemment pour qu’il est le même email qu’un des utilisateurs et metez à jour les données avec **updateUtilisateur()** (prend un utilisateur en paramettre). | Un message apparait comme quoi l’email est déjà utilisé.  Les modification n’ont pas était apporté. | Erreur dans les requet lorsque même nom qui a était corrigé par le développeur | OK |  |  |  |
| 6 | Essayez d’inserer cet utilisateur avec la méthode **insertUtilisateur().** | Un message apparait comme quoi l’email est déjà utilisé.  Les modification n’ont pas était apporté. |  | OK |  |  |  |
| 7 | Supprimer le dernier utilisateur de la liste avec **suppUtilisateur().** | Les modifications ont était apporté. |  | OK |  |  |  |
| 8 | Créer un objet produit et ajouter le dans la base avec la méthode **insertProduit()** (prend un objet produit en paramètre) | Les modifications ont était apporté. | Erreur dans les requet pour prixUni qui est un float qui a était corrigé par le développeur | OK |  |  |  |
| 9 | Afficher les produits avec la méthode **getProduits()** | Les produits sont afficher et trier par categorie. |  | OK |  |  |  |
| 10 | Prenez un des produits renvoyer et modifier ses informations.  Mettez le à jour dans la base avec la méthode **updateProduit()** (prend un objet produit en paramètre) | Les modifications ont était apporté. |  | OK |  |  |  |
| 11 | Supprimer un produits au hasard dans la liste avec la méthode **suppProduit()** | Les modifications ont était apporté. |  | OK |  |  |  |
| 12 | Créer un objet ticket et ajouter le dans la base avec la méthode **insertTicket()** (prend un objet ticket en paramètre) | Les modifications ont était apporté. | Erreur dans les requet pour prixTotal qui est un float qui a était corrigé par le développeur | OK |  |  |  |
| 13 | Afficher les tickets d’un client avec la méthode **getTicketClient()** (prend l’objet client en paramètres) | Les modifications ont était apporté.  Affiche les bonnes données. |  | OK |  |  |  |
| 14 | Prenez un des ticket renvoyer et modifier ses informations.  Mettez le à jour dans la base avec la méthode **updateTicket()** (prend un objet ticket en paramètre) | Les modifications ont était apporté.  (je ne teste pas les valeurs négatifs, c’est à coté de verifié) |  | OK |  |  |  |
| 15 | Supprimer un produits au hasard dans la liste avec la méthode **suppTicket ()** (prend un objet ticket en paramètre) | Les modifications ont était apporté. |  | OK |  |  |  |
| 16 | Créer un objet historique et ajouter le dans la base avec la méthode **insertHistorique()** (prend un objet historique en paramètre) | Les modifications ont était apporté.  (type uniquement : TICKET ou SECU, je verifie pas car ca sera automatique cette partie) |  | OK |  |  |  |
| 17 | Afficher les historique des fiche de secu avec la méthode **getHistoriqueSecu()** | Affiche les bonnes données. |  | OK |  |  |  |
| 18 | Afficher les historique des ticket avec la méthode **getHistoriqueTicket()** | Affiche les bonnes données |  | OK |  |  |  |
| 19 | Afficher les historique des fiche de secu d’un client avec la méthode **getHistoriqueSecuClient()** (prend l’objet client en paramètres) | Affiche les bonnes données. |  | OK |  |  |  |
| 20 | Afficher les historique des tickets d’un client avec la méthode **getHistoriqueTicketClient()** (prend l’objet client en paramètres) | Affiche les bonnes données. |  | OK |  |  |  |
| 21 | Supprimer un historique au hasard dans la liste avec la méthode **suppHistorique ()** (prend un objet historique en paramètre) | Les modifications ont était apporté. |  | OK |  |  |  |
| 22 | Récupérer tous les droits de la base avec la méthode **getDroits()** | Affiche les bonnes données. |  | OK |  |  |  |
| 23 | Récupérer les droits d’un utilisateur avec **getDroitUtilisateur()** (prend en paramètre utilisateur) | Affiche les bonnes données. |  | OK |  |  |  |
| 24 | Ajouter un droit à un utilisateur avec la méthode **insertDroitUtilisateur()** (prend en paramètre utilisateurdroit) | Les modifications ont était apporté. |  | OK |  |  |  |
| 25 | Supprimer un droit à un utilisateur avec la méthode **suppDroitUtilisateur()** (prend en paramètre utilisateurdroit) | Les modifications ont était apporté. |  | OK |  |  |  |
| 26 | Crée un client avec le nom de club *nomClubDeTest* en passant un objet Client à la méthode **insertClient().**  Verifier sa présence en affichant la liste des client avec la méthode **getClients()** | Les modifications ont était apporté. |  | OK  OK |  |  |  |
| 27 | Prenez un client au hasard de la liste donner, et modifier son nom de club pour qu’il est le même nom de club qu’un des client. Metez à jour les données avec **updateClient()** (prend un client en paramettre). | Un message apparait comme quoi le nom de club existe déjà.  Les modification n’ont pas était apporté. |  | OK |  |  |  |
| 28 | Essayez d’inserer cet utilisateur avec la méthode **insertClient().** | Un message apparait comme quoi le nom de club existe déjà.  Les modification n’ont pas était apporté. |  | OK |  |  |  |
| 29 | Remetez un nom de club différent et modifier des données. Mettez le à jour dans la base avec la méthode **updateClient()** (prend un client en paramettre). | Les modifications ont était apporté. |  | OK |  |  |  |
| 30 | Supprimer le dernier client de la liste avec **suppClient().** | Les modifications ont était apporté. |  | OK |  |  |  |

Remarks:

|  |
| --- |
| Des erreurs dans des méthodes qui ont tout fait planté et qui nous ont paralizé pour le développement du backend. |