**Manuel utilisateur pour application #GYM**

* Introduction

C’est une application qui sert à gérer les opérations quotidiennes des utilisateurs au #GYM. Elle est divisée en deux parties :

1. **Partie « menu » :** sert à aider l’agent au comptoir pour gérer les demandes des clients du #GYM.
2. **Partie mobile :** sert à faciliter la vie aux clients du #GYM et éviter la perte du temps des clients en attendant un agent au comptoir pour répondre à leurs demandes.

* Séquencement
* À partir de la classe GYM qui fait représente la classe main de notre logiciel. Cette classe créée une instance « MenuGYM » qui sert à appeler la classe portant le même nom pour commencer le déroulement de l’application.
* La classe « MenuGYM » à son tour va créer une instance « BaseDonnées » qui représentera la base de données de notre application.
* Le but de la création de cette base de données sera d’enregistrer toutes les données du #GYM dans des « Dictionnaires » ou « Arraylists » qui contiendront les « utilisateurs » en général du #GYM, « les professionnels », « les membres », « les séances » créées, « les transaction », « les employées » et « les courriels et les codes QR des utilisateurs ».
* L’application va demander si on est sur l’application mobile ou non.
  1. **Si « oui » (MenuMob) :** 
     + Le menu mobile sera affiché et le système va demander un courriel facebook valide du client. À travers ce processus, deux méthodes seront appelées :

**« verifierCourriels » :** Pour vérifier la présence de ce courriel dans la base de données des clients enregistrés.

**« verifierCourriel » :** Pour vérifier la syntaxe de l’adresse courriel elle-même.

* + - Ensuite, le système va vérifier l’état de l’utilisateur si c’est un « pro » ou « membre » pour afficher le bon menu.
      * 1. **Membre :** Il affichera un menu à travers « afficherMenuMob » pour les services dont le membre aura besoin (inscrire séance, confirmer présence à une séance).

1. **Inscrire séance :** À l’aide de l’opération « Seance.inscrireSeance ».

2. **confirmer présence :** En utilisant la méthode

**« Seance.confirmerPresence »** : détectera l’état de l’utilisateur, si membre va afficher son QR code si sa confirmation est validée. Si c’est un pro, un code membre sera demandé afin de valider son accès à la séance.

En plus, si le membre n’est pas inscrit dans la séance, le professionnel pourra l’inscrire à travers « Seance.inscrireSeance » et puis il confirmera sa présence.

* + - * 1. **Professionnel :** Il pourra confirmer la présence d’un membre ou consulter ses inscriptions pour la semaine courante à travers « Seance.consulterInscriptions ».
        2. **Il y en des options communes :** accèder au #GYM (« BaseDonnees.validerAccesMob »), changer utilisateur (« verifierClient ») et quitter (« quitOrContinue »).
  1. **Sinon (menuReception) :**
* À travers « menuReception » l’agent pourra gérer un utilisateur si c’est nouveau ou pas (« gérer ») dont les méthodes seront « Utilisateur.adherer(Service.creerSeancesMenu), actualiser et supprimer ».
* La validation des données sera faite à travers la base des données en utilisant la méthode « verifierData » et ses vérificateurs déversés. (426)
* « ProceduresComptable » : Méthode responsable à envoyer les rapports aux utilisateurs et à RnB.
* La création d’un nouveau service va créer un ensemble des séances dont le nombre sera la période totale en jours divisé par le nombre de récurrence.
* Une nouvelle inscription ajoutera (séance dans séances pris du membre), (inscription dans inscriptions de la séance) et (transaction dans transactions quotidienne afin de l’envoyer à RnB).
* Les dictionnaires de la base de données ont les clés comme le nombre de la semaine actuelle à travers l’année et lorsqu’on fait les comparaisons pour extraire les données d’une telle semaine, il suffit juste de comparer la clé avec la semaine actuelle.

Le total des mots : 533 mots.