Projet de fin de session

Création d'un réseau social de la classe

Dans ce projet, vous allez devoir créer le réseau social **TeccSocial**
Pour obtenir la totalité des points, vous devrez définir toutes les fonctions du projet.
Vous pouvez aussi gagner des points bonus en faisant l'une des tâches "bonus"

Déscription globale du projet

Le **TeccSocial** permet de connecter les étudiants de la classe.
Vous allez devoir permettre l'inscription des utilisateurs, l'ajout d'ami, l'ajout des postes, les conversations entre amis et d'autres fonctionnalités intéressantes.
Le but de ce projet est de vous familiariser avec les différentes notions apprises en classe, le développement de vos compétences de programmation et la découvertes de nouvelles choses intéressantes en **python**

La structure du projet

- Le projet va utiliser des fichiers **json** pour stocker les informations (base de données)
- Le projet aura des classes pour gérer la lecture/écriture des fichiers
- Le fichier sera composé de plusieurs fichiers, chaque fichier contiendra un groupe de fonctionnalités précises

Le menu

Quand on lance l'application, nous devrons trouver le menu ci-dessous:

```
1- Se connecter
2- S'inscrire
3- Réinitialiser les données
4- Quitter
```

1- Se connecter

Ce menu permet de connecter un utilisateur qui est déjà inscrit dans le système

```
Entrer un nom d'utilisateur pour se connecter:
```

Une fois connecté, vous devrez voir le menu suivant:

```
1- Mes amis
2- Poster
```

```
3- Voir le mur
4- Mes conversations
5- Se déconnecter
```

1- Mes amis

Dans ce menu, nous allons voir un autre menu comme celui ci:

```
1- Voir la liste des amis
2- Ajouter un ami
3- Supprimer un ami
4- Retourner
```

Voir la liste des amis

Dans ce choix, nous allons voir un autre menu qui permet de choisir la façon d'affichage de ma liste d'amis

```
1- Affichage normal
2- Affichage par nom d'utilisateur
3- Affichage par nom de famille
4- Affichage par prénom
```

Selon le choix, nous allons voir la liste selon l'ordre souhaité

Ajouter un ami

Cela va nous permettre d'ajouter un ami en demandant son nom d'utilisateur

```
------Ajouter un nouveau ami------
Entrer le nom d'utilisateur du nouveau ami:
```

Supprimer un ami

Cela va nous permettre de supprimer un ami de ta liste d'amis

```
------Supprimer un ami------
Entrer le nom d'utilisateur à supprimer:
```

2- Poster

Cela va permettre d'ajouter un nouveau post dans le mur de tes amis.
Une fois on clique sur Poster, un text va nous demander de poster un message:

```
------Ajouter un nouveau post------
Entrer le contenu du post:
```

On ne devrait pas accépter un post vide

3- Voir le mur

Ce menu nous permet de voir le mur qui contient les posts de nos amis. Attention, dans cette application, l'utilisateur ne voit pas ses propres posts.
Un menu nous permet de choisir l'ordre d'affichage des posts

```
** --- Choisir l'une des options suivantes pour l'ordre d'affichage: **
1- Affichage du plus récent au plus ancien
2- Affichage du plus ancien au plus récent
```

Si le mur est vide, nous allons voir un message qui le précise. Sinon nous allons afficher les posts selon l'ordre choisi.

4- Mes conversations

Ce menu me permet de voir la liste des conversations que je peux voir.
L'utilisateur peut voir tous ses amis dans une liste, et choisir à qui il veut envoyer des messages ou voir sa conversation avec.

```
** --- Choisir la conversation à continuer: **
1- pnegri1
2- lplaxton2
3- wraiman3
4- mfilanx
5- cboonec
```

Une fois une conversation est choisi, on peut la lire, écrire un message ou bien tapez FIN pour sortir.
Si par exemple on choisi une conversation:

```
-> fgentreau0: Salut
-> fgentreau0: Comment ça va
-> pnegri1: Salut, je vais bien et toi
-> fgentreau0: Je vais bien merci. Comment tu vas aujourd'hui ?
-> fgentreau0: Je suis au café
```

5- Se déconnecter

Ce choix nous permet de se déconnecter et revenir au menu initial

La suite du menu initial

```
1- Se connecter
2- S'inscrire
3- Réinitialiser les données
4- Quitter
```

2- S'inscrire

Ce menu permet d'inscrire un nouveau utilisateur en demandant ses informations.

Il faut bien sûr vérifier si le nom d'utilisateur demandé est disponible

3- Réinitialiser les données

Ce choix permet de simplement de vider tous les fichiers de l'application pour pouvoir commencer du 0.
Faites attention ! Cela va vider tous les données que vous avez enregistré sur l'application

4- Quitter

Cela devrait mettre fin à l'application et quitter

Les données

Les fichiers qui préservent les données se trouvent sous le dossier **classes/database**

Nous avons 4 fichier au total:

- users.json
- friends.json
- posts.json
- conversations.json

Le fichier users.json

Dans ce fichier, nous stockons tous les utilisateurs qui se trouvent dans le système.
Le fichier contient une liste JSON sous la forme suivante:

```
[
    {
        "username": "user1",
        "firstName": "prenom1",
        "lastName": "nom1",
        "dateOfBirth": "2000-01-01"
    },
    {
        "username": "user2",
        "firstName": "prenom2",
        "lastName": "nom2",
        "dateOfBirth": "2000-01-02"
    }
]
```

Chaque utilisateur a:

- username: Nom d'utilisateur unique
- firstName: Le prénom de l'utilisateur
- lastName: Le nom de famille de l'utilisateur

- dateOfBirth: La date de naissance de l'utilisateur

Le fichier friends.json

Dans ce fichier, nous préservons les amis de chaque utilisateurs sous la forme suivante:

```
[
    {
        "username": "user1",
        "friends": [
            "user2",
            "user3"
        ]
    },
    {
        "username": "user2",
        "friends": [
            "user1"
        ]
    },
    {
        "username": "user3",
        "friends": [
            "user1"
        ]
    },
]
```

Donc, chaque utilisateur a une entrée unique dans cette liste:

- username: L'utilisateur à qui la liste d'amis appartient
- friends: La liste des amis de cet utilisateur

Le fichier posts.json

Ce fichier contient tous les posts que n'importe quel utilisateur a publié sous la forme:

```
[
    {
        "username": "user1",
        "content": "Premier poste de l'application",
        "postDate": "2022-07-24T15:59:56.641543"
    },
    {
        "username": "user1",
        "content": "Il fait beau aujourd'hui",
        "postDate": "2022-07-24T18:59:56.641543"
    },
    {
        "username": "user2",
        "content": "Belle visite au cinéma",
        "postDate": "2022-07-25T12:59:56.641543"
    },
    {
        "username": "user1",
        "content": "Les exams s'approchent...",
        "postDate": "2022-07-26T12:59:56.641543"
    },
]
```


Dans cette liste de posts, chacun contient:

- username: Le nom d'utilisateur qui a publié le post
- content: Le contenu (message) du post
- postDate: Le moment (date et heure) de publication du post

Le fichier conversations.json

Ce dernier fichier contient toutes les conversations de tous les utilisateurs sous la forme:

```
[
    {
        "user1": "goku",
        "user2": "vegeta",
        "messages": [
            {
                "sender": "goku",
                "message": "Salut, comment ça va ?"
            },
            {
                "sender": "vegeta",
                "message": "Yo !"
            },
            {
                "sender": "vegeta",
                "message": "Je vais bien et toi ?"
            },
            {
                "sender": "goku",
                "message": "Trop occupé, je prépare pour les examens"
            },
            {
                "sender": "vegeta",
                "message": "Bonne chance !"
            },
        ]
    },
]
```

Chaque conversation entre 2 utilisateurs est unique. Donc user1, user2 == user2, user1.

- user1: utilisateur qui fait partie de la conversation
- user2: utilisateur qui fait partie de la conversation
- messages: La liste des messages dans la conversation. Chaque message contient:
  - sender: L'utilisateur qui a envoyé le message.
  - message: Le message écrit

Le travail à faire

Tout ce qui est décrit dans cette section représente le travail que vous devrez faire pour réussir le projet.
Dans le code, il y aura toujours un commentaire avec le signe **TODO:** qui indique qu'est ce qu'il faut compléter
Vous devrez soit compléter ou définir les fonctions suivantes:

- Compléter la méthode **emptyFiles()** dans la classe **DatabaseConnector**
- Compléter la méthode **signup()** dans la classe **Application**
- Implémenter la méthode **_friends()** dans la classe **Application**

- Définir la classe **Post** dans le fichier **post.py** tel que décrit dans le fichier
- Implémenter les deux méthodes **showByLastName() et showByFirstName()** pour afficher la liste dans l'ordre par nom de famille ou par prénom dans la classe **FriendList**
- Compléter le bout manquant dans la méthode **_showConversation()** pour ajouter un message dans la conversation dans la classe **Application**
- Compléter les méthodes **readFriends()** et **writeFriends()** dans la classe **DatabaseConnector** qui permettent de lire de et écrire sur le fichier friends.json
- Implémenter la méthode **_deleteFriend()** qui permet de supprimer un ami et qui se trouve dans la classe **Application**
- Définir la classe **ConversationMessage** qui se trouve dans le fichier **conversation.py**
- Implémenter la méthode **__showList()** dans la classe **PostList** qui permet d'afficher la liste des posts

Les tâches bonus

Pour obtenir le 10% de bonus. Vous devrez soit effectuer les deux tâches suivantes:

- Mettre votre projet dans un repository privé GIT et le partager avec moi: achaara@teccart.online (mailto:achaara@teccart.online) (5%)
- Un rapport qui décrit les difficultés et les recherches/choses que vous aviez fait pour les résoudre durant ce projet (5%)

Ou bien faire la tâche suivante:

- Ajouter la possibilité de poster une image dans le post (10%). Ajouter un nouveau menu: Poster image

Le total du bonus ne dépassera jamais 10%, donc même si vous faites les 3 tâches, vous n'allez pas obtenir 20% de bonus.