

Compilation et lancement

Afin de lancer le programme, exécuter dans l'ordre les commandes suivantes:

1. Exécuter la commande : `make`
2. Puis lancer le server avec : `./bin/server`
3. Puis lancer le client avec : `./bin/client 127.0.0.1`

Côté client

Une fois que la connexion est établie avec le serveur, le client aura la possibilité de s'authentifier en se connectant à son compte ou en créant un nouveau compte.

Pour créer un nouveau compte :

```
/register <username> <password>
```

Pour se connecter à son compte :

```
/login <username> <password>
```

Liste des fonctionnalités implémentées

Fonctionnalités	Commandes	Détails
Envoi d'un message public	<code>/public <message></code>	Un message est envoyé à tous les utilisateurs connectés
Envoi d'un message privé	<code>/private <username> <message></code>	Un message est envoyé à un seul utilisateur. Si le destinataire est déconnecté au moment de l'envoi, celui ne recevra le message qu'une fois reconnecté
Envoi d'un message dans un groupe	<code>/group <groupname> <message></code>	Un message est envoyé à tous les membres du groupe connectés au moment de l'envoi
Création d'un groupe	<code>/create <groupname></code>	Création d'un nouveau groupe

Fonctionnalités	Commandes	Détails
Invitation à rejoindre un groupe	<code>/invite <groupname> <username></code>	Toute personne membre d'un groupe, pourra inviter un autre utilisateur à rejoindre le groupe
Rejoindre un groupe	<code>/join <groupname></code>	Un utilisateur pourra rejoindre un groupe seulement si ce dernier y a été invité

La commande ci-dessous permet de lister toutes les commandes possibles :

```
/help
```

Protocole de communication

Afin de réaliser la communication entre le serveur et le client, nous avons procédé de la manière suivante :

- Le client peut envoyer une requête au serveur via l'objet **Request** :

```
typedef struct{
    request_type type;           // Le type de la requête
    int paramCount;             // Le nombre de paramètres envoyés
    char params[PARAM_NUMBER][PARAM_SIZE]; // Les paramètres de la requête
    Message message;           // Le message à envoyer si nécessaire
} Request;
```

- Le serveur répondra au client avec un objet **Response** :

```
typedef struct{
    response_type type;         // Le type de la réponse
    int paramCount;            // Le nombre de paramètres envoyés
    char params[PARAM_NUMBER][PARAM_SIZE]; // Les paramètres de la réponse
    Message message;          // Le message à envoyer si nécessaire
} Response;
```