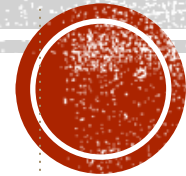




ECOLE MAROCAINE DES  
SCIENCES DE L'INGENIEUR  
Membre de  
HONORIS UNITED UNIVERSITIES

# DÉVELOPPEMENT D'UNE APPLICATION CHATSERVER

*Année Universitaire 2022/2023*  
*Réalisée par: CHAFIKI ZINEB*



# TABLE DE MATIÈRE

01

Introduction

02

Cahier de charge

03

Réalisation

04

Conclusion

01

# INTRODUCTION

# PRÉSENTATION DE L'APPLICATION

- & Une application client-serveur est un modèle d'architecture dans lequel les tâches sont réparties entre deux parties distinctes : le client et le serveur.
- & Le client est responsable de l'interaction avec l'utilisateur et envoie des requêtes au serveur.
- & Le serveur reçoit les requêtes des clients, les traite et renvoie les résultats appropriés aux clients.

# RÔLES DU SERVEUR:

**Gestion des connexions par des sockets**

**Coordination des échanges à l'aide des threads**

**Contrôle des clients connectés**

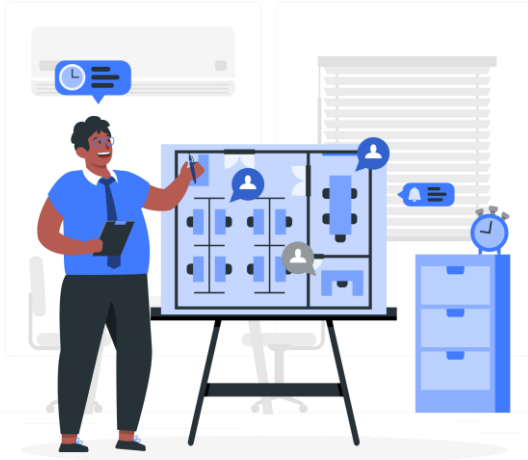


02

# CAHIER DE CHARGE



# SPÉCIFICATIONS FONCTIONNELLES



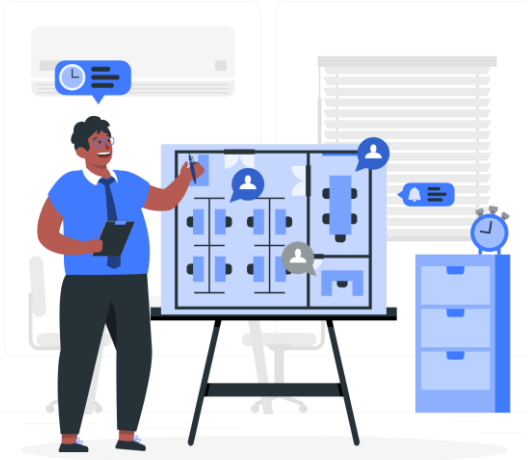
## ⑧ Établissement des connexions :

Le serveur utilise la classe `ServerSocket` pour écouter les connexions entrantes sur le port 8080.

Lorsqu'un client se connecte, le serveur accepte la connexion et lui attribue un identifiant unique en augmentant le compteur `clientNbre`.

Le serveur crée une nouvelle instance de la classe `Communication` pour gérer la communication avec le client.

# SPÉCIFICATIONS FONCTIONNELLES



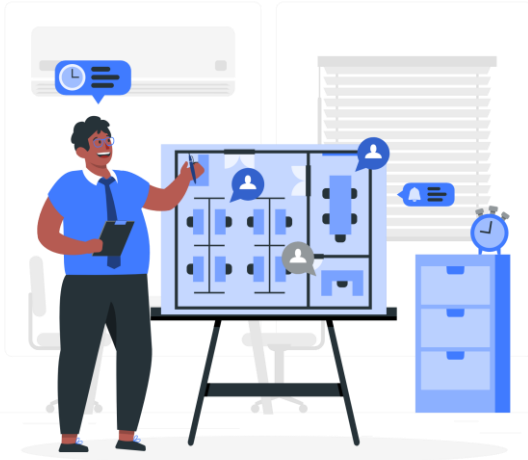
## ③ Communication entre le serveur et les clients :

Le serveur utilise des flux d'entrée et de sortie pour recevoir et envoyer des messages entre le serveur et les clients.

Les messages envoyés par les clients sont relayés à tous les autres clients connectés, permettant ainsi une communication de groupe.



# SPÉCIFICATIONS FONCTIONNELLES



## ⑧ Gestion des erreurs:

Le code gère les erreurs de connexion et de communication en capturant les exceptions `IOException` et en les traitant de manière appropriée.

Lorsqu'une erreur se produit, le serveur affiche une trace de la pile d'erreurs et continue d'écouter les connexions entrantes.

# SPÉCIFICATIONS **NON FONCTIONNELLES**

1. Langage de programmation Java
2. Code bien structuré, maintenable et respecte les bonnes pratiques de développement.
3. Capacité de gérer un nombre important de connexions clientes sans compromettre les performances.
4. Sécurité des communications et l'intégrité des données échangées.



03

# RÉALISATION DE L'APPLICATION



01

## Outils utilisés :



**IntelliJ IDEA**

un IDE puissant et polyvalent utilisé pour le développement en Java.



**Scene Builder**

un outil visuel permettant de créer des interfaces graphiques JavaFX.

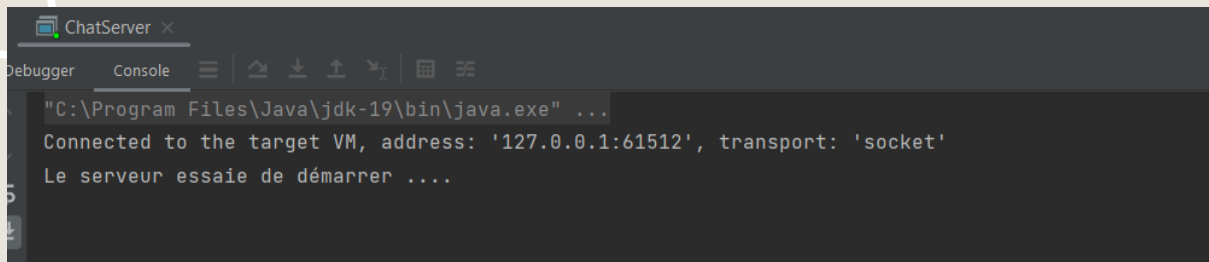
**JavaFX**



un Framework graphique moderne utilisé pour la création d'applications de bureau riches en fonctionnalités.

02

## Enregistrements Serveur



```
"C:\Program Files\Java\jdk-19\bin\java.exe" ...  
Connected to the target VM, address: '127.0.0.1:61512', transport: 'socket'  
Le serveur essaie de démarrer ....
```



Serveur exécute





02

## Espace Client :

Surface de connexion du client

The screenshot shows a web browser window with the title 'Client's Chat APP!'. The page content is as follows:

# CHATSERVER

## Login Account

Username

Password

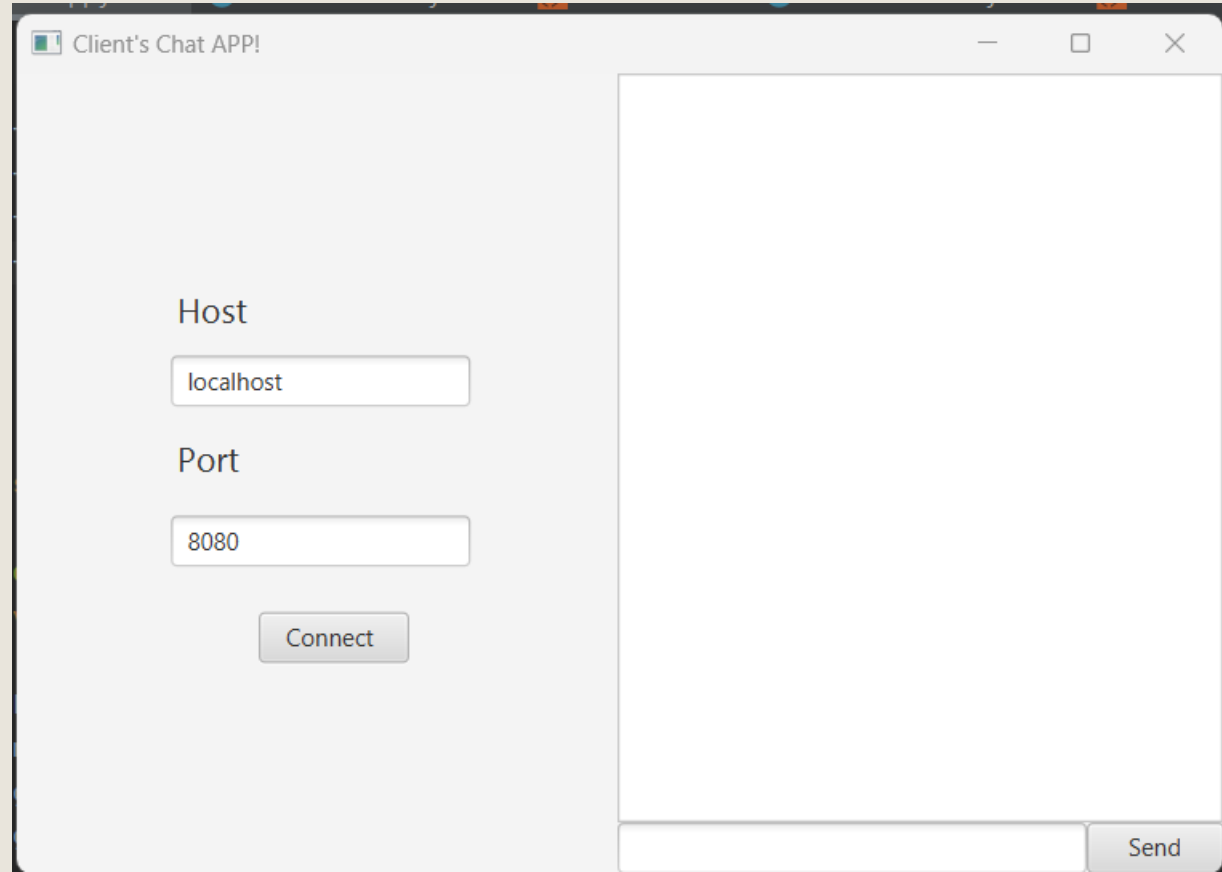
Login

[Forget Password](#)

02

## Espace Client :

Interface de connexion  
au serveur local et au  
port



The screenshot shows a web application window titled "Client's Chat APP!". The interface is light gray and contains the following elements:

- Host:** A text input field containing the value "localhost".
- Port:** A text input field containing the value "8080".
- Connect:** A gray button with the text "Connect" centered below the port field.
- Chat Area:** A large, empty white rectangular area on the right side of the window.
- Input and Send:** At the bottom, there is a white input field and a gray "Send" button.

02

## Espace Client :

Interface de client  
d'envoi des  
messages

Client's Chat APP!

Host

localhost

Port

8080

Connect

Vous êtes le client 1

Envoyez le message que vous voulez .... :D

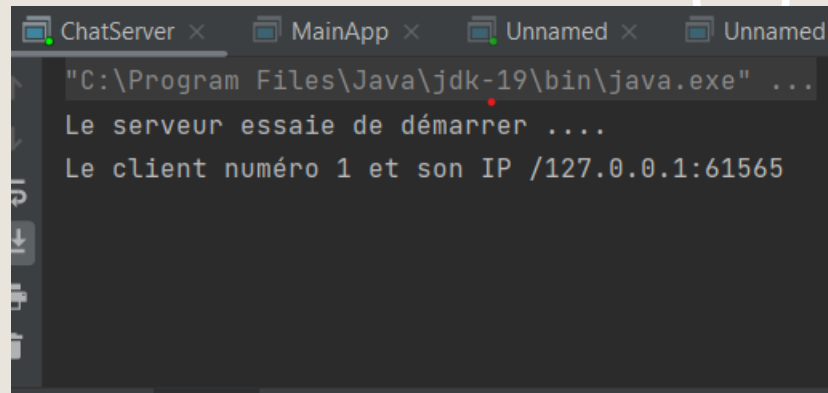
Send



02

## Espace Serveur:

Visualisation du console serveur lors de la connexion du 1er client



The screenshot shows a Java IDE with four tabs: ChatServer, MainApp, Unnamed, and Unnamed. The ChatServer tab is active, displaying the following text in its console:

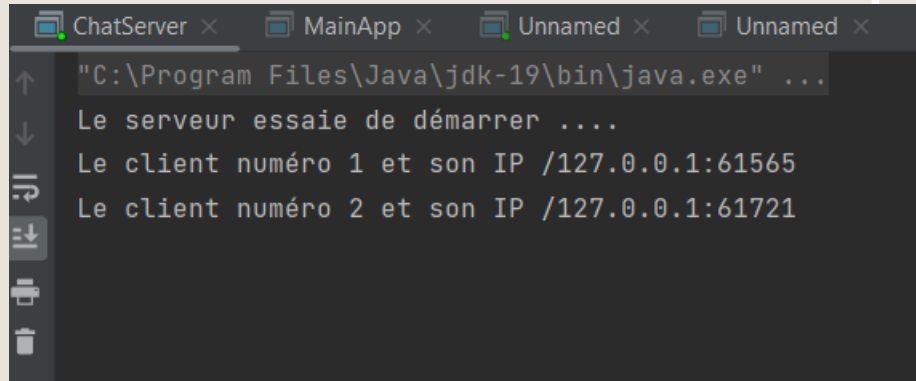
```
"C:\Program Files\Java\jdk-19\bin\java.exe" ...  
Le serveur essaie de démarrer ....  
Le client numéro 1 et son IP /127.0.0.1:61565
```



02

## Espace Admin :

**Visualisation du console serveur lors de la connexion du 2eme client**



The screenshot shows a Java IDE with four tabs: ChatServer, MainApp, Unnamed, and Unnamed. The ChatServer tab is active, displaying the following text in the console:

```
"C:\Program Files\Java\jdk-19\bin\java.exe" ...  
Le serveur essaie de démarrer ....  
Le client numéro 1 et son IP /127.0.0.1:61565  
Le client numéro 2 et son IP /127.0.0.1:61721
```

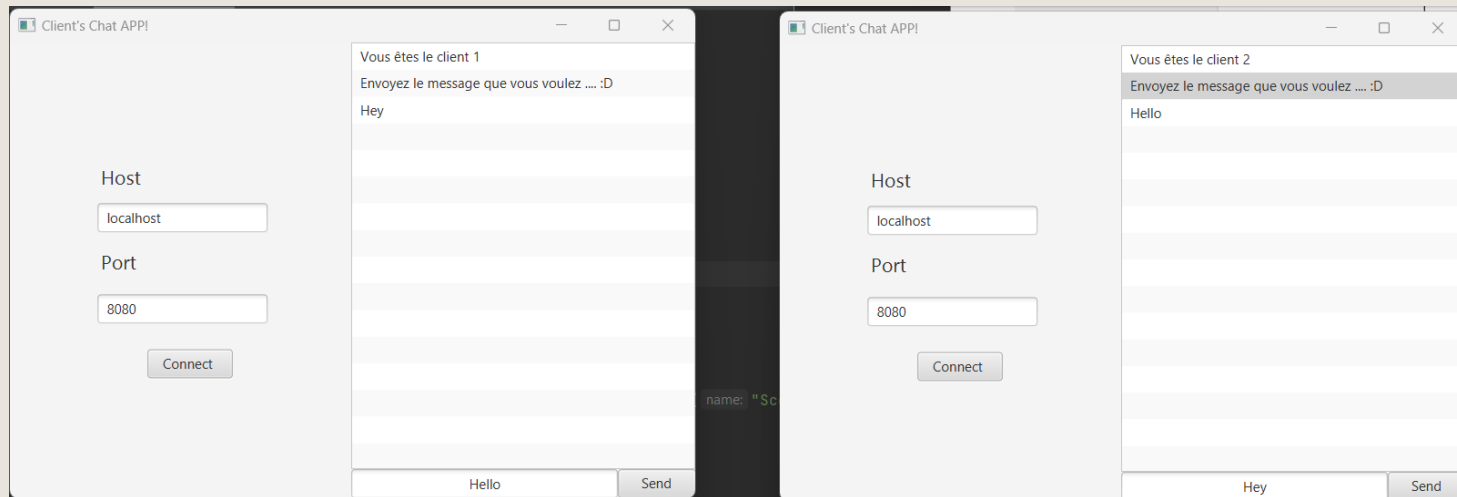
A white circle highlights the top right corner of the IDE window.





## Espace Admin :

**Visualisation des interfaces d'envoi des messages**





04

# CONCLUSION



**MERCI DE VOTRE ATTENTION**