

Cahier des Charges

Sommaire

Cahier des Charges

Sommaire

0. Légende

1. Contexte du Projet

2. Objectifs du Projet

3. Fonctionnalités Principales

3.1 Messagerie en temps réel

3.2 Architecture modulaire

4. Technologies utilisées

5. Architecture du Projet

6. Présenté par

0. Légende

 → Optionnel

1. Contexte du Projet

L'application développée est une plateforme de messagerie instantanée de type **Discord-like**, destinée aux utilisateurs souhaitant échanger en temps réel via une interface de bureau moderne et intuitive.

2. Objectifs du Projet

- Développer une application de chat modulaire et professionnelle.
- Permettre une communication en temps réel via WebSocket.
- Proposer une interface utilisateur simple et intuitive.
- Implémenter une architecture claire de type client-serveur.
- Gérer l'authentification des utilisateurs et la création de salons de discussion.

3. Fonctionnalités Principales

3.1 Messagerie en temps réel

- Utilisation de STOMP sur WebSocket pour les échanges en temps réel.
- Prise en charge de plusieurs salons (multi-room).
- Authentification des utilisateurs.
- Système d'épingle et suppression d'épingle d'un message.
- Système de status (En ligne, Inactif, Ne Pas Déranger, Hors ligne)
- Système de demande d'amis avec possibilité d'approbation ou refus en temps réel

- Ajout et personnalisation de l'avatar utilisateur avec un système de presets disponibles

3.2 Architecture modulaire

- Séparation claire entre les modules client, serveur et communs.
- Utilisation de bibliothèques partagées (modèle, utilitaires, etc.).

4. Technologies utilisées

- Java → Back-End
- JavaFX → Front-End
- CSS → Front-End
- SQL (MariaDB) → Back-end / Base de données
- Spring Boot → Back-End (Serveur WebSocket)
- STOMP/WebSocket → Communication en temps réel
- Log4j2 → Journalisation
- JUnit 5 & Mockito → Tests unitaires

5. Architecture du Projet

L'application est divisée en trois modules principaux :

- **shared** : Bibliothèques partagées, modèles de données, utilitaires communs.
- **server** : Application Spring Boot qui gère les connexions WebSocket et le routage des messages.
- **client** : Application JavaFX côté client, permettant aux utilisateurs d'interagir avec le serveur.

6. Présenté par

PIECOURT ESTÉBAN, YANIS ZINE, PALANGA GAUVAIN