

Descriptif de la Réalisation Professionnelle, y compris les productions réalisées et schémas explicatifs

Objectifs et besoins du projet

Ce projet ayant pour but de réaliser une application de messagerie en temps réel avec une architecture client-serveur moderne, nous avons défini les besoins tels que ci-dessous :

- **Gestion des utilisateurs**
 - Création et authentification des comptes utilisateurs
 - Profils utilisateurs personnalisables
 - Gestion des droits et permissions
- **Gestion des conversations**
 - Support pour conversations individuelles entre deux utilisateurs
 - Support pour conversations de groupe avec plusieurs participants
 - Support pour canaux de diffusion publique
 - Gestion des participants et de leurs rôles
- **Gestion des messages**
 - Envoi et réception de messages en temps réel
 - Support pour différents types de contenus (texte, fichiers)
 - Épinglage de messages importants
 - Suppression de messages
 - Option de messages anonymes
- **Interface utilisateur**
 - Application desktop intuitive et réactive
 - Affichage en temps réel des statuts et messages
 - Navigation fluide entre les différentes conversations
 - Support multi-plateforme (Windows et Linux)
- **Sécurité et fiabilité**
 - Protection des données utilisateurs
 - Transmission sécurisée des messages
 - Stabilité et performance des connexions en temps réel

Réalisations et implémentations

Dans la réalisation de ce projet, j'ai eu la chance de développer les éléments suivants :

Architecture du système

- **Mise en place d'une architecture modulaire à trois composants**

- Module commun (chat-common) contenant les modèles et utilitaires partagés
- Module serveur (chat-server) gérant la logique backend et la persistance
- Module client (chat-client) fournissant l'interface utilisateur JavaFX
- **Implémentation du système de communication en temps réel**
 - Configuration du protocole WebSocket avec STOMP pour la messagerie structurée
 - Définition des différents canaux de communication pour les types de messages
 - Gestion des connexions, reconnexion et sessions utilisateurs
- **Mise en place de la persistance des données**
 - Conception du schéma de base de données PostgreSQL
 - Implémentation des opérations CRUD pour les entités principales
 - Optimisation des requêtes pour les performances

Fonctionnalités côté serveur

- **Développement des contrôleurs de communication**
 - MessageController pour la gestion des messages standards
 - SubscriptionController pour la gestion des abonnements aux conversations
 - PinController pour la gestion des messages épinglés
 - MessageManagementController pour la suppression et modification des messages
- **Mise en place de la gestion des utilisateurs**
 - Système d'authentification et de gestion de sessions
 - Vérification des permissions basée sur les rôles
 - Stockage sécurisé des informations utilisateurs
- **Configuration de l'environnement Spring Boot**
 - Paramétrage des WebSockets pour gérer de grands volumes de messages
 - Configuration des points de terminaison et du broker de messages
 - Mise en place du monitoring avec Spring Boot Actuator

Fonctionnalités côté client

- **Développement de l'interface utilisateur JavaFX**
 - Création des écrans de connexion et d'enregistrement
 - Interface de conversation avec liste des participants et messages
 - Panel de composition de messages avec support pour différents types de contenus
 - Indicateurs visuels pour les statuts de messages et utilisateurs
- **Implémentation du client WebSocket**
 - Création du WebSocketHandler comme façade pour les opérations WebSocket
 - Système de gestion des connexions avec logique de reconnexion
 - Pipeline de traitement des messages entrants et sortants

- **Optimisation des performances UI**

- Utilisation appropriée des threads JavaFX pour une interface réactive
- Chargement paresseux des données pour minimiser l'utilisation des ressources
- Gestion efficace des ressources pour une expérience fluide

Infrastructure et déploiement

- **Mise en place d'un pipeline CI/CD complet avec GitLab**

- Configuration des étapes de validation, test, build, packaging et déploiement
- Automatisation des tests unitaires et de la couverture de code
- Intégration des artefacts et rapports de test

- **Configuration du déploiement containerisé**

- Création des Dockerfiles pour le serveur
- Configuration de Docker Compose pour le déploiement en production
- Paramétrage des limites de ressources et health checks

- **Mise en œuvre du packaging natif multi-plateforme**

- Utilisation de JPackage pour créer des installateurs natifs
- Génération d'installateurs EXE et MSI pour Windows
- Génération de packages DEB et RPM pour Linux

Gestion de projet et documentation

- **Élaboration de la documentation technique**

- Documentation détaillée de l'environnement technologique
- Documentation des fonctionnalités et implémentations
- Diagrammes UML (composants, déploiement, séquence, activité)

- **Création des outils de développement**

- Makefile pour la simplification des commandes de build et déploiement
- Scripts d'automatisation pour les tâches récurrentes
- Documentation du processus de développement

- **Mise en place des bonnes pratiques**

- Structure de code modulaire et maintenable
- Tests unitaires pour les composants critiques
- Gestion des exceptions et logging complet