
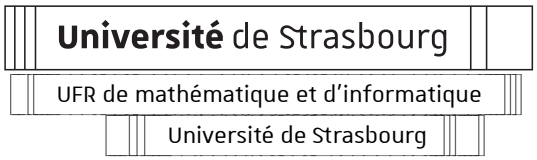


Master de Science, mention Informatique spécialité Science et Ingénierie des Réseaux, de l'Internet et des Systèmes

Mémoire de projet de Master, groupe nř2

Gabriel Poittevin
gabriel.poittevin@etu.unistra.fr

# ChaTalK, application de télécommunications sécurisées et centralisées

Projet encadré et commandité par

Julien Montavont montavont@unistra.fr et Cristel Pelsser pelsser@unistra.fr


7 janvier 2020

# Contents



# List of Figures

# List of Tables

**Abstract**

Ceci est le mémoire de projet du groupe nř2 de projet de Master de la promotion 2018-2020 du Master SIRIS de l'Université de Strasbourg.
L'objectif de ce document est de présenter le projet dans son ensemble, avec ses problématiques, les solutions qui leur ont été apportées et les difficultés rencontrées au cours de sa réalisation.

# 1 Introduction

## 1.1 Le projet

L'objectif de ce projet était de répondre à une série de problèmes rencontrés par l'organisation (fictive) Journalistes Sans Papiers. JSP souhaitait un moyen le plus sécurisé possible de permettre à ses membres de communiquer les uns avec les autres. La solution devait également supporter d'importants flux de données, et pouvoir fournir notamment un mode audio voire vidéo en plus du mode texte classique. Enfin, l'organisation souhaitait la possibilité de déployer à la volée le service sur des réseaux isolés, permettant à ses membres d'utiliser la solution de communication dans des situations coupées du réseau Internet et des serveurs centraux de l'organisation.

## 1.2 La solution

Notre équipe a conçu la solution ChaTalK afin de répondre aux besoin de l'entreprise. Il n'existait pas à notre connaissance de solution permettant les communication entièrement sécurisées de manière centralisée, dont un développement facile pour un néophyte puisse être possible. ChaTalK est une application Web 2.0, dont l'infrastructure repose sur des cluster Kubernetes sur des machines Ubuntu Server, le côté serveur de l'application est un ensemble de services Go et PostgresQL, et le côté client est une interface écrite en React JS pour la partie navigateur et en Kotlin pour l'application mobile.

# 2 Technique

## 2.1 Architecture de la solution

## 2.2 Infrastructure matérielle et logicielle

## 2.3 Protocoles de communication

## 2.4 Mesures et protocoles de sécurité

## 2.5 Base de données

## 2.6 Services logiciels, application côté serveur

## 2.7 Interface utilisateur, application côté client

# 3 Solution

La solution telle que nous l'avons conçue comprenait plusieurs éléments que nous allons rappeler ce-dessous.

**Déploiement facile** Notre application devait pouvoir être facilement déployée par un néophyte, sur n'importe quelle machine de type Linux, rapidement et sans connexion Internet. Cette tâche était d'une priorité maximale

**Sécurité** Notre application devait garantir la sécurité des flux de données, en particulier d'un point de vue intégrité et confidentialité. Seules les envoyeurs et récepteurs des messages devaient avoir accès aux données.
Nous avons partiellement rempli cet objectif, des contraints de temps liées à nos retard nous ayant forcés à abandonner le chiffrement des flux audiovisuels. L'authentification des utilisateurs a néanmoins été réalisée. Une couche de sécurité minimale basée sur SSL/TLS assure un niveau

de sécurité décent, tant que les utilisateurs du service font confiance au administrateur des serveurs sur lesquels fonctionne la solution.

**Tolérance aux pannes et passage à l'échelle** Grâce à sont infrasctucture basée sur Kubernete et à son architecture en micro-services, notre application est aisément capable de passer à l'échelle et de supporter la montée en charge, ainsi que de résister à la perte de machines dans les clusters. Les pires des cas entraînent des performances dégradées qui permettent de consulter les archives de conversation et donc de continuer une partie des activités des utilisateurs du service.

**Temps réel** Notre application devait assurer des conversations en temps réel entre les utilisateurs.
Nos tests actuels ne sont pas représentatifs d'une utilisation réelle, mais montrent que la latence des communications textuelles sur notre solution est imperceptible à l'humain pour un petit nombre d'utilisateurs.

**Interface utilisateur** Notre solution devait proposer une interface utilisateur compatible avec un maximum d'appareils.
Notre application frontale Web est utilisable sur tous les navigateurs modernes sans modification et sans distinction de système d'expoitation. De plus elle s'adapte à des tailles d'écrans diverses et à des appareils de puissance variée. Bien que cette interface Web puisse être utilisée en tant qu'application mobile, une application mobile pour Android en Kotlin est également disponible, plus adaptée à l'utilisation sur des téléphones intelligents.
L'interface utilisateur est épurée, présentant peu d'interactions possibles, documentées avec des icônographies habituelles telles qu'utilisées sur la majorité des autres services du même genre.

**Modularité** Notre solution devait permettre à un individu déployant un serveur de choisir les capacités qu'il souhaitait donner à celui-ci.
À cause des contraintes de temps, et puisque la tâche n'était pas prioritaire, elle a été la première abandonnée totalement. Les fonctionnalités supplémentaires audio et vidéo ont été incluses directement dans l'applications d'arrière-plan principale et le concept de modules serveur a été laissé à l'abandon.

**Compatibilité et standards** Notre application devait être compatible avec un maximum de standards et ne pas utiliser de technologies trop peu orthodoxes.
Puisque l'interface utilisée est une interface Web, les quatre systèmes d'exploitation les plus courants, Microsoft Windows, Apple macOS, Apple iOS et Android, sont couverts tant qu'ils possèdent un navigateur Internet à jour. De plus, les systèmes d'exploitation GNU/Linux et BSD sont également couverts tant qu'ils possèdent un navigateur Internet à jour. Enfin, s'il n'existe pas d'application native pour iOS, il en existe une pour Android.
D'un point de vue des standards, notre application est accessible à des clients utilisant indifféremment de l'IPv4, de l'IPv6 ou les deux.

**Internationalisation** Notre application se devait de supporter les principaux alphabets du monde et d'être disponible à minima en Anglais.
Puisque tous les textes que nous utilisons sont encodés en UTF-8, les alphabets divers sont gérés dans l'application. L'interface que nous avons développée l'a été en Anglais et est donc utilisable par des utilisateurs anglophones.

# 8 References