AVIGNON UNIVERSITÉ

Application Architectures distribuées

Jerome Isoard
Tom Leszczynski

11 avril 2022

Master d'Informatique
Ingénierie du Logiciel de la Société Numérique
UE Application architectures distribuées

Responsables
Mickael Rouvier

UFR
SCIENCES
TECHNOLOGIES
SANTÉ

AVIGNON UNIVERSITÉ
CERI

CENTRE
D'**ENSEIGNEMENT**
ET DE **RECHERCHE**
EN **INFORMATIQUE**
ceri.univ-avignon.fr

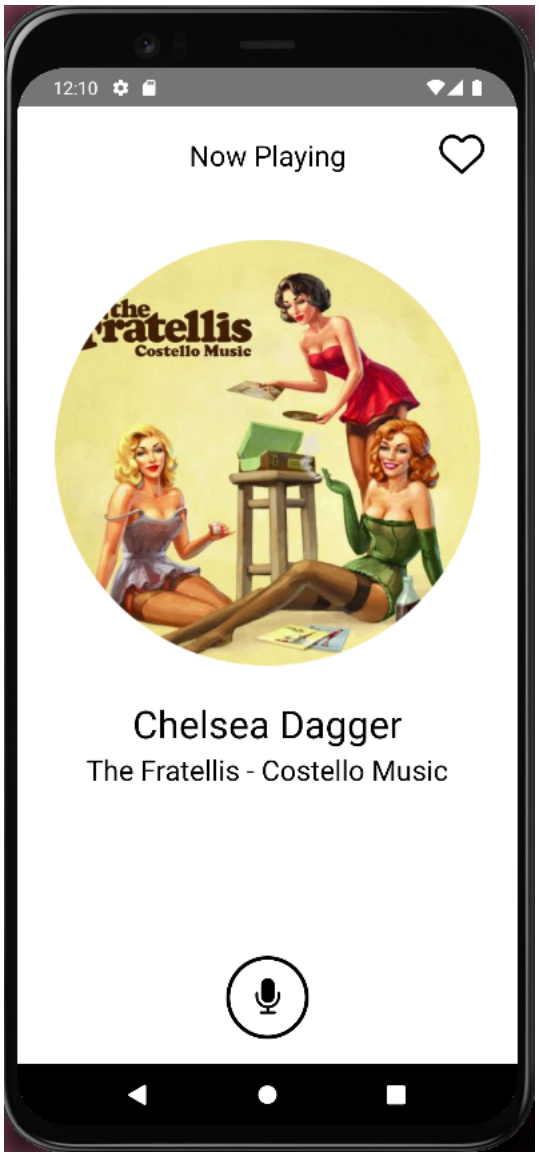
Sommaire

1 Introduction

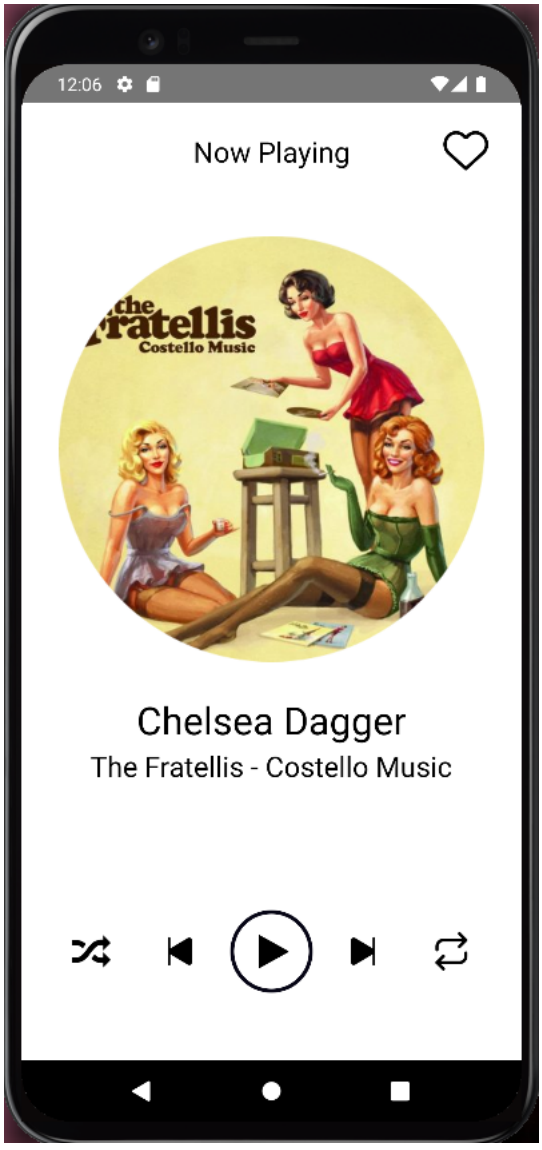
Pour l'ASR et le NLP on utilise `Java`. Le serveur ice est écrit en `Java` lui aussi. Le client ice est écrit en `Javascript` pour l'utiliser avec notre application Android développée avec `React Native`. Pour la bdd on utilise `DynamoDB` (AWS). Pour l'ajout de données dans la bdd on a pour le moment une application de bureau développée avec `ElectronJs`.
Nous avons un repo GitHub privé, nous pouvons vous ajouter si vous le souhaitez.

2 Interface mobile

L'application est développée avec react native. Le bouton de commande vocale est le premier à apparaître. Les commandes tactiles restes cependant disponibles avec un swipe sur la droite ou la gauche.



(a) Interface avec commande vocale



(b) Interface avec commandes tactiles

3 Automatic Speech Recognition

Notre ASR est développé en JAVA, on utilise Google Cloud Speech-To-Text. Il est possible de transcrire des fichiers audios aux formats mp3, WAV, Flac ou autre. La configuration de l'encodage pour la reconnaissance est indiquée sur la documentation (https://cloud.google.com/speech-to-text/docs/encoding). Pour son fonctionnement, un fichier audio local est envoyé au service Google Speech-To-Text sous forme de []bytes. Le service retourne le texte transcrit. Voici comment l'utiliser en ligne de commande :

```
cd path/to/ASR
mvn clean install
set GOOGLE_APPLICATION_CREDENTIALS=C:\path\to\jsonKey\key.json
mvn exec:java -Dexec.mainClass=SpeechToText
```

4 Natural Language Processing

Notre NLP est développée en JAVA, on utilise les regex. On attend certains mots clés en début de phrase pour l'action comme "Joue", "Reprend", "Stop", "Suivant" ...
Exemples d'utilisation en ligne de commande :

```
java main "Joue Come out and play de The offspring"
(output) <play, come out and play the offspring>

java main "Reprend la lecture"
(output) <resume, null>

java main "Passe ce morceau"
(output) <forward, null>
```

5 Diagramme

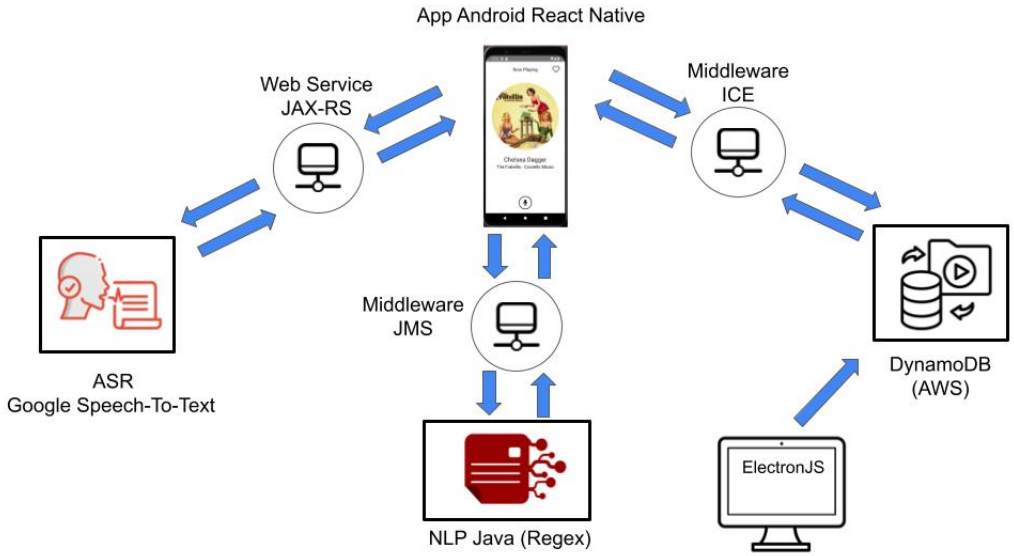


Figure 2. Architecture de la solution

La figure 2 décrit l'architecture globale de notre solution de lecteur audio piloter par la

voie. Tous les éléments distants décrits précédemment (ASR, NLP et Serveur de Streaming) sont reliés à l'application Android par des middlewares. Pour l'instant, nos choix se sont tournés vers un WebService pour l'ASR (pour rester dans une logique de requête vers un service distant), JMS pour le NLP (car il s'agit ici uniquement d'envoie de message) et ICE pour le serveur de Stream (comme cela est demandé dans les spécifications).