

1 octobre 2022

DOCUMENtATION

TECHNIQUE

Golden Webdav

Membres :

* Théo Paramelle
* GRaham
* Loïc

TABLE DES MATIERES



Première partie 3

Informations 3

1 - Résumé du document 3

2 - Rappel sur le fonctionnement de l’application 4

2.1 - Description du logiciel 4

2.2 - Décomposition du projet 4

2.3 - Architecture globale 5

3 - FRONT 6

3.1 - Architecture 6

3.2 - Technologies utilisées 6

3 - BACK 7

3.1 - Architecture 7

3.2 - Technologies utilisées 7

Première partie

Informations

| **Nom du projet** | Golden webdav |
| --- | --- |
| **Type de document** | Documentation technique |
| **Date** | 1 Octobre 20222 |
| **Version** | 1.1 |
| **Mots-clés** | AREA - Mobile - Web - Database - serveur |
| **Auteurs** | Théo Paramelle |

1 - Résumé du document

Ce document est la documentation technique officielle de la suite applicative Golden webdav. Il est divisé en quatre parties :

– La documentation technique du client : l'application web ;

– La documentation technique du serveur: backend et database ;

– La documentation technique de l'application mobile fonctionnant sur Android et iOS ;

2 - Rappel sur le fonctionnement de l’application

2.1 - Description du logiciel

Notre projet AREA a pour but la création d'un serveur web : goldenwebdav.

Plus que la diffusion de musique, il permet aux stations de radio de gérer un maximum d'éléments de leur quotidien, notamment les contrats publicitaires, les jingles, la récupération et l'exportation de flux de données.

Le logiciel a une interface moderne et ergonomique, permettant de gérer efficacement et simplement la diffusion tout en proposant de nombreuses manipulations avancées. Il est utilisable aussi bien sur des surfaces classiques que tactiles, et ce quel que soit le nombre d'écrans à disposition.

2.2 - Décomposition du projet

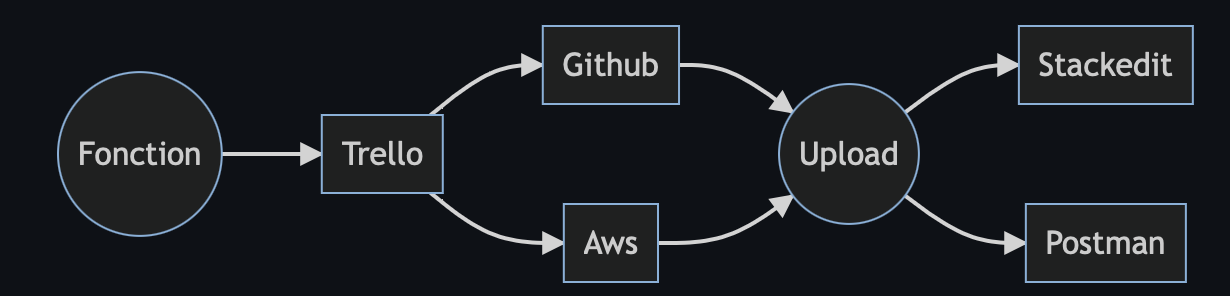
Notre projet se décompose en différentes parties :

• Le client web, qui est une application bureau fonctionnant sur Windows, Linux et MacOS

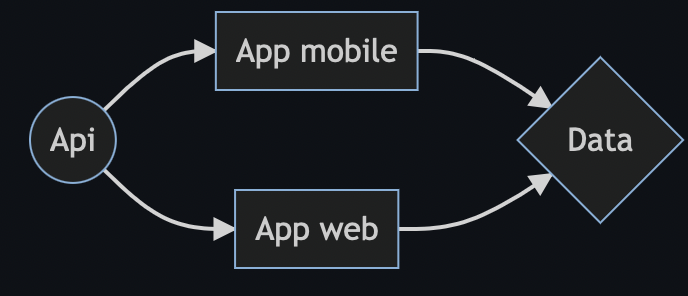
• Une application fonctionnant sur les tablettes tactiles équipées d'iOS ou d'Android, dialoguant avec le logiciel client et implémentant les fonctions de base.

Dans la suite de ce document, chacune de ces différentes parties sera développée.

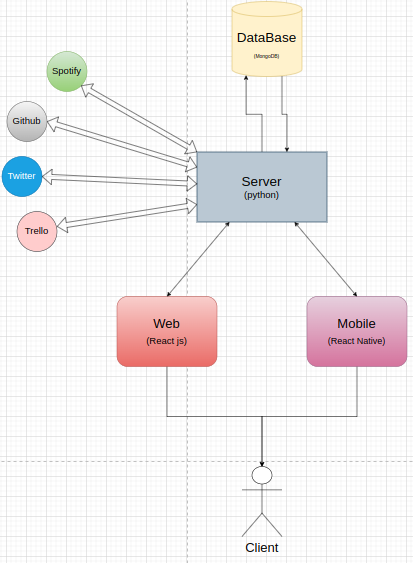
2.3 - Architecture globale



Environnement fonctionnel

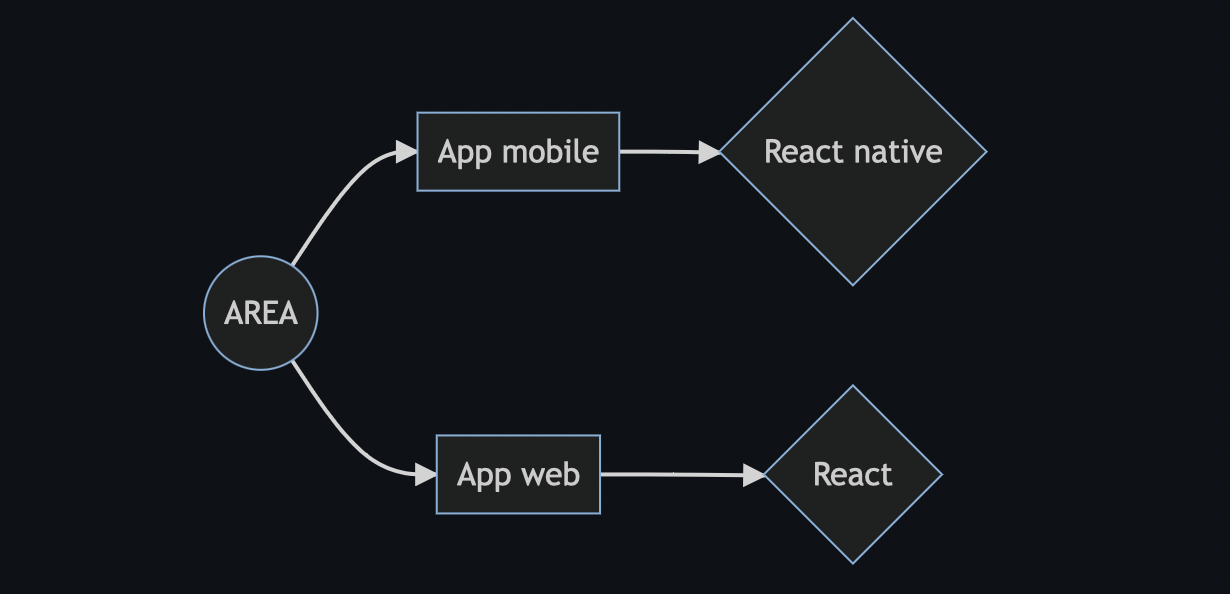


Environnement technique



3 - FRONT

3.1 - Architecture



Graphique technique Front

3.2 - Technologies utilisées

Le partie Front est codé en React et React Native.

Qt fournit tous les éléments nécessaires à la réalisation de notre application, et apporte plusieurs éléments essentiels :

• La portabilité : un code Qt compile indifféremment sous les trois systèmes d'exploitation ciblés, à savoir Windows, Linux et Mac OS X.

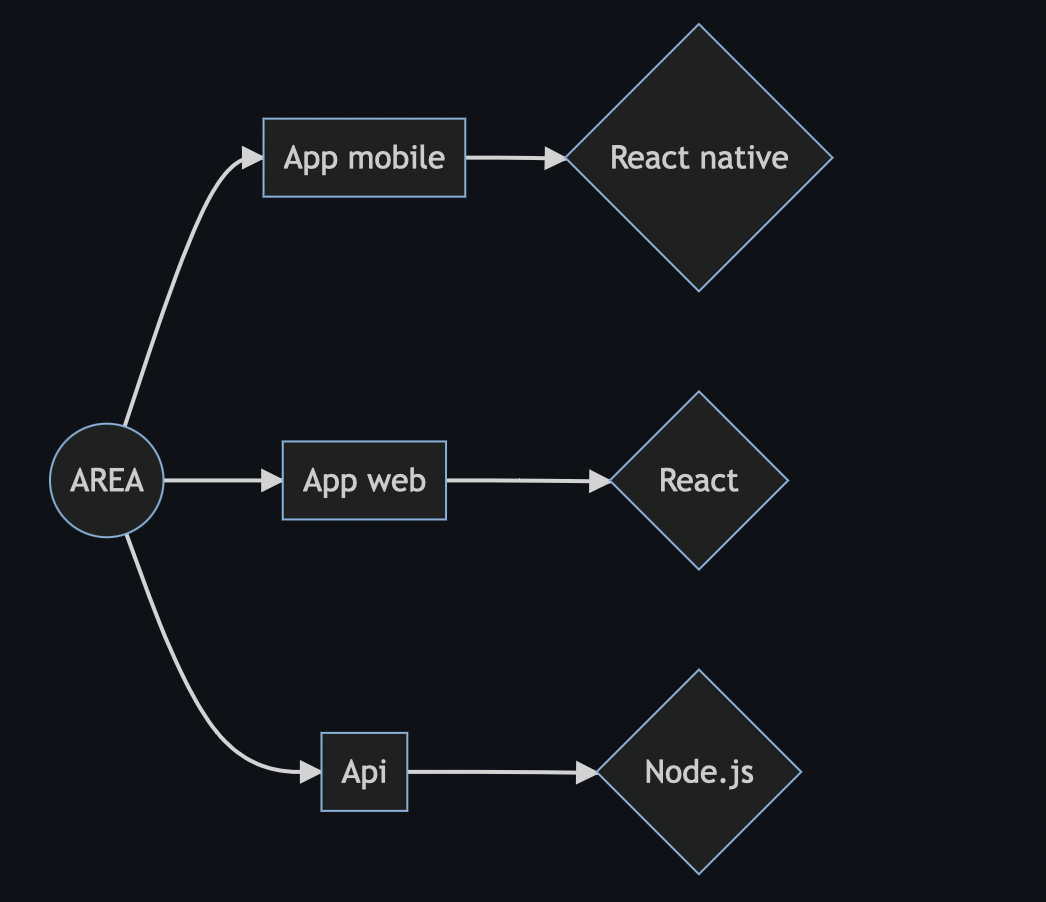
• Soutenu par Nokia, Qt est utilisé dans des projets professionnels de grande envergure (tels KDE ou Meego). Il est en développement perpétuel et possède une communauté active. Cela nous assure la viabilité à long terme de ce framework.

• Nokia et Qt fournissent depuis peu de nouveaux modules Qt ciblés sur une utilisation mobile et tactile. La version allégée du logiciel client utilise donc des technologies très récentes.

3 - BACK

3.1 - Architecture

Le back est lancer en Thread un pour le serveur et la réception des requêtes, l’autre pour le pooling sur les api.



Graphique technique Back

Chaque api a une classe correspondante contenant toutes les fonctions nécessaires.  
Chaque api possède également ses routes correspondante afin que ce soir plus simple d’utilisation.

*twitter*/addIdoles  
github/addIssue  
etc...

Pour ce qui est du pooling. La boucle fait appelle aux mêmes classe. Avec deux fonction commune « pullEvent » et « pushEvent ».

3.2 - Technologies utilisées

La partie BACK, a été développe en Python.

Ce langage a été choisis pour ses nombreuses librairie disponible mais également sa facilite d’utilisation.   
  
Pour les routes, nous avons utiliser la libraire Flask, qui est la plus adapter a notre projet.

Authentification :

Pour nous faciliter le processus d’authentification. Nous avons utiliser la librairie AuthLib pour : Twitter, Github.

Les api de Spotify et Trello étaient bien plus simple il n’a donc pas fallut de lib.

Api :

La communication avec les api étant faite uniquement par requêtes. Nous avons trouver également des lib (wrapper), nous permettant de faire des requestes plus facilement.

Spotify – Spotipy

Twitter – Tweepy

Trello – PyTrello

Github – PyGithub

Base de Données :

Nous avons utilise la base de données Mongo DB. Cette dernier utilise des fichiers type JSON. Sachant que en python nous pouvons faire des dictionnaire. Ce choix de DB était le plus logique. Nous pouvions donc envoyer des dictionnaire en DB et recevoir des dictionnaire. Tout était donc plus lisible et simple.

Également MongoDB possède un outil : MongoDB Compasse, qui nous permet de visualise la DB très clairement.

PyMongo est la librairie que nous avons utiliser pour communiquer avec la DB.