Document

# Client

Technologie : AngularJS

Port : 4000

Méthode de communication avec le back : Routes / API rest

Méthode d’authentification : OAuth2.0

Architecture :

***DEV\_dashboard\_2019****/*

***Src****/*

***App****/*

*App.Component.js*

*App.Component.html*

*App.Component.css*

***Components****/*

***Service****/*

*Service.component.html*

*Service.component.js*

*Service.component.css*

***Widget****/*

*widget.component.html*

*widget.component.js*

*widget.component.css*

***login****/*

*login.component.html*

*login.component.js*

*login****.****component****.****css*

***register****/*

*register.component.html*

*register.component.js*

*register.component.css*

***Services****/*

*Login.service.ts*

*Login.service.spec.ts*

*widget.service.ts*

*widget.service.spec.ts*

Les widgets avec authentication :

1. Service Steam :
   1. Récupérer la liste d’amis de steam
   2. Vérifier l’état de connexion des amis
   3. Afficher le jeux sur lequel ils jouent.
2. One drive :
   1. Affichage des fichiers
   2. Téléchargements des fichiers
3. Spotify :
   1. Sélectionner et jouer un playlist utilisateur

Les widgets sans authentication :

1. Widget Météo
2. Widget cours boursier
3. Widget google map
4. Widget youtube
5. Widget twitch
6. Widget Reddit

Le Timer :

Toutes les X minutes, choisi par l’utilisateur, le client, les widgets se mettent à jour.

# Serveur

Technologie : NodeJS

Port : 8080

Méthode de communication avec le front : Routes / API rest

Méthode authentification : Token ( username / password )

Liste des routes :

* ***Route :*** 
  + ***Methode*** 
    - ***Paramètre***
  + ***Réponse renvoyé***
* about.json :
  + GET
* Login :
  + POST :
    - Username
    - Password
    - Réponse :
      * token ( sauvegardé en cookie )
      * Erreur 400 ( login invalide )
  + DELETE /account :
    - Token
    - Réponse :
      * 300, OK
      * 400, NOK
* Logout :
  + POST :
    - Token
    - Réponse :
      * 300, OK
      * 400, NOK
* Register :
  + POST :
    - Username
    - Password
    - Réponse :
      * 300 , utilisateur créé
      * 400 , utilisateur déjà existant
* Services & Widget :
  + POST :
    - Id de service
    - Paramètres optionnels
    - Réponse :
      * 300, OK + id de widget
      * 400, NOK
  + GET :
    - Token
    - Réponses :
    - Widgets avec leurs id ect…
  + UPDATE :
    - Token
    - Id\_widget
    - Paramètres
    - Réponse :
      * 300, OK
      * 400, NOK
  + DELETE :
    - Token
    - Id\_widget
    - Réponse :
      * 300, OK
      * 400, NOK

Architecture du projet :

* Class Main
* Class Account
* Class Widgets