



README - Architecture micro service DoesThisPersonExist

⌚ Class	Micro-services
🕒 Created	@Feb 21, 2021 7:24 PM
👤 Membres	Ⓐ Vincent Rospini 🧑 Axel Buée 🧑 Julie Coustenoble Ⓢ Angélique Souvant
⌚ Type	Projet
📎 resources	
⌚ semester	

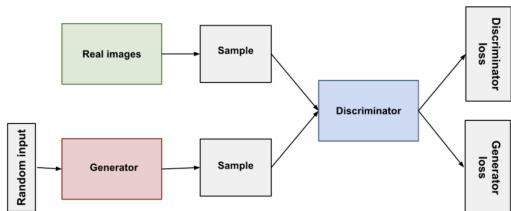
Introduction

L'objectif du projet est de développer un jeux avec en architecture micro conteneuriser sur docker.

Le but du jeu est de deviner si l'image correspond à un vrai visage ou un faux visage généré.

Does this person exist <https://thispersondoesnotexist.com/>

L'utilisateur doit deviner si l'image qui lui est présenté existe vraiment.



L'idée GAN

https://developers.google.com/machine-learning/gan/gan_structure



L'équipe

Julie

Images
Front-end

Vincent

Game

Axel

Front-end

Angélique

Authentification
profil

Technologies utilisées

Front

- React.JS langage js front
- Next.JS framework react.js front
- React-bootstrap framework css
- Docker

Profils

- nodeJS langage javascript back
- MongoDB base de données non relationnelle
- Docker

Images

- Python
- Django framework python
- Docker

Authentification

- nodeJS langage javascript back
- PostgresSQL base de données relationnelle
- Docker

Game

- Python
- Flask framework python
- SQLite base de données relationnelle
- Docker

Démarrer le projet

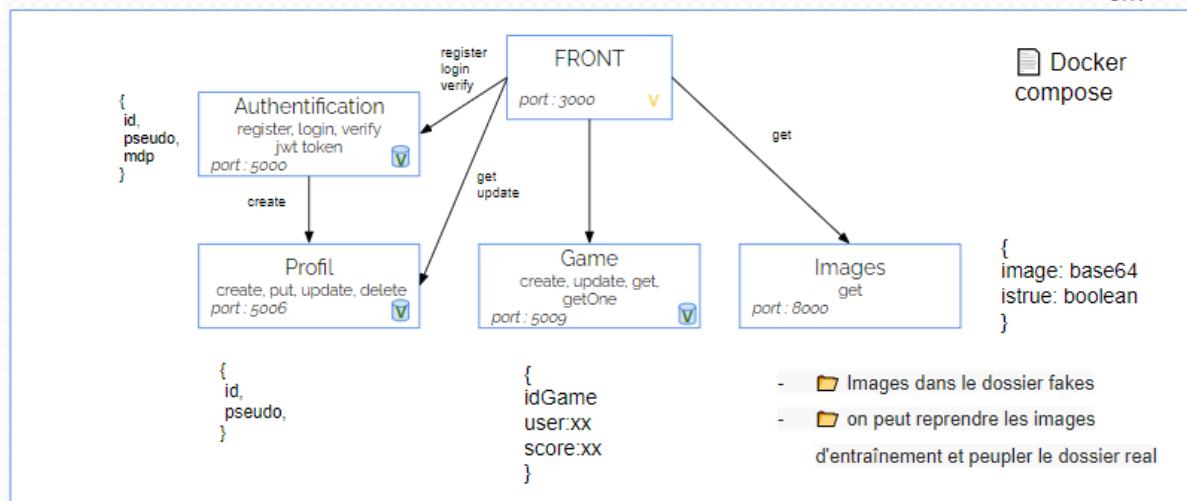
- Démarrer docker
- Se placer dans le dossier du projet et exécuter la commande suivante

```
docker-compose up -d
```

Fonctionnement de l'application

- URL de l'application : <http://localhost:3000>
- Cliquer sur **Sign up** en haut à gauche de la page
- Créer un profil utilisateur (vous serez redirigé sur la page **Login**)
- Se connecter en entre nom d'utilisateur et mot de passe
- On arrive sur le profil ou les informations nom utilisateur et email sont affichées ainsi que le meilleur score du joueur
- Cliquer sur **PLAY** pour jouer ! Une fois la partie perdue
- Appuyer de nouveau sur **PLAY** pour jouer
- Vous avez la possibilité de vous déconnecté une fois terminé avec le bouton **Logout** du menu

Architecture projet



Swagger

Les points d'api suivants sont affichés avec swagger sur les URL suivantes:

- | Profil : <http://localhost:5006/doc/>
- | Authentification : <http://localhost:5000/doc/>
- | Faces : DoesThisPersonExist/Faces/swagger_api.json

Micro services disponibles

- faces : Permet de récupérer des images
- authentification : Permet de gérer la l'authentification à l'aide de jwt token
- profil : Permet de gérer le profil d'un utilisateur
- games : Permet de gérer le score pour une partie
- Front : Permet de gérer l'affichage du projet et interagit avec l'ensemble des API

Documentation API pour Authentification

Register

```

METHOD: POST
http://localhost:5000/api/auth/register
body : {
    "username" : "username",
    "email" : "email",
    "password" : "admin",
}

```

login

```
METHOD: POST
http://localhost:5000/api/auth/login
body : {
  "username" : "Angelique",
  "password" : "admin"
}
```

Verify

```
METHOD: POST
http://localhost:5000/api/verify/user
headers : {
  "x-access-token" : "token"
}

retourn true ou false en fonction de si l'utilisateur est connecté
```

Documentation API pour Profil

create

```
METHOD: POST
http://localhost:5006/profil
body: {
  "userId" : "test",
  "email" : "test",
  "nom" : "test",
  "prenom" : "test",
}
```

getall

```
METHOD: GET
http://localhost:5006/profil
```

getone

```
METHOD: GET
http://localhost:5006/profil/:code
```

update

```
METHOD: PUT
http://localhost:5006/profil/:code
body: {
  "email" : "test",
  "nom" : "test",
  "prenom" : "test",
}
```

delete

```
METHOD: DELETE  
http://localhost:5006/profil/:code
```

Documentation API pour Faces

Get a random face generated or not

```
METHOD: GET  
http://localhost:8000/face/
```

Documentation API pour Games

```
/api/game/best/user/:id  
La meilleur game d'un joueur  
  
GET  
/api/game/user/:id  
Toutes les game d'un joueur  
  
GET  
/api/game/:id  
Information d'une game  
  
POST  
/api/game  
Création d'une game  
  
PUT  
/api/game/:id  
Update d'une game  
  
DELETE  
/api/game/:id  
Suppression d'une game
```