Documentation api rest Area

# 

# Introduction:

### Dans le cadre du projet de fin de 3ème année d’Epitech Area, nous avons développé une api rest ainsi qu’un client web et un client mobile.

### Ainsi cette api permet de se connecter, créer des Area et les gérer.

### Cette api est disponible en lançant le serveur grâce à la commande “docker-compose up” (docker prérequis).

### Ensuite vous pouvez accéder librement au serveur grâce aux clients mobile ou web, ou bien directement par call sur [localhost:8080/](http://localhost:8080/)

### Dans la suite du document vous trouverez la liste des routes dans le sommaire puis la description et le fonctionnement de chaque route.

# 

# Sommaire:

## Home

## /

## /about.json

## Authentification

## /register

## /login

## /update

## /logout

## Area

## 

## Autre

## 

# Home:

## GET: localhost:8080/

#### description: simple home, renvoie simplement “hello world”

#### body: {}

#### response: “hello world”

## GET: localhost:8080/about.json

#### description: to do

#### body: {}

#### response:

## Authentification:

L’authentification est disponible dans les routes /user/

## POST: localhost:8080/user/register

#### description: inscription dans la base de données.

#### body: {password, username, email}

#### response: {error: false, user: { id, username, email, google} }

#### response error: {error: true, message: “error message”}

## GET: localhost:8080/user/login

#### description: connexion à la base de données.

#### body: {password, email}

#### réponse: {error: false, user: { id, username, email, google} }

#### réponse erreur: {error: true, message: “error message”}

## PUT: localhost:8080/user/update

#### description: modifie la base de donnée

#### body: {id, username, email, google}

#### réponse: {error: false, user: { id, username, email, google} }

#### réponse erreur: {error: true, message: “error message”}

## GET: localhost:8080/user/logout

#### description: déconnection et redirige sur /

#### 

## Area:

Les area sont disponible dans les routes /area/

## POST: localhost:8080/area/create

#### description: créer une area.

#### body: {userId, actionId, actionDesc, reactionId, reactionDesc}

#### response: {error: false, area: { id, userId, actionId, actionDesc, reactionId, reactionDesc} }

#### response error: {error: true, message: “error message”}

## GET: localhost:8080/user/getbyid

#### description: cherche les ares dont l’user id est celui demandé.

#### body: {userId}

#### réponse: {error: false, areas: [{ id, username, email, google}] }

#### réponse erreur: {error: true, message: “error message”}

## PUT: localhost:8080/user/update

#### description: modifie la base de donnée

#### body: { id, userId, actionId, actionDesc, reactionId, reactionDesc}

#### response: {error: false, area: { id, userId, actionId, actionDesc, reactionId, reactionDesc} }

#### réponse erreur: {error: true, message: “error message”}

## DEL: localhost:8080/user/delete

#### PUT: localhost:8080/user/update

#### description: supprime l’élément de la base de donnée

#### body: { id}

#### response: {error: false, deleted: { rowCount, ...} }

#### réponse erreur: {error: true, message: “error message”}