

# API AREA

## Preface

This Project is made by Benjamin Piriac, Mark-Antoine Wittling, Matieu Dèru, Stànisslass Hérron, Jean Bougnarel and Mikael Gerard, for Epitech.

## Introduction

The AREA is a simple dashboard with services. Each services have action or reaction. Some of them will need OAuth2. It is build with docker. The purpose of the project is to creating AREA :

- **Action** : A predefined event triggering a reaction (like a new post from someone on reddit)
- **REAction** : Predefined event supposed to be trigger by a action (like post a new image on imgur)

The project is compose of :

- **Client** : call the api and display informations, run on port 8081
- **API / Server**: manage database and call externe api and send information to the client, running on port 8080
- **Mobile client** : available on localhost:8081/client.apk
- **Mongo data base**: A data base by mongodb, available on the port 27017

## API Routes

The following commands explain there ous use

### Connect - POST

[/connectUser](#)

**Body :**

**username** : username of the user  
**password** : password of the user

Connect a user, return success or fail.

**return exemple :**

```
{  
  success : True  
  message : "User connected"  
  token  : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...t2ot3Urfre4egTFGEE"  
}
```

**Status Success :** 201

**Status Error :** 401

## Create user - POST

[/createUser](#)

### Body :

**username** : username of the user  
**password** : password of the user

Connect a user, return success or fail.

### return exemple :

```
{  
  success : True  
  message : "User created"  
  token : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...t2ot3Urfre4egTFGEE"  
}
```

**Status Success :** 201

**Status Error :** 401

All next request need a specific header :

**Authorization : 'token {token received on the connection}'**

## Create area - POST

[/CreateArea](#)

### Body :

**action** : Action for the area  
**reaction** : Reaction for the area  
**optional name** : Name of the area  
**optional color** : Color of the area

All parameter needed by the action and the reaction, exemple :

**city** : City for the weather  
**channel\_id**: Channel to post a message  
**message**: message to post

Create a AREA.

### return exemple :

```
{  
  success : True  
  message : "Area created successfully"  
}
```

**Status Success :** 201

**Status Error :** 401

## Get areas - GET

[/GetAreas](#)

Get all AREA of the user.

return exemple :

```
[
  {
    area_id: "405ope7bk5mpo3hi",
    user_id: "405ope1mk5mpevm0",
    action: "weather_time",
    area_name : "Area racoon",
    action_params: {
      city: "Nantes",
      time: "10:10"
    },
    reaction: "discord_send_message",
    reaction_params: {
      channel_id: "12435",
      message: "the weather is {weather}"
    }
  }
]
```

Status Success : 201

Status Error : 401

## Get area - GET

[/GetArea/:Areald](#)

Get all AREA of the user.

return exemple :

```
{
  area_id: "405ope7bk5mpo3hi",
  user_id: "405ope1mk5mpevm0",
  area_name : "Area racoon",
  action: "weather_time",
  action_params: {
    city: "Nantes",
    time: "10:10"
  },
  reaction: "discord_send_message",
  reaction_params: {
    channel_id: "12435",
    message: "the weather is {weather}"
  }
}
```

Status Success : 201

Status Error : 401

## Get area's name - GET

[/GetAreas/Name](#)

Get AREA's name and id of the user.

return exemple :

```
{
  Area: [
    {
      area_id: "405ope7bk5mpo3hi",
      action: "Github",
      reaction: "Discord",
      area_name : "Area racoon"
    },
    ...
  ]
}
```

Status Success : 201

Status Error : 401

## Delete area - Delete

[/DeleteArea](#)

Body :

area\_id : Id of the area

Delete a AREA.

return exemple :

```
{
  success : True
  message : "Area deleted"
}
```

Status Success : 201

Status Error : 401

## Add token - POST

</auth/addToken>

### Body :

**service** : The name of the service  
-- All the token given by the Oauth (depend on the service)

Save token from authentication in differents services.

### return exemple :

```
{  
  success : True  
  message : "Token saved"  
}
```

Status Success : 201

Status Error : 401

## Delete token - DELETE

</auth/token>

### Body :

**service** : The name of the service

Delete token from authentication in differents services.

### return exemple :

```
{  
  success : True  
  message : "Token delete"  
}
```

Status Success : 201

Status Error : 401

## Get services - GET

</auth/getServices>

Get the services the user is connected to.

return exemple :

```
{
  Discord : True,
  Reddit : True,
  Imgur : False,
  Steam : True
}
Status Success :      201
Status Error :       401
```

## Get all actions - GET

</getActions>

Get all the actions available for the user.

return exemple :

```
{
  actions : [
    {
      name : "github_new_push",
      service : "Github",
      title : "Repository push",
      description : "Trigger when someone push on a repo",
      params : [
        {
          name : "repository",
          description : "Name of the repository"
        },
        {
          name : "username",
          description : "Owner's name of the repository"
        }
      ]
    },
    ...
  ],
}
Status Success :      201
Status Error :       401
```

## Get all actions and reactions - GET

### [/getReactions](#)

Get all the reactions available for the user.

return exemple :

```
{
  reactions : [
    {
      name : "slack_send_message",
      service : "Slack",
      title : "Send message",
      description : "Send a message to a channel",
      params : [
        {
          name : "channel_id",
          description : "Id of the channel"
        },
        {
          name : "message",
          description : "Message to send"
        }
      ]
    },
    ...
  ],
}
```

Status Success : 201

Status Error : 401