The API reference

Post /verify

This route verify `token` for login and show data from logged user if him logged, othewise it send a error message.

data:

```
{
   "auth": "<token>"
}
```

response:

```
{
   "type": "sucess",
   "data": {
      "auth": "<token>",
      "user": {
          "id": "<id>",
          "name": "<name>",
          "email": "<email>",
          "genre": "<genre>",
          "phone": "<phone>",
          "born": "<born_date>",
          "state": "<civil_state>",
          "profile": {
              "data": "<image>",
              "alt_text": "<text>"
          }
      }
   }
}
```

Post /login

This route is the way to make a login on server, the return will be the user data and the login `token`.

data:

```
{
    "data": {
       "email": "<email>",
       "pass": "<password>"
    }
}
```

response:

```
{
   "type": "sucess",
   "data": {
      "auth": "<token>",
      "user": {
          "id": "<id>",
          "name": "<name>",
          "email": "<email>",
          "genre": "<genre>",
          "phone": "<phone>",
          "born": "<born_date>",
          "state": "<civil_state>",
          "profile": {
              "data": "<image>",
              "alt_text": "<text>"
          }
      }
   }
}
```

Post /logout

This route is the way to make a login on server, the return will be the user data and the login `token`.

data:

```
{
    "auth": "<token>"
}
```

response:

```
{
   "type": "sucess",
   "message": "Token \"<Token>\" removed"
}
```

Post /user

This route is for to create users on system, in this case the api will to send a email for atual people and him with `token` for verify before to create the user and returns a default sucess message.

data:

```
{
   "auth": "<token>",
   "data": {
      "id": "<id>",
      "name": "<name>",
      "email": "<email>",
      "pass": "<password>",
      "genre": "<genre>",
      "phone": "<phone>",
      "born": "<born_date>",
```

```
        "state": "<civil_state>",
        "profile": {
             "data": "<image>",
             "alt_text": "<text>"
        }
    }
}
```

response:

```
{
   "type": "sucess",
    "message": "See you mail box for verify your account"
}
```

Get /user?p=`{page}`&l=`{limit}`

Getting the people list from server (the logged user must have high privileges), the querys `p` if for page number, and `l` is the limit of events per page.

data:

```
{
   "auth": "<token>"
}
```

response:

```
{
   "status": "sucess",
   "data": [
     {
        "id": "<id>",
        "name": "<name>",
        "profile": {
           "data": "<image>",
           "alt_text": "<text>"
        }
     },
      ...
  ]
}
```

##

Get /user/`{id}`

Getting user data, if logged user have'nt high privileges or is not the user it returns a error response

data:

```
{
   "auth": ""
}
```

response:

```
{
   "type": "sucess",
   "data": {
      "id": "<id>",
      "name": "<name>",
      "email": "<email>",
      "genre": "<genre>",
      "phone": "<phone>",
      "born": "<born_date>",
      "state": "<civil_state>",
      "profile": {
           "data": "<image>",
           "alt_text": "<text>"
      },
      "auth": "<token>"
   }
}
```

Post /user/`{id}`

Route for edit existent users, the logged user must be the user or have high privileges, otherwise it returns a error.

data:

```
{
   "auth": "<token>",
   "data": {
      "id": "<id>",
      "name": "<name>",
      "email": "<email>",
      "pass": "<password>",
      "genre": "<genre>",
      "phone": "<phone>",
      "born": "<born_date>",
      "state": "<civil_state>",
      "profile": {
           "data": "<image>",
           "alt_text": "<text>"
      },
   }
}
```

response:

```
{
   "type": "sucess",
   "data": {
      "id": "<id>",
      "name": "<name>",
      "email": "<email>",
      "genre": "<genre>",
      "phone": "<phone>",
      "born": "<born_date>",
      "state": "<civil_state>",
      "profile": {
           "data": "<image>",
           "alt_text": "<text>"
      },
      "auth": "<token>"
   }
}
```

Get /event?p=`{page}`&l=`{limit}`

Getting the event list from server, the querys `p` if for page number, and `l` is the limit of events per page.

data:

```
{
   "auth": "<token>"
}
```

response:

```
{
   "status": "sucess",
   "data": [
     {
        "id": "<id>",
        "name": "<name>",
        "cover": {
           "data": "<image>",
           "alt_text": "<text>"
        }
     },
      ...
  ]
}
```

Get /event/`{id}`

Getting the event informations by event id

data:

```
{
   "auth": "<token>"
}
```

response:

Get /celule/`{id}`

Getting the celule informations by celule id, user logged must be on parent of celule if it exists

data:

```
{
   "auth": "<token>"
}
```

response:

```
{
   "status": "sucess",
   "data": {
      "id": "<id>",
      "name": "<name>",
      "desc": "<description>",
      "type": "church|celule",
      "address": {
          "street": "<street>",
          "neighborhood": "<neighborhood>",
          "city": "<city>",
          "state": "<state>",
          "cep": "<postal code>"
      },
      "cover": {
         "data": "<data>",
         "alt_text": "<text>"
      },
      "Parent": "<celule_id>",
      "members": [
          {
              "user": "<user_id>",
          },
          ...
      ],
   }
}
```

Get /celule?p=`{page}`&l=`{limit}`

Getting the celule list from server, the querys `p` if for page number, and `l` is the limit of events per page.

data:

```
{
   "auth": "<token>"
}
```

response:

```
{
   "status": "sucess",
   "data": {
      "id": "<id>",
      "name": "<name>",
      "cover": {
         "data": "<data>",
         "alt_text": "<text>"
      }
   }
}
```