Wodabest API

SUMMARY

Protected routes 18

GET	api/user/names	Retrieve information about registered users
GET	api/user	Retrieve information about the authenticated user
POST	api/logout	Logout the authenticated user from the application
PATCH	api/settings/profile	Update information about the authenticated user
PATCH	api/settings/password	Update the password of the authenticated user
GET	api/projects	Retrieve all projects
GET	api/projects/{id}	Retrieve one project
POST	api/projects	Create a new project
PATCH	api/projects/{project}	Update a project
DELETE	api/projects/{project}	Delete a project
POST	api/projects/ranking/{category}	Retrieve the best projects
GET	api/categories	Retrieve all categories
GET	api/likes	Retrieve all likes
POST	api/likes	Add a like
DELETE	api/likes/{like}	Delete a like
GET	api/games	Retrieve all games played
POST	api/games/init/{category}	Retrieve two projects to play a match
PATCH	api/games/update	Update projects and create match history

Unprotected routes 4

POST	api/login	Authenticate a user to access the application
POST	api/register	Register a new user to the application
POST	api/oauth/{driver}	Contact a third-party application for user information
GET	api/oauth/{driver}/callback	Obtain an authentication token from the third-party application and create a new user

Authentication .0

POST **api/login** Authenticate a user to access the application

Body

```
{
        email: String                  The email entered by the user
        password: String               The password entered by the user

}
```

Response

```
{
        token: String
        token_type: String
        expires_in: int
}
```

POST **api/logout** Logout the authenticated user from the application

POST **api/register** Register a new user to the application

Body

```
{
        name: String                              The name entered by the user
        email: String                             The email entered by the user
        password: String                          The password entered by the user
        password-confirmation: String             The password confirmation
}
```

Response

User ⌄

```
{
        id: int
        name: String
        email: String
        email_verified_at: timestamp
        photo_url: String
        created_at: timestamp
        updated_at: timestamp
}
```

Authentication .1

POST api/oauth/{driver} Contact a third-party application for user information

Parameter

{driver}	The third-party application name
Example	google

Response

```
{
        url: String          The callback url from the third-party application
}
```

GET api/oauth/{driver}/callback Obtain an authentication token from the third-party application and create a new user

Parameter

{driver}	The third-party application name
Example	github

Response

```
{
        token: String
        token_type: String
        expires_in: int
}
```

02 User .0

GET **api/user** Retrieve information about the authenticated user 🔒

Response

User ⌄

```
{
        id: int
        name: String
        email: String
        email_verified_at: timestamp
        photo_url: String
        created_at: timestamp
        updated_at: timestamp
}
```

GET **api/user/names** Retrieve information about registered users 🔒

Response

[

User ⌄

```
{
        id: int
        name: String
        photo_url: String
}
```

]

PATCH **api/settings/profile** Update information about the authenticated user 🔒

Body

```
{
        name: String
        email: String
}
```

Réponse

User ⌄

```
{
        id: int
        name: String
        email: String
        email_verified_at: timestamp
        photo_url: String
        created_at: timestamp
        updated_at: timestamp
}
```

PATCH api/settings/password Update the password of the authenticated user 🔒

Body

```
{
        password: String
        password_confirmation: String
}
```

03 Projects .0

GET **api/projects** Retrieve all projects

Response

```
[
 Project ⌵
         {
                 id: int
                 title: String
                 description: String
                 filename: String
                 wins: int
                 losses: int
                 rank: int
                 score: int
                 created_at: timestamp
                 updated_at: timestamp
                 user_id: int
                 category_id: int
         }
 ]
```

GET **api/projects/{id}** Retrieve one project

Parameter

`{id}` The id of the project to retrieve

Response

```
Project ⌵
         {
                 id: int
                 title: String
                 description: String
                 filename: String
                 wins: int
                 losses: int
                 rank: int
                 score: int
                 created_at: timestamp
                 updated_at: timestamp
                 user_id: int
                 category_id: int
         }
```

03 Projects .1

POST **api/projects** Create a new project

Body

```
    {
            user_id: int                The id of the authneticated user
            title: String               The title of the project
            description: String         The description of the project
            image: String               The name of the encoded project file
            category: String            The name of the category to which the project belongs
    }
```

Response

```
Project ⌵
        {
                id: int
                title: String
                description: String
                filename: String
                wins: int
                losses: int
                rank: int
                score: int
                created_at: timestamp
                updated_at: timestamp
                user_id: int
                category_id: int
        }
```

DELETE **api/projects/{project}** Delete a project

Parameter

`{project}` The id of the project to delete

Response

`Statut` `String`

Example `"Deleted project item"`

03 Projects .2

PATCH api/projects/{project} Update a project

Parameter

```
{project}        The id of the project to retrieve
```

Body

```
{
          title: String                   The title of the project
          description: String             The description of the project
}
```

Response

```
Project ⌄
    {
            id: int
            title: String
            description: String
            filename: String
            wins: int
            losses: int
            rank: int
            score: int
            created_at: timestamp
            updated_at: timestamp
            user_id: int
            category_id: int
    }
```

POST api/projects/ranking/{category} Retrieve the best projects

Parameter

```
{category}       The slug of the requested category
```

Response

```
[
 Project ⌄
     {
             id: int
             title: String
             description: String
             filename: String
             wins: int
             losses: int
             rank: int
             score: int
             created_at: timestamp
             updated_at: timestamp
             user_id: int
             category_id: int
     }
 ]
```

04 Categories .0

GET **api/categories** Récupérer l'ensemble des catégories

Response

```
[
 Category ⌄
           {
                   id: int
                   name: String
                   slug: String
                   created_at: timestamp
                   updated_at: timestamp
           }
 ]
```

GET **api/games** Récupérer l’ensemble des matchs joués

Response

```
[
 Game ⌄
      {
               id: int
               winner: id
               loser: id
               created_at: timestamp
               updated_at: timestamp
      }
 ]
```

POST **api/games/init/{category}** Récupérer deux projets pour participer à un match

Parameter

`{category}` The slug of the requested category

Response The two projects going to participate in the match

```
[
 Project ⌄
          {
                   id: int
                   title: String
                   description: String
                   filename: String
                   wins: int
                   losses: int
                   rank: int
                   score: int
                   created_at: timestamp
                   updated_at: timestamp
                   user_id: int
                   category_id: int
          }
 ]
```

PATCH **api/games/update** Update projects and create match history

Body

```
{
        winner: int        The id of the winning project

        loser: int         The id of the losing project
}
```

Response The two projects that participated in the match

```
[
 Project ⌄
        {
                id: int
                title: String
                description: String
                filename: String
                wins: int
                losses: int
                rank: int
                score: int
                created_at: timestamp
                updated_at: timestamp
                user_id: int
                category_id: int
        }
]
```

06 Likes .0

GET api/likes Récupère la totalité des mentions like

Response

```
[
 Like ⌄
     {
             id: int
             created_at: timestamp
             updated_at: timestamp
             project_id: id
             user_id: id
     }
 ]
```

POST api/likes Ajouter une mention like

Body

```
    {
            project: int                The id of the project
            user: int                   The id of the user
    }
```

Response

```
Like ⌄
    {
            id: int
            created_at: timestamp
            updated_at: timestamp
            project_id: id
            user_id: id
    }
```

DELETE api/likes/{like} Supprimer une mention like

Parameter

`{like}` The id of the like to delete

Response

`Statut` `String`

Example `"Deleted like"`