

J. Harrault

Music Sheet Writer

Music Sheet Writer

Spécification de l’API

Objectifs du document

Résumé

Ce document spécifie l’API de Music Sheet Writer. Cette API respecte une architecture REST et est utilisée communément par le site internet et les applications mobiles afin de stocker, récupérer et modifier des informations relatives à l’aspect communautaire de Music Sheet Writer.

L’aspect sécuritaire est respecté grâce à la mise en place de rôles et permissions associés aux différents utilisateurs. Chaque rôle donne accès à un ensemble de fonctionnalités au sein de l’API. Ces fonctionnalités sont accessibles en appelant l’API via des requêtes HTTP.

Chacune de ces requêtes est composée d’une URL, d’une méthode HTTP et optionnellement d’un corps et génère une réponse renvoyée par l’API. Ces réponses peuvent signaler soi la bonne réaction de l’API vis-à-vis d’une requête soi un problème au niveau du serveur ou de la requête HTTP.

Glossaire

***– A –***

API (Application Programming Interface) : une API est un ensemble d’artéfacts servant de façade pour permettre à un logiciel d’offrir des services à d’autres logiciels. C’est une porte d’accès à des fonctionnalités en faisant abstraction de leur fonctionnement.

***– C –***

Client HTTP : un client HTTP est un logiciel conçu pour se connecter à un serveur HTTP, comme les navigateurs internet par exemple.

***– H –***

HTTP (*Hypertext Transfer Protocol*) : HTTP est un protocole de communication client-serveur. Il peut fonctionner sur n'importe quelle connexion fiable. Les clients HTTP les plus connus sont les navigateurs Web permettant à un utilisateur d'accéder à un serveur contenant les données.

HTTPS (*Hypertext Transfer Protocol Secure*) : HTTPS est l’agrégation du HTTP et d’une couche de chiffrement. Il garantit la fiabilité du serveur en assurant notamment une sécurité et une confidentialité des données envoyées par l’utilisateur.

***– J –***

JSON (JavaScript Object Notation) : est un format de données textuelles dérivé de la notation des objets du langage JavaScript.

***– R –***

REST (*Representation State Tranfer*) : REST est un style d’architecture logiciel qui, la plupart du temps, est utilisé pour implémenter un Web Service. La communication se fait donc entre un client et serveur et utilise le protocole HTTP.

Ressource (REST) : Une ressource est une entité identifiée au sein d’une arborescence.

***– S –***

Serveur HTTP : un serveur HTTP est un logiciel servant des requêtes à un ou plusieurs clients en respectant le protocole HTTP.

***– U –***

URI (Uniform Resource Identifier) : une URI est l’identifiant d’une ressource sur un réseau.

URL (Uniform Resource Locator) : une URL est l’emplacement d’une ressource sur un réseau. Une URL est un sous-ensemble d’URI.

URN (Uniform Resource Name) : une URN est le nom d’une ressource sur un réseau. Une URN est un sous-ensemble d’URI.

*– W –*

Web Service : un Web Service est un programme informatique de la famille des technologies web permettant la communication et l'échange de données entre applications. Il s'agit donc d'un ensemble de fonctionnalités exposées sur internet ou sur un intranet.

Description du document

|  |  |
| --- | --- |
| Titre | Music Sheet Writer : Spécification de l’API |
| Date de création | 24/10/2015 |
| Date de publication | 25/10/2015 |
| Auteur | J. Harrault |
| Responsable | Jonathan Racaud |
| E-mail | musicsheetwriter\_2017@labeip.epitech.eu |
| Sujet | Spécification de l’API |
| Version du modèle | 2.0 |

Tableau des révisions

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Auteur | Section(s) | Commentaire |
| 24/10/2015 | Jeremy HARRAULT | Toutes | Création du document |
| 25/10/2015 | Jeremy HARRAULT | Toutes | Première version |
| 30/10/2015 | Jeremy HARRAULT | 3.2. Interface (/login) | Changement de « username » à « login » |
| 02/11/2015 | Jeremy HARRAULT | 3.2. Interface (/test) | Ajout de la route /test |
| 12/11/2015 | Jeremy HARRAULT | Glossaire  3.2. Interface (/users GET) | Ajout de « Ressource (REST) »  Limite les utilisateurs renvoyés aux comptes non clôturés. |
| 16/11/2015 | Jeremy HARRAULT | 2.2. Vue en ressource  3.2. Interface (/contact POST) | Le contact est possible en mode visiteur. |
| 18/11/2015 | Jeremy HARRAULT | 3.2. Interface (/users GET)  3.2. Interface (/users/[id] GET)  3.4.2. Liste des erreurs | Correction « status != CLOSED »  Ajout explication sur « is\_favourite »  Ajout des erreurs GLO-USERNOTFOUND, GLO-SCORENOTFOUND, GLO-SUBNOTFOUND, GLO-FAVNOTFOUND |
| 20/11/2015 | Jeremy HARRAULT | 3.2. Interface (/users/[id] GET) | Ajout de “is\_subscription” |
| 22/11/2015 | Jeremy HARRAULT | 3.2. Interface | « is\_subscription » et « is\_favourite » ne sont 0/1 mais true/false. |
| 25/11/2015 | Jeremy HARRAULT | 3.2 Interface (/users/[id] PUT) | Correction du JSON sur le retour 200de la requête |
| 26/11/2015 | Jeremy HARRAULT | 3.2 Interface (/logout) | Changement de permission (MEM => VIS) |
| 05/12/2015 | Jérémy HARRAULT | 3.2 Interface  3.2 Interface (/users/[id]/scores/own/[id] PUT) | Reponse vide et status code à 204 pour les requêtes DELETE  Ajout de l’URL |
| 11/01/2016 | Jeremy HARRAULT | 3.2. Interface (/users/[id]/password PUT) | Modification de route et méthode pour le changement de mot de passe |
| 06/02/2016 | Jeremy HARRAULT | 2.2. Vue en ressources  3.2. Interface (/users/[id]/scores/own POST, /scores/[id] GET et /scores/[id]/preview GET)  3.3. Le téléchargement de fichier  3.4.2. Liste des erreurs | Ajout de la ressources /scores/[id]/preview GET  Modification de la réponse de /scores[id] GET.  Modification du content-type autorisé pour cette ressource /users/[id]/scores/own POST  Ajout du content-type *content\_type\_msw* |
| 13/02/2016 | Jeremy HARRAULT | 3.2 Interface (/users POST et /users/[id]/personal\_data PUT) | Ajout des explications sur firstname et surname  Suppression du champ “photo” en entrée de la requête. |
| 15/02/2016 | Jeremy HARRAULT | 2.2. Vue en ressources  3.2. Interface (/scores/[id] GET et /scores/[id]/preview GET)  3.4. Liste des erreurs  3.3. Le téléchargement de fichier | Suppression des requêtes /scores/[id] GET et /scores/[id]/preview GET  Modification de l’objet « score » : « location » => « location\_project » et « location\_preview »  Ajout de l’erreur « SCO-FAVALREADYADD »  Suppression du content-type pour les partitions |
| 21/02/2016 | Jeremy HARRAULT | 3.2. Interface | Ajout des champs de notoriété pour les utilisateurs, abonnements, abonnés et les partitions |

Sommaire

[1. Présentation de l’API 1](#_Toc436576364)

[1.1. But de l’API 1](#_Toc436576365)

[1.2. Choix architecturaux et composants 1](#_Toc436576366)

[1.2.1. REST 1](#_Toc436576367)

[1.2.2. Le Framework Symfony 1](#_Toc436576368)

[2. Les besoins métiers 2](#_Toc436576369)

[2.1. Utilisateurs et permissions 2](#_Toc436576370)

[2.1.1. Les utilisateurs 2](#_Toc436576371)

[2.1.2. Permissions 3](#_Toc436576372)

[2.2. Vue en ressources 3](#_Toc436576373)

[3. Implémentation 6](#_Toc436576374)

[3.1. Les règles métiers du serveur 6](#_Toc436576375)

[3.2. L’interface 6](#_Toc436576376)

[3.3. Le téléchargement de fichier 25](#_Toc436576377)

[3.4. Les erreurs 25](#_Toc436576378)

[3.4.1. Le formatage 25](#_Toc436576379)

[3.4.2. Liste des erreurs 26](#_Toc436576380)

Liste des Tableaux

[Tableau 1 : Composition d'une requête-réponse HTTP 1](#_Toc436576310)

[Tableau 2 : Description des utilisateurs de l’API 2](#_Toc436576311)

[Tableau 3 : Description des différents niveaux de privilège 3](#_Toc436576312)

[Tableau 4 : Vue en ressources de l'API 5](#_Toc436576313)

[Tableau 5 : Règle métier de l’API 6](#_Toc436576314)

[Tableau 6 : Détails des entrées-sorties des routes de l'API 25](#_Toc436576315)

[Tableau 7 : Les différents "Content-Type" acceptés par l'API 25](#_Toc436576316)

[Tableau 8 : Liste des erreurs possibles renvoyées par l'API 27](#_Toc436576317)

Liste des Figures

[Figure 1 : Cas d'utilisation en fonction des utilisateurs de l’API 2](#_Toc436576318)

[Figure 2 : Schémas des erreurs renvoyées par l'API 25](#_Toc436576319)

# Présentation de l’API

## But de l’API

L’API Music Sheet Writer rend accessible les fonctionnalités de partage et d’accès à la communauté à tous les clients HTTP à savoir les applications mobiles sur iPhone, Android et Windows Phone ainsi qu’aux navigateurs internet. L’unicité de l’API et le regroupement des fonctionnalités au sein d’un seul et même composant facilite la maintenance de l’application.

## Choix architecturaux et composants

### REST

L’API respecte une architecture REST garantissant une modularité élevée et une facilité d’implémentation. REST respecte une interface uniforme sans tenir compte d’un contexte particulier dans la requête envoyée. L’interface d’une architecture REST se définie par un ensemble de ressources indépendant permettant d’accéder ou d’effectuer des opérations sur des données spécifiques. Chaque requête s’effectue sur une et une seule ressource. L’opération à effectuer se détermine par la méthode HTTP. Enfin, des données supplémentaires peuvent être envoyées au serveur dans le « corps » de la requête. Une fois reçue, le serveur traite la requête et renvoie une réponse HTTP. Cette réponse est notamment caractérisée par un code de terminaison HTTP, définissant le type de réponse renvoyée, et, comme la requête, de données optionnelles dans le « corps » de la réponse.

Composition d’une :

|  |  |
| --- | --- |
| Requête | Réponse |
| - URI  - Méthode HTTP  - donnée dans le corps | - Code de terminaison  - données dans le corps |

Tableau 1 : Composition d'une requête-réponse HTTP

### Le Framework Symfony2

*TODO*

*Pourquoi utiliser symfony2 ? Avantage pour le projet ?*

*Quelle sont les fonctionnalités de symfony2 utilisées dans le projet ? (Élément de réponse : Gestion des routes, gestion des permissions, abstraction de la couche données et liaison avec la BDD, gestion de l’authentification de l’utilisateur)*

# Les besoins métiers

## Utilisateurs et permissions

### Les utilisateurs

Il existe plusieurs rôles parmi les utilisateurs de l’API.

|  |  |
| --- | --- |
| *Nom* | *Description* |
| Visiteur | - Un utilisateur non connecté sur le système |
| Membre | - Un utilisateur connecté avec un compte Music Sheet Writer |
| Administrateur | - Un utilisateur connecté avec un compte administrateur Music Sheet Writer |

Tableau 2 : Description des utilisateurs de l’API

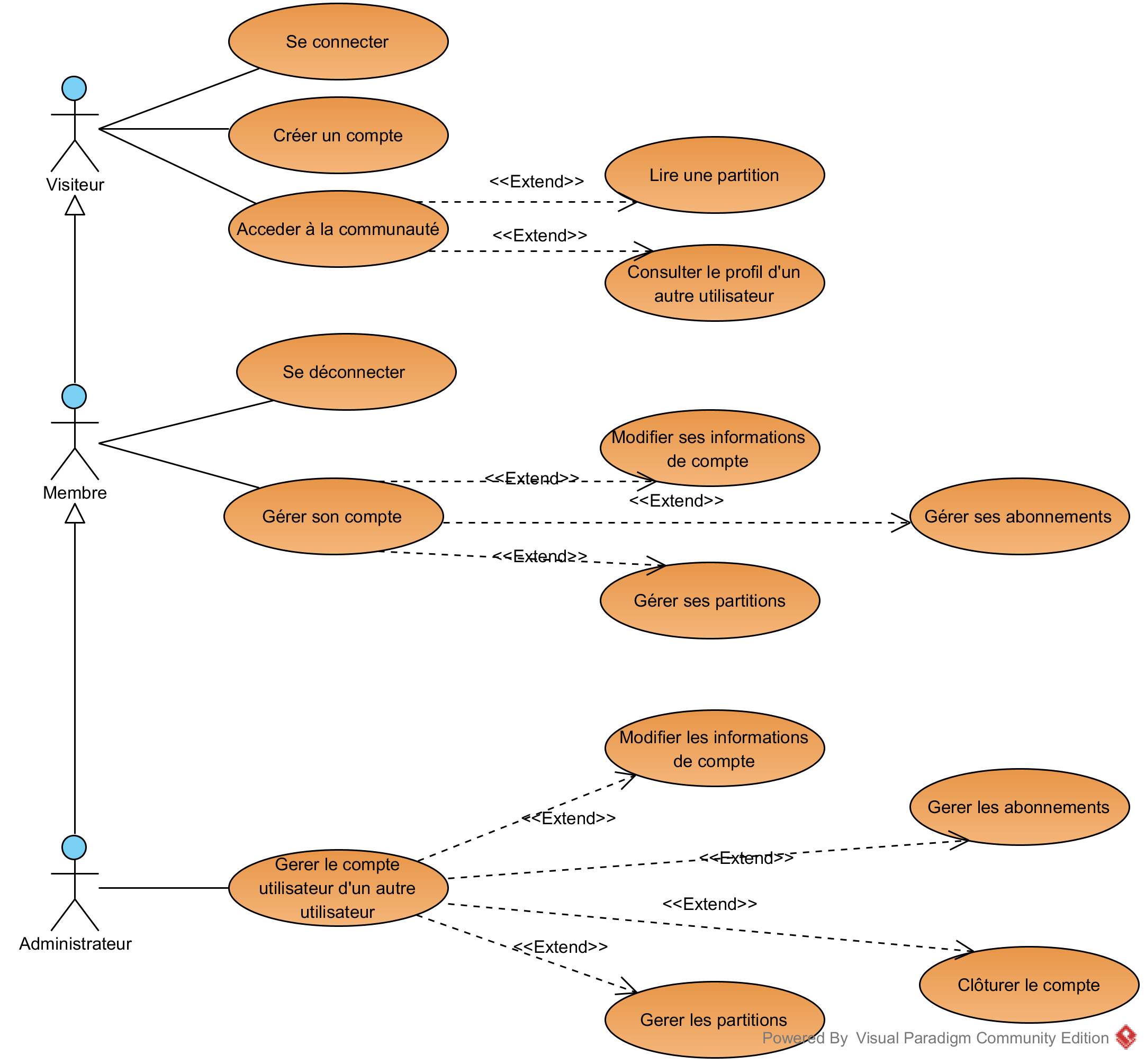


Figure 1 : Cas d'utilisation en fonction des utilisateurs de l’API

### Permissions

Puisqu’il existe différents type d’utilisateurs, certains ayant accès à plus de fonctionnalités que d’autres d’utilisateur, l’API devra supporter plusieurs niveaux de permissions.

|  |  |  |
| --- | --- | --- |
| *Nom court* | *Nom* | *Description* |
| VIS | Droit Visiteur | L’utilisateur peut effectuer l’action, qu’il soit connecté ou pas. |
| MEM | Droit membre | L’utilisateur doit être connecté pour effectuer l’action. S’il est connecté à un compte membre, l’action ne peut être effectuée que sur son compte. L’identifiant utilisateur doit être vérifié. En revanche, si l’utilisateur est connecté sur un compte administrateur, l’action peut être effectuée sur n’importe quel compte. |
| ADM | Droit Administrateur | L’utilisateur doit être connecté à un compte administrateur pour effectuer l’action. |

Tableau 3 : Description des différents niveaux de privilège

## Vue en ressources

La liste ci-dessous définie les ressources disponibles sur le serveur, leur méthodes HTTP autorisées et le niveau de permission associé nécessaires à l’appel de la requête. Les permissions sont décrites dans la partie [2.1-Utilisateurs et permissions](#_Utilisateurs_et_permissions).

|  |  |  |  |
| --- | --- | --- | --- |
| Ressource | Méthode HTTP | Permission | Description |
| /login | POST | VIS | Connecte un utilisateur à partir de ses identifiants de connexion. |
| /logout | POST | VIS | Déconnecte un utilisateur. |
| /forgotten\_password | POST | VIS | Envoie un mail à un utilisateur qui a perdu son mot de passe. |
| PUT | VIS | Réinitialise le mot de passe d’un utilisateur en utilisant un jeton de réinitialisation. |
| /users | GET | VIS | Récupère la liste des utilisateurs. Des filtres peuvent être ajoutés. |
| POST | VIS | Enregistre un nouvel utilisateur. |
| /users/[id] | GET | VIS | Récupère toutes les informations d’un utilisateur (informations personnelles, abonnement, abonnées et partitions). |
| DELETE | ADM | Clôture un compte. |
| /users/[id]/password | PUT | MEM | Change le mot de passe d’un utilisateur en utilisant son mot de passe actuel. |
| /users/[id]/personal\_data | GET | VIS | Récupère toutes les informations personnelles d’un utilisateur. |
| PUT | MEM | Change les informations personnelles d’un utilisateur. |
| /users/[id]/photo | GET | VIS | Récupère la photo de profil d’un utilisateur. |
| PUT | MEM | Change la photo de profil d’un utilisateur. |
| /users/[id]/subscriptions | GET | VIS | Récupère la liste des abonnements d’un utilisateur. |
| POST | MEM | Ajoute un nouvel abonnement à un utilisateur. |
| /users/[id]/subscriptions/[uid] | DELETE | MEM | Retire un abonnement d’un utilisateur. |
| /users/[id]/subscribers | GET | VIS | Récupère la liste des abonnés d’un utilisateur. |
| /users/[id]/scores/own | GET | VIS | Récupère la liste des partitions d’un utilisateur. |
| POST | MEM | Ajoute une nouvelle partition à un utilisateur. |
| /users/[id]/scores/own/[sid] | PUT | MEM | Modifie les informations d’une partition (titre). |
| DELETE | MEM | Retire une partition d’un utilisateur. |
| /users/[id]/scores/favourites | GET | VIS | Récupère la liste des partitions favorites d’un utilisateur. |
| POST | MEM | Ajoute une nouvelle partition favorite à un utilisateur. |
| /users/[id]/scores/favourites/[sid] | DELETE | MEM | Retire une partition favorite d’un utilisateur. |
| /users/[id]/purchases | GET | MEM | Récupère la liste des achats d’un utilisateur. |
| /scores | GET | VIS | Récupère la liste des partitions. Des filtres peuvent être ajoutés. |
| /scores/[id] | DELETE | ADM | Supprime le fichier contenant la partition. |
| /purchases | GET | ADM | Récupère la liste des achats. Des filtres peuvent être ajoutés. |
| /contact | POST | VIS | Envoie un email à l’équipe MSW. |
| /test | ANY | VIS | Test la disponibilité de l’API. |

Tableau 4 : Vue en ressources de l'API

# Implémentation

## Les règles métiers du serveur

Ci-dessous se trouve la liste des règles métiers que le serveur doit implémenter.

|  |  |
| --- | --- |
| Nom | Règle |
| Unicité des comptes | Le nom d’utilisateur et l’adresse email doivent être uniques à chaque compte. De plus, chaque compte est identifié par un « id » (numérique). |
| Mot de passe - format | Le mot de passe doit contenir plus de 6 caractères avec au moins 1 chiffre |
| Statuts de compte | Lorsqu’un compte est créé, son statut est à « PENDING ».  Une fois validé, il passe en « ACTIVATED ».  Si un compte a été clôturé, son statut est à « CLOSED ». |

Tableau 5 : Règle métier de l’API

## L’interface

Les données échangées dans les requêtes et les réponses de l’API sont au format JSON.

Ci-dessous se trouve le détail des requêtes prises en charges par le serveur.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requête | | | Réponse | |
| Ressource | Méthode HTTP | Corps | Code HTTP | Corps |
| /login | POST | *{*  *“login”: ”bob”,*  *“password”: ”azerty123”*  *}*  “login” can be either the username or the email | 200 | *{*  *“id”: 1,*  *“username”: “bob”,*  *“last\_activity\_date”: “10-29-2015”*  *}*  La date est sous la forme MM-DD-YYYY |
| 401 | *{*  *“shortcode”: “LOG-BADCREDENTIALS”,*  *“message” : “Credentials not valid”*  *}* |
| 403 | *{*  *“shortcode”: “LOG-ACCNOTACTIVE”,*  *“message”: “Account not activated”*  *}* |
| 403 | *{*  *“shortcode”: “LOG-CLOSEDACC”,*  *“message”: “Account closed”*  *}* |
| /logout | POST | EMPTY | 204 | EMPTY |
| /forgotten\_password | POST | *{*  *“email”: “bob.marley@jamaica.jm”*  *}* | 204 | EMPTY |
| 404 | *{*  *“shortcode”: “FOR-EMAILNOTFOUND”,*  *“message”: “Email not found”*  *}* |
| PUT | *{*  *“token”: “[token]”,*  *“password”: “azerty123”*  *}* | 204 | EMPTY |
| 400 | *{*  *"shortcode": "GLO-BADFIELD",*  *“message": "Field(s) not correct",*  *"data": {*  *"password": "password not valid",*  *}*  *}* |
| 404 | *{*  *“shortcode”: “FOR-TOKENNOTFOUND”,*  *“message”: “Token not found”*  *}* |
| /users  Des filtres peuvent être ajoutés en tant que paramètre GET  - uname  - fname  - sname  - email | GET | EMPTY | 200 | *[*  *{*  *“id”: 1,*  *“username”: “bob”,*  *“firstname”: “bob”,*  *“lastname”: marley”,*  *“email”: “bob.marley@lamaica.jm”,*  *“message”: “I have only one love”,*  *“status”: “ACTIVATED”,*  *“last\_activity\_date’: {*  *"date": "2015-09-04 16:23:48",*  *"timezone\_type": 3,*  *"timezone": "Europe/Paris"*  *},*  *“creation\_date”: {*  *"date": "2015-09-04 16:23:48",*  *"timezone\_type": 3,*  *"timezone": "Europe/Paris"*  *},*  *“photo”: “[url\_photo]”,*  *“nb\_subscriptions”: [nb\_subscriptions],*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_favourites”: [nb\_favourites],*  *“nb\_scores”: [nb\_scores],*  *“is\_subscription”: false*  *},*  *{*  *“id”: 2,*  *“username”: “bobr”,*  *“firstname”: “bobr”,*  *“lastname”: marleyr”,*  *“email”: “bob.marleyr@lamaica.jm”,*  *“message”: “I have only two love”,*  *“status”: “PENDING”,*  *“last\_activity\_date’: {*  *"date": "2015-09-04 16:23:48",*  *"timezone\_type": 3,*  *"timezone": "Europe/Paris"*  *},*  *“creation\_date”: {*  *"date": "2015-09-04 16:23:48",*  *"timezone\_type": 3,*  *"timezone": "Europe/Paris"*  *},*  *“photo”: “[url\_photo]”,*  *“nb\_subscriptions”: [nb\_subscriptions],*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_favourites”: [nb\_favourites],*  *“nb\_scores”: [nb\_scores],*  *“is\_subscription”: true*  *}*  *…*  *]*  Seuls les comptes non-clôturés (status != « CLOSED ») sont renvoyés.  Le tableau d’utilisateurs contient les utilisateurs correspondants aux filtres de recherche.  Le champ « is\_subscription » est à *true* si l’utilisateur retourné est un abonnement de l’utilisateur connecté. Sinon, il est à *false*. Si aucun utilisateur n’est connecté, le champ est à *false*. |
| POST | *{*  *"username":"bob",*  *"firstname":"bob",*  *"lastname":"marley",*  *"email":"bob.marley@jaimaica.jm",*  *"password":"azerty123",*  *}*  Les champs “firstname” et “surname” sont facultatifs | 200 | *{*  *“id”: 1,*  *“username”: “bob”,*  *“firstname”: bob”,*  *“lastname”: “marley”,*  *“email”: “bob.marley@jamaica.jm”,*  *“message”: “I have only one love”,*  *“status”: “PENDING”,*  *“last\_activity\_date’: null,*  *“creation\_date”: {*  *"date": "2015-09-04 16:23:48",*  *"timezone\_type": 3,*  *"timezone": "Europe/Paris"*  *},*  *“photo”: “[url\_photo]”,*  *}* |
| 400 | *{*  *"shortcode": "GLO-BADFIELD",*  *"message": "Field(s) not correct",*  *"data": {*  *"username": "username not valid",*  *"email": "email not valid"*  *}*  *}* |
| 409 | *{*  *"shortcode": "REG-EALRUSED",*  *"message": "Username or email already used",*  *"data": {*  *"username": "username already used.",*  *"email": "email already used."*  *}*  *}* |
| /users/[id] | GET | EMPTY | 200 | *{*  *“is\_subscription”: true*  *“personal\_data”: {*  *"id": 1,*  *"username": "bob",*  *"firstname": "bob",*  *"lastname": "marley",*  *"email": "bob.marley@jamaica.jm",*  *"message": "I have only one love",*  *"photo": “[url\_photo]”,*  *“nb\_subscriptions”: [nb\_subscriptions],*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_favourites”: [nb\_favourites],*  *“nb\_scores”: [nb\_scores]*  *},*  *“subscription”: [*  *{*  *“id”: 2,*  *“username”: “bobr”,*  *“photo”: “[url\_photo]”,*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_scores”: [nb\_scores],*  *“is\_subscription”: true*  *},*  *{*  *“id”: 3,*  *“username”: “bobs”,*  *“photo”: “[url\_photo]”,*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_scores”: [nb\_scores],*  *“is\_subscription”: true*  *}*  *…*  *],*  *“subscriber”: [*  *{*  *“id”: 2,*  *“username”: “bobr”,*  *“photo”: “[url\_photo]”,*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_scores”: [nb\_scores],*  *“is\_subscription”: true*  *},*  *{*  *“id”: 4,*  *“username”: “bobt”,*  *“photo”: “[url\_photo]”,*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_scores”: [nb\_scores],*  *“is\_subscription”: true*  *}*  *…*  *],*  *“score”: {*  *“own”: [*  *{*  *“id”: 1,*  *“title”: “One love”,*  *“by”: {*  *“id”: 1,*  *“username”: bob”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *“is\_favourite”: true*  *},*  *{*  *“id”: 2,*  *“title”: “Generation”,*  *“by”: {*  *“id”: 1,*  *“username”: “bob”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *“is\_favourite”: false*  *},*  *…*  *],*  *“favourite”: [*  *{*  *“id”: 1,*  *“title”: “One love”,*  *“by”: {*  *“id”: 1,*  *“username”: bob”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *“is\_favourite”: true*  *},*  *{*  *“id”: 2,*  *“title”: “Macumba”,*  *“by”: {*  *“id”: 6,*  *“username”: “Renand”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *“is\_favourite”: false*  *},*  *…*  *]*  *}*  *}*  Le champ « is\_subscription » est à *true* si l’utilisateur retourné est un abonnement de l’utilisateur connecté. Sinon, il est à *false*. Si aucun utilisateur n’est connecté, le champ est à *false*.  Le champ « is\_favourite » est à *true* si la partition retournée est une partition favorite de l’utilisateur connecté. Sinon, il est à *false*. Si aucun utilisateur n’est connecté, le champ est à *false*. |
| DELETE | EMPTY | 200 | *{*  *"id": 1,*  *“closure\_date”: {*  *"date": "2015-09-11 18:26:56",*  *"timezone\_type": 3,*  *"timezone": "Europe/Paris"*  *},*  *"status": "CLOSED"*  *}* |
| 403 | *{*  *"shortcode": "CLO-USERALRCLO",*  *"message": "User account already closed"*  *}* |
| /users/[id]/password | PUT | *{*  *“current\_password”: “azerty123”,*  *“new\_password”: “qwerty456”*  *}* | 204 | EMPTY |
| 400 | *{*  *"shortcode": "GLO-BADFIELD",*  *"message": "Field(s) not correct",*  *"data": {*  *"password": "password not valid"*  *}*  *}* |
| 403 | *{*  *"shortcode": "USR-WRONGPASS",*  *"message": "Current password not correct”*  *}* |
| /users/[id]/personal\_data | GET | EMPTY | 200 | *{*  *"id": 1,*  *"username": "bob",*  *"firstname": "bob",*  *"lastname": "marley",*  *"email": "bob.marley@jamaica.jm",*  *"message": "I have only one love",*  *"photo": “[url\_photo]”,*  *“nb\_subscriptions”: [nb\_subscriptions],*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_favourites”: [nb\_favourites],*  *“nb\_scores”: [nb\_scores]*  *}* |
| PUT | *{*  *"firstname":"bob",*  *"lastname":"marley",*  *"email": "bob.marley@jamaica.jm",*  *"message": "I have only one love”*  *}* | 200 | *{*  *"id": 1,*  *"username": "bob",*  *"firstname": "bob",*  *"lastname": "marley",*  *"email": "bob.marley@jamaica.jm",*  *"message": "I have only one love",*  *"photo": “[url\_photo]”,*  *“nb\_subscriptions”: [nb\_subscriptions],*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_favourites”: [nb\_favourites],*  *“nb\_scores”: [nb\_scores]*  *}* |
| 400 | *{*  *"shortcode": "GLO-BADFIELD",*  *"message": "Field(s) not correct",*  *"data": {*  *"email": "email not valid"*  *}*  *}* |
| /users/[id]/photo | GET | EMPTY | 200 | *{*  *“photo”: “[url\_photo]”*  *}* |
| PUT | En-tête :  *Content-type: [content\_type\_picture]*  Corps :  Le contenu du fichier | 200 | *{*  *“photo”: “[url\_photo]”*  *}* |
| 406 | *{*  *"shortcode": "UPL-BADCONTENTTYPE",*  *"message": "ContentType not acceptable"*  *}* |
| /users/[id]/subscriptions | GET | EMPTY | 200 | *[*  *{*  *“id”: 1,*  *“username”: “bob”,*  *“photo”: “[url\_photo]”,*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_scores”: [nb\_scores],*  *“is\_subscription”: true*  *},*  *{*  *“id”: 2,*  *“username”: “bobr”,*  *“photo”: “[url\_photo]”,*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_scores”: [nb\_scores],*  *“is\_subscription”: true*  *}*  *…*  *]*  Le champ « is\_subscription » est à *true* si l’utilisateur retourné est un abonnement de l’utilisateur connecté. Sinon, il est à *false*. Si aucun utilisateur n’est connecté, le champ est à *false*. |
| POST | *{*  *“id”: 2*  *}* | 200 | *{*  *“id”: 2,*  *“username”: “bobr”,*  *“photo”: “[url\_photo]”*  *}* |
| 404 | *{*  *"shortcode": "SUB-BADUSERID",*  *"message": "User not found"*  *}* |
| /users/[id]/subscriptions/[uid] | DELETE | EMPTY | 204 | EMPTY |
| /users/[id]/subscribers | GET | EMPTY | 200 | *[*  *{*  *“id”: 1,*  *“username”: “bob”,*  *“photo”: “[url\_photo]”,*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_scores”: [nb\_scores],*  *“is\_subscription”: true*  *},*  *{*  *“id”: 2,*  *“username”: “bobr”,*  *“photo”: “[url\_photo]”,*  *“nb\_subscribers”: [nb\_subscribers],*  *“nb\_scores”: [nb\_scores],*  *“is\_subscription”: false*  *}*  *…*  *]*  Le champ « is\_subscription » est à *true* si l’utilisateur retourné est un abonnement de l’utilisateur connecté. Sinon, il est à *false*. Si aucun utilisateur n’est connecté, le champ est à *false*. |
| /users/[id]/scores/own | GET | EMPTY | 200 | *[*  *{*  *“id”: 1,*  *“title”: “One love”,*  *“by”: {*  *“id”: 1,*  *“username”: bob”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *“is\_favourite”: true*  *},*  *{*  *“id”: 2,*  *“title”: “Generation”,*  *“by”: {*  *“id”: 5,*  *“username”: “Simple surface”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *“is\_favourite”: false*  *},*  *…*  *]*  Le champ « is\_favourite » est à *true* si la partition retournée est une partition favorite de l’utilisateur connecté. Sinon, il est à *false*. Si aucun utilisateur n’est connecté, le champ est à *false*. |
| POST | En-tête :  *Content-type: [content\_type\_msw]*  Corps :  Le contenu du fichier | 200 | *{*  *“id”: 1,*  *“title”: “Generation”,*  *“by”: {*  *“id”: 5,*  *“username”: “Simple surface”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *}* |
| 406 | *{*  *"shortcode": "UPL-BADCONTENTTYPE",*  *"message": "ContentType not acceptable"*  *}* |
| /users/[id]/scores/own/[sid] | PUT | {  “title”: “Generation”  } | 200 | *{*  *“id”: 1,*  *“title”: “Generation”,*  *“by”: {*  *“id”: 5,*  *“username”: “Simple surface”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *}* |
| DELETE | EMPTY | 204 | EMPTY |
| /users/[id]/scores/favourites | GET | EMPTY | 200 | *[*  *{*  *“id”: 1,*  *“title”: “One love”,*  *“by”: {*  *“id”: 1,*  *“username”: bob”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *“is\_favourite”: true*  *},*  *{*  *“id”: 2,*  *“title”: “Generation”,*  *“by”: {*  *“id”: 5,*  *“username”: “Simple surface”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *“is\_favourite”: false*  *},*  *…*  *]*  Le champ « is\_favourite » est à *true* si la partition retournée est une partition favorite de l’utilisateur connecté. Sinon, il est à *false*. Si aucun utilisateur n’est connecté, le champ est à *false*. |
| POST | *{*  *“id”: 2*  *}* | 200 | *{*  *“id”: 1,*  *“title”: “Generation”,*  *“by”: {*  *“id”: 5,*  *“username”: “Simple surface”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *}* |
| 404 | *{*  *"shortcode": "FAV-BADSCOREID”,*  *"message": "Score not found"*  *}* |
| 409 | *{*  *"shortcode": "FAV-SCOREALREADYFAV",*  *"message": "Score already in favourites"*  *}* |
| /users/[id]/scores/favourites/[fid] | DELETE | EMPTY | 204 | EMPTY |
| /users/[id]/purchases | GET | EMPTY | 200 | *[*  *{*  *“id”: 1,*  *“date”: {*  *"date": "10-29-2015 00:00:00",*  *"timezone\_type": 3,*  *"timezone": "Europe/Paris"*  *}*  *},*  *{*  *“id”: 1223,*  *“date”: {*  *"date": "10-30-2016 00:00:00",*  *"timezone\_type": 3,*  *"timezone": "Europe/Paris"*  *}*  *},*  *…*  *}* |
| /scores  Des filtres peuvent être ajoutés en tant que paramètre GET  - title  - by | GET | EMPTY | 200 | *[*  *{*  *“id”: 1,*  *“title”: “One love”,*  *“by”: {*  *“id”: 1,*  *“username”: bob”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *“is\_favourite”: true*  *},*  *{*  *“id”: 2,*  *“title”: “Generation”,*  *“by”: {*  *“id”: 5,*  *“username”: “Simple surface”*  *}*  *“location\_project”: “[url\_project]”,*  *“location\_preview”: “[url\_preview]”,*  *“nb\_favourites”: [nb\_favourites],*  *“is\_favourite”: false*  *},*  *…*  *]*  Le tableau de partition contient les partitions correspondantes aux filtres de recherche.  Le champ « is\_favourite » est à *true* si la partition retournée est une partition favorite de l’utilisateur connecté. Sinon, il est à *false*. Si aucun utilisateur n’est connecté, le champ est à *false*. |
| /scores/[id] | DELETE | EMPTY | 204 | EMPTY |
| /purchases | GET | EMPTY | 200 | *[*  *{*  *“id”: 1,*  *“by”: {*  *“id”: 1,*  *“username”: “bob”*  *},*  *“date”: {*  *"date": "10-29-2015 00:00:00",*  *"timezone\_type": 3,*  *"timezone": "Europe/Paris"*  *}*  *},*  *{*  *“id”: 1223,*  *“by”: {*  *“id”: 1,*  *“username”: “bob”*  *},*  *“date”: {*  *"date": "10-30-2016 00:00:00",*  *"timezone\_type": 3,*  *"timezone": "Europe/Paris"*  *}*  *},*  *…*  *]* |
| /contact | POST | *{*  *“email”: “bob.marley@jamaica.jm”,*  *“subject”: ”Connexion Impossible”,*  *”message”: ”je n’arrive pas à me connecter”*  *}* | 201 | EMPTY |
| /test | ANY | Whatever | 200 | {  “status”: “1”  “method”: “ANY”  “body”: “Whatever”  }  Le champ “status” est à un 1 si l’API est fonctionnelle. Sinon, 0.  Le champ “method” est égal à la méthode de la requête.  Le champ “body” est égal aux données envoyées dans le corps de la requête, sous forme de chaine de caractères. Chaine vide si le corps de la requête est vide. |

Tableau 6 : Détails des entrées-sorties des routes de l'API

## Le téléchargement de fichier

Lors d’un téléchargement de fichier, il est demandé de spécifier un « Content-type ». Ci-dessous se trouve la liste des types de contenu acceptés.

|  |  |  |  |
| --- | --- | --- | --- |
|  | *Valeur du « Content-type »* | *Description* | *Extension* |
| *content\_type\_msw* | application/msw | MusicSheetWriter | .msw |
| *content\_type\_picture* | image/jpeg | Image | .jpg .jpeg |
| Image/png | Image | .png |
| image/x-icon | Image | .ico |
| image/gif | Image | .gif |

Tableau 7 : Les différents "Content-Type" acceptés par l'API

## Les erreurs

### Le formatage

En cas d’erreur, la réponse de l’API est de la forme

Figure 2 : Schémas des erreurs renvoyées par l'API

*{*

*« shortcode » : [valeur],*

*« message » : [valeur],*

*« data » : [valeur/objet JSON]*

*}*

Le champ « shortcode » est l’identifiant de l’erreur. Il permet d’identifier précisément l’erreur survenue.

Le champ « message » est l’explication de l’erreur en anglais.

Le champ « data » est optionnel. Il n’est présent que lorsque l’erreur a besoin d’être plus spécifique que le message et/ou lorsqu’elle doit renvoyer des données.

### Liste des erreurs

|  |  |  |  |
| --- | --- | --- | --- |
| Short code | Code HTTP | Message | Explication (optionnel) |
| GLO-MISSING | 400 | The following required parameter are missing: [param1] [param2]… | Il manque des paramètres dans le corps de la requête. Seuls les paramètres manquants et obligatoires peuvent être retournés avec cette erreur |
| GLO-BADFIELD | 400 | *Field(s) not correct* | Le format de certains paramètres dans le corps de la requête est incorrect. |
| GLO-UNAUTHORIZED | 401 | You are not allowed to perform this action | L’utilisateur n’a pas les permissions requises pour effectuer cette action. |
| GLO-RESNFOUND | 404 | The requested resource is not found | L’URI de la requête est invalide. |
| GLO-USERNFOUND | 404 | The user ID is not found | L’ID d’un utilisateur inséré dans la requête /users/[**id**] (et dérivée) est introuvable |
| GLO-SCORENOTFOUND | 404 | The score ID is not found | L’ID d’une partition inséré dans la requête /users/[id]/scores/own/[**id**] ou /scores/[**id**] est introuvable. |
| GLO-SUBNOTFOUND | 404 | The subscription ID is not found | L’ID d’un abonnement inséré dans la requête /users/[id]/subscription/[**id**] est introuvable. |
| GLO-FAVNOTFOUND | 404 | The favourite score ID is not found | L’ID d’une partition favorite inséré dans la requête /users/[id]/scores/favourites/[**id**] est introuvable. |
| GLO-METHNALLOW | 405 | The method used is not allowed on the resource | L’URI est correct mais la méthode HTTP associée n’est pas autorisée. |
| GLO-BADJSON | 415 | The JSON sent is not correct | Le corps de la requête n’est pas sous format JSON |
| GLO-DBSERERR | 502 | The database server does not respond | La base de données n’est accessible |
| GLO-DBSCHERR | 500 | An SQL error occurred | Une anomalie dans le schéma de la base de données est survenue. Elle ne devrait pas être retournée en production |
| *LOG-BADCREDENTIALS* | 401 | *Credentials not valid* | Les identifiants de connexion sont incorrectes |
| *LOG-ACCNOTACTIVE* | 403 | *Account not activated* | Le compte n’a pas été validé via le mail de confirmation d’inscription |
| *LOG-CLOSEDACC* | 403 | *Account closed* | Le compte a été clôturé |
| *FOR-EMAILNOTFOUND* | 404 | Email not found | L’adresse email n’appartient à aucun compte |
| *FOR-TOKENNOTFOUND* | 404 | Token not found | Le jeton de réinitialisation de mot de passe est introuvable |
| *REG-EALRUSED* | 409 | *Username or email already used* | Le nom d’utilisateur et/ou l’adresse email appartiennent déjà à un autre utilisateur |
| *USR-WRONGPASS* | 403 | *Current password not correct* | Le mot de passe actuel du compte n’est pas correct |
| *CLO-USERALRCLO* | 403 | *User account already closed* | Le compte est déjà clôturé |
| *UPL-BADCONTENTTYPE* | 406 | *ContentType not acceptable* | Le champ « Content-type » dans l’en-tête de la requête n’est pas accepté par le serveur pour cette requête. |
| *SUB-BADUSERID* | 404 | *User not found* | L’utilisateur identifié n’existe pas. |
| *FAV-BADSCOREID* | 404 | Score not found | La partition identifiée n’existe pas. |
| *FAV-SCOREALREADYFAV* | 409 | Score already in favourites | La partition est déjà marquée comme favoris. |

Tableau 8 : Liste des erreurs possibles renvoyées par l'API