**Documentation API-AMR**

Table des matières

[INTRODUCTION 3](#_Toc132871261)

[METHOD /nom\_route : Description de l’usage de cette route 3](#_Toc132871262)

[ROUTES /session 5](#_Toc132871263)

[POST /session : Se connecter 5](#_Toc132871264)

[DELETE /session : Se déconnecter 7](#_Toc132871265)

[ROUTES /account 8](#_Toc132871266)

[GET /account : Obtenir les informations de l’utilisateur connecté (deux modes : Normale et Approndies) 8](#_Toc132871267)

[POST /account : Créer un compte pour MembreMr, MembreVolontaire (ne nécessitant pas d’authentification). 10](#_Toc132871268)

[PUT /account : Modifier les informations de l’utilisateur connecté 11](#_Toc132871269)

[DELETE /account : Supprimer son propre compte 12](#_Toc132871270)

[ROUTES /users 13](#_Toc132871271)

[GET /users/{user\_id} : Obtenir les informations d’un utilisateur (deux modes : Normale et Approndies) 13](#_Toc132871272)

[GET /users : Obtenir les informations de tous les utilisateurs avec filtre 16](#_Toc132871273)

[POST /users : Créer un nouvel utilisateur 17](#_Toc132871274)

[POST /users/{user\_id}/favori : Ajouter un utilisateur en favori → [Membre MR], [Membre Volontaire] 19](#_Toc132871275)

[PUT /users/{user\_id} : Modifier les données d’un utilisateur 20](#_Toc132871276)

[PUT /users/{user\_id}/status : Modifier le statut de l’utilisateur (Actif, Désactivé, Refusé) 21](#_Toc132871277)

[DELETE /users/{user\_id} : Supprimer un utilisateur 22](#_Toc132871278)

[DELETE /users/{users\_id}/favori : Supprimer un utilisateur de ses favoris → [Membre MR], [Membre Volontaire] 23](#_Toc132871279)

[ROUTES /helprequests 24](#_Toc132871280)

[GET /helprequests/{helprequest\_id} : Récupérer une demande 24](#_Toc132871281)

[GET /helprequests : Récupérer toutes les demandes 26](#_Toc132871282)

[POST /helprequests : Créer une nouvelle demande → [Membre MR] 28](#_Toc132871283)

[POST /helprequests/{helprequest\_id}/comments : Ajouter un commentaire sur une demande 29](#_Toc132871284)

[POST /helprequests/{helprequest\_id}/comments/{comment\_id}/report : Signaler un commentaire 30](#_Toc132871285)

[PUT /helprequests/{helprequest\_id}/treatment : Modifier un traitement (accepter/refuser/mettre en favori) sur une demande 32](#_Toc132871286)

[PUT /helprequests/{helprequest\_id}/accept/{rmmember\_id} : Accepter ou Refuser le traitement d’une demande par un membre volontaire 33](#_Toc132871287)

[DELETE /helprequests/{helprequest\_id}/comments/{comment\_id} : Supprimer un commentaire 34](#_Toc132871288)

[ROUTES /messages 35](#_Toc132871289)

[GET /messages : Récupérer tous mes messages incluant mes réponses 35](#_Toc132871290)

[POST /messages/{user\_id} : Envoyer un message 36](#_Toc132871291)

# INTRODUCTION

Cette documentation continent les informations précises décrivant les différentes routes composant l’API amr, leur fonctionnement et le cadre de leur usage. L’accès à certaines routes demande l’authentification et parfois même un niveau d’habilitation.

## METHOD /nom\_route : Description de l’usage de cette route

### Description

On décrit simplement ici si l’appel de cette route requiert une authentification et un niveau d’habilitation particulier.

### Droits d’accès :

C’est simplement une description des droits d’accès, s’il faut être authentifié et si oui, avec quel niveau d’habilitation.

Si l’utilisateur n’est pas authentifié alors qu’il le devrait, le code http renvoyé sera 401. Si l’utilisateur ne possède pas les droits suffisants pour accéder à la route ou faire l’opération souhaitée, le code http renvoyé sera 403

### Input [Données/Paramètres d’entrées de la route]

Les données d’entrées sont envoyées :

* En tant que paramètres URL si on accède à la route via la méthode GET
* En tant que données JSON dans le corps de la requête via les méthodes POST, PUT et DELETE

|  |  |  |
| --- | --- | --- |
| Nom paramètre | Type de valeur/Valeurs possibles | Requis ou Facultatif [Si Facultatif, mention de la valeur par défaut] |
| … | … | … |

### Output [Données attendues en retour au format JSON]

Des données en JSON sont systématiquement renvoyées avec un code http lors de l’appel d’une route de l’API.

Les retours de codes http possibles pour cette API sont :

* 200 OK : Tout s’est passé sans erreur
* 400 BADREQUEST : Les données envoyées en entrée ne sont pas valides pour le traitement de celles-ci par l’API.
* 401 UNAUTHORIZED : L’utilisateur devrait être authentifié pour accéder à cette route alors que ce n’est pas le cas (il manque sûrement le jeton d’authentification dans les en-têtes de la requête http)
* 403 FORBIDDEN : L’utilisateur n’a pas les droits suffisant pour accéder à la route. Par exemple, un utilisateur standard appelle une route réservée aux administrateurs
* 404 NOTFOUND : La route ou la ressource n’existe pas.
* 429 TOOMANYREQUEST : L’accès à la route est bloqué pour l’utilisateur car trop de tentatives d’accès ont été effectuées.
* 500 INTERNALERRORSERVER : Une erreur a été rencontré durant l’exécution du code associé à la requête. L’exécution du code a généré un plantage non géré dans l’API.

Dans le corps de la réponse JSON renvoyés, deux champs existent :

* Message : (Présent systématiquement) Description de l’exécution
* Data : (Présent si besoin, et uniquement si le code http est 200 OK) Résultat de données attendues par l’appel de la route.

Plus précisément, un tableau décrit ensuite le format précis des données reçues pour Data :

|  |  |
| --- | --- |
| Nom du champ renvoyé | Type de données de ce champ |
| … | … |

# ROUTES /session

## POST /session : Se connecter

### Description :

Cette route permet de s’authentifier pour récupérer le token JWT, token qui sera utiles pour les routes qui requirent l’authentification.

### Droits d’accès :

Authentification : Non

### Input :

La route étant en POST, les données sont à envoyés au format JSON dans le body de la requête http.

|  |  |  |
| --- | --- | --- |
| Email | String | Requis |
| Password | String | Requis |

### Output (Statut OK) :

Message : ‘Authentification réussie’

Data :

|  |  |
| --- | --- |
| Tokenapi | String |
| User | User |

User :

|  |  |
| --- | --- |
| Id | Integer |
| Email | String |
| Firstname | String |
| Surname | String |
| City | String |
| Postalcode | String |
| Usertype | String :  ‘Administrateur’  ‘Superadministrateur’  ‘Modérateur’  ‘Partenaire’  ‘MembreMr’  ‘MembreVolontaire’ |
| Userstatus | String :  ‘Actif’  ‘En demande d’activation’  ‘Désactivé’  ‘Refusé’ |

Une image contenant texte

Description générée automatiquement

## DELETE /session : Se déconnecter

### Description :

Cette route permet de se déconnecter. Le jeton n’expire mais la date de fin de connexion est enregistrée.

### Input :

Pas de paramètres

### Droits d’accès :

Authentification : Oui

### Output (Statut OK) :

Message : ‘Déconnexion réussie’

Data : [Pas de données]

Une image contenant texte

Description générée automatiquement

# ROUTES /account

## GET /account : Obtenir les informations de l’utilisateur connecté (deux modes : Normale et Approndies)

### Description :

Cette route permet de récupérer les informations de l’utilisateur connecté. Il existe deux modes :

* Normale : Récupération des informations standard (id, nom, prénom, ville, code postale, email, point, …) → informations renseignées lors de la création du compte
* Approfondie : Récupération de toutes les informations reliées au compte. En plus des informations standard, on récupère toutes les connexions, toutes les demandes d’aides, tous les achats, tout ce qui relie l’utilisateur à AMR.

### Droits d’accès :

Authentification : Oui

### Input :

|  |  |  |
| --- | --- | --- |
| Mode | Integer :  ‘0’ : Normale  ‘1’ : Approfondie | Requis |

### Output (Statut OK) :

Message : ‘Utilisateur récupéré’

Data :

|  |  |
| --- | --- |
| Id | Integer |
| Email | String |
| Surname | String |
| Firstname | String |
| City | String |
| Postalcode | String |
| Point | Integer |

Une image contenant texte

Description générée automatiquement

## POST /account : Créer un compte pour MembreMr, MembreVolontaire (ne nécessitant pas d’authentification).

### Description :

Cette route permet de créer un nouveau compte MembreMr et MembreVolontaire seulement. Les nouveaux utilisateurs d’AMR passeront donc par cette route pour créer leur compte.

### Droits d’accès :

Authentification : Non

### Input :

|  |  |  |
| --- | --- | --- |
| Email | String (Email) | Requis |
| Password | String | Requis |
| Surname | String | Requis |
| Firstname | String | Requis |
| Postalcode | String | Requis |
| City | String | Requis |
| Usertype | String :  ‘MembreMr’  ‘MembreVolontaire’ | Requis |

### Output (Statut OK) :

Message : ‘Utilisateur crée avec succès’

Data : [Pas de données]

Une image contenant texte

Description générée automatiquement

## PUT /account : Modifier les informations de l’utilisateur connecté

### Description :

Avec cette route, l’utilisateur connecté pourra modifier ces propres informations standard (nom, prénom, ville, code postale). Le champ email n’est pas modifiable car il s’agit du login de l’utilisateur.

### Droits d’accès :

Authentification : Oui

### Input :

|  |  |  |
| --- | --- | --- |
| Surname | String | Facultatif, aucune valeur par défaut |
| Firstname | String | Facultatif, aucune valeur par défaut |
| City | String | Facultatif, aucune valeur par défaut |
| Postalcode | String | Facultatif, aucune valeur par défaut |

### Output (Statut OK) :

Message : ‘Données de l’utilisateur connecté modifiées avec succès’

Data : [Pas de données]

Une image contenant texte

Description générée automatiquement

## DELETE /account : Supprimer son propre compte

### Description :

Avec cette route, l’utilisateur connecté pourra supprimer son propre compte. Il devra alors obligatoirement indiquer son mot de passe.

### Droits d’accès :

Authentification : Oui

### Input :

|  |  |  |
| --- | --- | --- |
| Password | String | Requis |

### Output (Statut OK) :

Message : ‘L’utilisateur a bien été supprimé’

Une image contenant texte

Description générée automatiquement

# ROUTES /users

## GET /users/{user\_id} : Obtenir les informations d’un utilisateur (deux modes : Normale et Approndies)

### Description :

Cette route permet de récupérer les informations de n’importe quel utilisateur de la base (même ceux inactif). Il existe deux modes :

* Normale : Récupération des informations standard (id, nom, prénom, ville, code postale, email, point, type d’utilisateur, statut d’utilisateur, …) → informations renseignées lors de la création du compte
* Approfondie : Récupération de toutes les informations reliées au compte. En plus des informations standard, on récupère toutes les connexions, toutes les demandes d’aides, tous les achats, tout ce qui relie l’utilisateur à AMR.

### Droits d’accès :

Authentification : Oui

* Administrateur
* SuperAdministrateur

### Input :

|  |  |  |
| --- | --- | --- |
| mode | 0 : ‘Normale’  1 : ‘Approfondie’ | Requis |

### Output (Statut OK) :

Message : ‘Utilisateur récupéré’.

Data :

|  |  |
| --- | --- |
| Id | Integer |
| Email | String |
| Surname | String |
| Firstname | String |
| City | String |
| Postalcode | String |
| Point | Integer |
| Usertype | String :  ‘Administrateur’  ‘MembreMr’  ‘MembreVolontaire’  ‘Modérateur’  ‘Partenaire’  ‘SuperAdministrateur’ |
| Userstatus | String :  ‘Actif’  ‘Demande d’activation’  ‘Non actif’ |

Une image contenant texte

Description générée automatiquement

## GET /users : Obtenir les informations de tous les utilisateurs avec filtre

### Description :

Cette route permet de récupérer les informations plusieurs utilisateurs avec une possibilité de filtre. Par défaut, tous les utilisateurs sont récupérés. Il est possible de filtrer par type d’utilisateur et par statut.

### Droits d’accès :

Authentification : Oui

* Administrateur
* SuperAdministrateur

### Input :

|  |  |  |
| --- | --- | --- |
| usertype | ‘Administrateur’  ‘MembreMr’  ‘MembreVolontaire’  ‘Modérateur’  ‘Partenaire’  ‘SuperAdministrateur’ | Facultatif, par défaut ‘Tout’ |
| userstatus | ‘Actif’  ‘Demande d’activation’  ‘Désactivé’ | Facultatif, par défaut ‘Tout’ |

### Output (Statut OK) :

Message : ‘Utilisateurs récupérés’.

Data :

|  |  |
| --- | --- |
| Id | Integer |
| Email | String |
| Surname | String |
| Firstname | String |
| City | String |
| Postalcode | String |
| Point | Entier |
| Dateinsert | DateTime |
| Usertype | String :  ‘Administrateur’  ‘MembreMr’  ‘MembreVolontaire’  ‘Modérateur’  ‘Partenaire’  ‘SuperAdministrateur’ |
| Userstatus | String :  ‘Actif’  ‘Demande d’activation’  ‘Désactivé’ |

## POST /users : Créer un nouvel utilisateur

### Description :

Avec cette route, l’administrateur pourra créer un nouveau compte de tout type. Un administrateur ne pourra pas créer de compte administrateur. C’est le rôle du superadministrateur.

### Droits d’accès :

Authentification : Oui

* Administrateur : Il peut créer des comptes MembreMr, MembreVolontaire, Partenaire, Modérateur. Il ne peut pas créer des comptes Administrateur ni SuperAdministrateur
* SuperAdministrateur : Il peut créer tout type de créer et n’a aucune limitation

### Input :

|  |  |  |
| --- | --- | --- |
| Email | String (Email) | Requis |
| Password | String | Requis |
| Firstname | String | Requis |
| Surname | String | Requis |
| Postalcode | String | Requis |
| City | String | Requis |
| Usertype | String :  ‘Administrateur’  ‘MembreMr’  ‘MembreVolontaire’  ‘Modérateur’  ‘Partenaire’  ‘SuperAdministrateur’ | Requis |

### Output (Statut OK) :

Message : ‘Utilisateur crée avec succès’

Data : [Pas de données]

Une image contenant texte

Description générée automatiquement

## POST /users/{user\_id}/favorites : Ajouter un utilisateur en favori → [Membre MR], [Membre Volontaire]

### Description :

Cette route permet à un utilisateur connecté de type MembreVolontaire ou MembreMr d’ajouter un autre utilisateur de type MembreVolontaire ou MembreMr dans ses favoris.

### Droits d’accès :

Authentification : Oui

* MembreVolontaire
* MembreMr

### Input :

User\_id

### Output (Statut OK) :

Message : ‘Utilisateur bien ajouté dans vos favoris’

Data : [Pas de données]

## PUT /users/{user\_id} : Modifier les données d’un utilisateur

### Description :

Cette route permet de modifier les informations standards de n’importe quel utilisateur. Les administrateurs ne peuvent pas modifier les données d’autres administrateurs.

### Droits d’accès :

Authentification : Oui

* Administrateur : Les administrateur ne peuvent modifier que les comptes MembreMr, MembreVolontaire, Partenaire, Modérateur
* SuperAdministrateur : Ils peuvent modifier tous type de compte

### Input :

|  |  |  |
| --- | --- | --- |
| Surname | String | Facultatif |
| Firstname | String | Facultatif |
| City | String | Facultatif |
| Postalcode | String | Facultatif |

### Output (Statut OK) :

Message : ‘Données de l’utilisateur modifiées’

Data : [Pas de données]

## PUT /users/{user\_id}/status : Modifier le statut de l’utilisateur (Actif, Désactivé, Refusé)

### Description :

Cette route permet de modifier le statut d’un compte. Ce sera par exemple utilisé pour valider la création d’un compte par l’état. Ce sera aussi utilisé pour les administrateurs et les superadministrateurs.

### Droits d’accès :

Authentification : Oui

* [Membre état] : Peut accepter ou refuser un MembreMR ou MembreVolontaire
* Administrateur : Peut modifier l’état d’un compte MembreMr, MembreVolontaire, Partenaire, Modérateur. Il lui est impossible de modifier l’état d’un compte Administrateur ou SuperAdministrateur
* Superadministrateur : Peut modifier l’état de tous les comptes.

### Input :

|  |  |  |
| --- | --- | --- |
| Userstatus | Entier :  ‘Activé’  ‘Désactivé’  ‘Refusé’ | Requis |

### Output (Statut OK) :

Message : ‘Utilisateur accepté/refusé’

Data : [Pas de données]

## DELETE /users/{user\_id} : Supprimer un utilisateur

### Description :

Cette route permet de supprimer un utilisateur à partir de son identifiant.

### Droits d’accès :

Authentification : Oui

* Administrateur : Les administrateur ne peuvent supprimer que des comptes MembreMr, MembreVolontaire, Partenaire, Modérateur
* SuperAdministrateur : Ils peuvent supprimer tous types de compte

### Input :

[Pas de données d’entrées]

### Output (Statut OK) :

Message : ‘Utilisateur supprimé’

Data : [Pas de données]

## DELETE /users/{users\_id}/favorites/{user\_id} : Supprimer un utilisateur de ses favoris → [Membre MR], [Membre Volontaire]

### Description :

Cette route permet à un utilisateur connecté de type MembreVolontaire ou MembreMr de supprimer un utilisateur de type MembreVolontaire ou MembreMr de ses favoris.

### Droits d’accès :

Authentification : Oui

* MembreVolontaire
* MembreMr

### Input :

[Pas de données d’entrées]

### Output (Statut OK) :

Message : ‘Utilisateur bien supprimé dans vos favoris’

Data : [Pas de données]

# ROUTES /helprequests

## GET /helprequests/{helprequest\_id} : Récupérer une demande

### Description :

Cette route permet à un utilisateur de récupérer la demande. Sinon les droits, les utilisateurs ne récupéreront pas les mêmes informations, et n’auront pas les mêmes droits sur les demandes.

Droits d’accès :

Authentification : Oui

* MembreVolontaire : Pourra accéder à toutes les demandes créées + les demandes qui lui sont reliées (qu’il a accepté et/ou terminé). S’il n’a pas les droits sur cette requête, Erreur 401
* MembreMr : Pourra accéder à tous les demandes qui lui sont reliées (demande qu’il a crée). Si n’a pas les droits sur cette requête, Erreur 401
* Administrateur, SuperAdministrateur : Pourront accéder à toutes les demandes sans aucune disctinction

### Input :

[Pas de données]

### Output (Statut OK) :

Message : ‘Demande récupéré

Data :

|  |  |
| --- | --- |
| title | String |
| Estimatedtime | Integer |
| Date | DateTime |
| City | String |
| Postalcode | String |
| Description | String |
| Mark | Integer ou Null |
| Helprequestcategory | String :  ‘Tâches ménagères’  ‘Espaces verts’  ‘Courses’  ‘Soutien informatique’  ‘Transports’  ‘Bricolage’  ‘Alimentation’ |
| Helprequeststatus | String :  ‘Créée’  ‘Accepté’  ‘Terminé’ |
| Helprequestvolunteermember | Null ou String => le nom et prénom du membre volontaire en charge de la demande |
| Helprequestrmmember | Null ou String => le nom et prénom du membremr qui a formulé la demande |

## GET /helprequests : Récupérer toutes les demandes

### Description :

Cette route permet à un utilisateur de récupérer toutes les demandes. Sinon les droits, les utilisateurs ne récupéreront pas toutes les demandes (seules ceux sur lesquelles ils possèdent des droits). Ils n’auront donc pas les mêmes droits sur tous les demandes.

### Droits d’accès :

Authentification : Oui

* MembreVolontaire : Pourra accéder à toutes les demandes créées + les demandes qui lui sont reliées (qu’il a accepté et/ou terminé).
* MembreMr : Pourra accéder à tous les demandes qui lui sont reliées (demande qu’il a crée).
* Administrateur, SuperAdministrateur : Pourront accéder à toutes les demandes sans aucune disctinction

### Input :

|  |  |  |
| --- | --- | --- |
| City | String | Facultatif, par défaut la ville de l’utilisateur |
| Postalcode | String | Facultatif, par défaut le code postal de l’utilisateur |
| Radius | Integer => Rayon en kilomètres au alentours des coordonnées géographiques de la ville et code postale | Facultatif, par défaut 100 |
| Helprequestcategory | String :  ‘Tâches ménagères’  ‘Espaces verts’  ‘Courses’  ‘Soutien informatique’  ‘Transports’  ‘Bricolage’  ‘Alimentation’ | Facultatif |
| Helprequeststatus | String :  ‘Créée’  ‘Accepté’  ‘Terminé’ | Facultatif |

### Output (Statut OK) :

Message : ‘Demande récupéré

Data [sous forme de tableau] :

|  |  |
| --- | --- |
| title | String |
| Estimatedtime | Integer |
| Date | DateTime |
| City | String |
| Postalcode | String |
| Description | String |
| Mark | Integer ou Null |
| Helprequestcategory | String :  ‘Tâches ménagères’  ‘Espaces verts’  ‘Courses’  ‘Soutien informatique’  ‘Transports’  ‘Bricolage’  ‘Alimentation’ |
| Helprequeststatus | String :  ‘Créée’  ‘Accepté’  ‘Terminé’ |
| Helprequestvolunteermember | Null ou String => le nom et prénom du membre volontaire en charge de la demande |
| Helprequestrmmember | Null ou String => le nom et prénom du membremr qui a formulé la demande |

## POST /helprequests : Créer une nouvelle demande → [Membre MR]

### Description :

Cette route permet à un membre MR de créer une nouvelle demande. A la création de la demande, le statut de celle-ci sera ‘Créée’. Elle ne sera reliée à aucun membre volontaire.

### Droits d’accès :

Authentification : Oui

* MembreMr

### Input :

|  |  |  |
| --- | --- | --- |
| title | String | Requis |
| Estimatedtime | Integer | Requis |
| Date | DateTime | Requis |
| City | String | Requis |
| Postalcode | String | Requis |
| Description | String | Requis |
| Helprequestcategory | String :  ‘Tâches ménagères’  ‘Espaces verts’  ‘Courses’  ‘Soutien informatique’  ‘Transports’  ‘Bricolage’  ‘Alimentation’ | Requis |

### Output (Statut OK) :

Message : ‘Demande créée’

Data : [Pas de données]

## POST /helprequests/{helprequest\_id}/comments : Ajouter un commentaire sur une demande

### Description :

Cette route permet à un membremr d’ajouter un commentaire sur une demande. Lors de sa création, le commentaire est reliée automatiquement au membre mr connecté.

### Droits d’accès :

Authentification : Oui

* MembreMr : Ne peux ajouter de commentaires que sur des demandes qui sont terminées qui lui seul lui a créé. Sinon Erreurr 401.

### Input :

|  |  |  |
| --- | --- | --- |
| content | String | Requis |

### Output (Statut OK) :

Message : ‘Commentaire ajoutée’

Data : [Pas de données]

## POST /helprequests/{helprequest\_id}/comments/{comment\_id}/report : Signaler un commentaire

### Description :

Cette route permet à un membrevolontaire ou un membremr de signaler un commentaire. Les modérateurs pourront avoir accès aux commentaires signalées par les membresmr et membrevolontaire.

### Droits d’accès :

Authentification : Oui

* MembreMr
* MembreVolontaire

### Input :

[Pas de données]

### Output (Statut OK) :

Message : ‘Commentaire signalée’

Data : [Pas de données]

## PUT /helprequests/{helprequest\_id}/treatment : Modifier un traitement (accepter/refuser/mettre en favori) sur une demande

### Description :

Cette route permet à un membre volontaire de traiter une demande d’aide. Il peut avec cette route soit l’accepter, soit la mettre en favori.

### Droits d’accès :

Authentification : Oui

* MembreVolontaire

### Input :

|  |  |  |
| --- | --- | --- |
| helprequesttreatment | String :  ‘Favoriser’  ‘Accepter’  ‘Refuser’ | Requis |

### Output (Statut OK) :

Message : ‘Demande bien refusée/accepté/enregistré en favori’

Data : [Pas de données]

## PUT /helprequests/{helprequest\_id}/accept/{rmmember\_id} : Accepter ou Refuser le traitement d’une demande par un membre volontaire

### Description :

Cette route permet à un membremr d’accepter ou de refuser le traitement de sa demande d’aide par un membrevolontaire. Une fois la demande acceptée, le MembreMr ne pourra plus accepter pour un autre membre volontaire.

Si la demande d’aide est acceptée, le statut dans cette demande change à ‘Acceptée’ et on met à jour le membre volontaire. Ainsi si cette même demande avait été acceptée par d’autres membres volontaires, ce sera refusé pour ces autres membres volontaires.

Si la demande d’aide est refusée, un message préconfiguré est envoyé au Membre Volontaire.

Le paramètre user\_id passé dans l’adresse URL devra obligatoirement être un utilisateur de type ‘MembreMr’ sinon Erreur 400.

### Droits d’accès :

Authentification : Oui

* MembreMr

### Input :

|  |  |  |
| --- | --- | --- |
| helprequestaccept | Integer :  0 => ‘Refusé’  1 => ‘Accepté’ | Requis |

### Output (Statut OK) :

Message : ‘Demande bien refusée/accepté/enregistré en favori’

Data : [Pas de données]

## PUT /helprequests/{helprequest\_id}/finish : Déclarer une demande d’aide comme terminée

### Description :

Cette route permet à un membremr de déclarer une requête d’aide comme terminée. C’est ici qu’il va inscrire le temps réel passée par le membre volontaire pour effectuer la tâche, la note attribuée. Des points seront donc calculés et attribuer au membre volontaire.

Il ne pourra déclarer comme terminée qu’une requête d’aide déclaré comme acceptée et qui lui est associé (c’est lui qui l’a créer), sinon Erreur 401.

### Droits d’accès :

Authentification : Oui

* MembreMr, il n’aura accès qu’aux requêtes d’aides que lui seul à créer et qui sont en statut acceptée.

### Input :

|  |  |  |
| --- | --- | --- |
| Realtimepassed | Datetime | Requis |
| Mark | Integer | Requis |

### Output (Statut OK) :

Message : ‘Demande enregistrée comme terminée’

Data : [Pas de données]

## DELETE /helprequests/{helprequest\_id}/comments/{comment\_id} : Supprimer un commentaire

### Description :

Cette route permet à un membremr de supprimer un commentaire fait sur une demande.

### Droits d’accès :

Authentification : Oui

* MembreMr

### Input :

[Pas de données]

### Output (Statut OK) :

Message : ‘Commentaire supprimé’

Data : [Pas de données]

[GET /helprequests/{helprequest\_id}/comments : Consulter les commentaires faites sur une demande]

# ROUTES /messages

## GET /messages : Récupérer tous mes messages incluant mes réponses

### Description :

Cette route permet à un utilisateur de récupérer tous les messages envoyés par lui-même et ceux qui étant destinées.

### Droits d’accès :

Authentification : Oui

* MembreVolontaire : Ne pourra accéder qu’aux messages qui le concerne
* MembreMr : Ne pourra accéder qu’aux messages qui le concerne

### Input :

[Pas de données]

### Output (Statut OK) :

Message : ‘Messages récupérées’

Data [sous forme de tableau] :

|  |  |
| --- | --- |
| messageorigin | Integer :  0 => Fromthisuser  1 => Fromanotheruser |
| Message | String |
| Date | DateTime |
| Usermessage | String => Nom et prénom du communiquant |

## POST /messages/{user\_id} : Envoyer un message

### Description :

Cette route permet à un utilisateur connecté d’envoyer un message à un destinataire.

### Droits d’accès :

Authentification : Oui

* MembreVolontaire : Ne pourra envoyer de message qu’à un membremr, sinon 401
* MembreMr : Ne pourra envoyer de message qu’à un membrevolontaire, sinon 401

### Input :

|  |  |  |
| --- | --- | --- |
| content | String | Requis |

### Output (Statut OK) :

Message : ‘Message envoyé’

Data : [Pas de données]

POST /gifts : Créer un nouveau cadeau

GET /gifts : Récupérer tous les cadeaux

GET /gifts/{gift\_id} : Récupérer les détails d’un cadeau

PUT : /gifts/{gift\_id} : Modifier un cadeau

DELETE /gifts/{gift\_id} : Supprimer un cadeau