

Note de cadrage



Groupomania

SOMMAIRE

Modèle de données	2
Technologies	4
Répertoire des routes de l'API	5

Modèle de données

user

Le modèle de données pour un utilisateur est le suivant :

- **id**: *number* — identifiant unique créé par Sequelize;
- **pseudonym**: *string* — pseudonyme de l'utilisateur;
- **registration**: *string* — numéro de matricule de l'utilisateur;
- **password**: *string* — mot de passe de l'utilisateur;
- **oldPassword**: *string* — première version du mot de passe, à la création du compte de l'utilisateur;
- **resetKey**: *string* — clé de réinitialisation de mot de passe;
- **image**: *string* — avatar de l'utilisateur (par rapport aux images disponible du côté client (front));
- **isAdmin**: *boolean* — statut du compte (Administrateur ou Utilisateur);
- **createdAt**: *string* — date de création (généré par Sequelize);
- **updatedAt**: *string* — date de dernière modification (généré par Sequelize);

message

Le modèle de données pour une publication est le suivant :

- **id**: *number* — identifiant unique créé par Sequelize CLI;
- **idUsers**: *number* — identifiant unique Sequelize pour l'utilisateur qui a créé la publication;
- **title**: *string* — titre de la publication;

message

Le modèle de données pour une publication est le suivant :

- **content**: *string* — texte principal de la publication;
- **image**: *string* — image de la publication téléchargée par l'utilisateur;
- **createdAt**: *string* — date de création (généré par Sequelize);
- **updatedAt**: *string* — date de dernière modification (généré par Sequelize);

comment

Le modèle de données pour un commentaire est le suivant :

- **id**: *number* — identifiant unique créé par Sequelize;
- **idUsers**: *number* — identifiant unique Sequelize pour l'utilisateur qui a créé le commentaire;
- **idMessages**: *number* — identifiant unique Sequelize pour la publication dont le commentaire est lié;
- **comment**: *string* — texte principal du commentaire;
- **createdAt**: *string* — date de création (généré par Sequelize);
- **updatedAt**: *string* — date de dernière modification (généré par Sequelize);

report

Le modèle de données pour une signalisation est le suivant :

- **id**: *number* — identifiant unique créé par Sequelize;
- **idUsers**: *number* — identifiant unique Sequelize pour l'utilisateur qui a créé la signalisation;

report

Le modèle de données pour une signalisation est le suivant :

- **idMessages:** *number* — identifiant unique Sequelize pour la publication dont la signalisation est lié (peut être null);
- **idComments:** *number* — identifiant unique Sequelize pour la commentaire dont la signalisation est lié (peut être null);
- **report:** *string* — texte principal de la signalisation;
- **createdAt:** *string* — date de création (généré par Sequelize);
- **updatedAt:** *string* — date de dernière modification (généré par Sequelize);

Technologies

Pour l'API:

- serveur: NodeJS
- framework: Express
- toutes les opérations (migrations, seeders) de la base de données doivent utiliser l'environnement Sequelize CLI avec des schémas (models) de données stricts.
- base de données: MySQL

Pour l'application front:

- serveur: NodeJS
- framework: Vue CLI (extensions: Vuex pour le stockage de données dynamique, Vue-router pour l'arborescence URL)
- toutes les opérations de requête vers l'API doivent utiliser le pack Axios.

Verb	Paramètres	Corps de la requête (demandé)	Corps de la réponse (attendu)	Fonction
POST	/api/user/login	{ registration: string, password: string }	{ user: {userId: number, pseudonym: string, image: string, newUser: boolean, isAdmin: boolean}, token: string }	Se connecter
PUT	/api/user/reset-password	{ registration: string, resetKey: string }	{ message: string }	Réinitialiser le mot de passe d'un compte précis
GET	/api/user/profiles	{userId: string }	{ {id: number, pseudonym: string, image: string}, {id: number, pseudonym: string, image: string}, ... }	Récupérer les infos de l'utilisateur, afin de les afficher sur sa fiche
GET	/api/user/profile/:id	{userId: string }	{id: number, pseudonym: string, image: string }	Récupérer les infos de l'utilisateur, afin de les afficher sur sa fiche
GET	/api/user/profile/:id /messages	{userId: string }	{ {id: number, idUsers: number, title: string, content: string, image: string, createdAt: string, updatedAt: string}, {id: number, idUsers: number, title: string, content: string, image: string, createdAt: string, updatedAt: string}, ... }	Récupérer tous les messages d'un utilisateur
PUT	/api/user/profile/:id	{userId: string, pseudonym: string, password: string, image: string }	{ user: {userId: number, pseudonym: string, image: string, newUser: boolean, isAdmin: boolean}, token: string }	Mettre à jour les informations du compte de l'utilisateur

Verb	Paramètres	Corps de la requête (demandé)	Corps de la réponse (attendu)	Fonction
POST	/api/admin/user	{userId: string, registration: string, password: string, resetKey: string } (\$ {userId} est celui de l'administrateur)	{ message: string }	Créer un compte d'utilisateur
DELETE	/api/admin/user/:id	{userId: string } (\$ {userId} est celui de l'administrateur)	{ message: string }	Supprimer le compte d'utilisateur précis
GET	/api/reports	{userId: string } (\$ {userId} est celui de l'administrateur)	{ {id: number, idUsers: number, idMessages: number, idComments: number, report: string, createdAt: string, up- datedAt: string}, {id: nu- mber, idUsers: number, idMessages: number, idComments: number, report: string, create- dAt: string, updatedAt: string}, ...}	Afficher les signalements envoyés par les utilis- ateurs
POST	/api/report	{userId: string, idMessages: string, idComments: string, report: string }	{ message: string }	Envoyer un signalement
DELETE	/api/report/:id	{userId: string } (\$ {userId} est celui de l'administrateur)	{ message: string }	Supprimer un signalement précis

Verb	Paramètres	Corps de la requête (demandé)	Corps de la réponse (attendu)	Fonction
GET	/api/messages/	{userId: string }	{ {id: number, idUsers: number, title: string, createdAt: string, updatedAt: string, user:{pseudonym:string}}, {id: number, idUsers: number, title: string, createdAt: string, updatedAt: string, user:{pseudonym:string}}, ...}	Récupérer les messages, publiés sur le serveur
GET	/api/messages/:id	{userId: string }	{id: number, idUsers: number, title: string, content: string, createdAt: string, updatedAt: string, user:{pseudonym:string, image:string}}	Récupérer un message précis, publié sur le serveur
POST	/api/messages/	{userId: string, title: string, content: string, image: file }	{ message: string }	Publier un message
PUT	/api/messages/:id	{userId: string, title: string, content: string, image: file }	{ message: string }	Modifier un message
DELETE	/api/messages/:id	{userId: string }	{ message: string }	Supprimer un message
GET	/api/messages/:id/comments	{userId: string }		Récupérer les commentaires rattachés à un message précis

Verb	Paramètres	Corps de la requête (demandé)	Corps de la réponse (attendu)	Fonction
POST	/api/messages/:id/comment	{userId: string, comment: string }	{ message: string }	Créer un commentaire, pour un message précis
PUT	/api/messages/:idMessages/comment/:id	{userId: string, comment: string }	{ message: string }	Modifier un commentaire précis, pour un message précis
DELETE	/api/messages/:idMessages/comment/:id	{userId: string }	{ message: string }	Supprimer un commentaire précis, pour un message précis