Jumi-Dorian Hoareau
Souksana Phetdara

Compte Rendu : Documentation Technique de l'API

Introduction

Dans cette documentation nous vous présentons en détail comment nous avons tester les différentes routes de l'API, leurs méthodes HTTP, les paramètres requis ainsi que les réponses que nous avons obtenues.

Routes de l'API

1. Tester le serveur

- **Méthode** : GET
- **Route** : `/api/test`
- **Accès** : Public
- **Réponse** : `"hello"`

2. Enregistrement d’un utilisateur

- **Méthode** : POST
- **Route** : `/api/register`
- **Accès** : Public
- **Body (JSON)** :
  {

  "nom": "Dupont",

  "prenom": "Jean",

  "email": "jean.dupont@example.com",

  "pseudo": "jeanD",

  "password": "monmotdepasse"

  }

- **Réponse attendue** :
  { "message": "OK" }
- **Erreurs possibles** :
  - `400` : Email déjà utilisé
  - `400` : Mot de passe trop court

Jumi-Dorian Hoareau
Souksana Phetdara

3. Connexion (Obtenir un token)

- **Méthode** : POST
- **Route** : `/api/login`
- **Accès** : Public
- **Body (JSON)** :
  {

  "email": "jean.dupont@example.com", // string, format email

  "password": "monmotdepasse" // string

  }

- **Réponse attendue** :
  { "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6..." }
- **Erreurs possibles** :
  - `401` : Email ou mot de passe incorrect

4. Profil utilisateur (Token requis)

- **Méthode** : GET
- **Route** : `/api/profil`
- **Accès** : Utilisateur authentifié
- **Headers** :
  `Authorization: Bearer [TOKEN]`
- **Réponse attendue** :
  {

  "nom": "Dupont",

  "prenom": "Jean",

  "email": "jean.dupont@example.com",

  "pseudo": "jeanD"

  }

- **Erreurs possibles** :
  - `401` : Token invalide ou expiré

5. Upload d'un fichier (Token requis)

- **Méthode** : POST
- **Route** : `/api/add-file`
- **Accès** : Utilisateur authentifié

- **Headers** :
  `Authorization: Bearer [TOKEN]`
- **Body** : Type : `form-data`
  - **Champ** : `file` (fichier à envoyer, formats autorisés : JPG, PNG, PDF, max 5 Mo)
- **Réponse attendue** :
  { "message": "Fichier uploadé", "filename": "exemple.jpg" }
- **Erreurs possibles** :
  - `400` : Format de fichier non autorisé
  - `400` : Fichier trop volumineux

6. Supprimer un utilisateur (Admin requis)

- **Méthode** : DELETE
- **Route** : `/api/users/rm/:id`
- **Accès** : Administrateur
- **Headers** :
  `Authorization: Bearer [TOKEN_ADMIN]`
- **Paramètres** :
  - `:id` (integer, ID valide d'un utilisateur existant)
- **Réponse attendue** :
  { "message": "Utilisateur supprimé" }
- **Erreurs possibles** :
  - `404` : Utilisateur non trouvé

7. Bloquer un utilisateur (Admin requis)

- **Méthode** : PUT
- **Route** : `/api/users/ban/:id`
- **Accès** : Administrateur
- **Headers** :
  `Authorization: Bearer [TOKEN_ADMIN]`
- **Réponse attendue** :
  { "message": "Utilisateur banni" }

8. Lister tous les utilisateurs (Admin requis)

- **Méthode** : GET
- **Route** : `/api/users/list`
- **Accès** : Administrateur
- **Headers** :
  `Authorization: Bearer [TOKEN_ADMIN]`
- **Réponse attendue** :
  [

  {

"id": 1,

"nom": "Dupont",

"prenom": "Jean",

"email": "jean.dupont@example.com",

"pseudo": "jeanD",

"role": "user"

}

]

9. Élever un utilisateur en admin (Admin requis)

- **Méthode** : PUT
- **Route** : `/api/user/up/:id`
- **Accès** : Administrateur
- **Headers** :
  `Authorization: Bearer [TOKEN_ADMIN]`
- **Réponse attendue** :
  { "message": "Utilisateur promu admin" }

10. Rétrograder un admin en utilisateur (Admin requis)

- **Méthode** : PUT
- **Route** : `/api/user/down/:id`
- **Accès** : Administrateur
- **Headers** :
  `Authorization: Bearer [TOKEN_ADMIN]`
- **Réponse attendue** :
  { "message": "Utilisateur rétrogradé" }