Application de gestion de réservations de restaurants

Documentation Technique

1. Introduction

Cette documentation technique décrit l'architecture et les choix techniques de l'application de gestion de réservations de restaurants. L'application permet aux utilisateurs de s'inscrire, de s'authentifier, de gérer les utilisateurs (ajouter, modifier, supprimer) et de récupérer la liste des utilisateurs.

2. Architecture

L'application est construite en utilisant une architecture client-serveur. Le frontend est une application web simple utilisant HTML, CSS et JavaScript, tandis que le backend est construit avec Node.js et Express.js. Les données sont stockées dans une base de données relationnelle (par exemple, MySQL).

2.1. Frontend

- **HTML** : Structure de la page web.
- **CSS** : Styles pour la mise en page.
- **JavaScript** : Gestion des événements et communication avec le backend via fetch.

2.2. Backend

- **Node.js** : Environnement d'exécution JavaScript côté serveur.
- **Express.js** : Framework web pour Node.js.
- **MySQL** : Base de données relationnelle pour stocker les informations des utilisateurs.

3. Choix Techniques

3.1. Node.js et Express.js

- **Node.js** est choisi pour sa performance et sa capacité à gérer un grand nombre de connexions simultanées.
- **Express.js** simplifie la création de routes et la gestion des requêtes HTTP.

3.2. MySQL

- **MySQL** est utilisé pour sa robustesse et sa capacité à gérer des transactions complexes. Il est bien adapté pour les applications nécessitant des relations entre les données.


```
{
  "nom": "John",
  "prenom": "Doe",
  "password": "password123",
  "adresse": "123 Main St",
  "email": "john.doe@example.com",
  "carte_bancaire": "1234567890123456"
```

Application de gestion de réservations de restaurants

5.2. Authentification

- **Route** : POST /authenticate
- **Description** : Permet à un utilisateur de s'authentifier en fournissant son email et son mot de passe.
- **Exemple de requête** :


```
{
  "email": "john.doe@example.com",
  "password": "password123"
}

}
```


5.3. Récupération des Utilisateurs

- **Route** : GET /users
- **Description** : Récupère la liste de tous les utilisateurs.
- **Exemple de réponse** :


```
[
 {
   "id": 1,
   "nom": "John",
   "prenom": "Doe",
   "email": "john.doe@example.com",
   "adresse": "123 Main St",
   "carte_bancaire": "1234567890123456"
 },
 ...
]
```


5.5. Suppression d'un Utilisateur

- **Route** : DELETE /users/:id

Application de gestion de réservations de restaurants

- **Description** : Supprime un utilisateur spécifique.
- **Exemple de réponse** :

{

"message": "Utilisateur supprimé"

}