🧾 Documentation Technique – GSB-MedOrder

📌 1. Introduction

🎯 Objectif

Développer une plateforme de **commande de médicaments** entre **praticiens** (visiteurs médicaux) et **pharmaciens**, gérée par des **administrateurs**.

👥 Public cible

- Administrateurs
- Pharmaciens
- Praticiens

🧰 Technologies utilisées

- **Frontend :** React + Vite
- **Backend :** Node.js + Express
- **Base de données :** MySQL

🛠️ 2. Installation & Configuration

🔄 Prérequis

- Node.js
- npm ou yarn
- MySQL

🖥️ Backend

```
# 1. Cloner le dépôt
git clone <repo_url>

# 2. Accéder au dossier Backend
cd Backend/

# 3. Installer les dépendances
npm install

# 4. Créer le fichier .env avec :
DB_HOST=
DB_USER=
DB_PASSWORD=
DB_NAME=
JWT_SECRET=

# 5. Lancer le serveur
npm start
# ou
node app.js
```

💻 Frontend

```
# 1. Aller à la racine du projet (frontend)
cd GSB-Medorder/

# 2. Installer les dépendances
npm install

# 3. Lancer le serveur
npm run dev
```

🗃️ Base de données

- Créer la base avec les **scripts SQL fournis**
- Importer les tables : `users`, `medicaments`, `commandes`, etc.

🏗️ 3. Architecture Générale

🔄 Schéma

Frontend (React) ↔ Backend (Express) ↔ MySQL

🗂️ Structure des Dossiers

🧩 Frontend

Dossier	Rôle
`assets/`	Images et icônes
`components/`	Composants réutilisables (`Header`, `Footer`, etc.)
`pages/`	Pages du site (`LoginPage`, `HomePage`, etc.)
`services/`	Appels API (`authService.js`, etc.)
`store/`	(optionnel) État global
`styles/`	Feuilles de style
`utils/`	Fonctions utilitaires
`App.jsx`, `main.jsx`	Entrée de l'app

🧱 Backend

Dossier	Rôle
`config/`	Connexion MySQL
`controllers/`	Logique métier
`routes/`	Endpoints API
`middleware/`	Authentification / rôles
`app.js`	Point d'entrée serveur
`.env`	Variables sensibles

🧩 4. Composants Frontend Principaux

- `LoginForm` : Connexion utilisateur
- `ProductList` : Liste des médicaments
- `OrderForm` : Création de commande
- `UserProfile` : Profil utilisateur
- `Header`, `Sidebar` : Navigation

🔌 5. Services Frontend

- `authService.js` : Login, logout, gestion du token
- `apiService.js` : Appels HTTP génériques
- `cartService.js` : Gestion du panier
- `orderService.js` : Création / récupération de commandes

🌐 6. API Backend

🔐 `authRoutes.js`

Méthode	Route	Action
POST	`/api/auth/login`	Connexion utilisateur
POST	`/api/auth/register`	Création utilisateur

👤 `userRoutes.js`

Méthode	Route	Action

GET	`/api/users`	Liste des utilisateurs
GET	`/api/users/:id`	Détails utilisateur
POST	`/api/users`	Ajouter utilisateur

📦 `commandeRoutes.js`

CRUD complet sur les commandes.

💊 `medicamentRoutes.js`

CRUD complet sur les médicaments.

📬 Exemple d'appel API

- **Méthode :** POST
- **URL :** `/api/commandes`
- **Body :**

```
{
  "userId": 1,
  "produits": [
    {
      "medicamentId": 2,
      "quantite": 3
    }
  ]
}
```

- **Réponse :** `201 Created` ou `400 Bad Request`

🗄️ 7. Base de Données (MySQL)

📚 Tables principales

Table	Champs
`users`	id, nom, email, mot_de_passe, role
`medicaments`	id, nom, description, prix, stock
`commandes`	id, user_id, date
`commande_details`	commande_id, medicament_id, quantite

🔗 Relations

- `users` 1⟶N `commandes`
- `commandes` N⟷N `medicaments` via `commande_details`

📖 8. Guides Utiles

➕ Ajouter une route

- Créer le **contrôleur**
- Ajouter dans le fichier **routes**
- Consommer dans le **service React**
- Créer la **page React** si nécessaire

🔒 Authentification JWT

- Le **token JWT** est généré au login
- Stocké dans le `localStorage`

- Protéger les routes avec `authMiddleware.js`

🚀 Déploiement

- **Backend :** utiliser PM2 ou Docker

Frontend :

npm run build

- Puis déployer sur un serveur web (Nginx, Apache…)

Souhaites-tu que je prépare aussi une version PDF ou un fichier Markdown à télécharger ?