📝 1. CAHIER DES CHARGES – APPLICATION MEDIGARDE

1. Présentation générale

Nom de l'application : MediGarde
Objectif :

- Localiser les pharmacies de garde à proximité selon la date et l'heure.
- Offrir une boutique en ligne pour chaque pharmacie afin de vendre des médicaments.
- Permettre la livraison à domicile pour les utilisateurs dans l'incapacité de se déplacer.

2. Technologies utilisées

Partie	Technologie
Front-end	React Native (Expo)
Backend	Node.js + Express.js (TypeScript)
Base de données	MongoDB (avec Mongoose)
Authentification	Firebase Authentication (OTP)
Notifications	Firebase Cloud Messaging (FCM)
Cartographie	Google Maps API
Stockage	Firebase Storage

3. Fonctionnalités principales

A. Utilisateurs

- Création de compte avec numéro de téléphone (OTP)
- Géolocalisation automatique
- Consultation des pharmacies de garde autour de soi (liste + carte)
- Détails d'une pharmacie
- Accès à la boutique d'une pharmacie
- Ajout d'articles au panier
- Validation de commande : adresse, heure, paiement (Cash on Delivery)
- Suivi et historique des commandes
- Notifications push (commandes et mises à jour)

B. Pharmacies (côté Admin)

- Authentification admin

- Gestion des infos pharmacie (nom, horaire, géoloc, image)
- Gestion de la boutique (ajout/suppression/modif. médicaments)
- Gestion des commandes reçues (statut, acceptation, livraison)
- Consultation des avis

4. Modèle de données MongoDB (simplifié)

Utilisateur

```
{
  _id,
  phone: string,
  name: string,
  address: string,
  favorites: [pharmacyId],
  orders: [orderId]
}
```

Pharmacie

```
{
  _id,
  name: string,
  location: {
    lat: number,
    lng: number,
    address: string
  },
  isOnCall: boolean,
  store: [medicineId],
  orders: [orderId],
  adminAccount: { email, password }
}
```

Médicament

```
{
  _id,
  name: string,
  price: number,
  stock: number,
  description: string,
  requiresPrescription: boolean,
  imageUrl: string,
  pharmacyId: ObjectId
}
```

Commande

```
{
  _id,
  userId: ObjectId,
  pharmacyId: ObjectId,
  medicines: [
    { medicineId, quantity }
  ],
```

```
  total: number,
  status: "pending" | "accepted" | "in_delivery" | "delivered" | "refused",
  deliveryAddress: string,
  createdAt,
  updatedAt
}
```

5. Écrans de l’application mobile

Authentification

- Splash screen
- Onboarding
- OTP screen

Utilisateur

- Home (pharmacies à proximité)
- Map View
- Détails pharmacie
- Boutique pharmacie
- Médicament (détail)
- Panier
- Checkout
- Commandes (liste + détail)
- Profil

Admin pharmacie

- Dashboard pharmacie
- Liste des médicaments
- Ajout / édition médicament
- Liste des commandes
- Détail commande

6. Livrables attendus

- Code source complet (frontend + backend)
- Base de données structurée avec données test
- Documentation technique
- Manuel utilisateur (pour testeurs)
- Vidéo démonstrative

7. Contraintes et recommandations

- Interface responsive et accessible
- Optimisation des ressources (images, stockage)
- Conformité RGPD (données perso)
- Sécurité : vérification d’inputs, OTP, protections backend

🎁 8. STRUCTURE DU BACKEND

✅ Technologies à installer :

```
npm install express mongoose cors dotenv body-parser
npm install -D typescript ts-node @types/node @types/express
```

📁 Arborescence recommandée :

```
backend/
├── controllers/
├── models/
├── routes/
├── services/
├── middlewares/
├── utils/
├── server.ts
├── .env
```

🛡️ .env

```
MONGODB_URI=mongodb+srv://...
API_KEY=jihane152003
API_PORT=5000
```

📧 server.ts

Configuration Express + MongoDB + routes + sécurité

📂 middlewares/apiKeyMiddleware.ts

Vérifie les requêtes avec API_KEY

📂 models/

Définit :

- Utilisateur
- Pharmacie
- Médicament
- Commande

📂 routes/

Contient :

- `pharmacy.routes.ts`
- `doctor.routes.ts`
- `auth.routes.ts`
- `order.routes.ts`

📂 services/

- `otp.service.ts` (Firebase)
- `geolocation.service.ts`
- `order.service.ts`

📂 utils/distance.ts

Fonction pour calculer la distance entre deux points GPS (Haversine)

💻 9. STRUCTURE DU FRONTEND (Expo)

📁 Écrans React Native :

- SplashScreen
- OnboardingScreen
- OTPVerificationScreen
- HomeScreen (liste + map)
- PharmacyDetailScreen
- StoreScreen
- MedicineDetailScreen
- CartScreen
- CheckoutScreen
- OrderHistoryScreen
- ProfileScreen

📁 Composants :

- `PharmacyCard.tsx`
- `DoctorCard.tsx`
- `MedicineCard.tsx`

📂 services/ApiService.ts

- `getPharmacies()`
- `getDoctors()`
- `getCurrentLocation()`
- `placeOrder()`

📂 utils/

- `formatDate.ts`, `validatePhone.ts`, etc.

📄 10. PLAN D'ACTION SUIVI

Phase 1 : Authentification OTP

- Configurer Firebase Auth avec téléphone
- Créer OTPVerificationScreen
- Stocker user dans AsyncStorage

Phase 2 : Backend setup

- Créer backend Express + models + routes sécurisées
- Tester API `/api/pharmacies`, `/api/doctors`, `/api/orders`

Phase 3 : Intégration frontend/backend

- HomeScreen : localisation + fetch data + affichage cards
- MapScreen : afficher pharmacie sur carte avec `react-native-maps`

Phase 4 : Fonctionnalités pharmacie

- StoreScreen : voir médicaments
- Panier, commande, paiement, suivi

Phase 5 : Admin pharmacie

- Dashboard React (web) ou mobile séparé

Phase 6 : Bonus

- Notifications Firebase (expo-firebase-notifications)
- Upload d'image (Firebase Storage)

Souhaites-tu que je crée les premiers fichiers de backend pour toi maintenant ?