**FormEase : Cahier des charges fonctionnel et technique pour développement dans Cursor**

**Objectif**

Développer une application web nommée **FormEase** qui permet à un utilisateur de créer un compte, de générer des formulaires d’inscription pour ses activités/événements, de collecter et de gérer les inscriptions, d’exporter des données, de valider ou rejeter des inscrits, d’envoyer des mails personnalisés, le tout avec un système de gestion d’abonnement Freemium (limitation à 1 formulaire et 100 inscrits en gratuit, illimité en Premium).

**Instructions à respecter impérativement**

* Ne développer que les fonctionnalités listées ci-dessous.
* Ne pas ajouter de fonctionnalités non spécifiées ou d’améliorations non demandées.
* Utiliser uniquement les technologies mentionnées.
* Respecter la structuration de la base de données donnée.
* Toutes les interfaces et fonctionnalités doivent être accessibles en français ET en anglais, selon la langue du navigateur par défaut, avec possibilité de changer dans le profil.
* L’interface doit rester simple, claire, responsive, et accessible.

**Fonctionnalités à développer**

**1. Gestion des utilisateurs**

* Création de compte et connexion par email/mot de passe.
* Stocker la langue préférée de l’utilisateur lors de l’inscription (langue du navigateur/OS par défaut).
* Profil utilisateur éditable (nom, prénom, email, mot de passe, langue).

**2. Abonnement Freemium**

* Par défaut, un utilisateur a droit à 1 seul formulaire actif et 100 inscrits visibles.
* Un utilisateur Premium (8 $/mois, paiement par Paypal, Stripe, ou Proxypay mobile money) a droit à un nombre illimité de formulaires et d’inscrits.
* Gestion du paiement et mise à jour automatique du statut d’abonnement.
* Archivage automatique des données pour les utilisateurs gratuits après 2 mois.

**3. Création et gestion des formulaires**

* Interface de création de formulaire en drag & drop (glisser-déposer).
* L’utilisateur choisit et configure parmi la liste suivante, uniquement :

**Champs de saisie**

* + Champ texte simple (input type="text")
  + Zone de texte longue (textarea)
  + Champ mot de passe (input type="password")
  + Champ email (input type="email")
  + Champ numéro (input type="number")
  + Champ téléphone (input type="tel")
  + Champ URL (input type="url")
  + Champ date (input type="date")
  + Champ heure (input type="time")
  + Champ datetime-local (input type="datetime-local")
  + Champ couleur (input type="color")
  + Champ recherche (input type="search")

**Champs de choix/options**

* + Boutons radio (input type="radio")
  + Cases à cocher (input type="checkbox")
  + Liste déroulante simple (select)
  + Liste déroulante multiple (select multiple)
  + Liste de suggestions/autocomplete (datalist)

**Fichiers**

* + Upload de fichier (input type="file")

**Contrôles**

* + Bouton submit (envoi du formulaire)
  + Bouton reset (réinitialisation)
  + Bouton normal (pour action personnalisée comme ouvrir une modale)

**Spécifiques**

* + Champ caché (input type="hidden")
  + Captcha/reCAPTCHA
  + Slider (input type="range")
  + Échelle (rating stars)
  + Signature numérique (capture via souris ou tactile)

**UI/Accessibilité**

* + Labels, fieldset, legend
  + Placeholder, messages d’erreur inline
  + Tooltip/info-bulle
* L’utilisateur peut configurer : l’ordre, l’obligation du champ, l’intitulé (français/anglais), options de réponse, valeurs par défaut.
* Customisation du style du formulaire (couleur, police, image de fond, logo).

**4. Gestion des inscriptions**

* Pour chaque formulaire, l’utilisateur voit un tableau des inscrits.
* Tableau avec filtres dynamiques selon les champs du formulaire (ex : sexe, profession…).
* Recherche, tri, pagination.
* Sélection multiple par cases à cocher, sélection globale.
* Actions groupées : suppression (corbeille), validation, envoi d’un mail personnalisé.
* Export des inscrits en PDF ou CSV : possibilité de choisir les champs à exporter.
* Les inscrits validés reçoivent un mail personnalisé (template custom, logo et infos contact insérés automatiquement).

**5. Dashboard et statistiques**

* Utiliser la librairie Tremor pour l’affichage du dashboard (statistiques sur les inscriptions, taux de validation, etc.).
* Un dashboard par utilisateur, récapitulant : nombre d’inscrits, nombre de formulaires, stats dynamiques par filtre.

**6. Gestion multi-langue**

* Toutes les interfaces, emails, notifications et exports doivent s’afficher en français ou anglais selon la langue détectée du navigateur (avec override possible dans le profil).
* Les labels de champs de formulaire doivent être saisis dans les deux langues.

**7. SuperAdmin**

* Un dashboard spécial accessible uniquement par les comptes SuperAdmin.
* Permet de : voir la liste des utilisateurs, gérer les abonnements, voir tous les formulaires, gérer les incidents, supprimer ou archiver un compte utilisateur.

**Structuration technique**

**Technologies imposées**

* **Front-End :** React (Next.js), TailwindCSS, Tremor UI, react-i18next, React DnD (pour le builder drag & drop)
* **Back-End :** Node.js (Express ou NestJS) OU Python (FastAPI)
* **Database :** PostgreSQL
* **Mailing :** Sendgrid ou équivalent (mails transactionnels uniquement)
* **Paiement :** Stripe, Paypal, Proxypay (mobile money)
* **Fichiers/Uploads :** Amazon S3 ou équivalent cloud

**Modèles de base de données**

User

- id

- first\_name

- last\_name

- email

- password\_hash

- role (user, premium, superadmin)

- language (fr, en)

- plan (free, premium)

- plan\_expiration

- created\_at

- updated\_at

Form

- id

- user\_id

- title

- description

- config (JSON des champs, options, style)

- is\_active

- created\_at

- updated\_at

FormField

- id

- form\_id

- type

- label\_fr

- label\_en

- options (JSON)

- is\_required

- order

- style (JSON)

- placeholder\_fr

- placeholder\_en

- default\_value

Submission

- id

- form\_id

- data (JSON)

- status (new, validated, trashed)

- validated\_at

- notified

- notified\_at

- mail\_opened (bool)

- created\_at

Payment

- id

- user\_id

- provider

- amount

- currency

- status

- transaction\_ref

- created\_at

AdminActionLog

- id

- admin\_id

- action\_type

- target\_type

- target\_id

- description

- created\_at

**Contraintes et exigences**

* **Respecter strictement la liste des champs de formulaire fournie, sans en ajouter.**
* **Le builder drag & drop ne doit permettre de configurer que ces éléments.**
* **Respecter la gestion multilingue, l’UI, l’export, les mails.**
* **Respecter la limitation Freemium : 1 formulaire et 100 inscrits pour gratuit, illimité en Premium.**
* **Archivage automatique des données après 2 mois pour le gratuit.**
* **Aucune fonctionnalité non demandée ne doit être développée.**

**Sécurité**

* Hashage des mots de passe, gestion sécurisée des tokens
* Limitation anti-spam sur les formulaires publics (Captcha)
* RGPD : droit à l’oubli, gestion claire des données archivées/supprimées

**Ce cahier des charges est strict :  
Cursor doit se limiter à ces fonctionnalités, cette stack, ces modèles de données et rien de plus.  
L’application doit être optimisée pour la simplicité, la sécurité, et le respect des limitations définies.**

Parfait !  
Je te propose **les deux : une arborescence de dossier claire** pour un projet Next.js + Node.js/Express + PostgreSQL, **et un schéma de workflow** pour illustrer le parcours utilisateur et la logique métier.

**1. Exemple d’arborescence de dossier**

*(pour un projet fullstack moderne, clean, évolutif)*

/formease

│

├── /frontend # Frontend React/Next.js

│ ├── /public # Images, assets statiques, favicon, logo

│ ├── /src

│ │ ├── /components # Composants UI réutilisables (Button, FormField, etc.)

│ │ ├── /modules # Fonctions spécifiques (builder, dashboard, etc.)

│ │ ├── /pages # Pages Next.js (index.js, login.js, dashboard.js, etc.)

│ │ ├── /hooks # Custom React hooks (useForm, useAuth, useLang)

│ │ ├── /context # Context API (auth, langue, abonnement)

│ │ ├── /styles # Feuilles de style (Tailwind config)

│ │ ├── /utils # Fonctions utilitaires (helpers, formatters)

│ │ ├── /locales # Fichiers de traduction (fr.json, en.json)

│ │ └── /lib # Librairies tierces et intégrations spécifiques

│ ├── next.config.js # Config Next.js

│ ├── tailwind.config.js # Config Tailwind

│ └── package.json

│

├── /backend # Backend Node.js/Express ou FastAPI

│ ├── /src

│ │ ├── /controllers # Logique métier (UserController, FormController…)

│ │ ├── /routes # Endpoints API REST

│ │ ├── /models # Modèles DB (ORM: Sequelize, Prisma…)

│ │ ├── /middleware # Auth, validation, logs

│ │ ├── /services # Services externes (mail, paiement, fichiers)

│ │ ├── /utils # Helpers, sécurité

│ │ └── app.js # Entrée principale backend

│ ├── /config # Config (env, base de données)

│ ├── package.json

│ └── README.md

│

├── /docs # Documentation projet (cahier des charges, Swagger, UML, etc.)

│

├── /scripts # Scripts de déploiement, migration DB

│

├── .env # Variables d’environnement globales

├── README.md # Documentation générale projet

└── docker-compose.yml # Pour lancer l’ensemble facilement

**2. Schéma de workflow utilisateur**

**Workflow de base “utilisateur”**

[Accueil]

│

▼

[Inscription / Connexion] ←─────────────────────────────────────────────────────┐

│ │

▼ │

[Détection langue du navigateur] → [Profil initialisé avec langue auto] │

│ │

▼ │

[Dashboard utilisateur] ─────────────────────────────────────────────────────────┘

│

├─► [Créer un formulaire (drag & drop)] ─► [Configurer les champs, styles]

│ │

│ ▼

│ [Publier et partager le formulaire (URL)]

│ │

│ ▼

│ [Collecte des inscriptions]

│ │

│ ▼

│ [Tableau des inscrits : filtrer, valider, exporter, envoyer mail]

│ │ │ │ │

│ ▼ ▼ ▼ ▼

│ [Suppression] [Validation] [Export PDF/CSV] [Envoi mail]

│ │

│ ▼

│ [Statut mis à jour]

│

├─► [Upgrade abonnement] (accès Premium, paiement en ligne)

│ │

│ ▼

│ [Formulaires illimités, stats avancées…]

│

└─► [Gérer son profil] (changer langue, infos, mot de passe)

(Si SuperAdmin : accès au dashboard admin pour surveiller et modérer)

**Workflow du builder de formulaire (drag & drop)**

[Dashboard utilisateur]

│

▼

[Créer un formulaire]

│

▼

[Drag & Drop des champs autorisés]

│

▼

[Configurer les propriétés (obligatoire, options, labels FR/EN, style…)]

│

▼

[Enregistrer / Publier le formulaire]

**Workflow abonnement Freemium**

[Utilisateur gratuit]

│

├─► [Crée 1 formulaire et collecte max 100 inscrits]

│

├─► [Archive automatique des données après 2 mois]

│

└─► [Invitation à passer Premium]

│

▼

[Page paiement (Paypal/Stripe/Proxypay)]

│

▼

[Accès Premium débloqué]