**Voici les spécifications fonctionnelles révisées, centrées sur la solution la plus simple (un seul fichier data.json) :**

**Spécifications Fonctionnelles – Interface Admin Blog (Solution Unifiée)**

**1. Contexte & Objectifs**

**But : proposer une interface d’administration simple pour gérer un blog statique basé sur un unique fichier JSON (data.json), accessible publiquement en lecture seule, et modifiable uniquement par les administrateurs.  
Cibles : administrateurs et rédacteurs ayant pour rôle de créer, modifier et publier du contenu.**

**2. Stockage & Flux de données**

* **Fichier unique /public/data/data.json  
  Contient trois collections :**
* **{**
* **"sections": [ … ],**
* **"posts": [ … ],**
* **"authors": [ … ]**
* **}**
* **Publication**
  + **L’interface admin écrase data.json à chaque sauvegarde d’un article, d’une section ou d’un auteur.**
  + **Stockage sur AWS S3 via Amplify Storage, niveau public, pour permettre au front public et aux sites tiers de consommer directement le JSON.**
* **Récupération côté Front**
* **import { Storage } from 'aws-amplify';**
* **async function fetchData() {**
* **const url = await Storage.get('data.json', { level: 'public' });**
* **const res = await fetch(url);**
* **return res.json(); // { sections, posts, authors }**
* **}**

**3. Définitions Clés**

* **Slug : version « URL-friendly » du titre, utilisée comme identifiant de l’article et nom de fichier (/blog/mon-article.json).**
* **Section : catégorie de haut niveau (ex. « React & Frontend »).**
* **Tag : mot-clé fin pour filtrer ou rechercher (ex. optimisation).**
* **PublishAt : date/heure de mise en ligne différée.**
* **search.json (optionnel) : index léger généré ultérieurement pour la recherche rapide (sans contenu complet).**

**4. Navigation (Menu Admin)**

* **Tableau de bord**
  + **Statistiques : nombre d’articles, brouillons, publications programmées.**
* **Articles**
  + **Liste / création / édition / suppression / publication différée.**
* **Sections**
  + **Liste / création / édition / tri par ordre.**
* **Auteurs**
  + **Liste / création / édition.**
* **Tags (si implémentés)**
* **Paramètres**
  + **Export CSV, configuration multisites.**

**5. Gestion des Articles**

**5.1. Liste des Articles**

* **Affichage en tableau :**
  + **Titre**
  + **Statut (draft | published | archived)**
  + **Type (tutoriel, annonce…)**
  + **Sections (noms)**
  + **Auteur**
  + **Tags**
  + **Date de création / MAJ / publication**
* **Fonctions :**
  + **Création d’un nouvel article**
  + **Filtre / tri / recherche (par titre, section, auteur, statut, tag)**
  + **Pagination ou chargement infini**

**5.2. Formulaire d’Édition/Création**

* **Champs de données :**
  + **Identité : id (auto-généré), slug (généré puis modifiable), title, excerpt**
  + **Contenu : content (éditeur Markdown/WYSIWYG), format (markdown|html), videoUrl**
  + **Organisation : sectionIds (multi-sélection), tagIds, type, status, visibility, isFeatured**
  + **SEO & Métadonnées : seo.title, seo.description, seo.image, publishAt, createdAt (auto), updatedAt (auto)**
* **Actions :**
  + **Sauvegarder brouillon**
  + **Publier / Dépublier**
  + **Supprimer**
  + **Prévisualiser (aperçu front)**

**6. Gestion des Sections & Auteurs**

* **Sections**
  + **Liste des sections avec id, title, description, order**
  + **Tri via champ order ou drag’n’drop**
  + **Création / édition / suppression**
* **Auteurs**
  + **Liste des auteurs avec id, name, bio, email**
  + **Création / édition / suppression**
* **Tags (si besoin)**
  + **Liste avec id, name, description, color**
  + **Création / édition / suppression**

**7. Génération du fichier JSON**

* **À chaque opération CRUD (création, modification, suppression) :**
  1. **L’API interne agrège les trois collections (sections, posts, authors).**
  2. **Écrit / écrase le fichier unique /public/data/data.json.**
  3. **(Optionnel) Met à jour search.json pour la recherche rapide :**
     + **Contient uniquement les métadonnées (sans content)**
     + **Structure d’élément :**
     + **{**
     + **"slug": "...",**
     + **"title": "...",**
     + **"excerpt": "...",**
     + **"authorId": 2,**
     + **"sectionIds": ["react"],**
     + **"tags": ["react", "optimisation"],**
     + **"type": "tutoriel",**
     + **"status": "published",**
     + **"seo": { "title": "...", "description": "..." },**
     + **"createdAt": "...",**
     + **"updatedAt": "...",**
     + **"publishAt": "..."**
     + **}**

**8. Validation & Gestion des Erreurs**

* **Frontend + Backend :**
  + **Champs obligatoires : title, content, sectionIds, authorId**
  + **Slug unique (vérification lors de la saisie)**
  + **URL vidéo bien formée**
* **Affichage des erreurs : messages inline, tooltips, mise en évidence des champs invalides.**

**9. Authentification & Sécurité**

* **Connexion via AWS Cognito ou NextAuth.js**
* **Rôles :**
  + **Admin : CRUD complet**
  + **Rédacteur : création et édition, pas de suppression**
* **Accès : interface admin accessible uniquement aux utilisateurs authentifiés**

**10. Évolutions Futures**

* **Découpage en plusieurs fichiers JSON (sections.json, posts.json, authors.json) si besoin de performance ou modularité.**
* **Moteur de recherche côté client avec search.json et typage avancé.**
* **Export multisites automatisé (FTP/SFTP, API distante).**
* **Tableau de bord enrichi (statistiques de lecture, analytics d’utilisation).**

**Cette version simplifiée te garantit un développement rapide et une mise en production immédiate. L’architecture reste extensible si tu souhaites découper ou ajouter des fonctionnalités plus tard.**