**Техническое задание “Платформа для просмотра PDF-отчётов”**

**1. Краткое описание**

**Разработать минималистичную платформу для хранения и просмотра PDF-отчётов с организацией по папкам и датам. Основной стек: React + TypeScript (frontend), NestJS + TypeScript (backend), PostgreSQL (БД).**

**2. Функциональные требования**

**2.1. Общие**

* **Поддержка хранения и просмотра PDF-файлов.**
* **Простая, чистая структура без лишнего визуального шума.**
* **Всё реализовано на TypeScript.**

**2.2. Основной функционал**

**2.2.1. Главная страница**

* **Левый сайдбар:**
  + **Кнопка-группа сортировки: “Дата” (по умолчанию) и “Папки”.**
  + **Список файлов:**
    - **При сортировке по дате - группировка по датам (на каждую дату - список файлов, заголовок - дата, пример: 18.03.2024, под ней файлы за этот день).**
    - **При сортировке по папкам - папки и файлы внутри (пример: “Общая”, “Финансы” и т.д.).**
* **Основная часть: просмотр выбранного PDF-файла (например, с помощью react-pdf).**

**2.2.2. Топбар**

* **Кнопка “Добавить папку” (справа вверху). При нажатии - модалка/форма для ввода имени новой папки.**

**2.2.3. Загрузка PDF**

* **Бэкенд: REST API для загрузки PDF в выбранную папку, сохранения файла на сервере и указанием имени, папки и даты загрузки в PostgreSQL.**

**3. Backend (NestJS + PostgreSQL)**

* **Модели/таблицы:**
  + **Папки: id, название, дата создания**
  + **Файлы: id, название, путь, дата загрузки, id\_папки**
* **REST API:**
  + **Загрузка файла (с указанием папки)**
* **Заранее прописать миграции с mock’овыми данными**
* **Файлы хранятся в файловой системе, метаданные - в PostgreSQL.**

**4. Frontend (React + TypeScript)**

* **Сайдбар:**
  + **Кнопка-группа сортировки: “Дата”, “Папки”**
  + **При смене режима - соответствующий вывод файлов (группировка по датам или по папкам)**
  + **Отображение вложенных файлов внутри выбранной папки (если выбран режим “Папки”)**
* **Топбар:**
  + **Одна кнопка “Добавить папку” (открывает форму/модалку)**
* **Центральная часть:**
  + **PDF viewer**

**5. Требования к коду**

* **Чистый TypeScript, логичная структура компонентов/модулей**
* **Валидация данных (папки/файла)**
* **Минималистичный дизайн (можно использовать shadcn/ui, tailwind)**
* **Обработка ошибок (например, если не выбрана папка, не загружен файл и т.д.)**
* **Readme с инструкцией по запуску**

**6. Запуск и деплой**

* **Docker (docker-compose для всего стека)**

**7. Ожидаемый результат**

* **Репозиторий с исходниками**
* **Краткая инструкция по запуску**
* **Заранее прогруженные mock’овые данные**
* **Проект готов к 08.08.2025**