Тестовое задание для позиции Junior Front-End Developer

Общие требования:

1. **Технологии:**
   - Использовать **React** (версии 18 и выше) для реализации интерфейса.
   - Для управления состоянием использовать **React-Redux** (https://react-redux.js.org/).
   - Для работы с формами использовать **React Hook Form** (https://www.react-hook-form.com/).
   - Использовать **React Router v6** для реализации маршрутизации и вложенных роутов.
   - Таблицы с данными и пагинацию реализовать, используя данные из публичного API **SWAPI** (https://swapi.dev/).
   - Проект должен быть написан на **TypeScript**.
2. **Требования к вёрстке:**
   - Придерживаться современных стандартов HTML5 и CSS3.
   - Адаптивность и отзывчивость интерфейса обязательны.
   - Для стилей можно использовать любые решения (CSS Modules, Styled Components или другие).
   - Допускается использование библиотек https://blueprintjs.com/ или https://getbootstrap.com/ на выбор.

Функциональные требования:

1. **Авторизация:**
   - Реализовать простую авторизацию через форму логина с использованием **React-Redux** для управления состоянием.
   - При успешной авторизации пользователь должен быть перенаправлен на главную страницу приложения (в качестве главной выберите любую страницу со списком сущностей).
   - Без авторизации доступ к остальным страницам должен быть заблокирован (доступна только страница логина), пользователь перенаправляется на логин.

- Авторизованный пользователь со страницы логина принудительно должен перенаправляться на главную.
- Данные авторизации могут быть фиктивными (например, логин: admin, пароль: password).

2. **Навигация:**
   - Приложение должно содержать не менее **трёх страниц** со списком сущностей.
   - Реализовать основную навигацию по страницам приложения (с использованием **React Router**).
   - Каждая страница должна выводить таблицу данных с использованием данных из **SWAPI** API.
3. **Страницы с таблицами данных:**
   - Таблицы должны выводить списки одной из сущностей API **SWAPI** (например: персонажи, планеты, корабли).
   - На каждой странице должна быть реализована **пагинация** (используя параметры запроса к API).
   - При клике на элемент таблицы должен происходить переход на отдельную страницу с деталями этой сущности (вложенные маршруты).
   - Набор полей для таблицы произвольный.
4. **Страница отдельной сущности:**
   - Реализовать отдельную страницу для отображения детальной информации о выбранной сущности (персонаж, планета или корабль).
   - На этой странице должна быть возможность редактирования данных сущности через форму (с использованием **React Hook Form**).
   - Изменения данных должны сохраняться только в локальном состоянии (редактирование может быть фиктивным, данные в API не изменяются).

Детали реализации:

1. **React-Redux:**
   - Использовать Redux для хранения данных авторизации и состояния приложения (например, текущий пользователь).
   - Настроить глобальный стор для управления доступом к страницам.
2. **React Hook Form:**
   - Все формы (форма логина и формы для редактирования сущности) должны быть реализованы с использованием **React Hook Form**.

- Валидация форм должна быть базовой (например, обязательные поля) с использованием библиотеки YUP (https://www.npmjs.com/package/yup).

3. **Пагинация:**
   - Реализовать пагинацию на страницах с таблицами, используя параметры запроса API для управления количеством выводимых элементов и страниц.

Дополнительные требования:

- Код должен быть чистым, с хорошей структурой и разделением логики.
- Использование современных хуков React (например, useState, useEffect, useSelector, useDispatch).
- Приветствуется использование **ESLint** с базовым набором правил.

Ожидаемый результат:

- Ссылка на репозиторий с кодом на GitHub.
- Инструкции по запуску приложения (установка зависимостей, запуск).

API:

Для работы с данными можно использовать следующие ресурсы API **SWAPI** (https://swapi.dev/documentation):

- **Персонажи**: https://swapi.dev/api/people/
- **Планеты**: https://swapi.dev/api/planets/
- **Космические корабли**: https://swapi.dev/api/starships/

Обратите внимание: необходимо использовать встроенные механизмы пагинации **SWAPI** для загрузки страниц с данными (параметры ?page=1, ?page=2, и т.д.).

*После выполнения задания отправьте, пожалуйста, письмо на **laba@solvatech.kz** с выполненным ТЗ (ссылка на GitHub), а также вашим резюме. Мы оценим вашу работу **в течение двух рабочих дней** и вернемся с обратной связью **в случае успешного выполнения задания.**