Проект. Перенос состояния в Redux. Изменение порядка ингредиентов с React DND

Для финальной сдачи работы придётся написать много кода и переосмыслить проект. Финальный этап состоит из двух частей: создание Redux-хранилища приложения и работа с пользовательским интерфейсом с помощью `react-dnd`.

Другими словами, вся бизнес-логика приложения, которая хранится в контексте или компонентах, попадёт в Redux. А пользователь должен получить возможность перетаскивать ингредиенты в конструктор.

1. Обновление инфраструктуры приложения

Установите в проект все необходимые пакеты, которые отвечают за работу Redux-хранилища приложения и dnd.

Вот список всех пакетов, которые вам понадобятся:

```
react-redux redux-thunk redux @reduxjs/toolkit react-dnd react-dnd-html5-backend
```

Измените структуру приложения под использование Redux. Мы рекомендуем добавить директорию `services/`, в которой будут лежать экшены и редьюсеры, разбитые по директориям.

2. Подготовка хранилища

Подготовьте начальное состояние в хранилище. На этом этапе мы рекомендуем собрать все редьюсеры в одном файле — сейчас все созданные компоненты связаны друг с другом.

В хранилище должны быть:

- список всех полученных ингредиентов,
- список всех ингредиентов в текущем конструкторе бургера,
- объект текущего просматриваемого ингредиента,
- объект созданного заказа.

Создайте хранилище и укажите начальное значение для каждого элемента хранилища. В отдельном файле создайте `rootReducer`, а позже — подключите хранилище к вашему приложению.

Важно: обязательно подключите Redux Devtools к проекту.

3. Создание первых экшенов и редьюсеров

Когда хранилище создано и инициализировано — время наполнить его бизнес-логикой приложения. Опишите экшены и редьюсеры для следующей функциональности:

- Получение списка ингредиентов от API. Используется в компоненте `BurgerIngredients`.
- Получение списка ингредиентов для конструктора бургера. Используется в компоненте `BurgerConstructor`.
- Добавление данных о просматриваемом в модальном окне `IngredientDetails` ингредиенте.
- Удаление данных о просматриваемом в модальном окне ингредиенте при закрытии модального окна.
- Получение и обновление номера заказа в модальном окне `OrderDetails`.

Все действия, которые сопровождаются или зависят от API, должны проходить через усилители. Не забудьте подключить усилитель к вашему хранилищу.

4. Доработка интерфейса навигации по ингредиентам

В компоненте `BurgerIngredients` есть три переключателя: «Булки», «Соусы» и «Начинки». Используйте эти переключатели как панель навигации. По мере пользовательского скролла ингредиентов в компоненте

`BurgerIngredients` выделяйте активным тот переключатель, заголовок которого в самом контейнере ближе всего к верхней левой границе контейнера компонента `BurgerIngredients`.

Другими словами, заголовок не обязательно должен быть в поле зрения пользователя, но при этом он находится ближе всего к html-элементу с ингредиентами. Только в этом случае переключатель становится активным. Нажатие на переключатель реализовывать не обязательно.

5. Реализация перетаскивания ингредиентов

С помощью библиотеки `react-dnd` реализуйте такую функциональность:

1. Пользователь может добавить ингредиент из `BurgerIngredients` в компонент `BurgerConstructor`.
2. При успешном перетаскивании у ингредиента в `BurgerConstructor` увеличивается счётчик. Перетаскивать ингредиент (не являющийся булкой) можно многократно.
3. Пользователь может нажать на иконку удаления ингредиента в компоненте `BurgerConstructor`. Ингредиент удалится из `BurgerConstructor`, а счётчик количества ингредиентов в компоненте `BurgerIngredients` уменьшится на один.
4. Если в `BurgerConstructor` добавлено несколько одинаковых ингредиентов — удаление одного ингредиента не влияет на остальные ингредиенты в `BurgerConstructor` с тем же `_id`.
5. При успешном перетаскивании ингредиента с типом `bun` из `BurgerIngredients` в `BurgerConstructor` текущий элемент с типом `bun` должен удалиться и замениться на перетаскиваемый. Все эти действия также должны отразиться на счётчике количества ингредиентов в `BurgerIngredients`. В итоговой стоимости заказа нужно учитывать цены двух булок: одна снизу бургера, другая — сверху.

Эта функциональность связана и с обновлением хранилища. Добавьте необходимые экшены и редьюсеры на удаление и добавление ингредиентов из каждого состояния:

- списка полученных от API ингредиентов,
- списка ингредиентов конструктора бургера.

6. Вложенная сортировка ингредиентов в `BurgerConstructor`

Модифицируйте компонент `BurgerConstructor` и добавьте каждому ингредиенту, за исключением ингредиентов с типом `bun`, функциональность перетаскивания внутри контейнера `BurgerConstructor`:

- пользователь может изменить порядок ингредиентов в `BurgerConstructor` перетаскиванием конкретного элемента;
- при попытке «бросить» ингредиент за пределы `BurgerConstructor` ничего происходить не должно — ингредиент возвращается в исходное положение.

Изменение порядка ингредиентов должно влиять на бизнес-логику приложения. Поэтому доработайте экшены и редьюсеры, добавив возможность обновления порядка элементов в списке ингредиентов конструктора бургера.

Визуально можете сопроводить процесс сортировки на свой вкус: попробовать сортировать элементы прямо в процессе перетаскивания или добавить лишь небольшие визуальные изменения всего контейнера, когда пользователь может «бросить» ингредиент и изменить порядок.