Проект. Перенос состояния в Context

Мы добавили отдельную страницу в Figma со схемами и описанием шагов, которые вам предстоит сделать на этом этапе. Эта часть доработки проекта не очень объёмная и нацелена на практику в работе с хранилищем данных.

1. Использование Context в конструкторе бургеров

Доработайте компонент `BurgerConstructor` и данные в нём. В дальнейшем этот компонент станет более интерактивным — в него можно будет добавлять новые ингредиенты и удалять уже существующие. Поэтому важно заранее убрать весь хардкод из компонента. Так мы сможем отредактировать логику компонента и отрисовать данные, которые получены извне.

Вы можете использовать те данные, которые уже получаете из API для компонента `BurgerIngredients`. Сохраняйте данные в Context и подпишите на него компонент `BurgerConstructor`. Данные из контекста должны быть доступны при нажатии на кнопку «Оформить заказ» и в блоке с итоговой стоимостью.

2. Набор ограничений внутри конструктора бургера

Измените условия отрисовки ингредиентов внутри `BurgerConstructor` по таким правилам:

1. В отрендеренных данных может быть только один ингредиент с типом `bun`. Это булки, возможность перетаскивать которые в дальнейшем будет заблокирована.
2. Рендерить данные с типом `bun` потребуется дважды. Ведь булка должна быть сверху и снизу бургера. Добавляйте к названию булки соответствующее упоминание: «верх» и «низ».
3. Учитывайте, что для расчётов итоговой стоимости потребуется цена каждой из этих булок. Подумайте, где в дальнейшем расположите вычисление итоговой стоимости.

Результатом первых двух шагов будет изменение способа хранения и отрисовки данных компонента `BurgerConstructor`. Это поможет в следующих шагах, а также на других этапах работы с проектом.

3. Подсчёт итоговой стоимости бургера

Выводите стоимость бургера динамически в зависимости от тех ингредиентов, которые находятся в конструкторе. Мы рекомендуем попрактиковаться с использованием хука `useReducer`, но это необязательное требование, а лишь временное решение.

Проверьте корректность расчётов: генерируйте разные ингредиенты в компоненте `BurgerConstructor`.

4. Возможность создавать заказ

Ранее вы использовали захардкоженный номер заказа в модальном окне `OrderDetails`. Как и в случае с конструктором бургеров — захардкоженных данных больше не будет — это логика работы с бэкендом.

При нажатии на кнопку «Оформить заказ» отправляйте запрос к API на следующий эндпоинт:

```
// Эндпоинт
// POST <https://norma.nomoreparties.space/api/orders>

// Тело запроса
{
  "ingredients": ["609646e4dc916e00276b286e","609646e4dc916e00276b2870"]
}
```

В теле запроса нужно передать `_id` всех ингредиентов, которые находятся в компоненте `BurgerConstructor`. Пример ответа:

```
{
  "name": "Краторный метеоритный бургер",
  "order": {
      "number": 6257
  },
  "success": true
}
```

По возможности старайтесь обрабатывать ошибки при работе с API.

Если запрос прошёл успешно, сохраняйте номер заказа и отображайте его в `OrderDetails`.

5. Проверка типизации

Если модифицировали `OrderDetails` и `BurgerConstructor`, насыщая их новыми пропсами, — не забудьте проверить PropTypes для этих компонентов.