В этом уроке мы закончим с вами страницу basket, добавив к ней недостающие екшены.

Давайте добавим екшен удаления товара из корзины. Для этого добавим синхронный екшен removePhoneFromBasket

<td>

<span

className='delete-cart'

onClick={() => removePhoneFromBasket(phone.id)}

/>

</td>

Импортируем этот екшен и добавим в mapStateToProps.

import {

removePhoneFromBasket

} from 'actions'

const mapDispatchToProps = {

removePhoneFromBasket

}

export default connect(mapStateToProps, mapDispatchToProps)(Basket)

Теперь давайте добавим actionType

export const REMOVE\_PHONE\_FROM\_BASKET = 'REMOVE\_PHONE\_FROM\_BASKET'

И добавим екшен, не забыв импортировать actionType.

export const removePhoneFromBasket = id => async dispatch => {

dispatch({

type: REMOVE\_PHONE\_FROM\_BASKET,

payload: id

})

}

Теперь осталось добавить actionType в наш basket reducer.

import R from 'ramda'

import {

ADD\_PHONE\_TO\_BASKET,

REMOVE\_PHONE\_FROM\_BASKET

} from 'actionTypes'

const initialState = []

export default (state = initialState, {type, payload}) => {

switch (type) {

case ADD\_PHONE\_TO\_BASKET:

return R.append(payload, state)

case REMOVE\_PHONE\_FROM\_BASKET:

return R.without(R.of(payload), state)

default:

return state

}

}

Мы используем метод without, который возвратит нам массив без елементов, которые мы укажем. Но эти елементы нужно указывать в виде массива, поэтому я написал R.of, чтобы обернуть payload в массив.

Если мы посмотрим в браузер и кликнем удалить в корзине, то елемент у нас удалится, так как мы удалили все вхождения этого id в массив.

Теперь давайте с вами добавим sidebar

const renderSidebar = () => (

<div>

<Link

className='btn btn-info'

to='/'

>

<span className='glyphicon glyphicon-info-sign'/>

<span>Continue shopping!</span>

</Link>

{

R.not(isBasketEmpty) &&

<div>

<button

onClick={cleanBasket}

className='btn btn-danger'

>

<span className='glyphicon glyphicon-trash' />

Clear cart

</button>

<button

className='btn btn-success'

onClick={() => basketCheckout(phones)}

>

<span className='glyphicon glyphicon-envelope' />

Checkout

</button>

</div>

}

</div>

)

И импортируем все екшены

import {

removePhoneFromBasket,

basketCheckout,

cleanBasket

} from 'actions'

const Basket = ({

phones,

totalPrice,

basketCheckout,

removePhoneFromBasket,

cleanBasket

}) => {

...

}

И добавим их в mapDispatchToProps

const mapDispatchToProps = {

basketCheckout,

removePhoneFromBasket,

cleanBasket

}

Теперь у нас есть 2 интересных кнопки, на которые мы повесили екшены и их нужно создать. Это очистить корзину и перейти к оплате. Давайте добавим actionType CLEAN\_BASKET.

export const CLEAN\_BASKET = 'CLEAN\_BASKET'

И опишем эти 2 екшена. cleanBasket будет просто диспатчить евент. А basketCheckout будет алертом выводить в строку все данные товаров какие мы получили. Так как мы не будет в этой серии с вами реализовывать checkout, то будем считать, что при алерте мы передали данные в checkout.

export const cleanBasket = () => dispatch => {

dispatch({

type: CLEAN\_BASKET

})

}

export const basketCheckout = phones => () => {

alert(JSON.stringify(phones))

}

Теперь нам осталось добавить CLEAN\_BASKET в редьюсер basket. Так как мы просто хотим удалить все, то можем вернуть пустой массив.

case CLEAN\_BASKET:

return []

Если мы посмотрим в браузер, то у нас в пустой корзине есть кнопка Continue shopping, которая ведет на главную, а когда добавлены товары есть возможность очистить корзину либо нажать checkout.

Если у вас что-то не получается, либо возникли вопросы и комментарии, пишите их прямо под этим видео и я обязательно на них отвечу.