OneBitFood V2 – Aula 3

{} onebitcode.com/onebitfood-v2-aula-3

[Aviso importante: Caso você copie e cole os códigos, alguns caracteres invisíveis podem vir junto e dar alguns Bugs, então remova os espaços entre as linhas (e os crie novamente com a barra)]

Bons códigos

0 – Links importantes

- Guia para instalar as dependências: https://onebitcode.com/guia-de-instalacao-do-ruby-on-rails
- API completa no GitHub: https://github.com/OneBitCodeBlog/onebitfoodV2-api
- Cliente Web no GitHub: https://github.com/OneBitCodeBlog/onebitfoodV2-nextjs

1 – Formulário de Endereço

Nesta parte vamos criar a base do Formulário de endereço que é onde o usuário vai selecionar o endereço de busca de restaurantes e de entrega dos produtos.

1 – Vamos atualizar a API para que ela devolva as cidades disponíveis no padrão adequado, na API em views/available_cities/index.json.jbuilder coloque:

Default

```
1  json.array! @available_cities
```

2 – Vamos atualizar o _app para que o RecoilRoot também envolva o Header (que vai usar estados dele):

Default

```
1   import Head from 'next/head';
2   import '../styles/globals.scss'
3   import Header from '../components/Header';
4   import Container from 'react-bootstrap/Container';
5   import { RecoilRoot } from 'recoil';
6
7   function MyApp({ Component, pageProps }) {
8    return (
9      <>
10       <Head>
11         <title>OneBitFood V2</title>
12         <link ref="icon" href="/favicon.icon" />
13       </Head>
14
15       <main>
16         <RecoilRoot>
17           <Header />
18           <Container className='mt-6'>
19               <Component {...pageProps} />
20           </Container>
21         </RecoilRoot>
22       </main>
23     </>
24    )
25  }
26
27  export default MyApp
```

3 – Crie o componente para o form de endereço:

Default

```
mkdir components/AddressModal
touch components/AddressModal/index.js
mkdir components/AddressModal/FormAddress
touch components/AddressModal/FormAddress/index.js
```

4 – Para criar o modal de endereço coloque o seguinte código em components/AddressModal/index.js`:

Default

```
1   import Modal from 'react-bootstrap/Modal';
2
3   export default function AddressModal(props) {
4
5    return (
6      <Modal
7        show={props.show}
8        size="sm"
9        aria-labelledby="contained-modal-title-vcenter"
10       centered
11       backdrop="static"
12       keyboard={false}
13     >
14       <Modal.Header>
15         <h5 className='fw-bold mt-2'>Endereço de entrega</h5>
16       </Modal.Header>
17       <Modal.Body>
18        Hello
19       </Modal.Body>
20     </Modal>
21    )
22  }
```

5 – Para incluir a chamada do modal no Header, faça as seguintes inclusões:

Default

```
import { useState } from 'react';
import { Navbar, Nav } from 'react-bootstrap';
...
import AddressModal from '../AddressModal';
import { FaCrosshairs } from 'react-icons/fa';

export default function Header() {
 const [addressModalShow, setAddressModalShow] = useState(false);

 return (
   ...
    <Navbar.Collapse id='responsive-navbar-nav' className='justify-content-end'>
      <Nav className="py-2 text-center">
        <span className="clickable_effect" onClick={() =>
setAddressModalShow(true)}>
          <FaCrosshairs className='mb-1'/> Endereço
        </span>
        <AddressModal
          show={addressModalShow}
          onHide={() => setAddressModalShow(false)}
          onShow={() => setAddressModalShow(true)}
        />
      </Nav>
      <SearchBox/>
    </Navbar.Collapse>
  </Navbar>
 )
}
```

6 – Crie um service chamado getAvailableCities.js e coloque nele:

Default

```
import useSWR from 'swr';

export default function getAvailableCities() {
 const fetcher = (...args) => fetch(...args).then((res) => res.json());

 const { data, error } = useSWR(
   `${process.env.apiUrl}/api/available_cities`,
   fetcher, { revalidateOnFocus: false }
)
  console.log(data)
 return { available_cities: data, isLoading: !error && !data, isError: error }
}
```

7 – Agora, no component FormAddress coloque:

Default

```
 import { useState, useEffect } from 'react';
 import { Row, Col, Button, Form, Spinner, Alert } from 'react-bootstrap';
 import { useRouter } from 'next/router'

 import { useRecoilState } from 'recoil';
 import addressState from '../../../store/atoms/addressAtom';

 import getAvailableCities from '../../../services/getAvailableCities';

 export default function FormAddress(props) {
  const { available_cities, isLoading, isError } = getAvailableCities();
  const [address, setAddress] = useRecoilState(addressState);
  const [cityChanged, setCityChanged] = useState(false);
```

```
14    const router = useRouter();
15
16
17    if(isError)
18      return <Alert variant='custom-red'>Erro ao carregar</Alert>
19    else if(isLoading)
20      return <Spinner animation='border'/>
21
22
23    const updateAddress = (e) => {
24      if(e.target.name == 'city')
25        setCityChanged(true);
26      setAddress({ ...address, [e.target.name]: e.target.value });
27    }
28
29    const confirmAddress = (e) => {
30      e.preventDefault();
31      props.onHide();
32      if(cityChanged)
33        router.push('/restaurants');
34    }
35
36    return (
37      <Row>
38        <Col md={12}>
39          <Form onSubmit={e => confirmAddress(e)}>
40            <Form.Group>
41              <Form.Label>Sua cidade</Form.Label>
42              <Form.Control
43                required as="select"
```

```
44              onChange={updateAddress}
45              value={address.city}
46              name="city"
47            >
48              {address.city == '' && <option key={0}>Escolher cidade</option>}
49              {available_cities.map((state, i) => {
50                return <option key={i} value={state}>{state}</option>
51              })}
52            </Form.Control>
53          </Form.Group>
54          {address.city != '' &&
55            <div>
56              <Form.Group className='mt-3'>
57                <Form.Label>Bairro</Form.Label>
58                <Form.Control
59                  required
60                  type="text"
61                  placeholder="Bairro"
62                  onChange={updateAddress}
63                  value={address.neighborhood}
64                  name="neighborhood"
65                />
66              </Form.Group>
67              <Form.Group className='mt-3'>
68                <Form.Label>Logradouro</Form.Label>
69                <Form.Control
70                  required
71                  type="text"
72                  placeholder="Rua/Avenida/Alameda"
73                  onChange={updateAddress}
```

```
74                 value={address.street}
75                 name="street"
76               />
77             </Form.Group>
78             <Form.Group className='mt-3'>
79               <Form.Label>Nûmero</Form.Label>
80               <Form.Control
81                 required
82                 type="text"
83                 placeholder="Nûmero"
84                 onChange={updateAddress}
85                 value={address.number}
86                 name="number"
87               />
88             </Form.Group>
89             <Form.Group className='mt-3'>
90               <Form.Label>Complemento</Form.Label>
91               <Form.Control
92                 type="text"
93                 placeholder="Complemento"
94                 onChange={updateAddress}
95                 value={address.complement}
96                 name="complement"
97               />
98             </Form.Group>
99             <div className="text-center pt-4">
100              <Button variant="custom-red" className='text-white' type="submit"
    size="md">
101
                Confirmar endereço
102
              </Button>
103
```

```
104               </div>
105             </div>
106           }
107         </Form>
108       </Col>
109     </Row>
110   )
    }
```

8 – Inclua ele no AddressModal inserindo:

Default

```
1   ...
2   import FormAddress from './FormAddress'
3
4   export default function AddressModal(props) {
5
6    return (
7      ...
8        <Modal.Body>
9          <FormAddress
10           onHide={() => props.onHide()}
11           onShow={() => props.onShow()}
12         />
13       </Modal.Body>
14     </Modal>
15   )
16 }
```

9 – Ainda no AddressModal agora vamos exibir o modal de endereço sempre que ele não tenha sido preenchido ainda (exceto na página inicial):

Default

```
1   import { useEffect } from 'react';
2   import Modal from 'react-bootstrap/Modal';
3   import FormAddress from './FormAddress';
4   import { useRecoilState } from 'recoil';
5   import addressState from '../../store/atoms/addressAtom';
6   import { useRouter } from 'next/router';
7
8   export default function AddressModal(props) {
9    const [address, setAddress] = useRecoilState(addressState)
10   const router = useRouter();
11
12   useEffect(() => {
13     if(router.asPath != '/' && address.city == '')
14       props.onShow();
15  }, [router])
16
17   return(
18     <Modal
19       show={props.show}
20       size='sm'
21       aria-labelledby='contained-modal-title-vcenter'
22       centered
23       backdrop='static'
24       keybord={false}
25     >
26       <Modal.Header>
27         <h5 className='fw-bold mt-2'>Endereço de entrega</h5>
28       </Modal.Header>
```

```
29        <Modal.Body>
30          <FormAddress
31            onHide={() => props.onHide()}
32          />
33        </Modal.Body>
34      </Modal>
35    )
36  }
```

10 – Para finalizar, atualize o getRestaurants para fazer o filtro por cidade:

Default

```
1   ...
2   import { useRecoilState } from 'recoil';
3   import addressState from '../store/atoms/addressAtom';
4
5   export default function getRestaurants() {
6    const [address, setAddress] = useRecoilState(addressState);
7
8    ...
9    if(address.city != '')
10     params = `${params == '' ? '?' : `${params}&`}city=${address.city}`
11   ...
12  }
```

2 – Modal para incluir no carrinho

1 – Crie o Atom cartAtom.js dentro de store/atoms e coloque nele:

Default

```
import { atom } from 'recoil';
import { recoilPersist } from 'recoil-persist'

const { persistAtom } = recoilPersist()

const cartState = atom({
 key: 'cartState',
 default: {restaurant: {}, products: []},
 effects_UNSTABLE: [persistAtom]
});

export default cartState;
```

2 – Crie o component AddProductModal rodando:

Default

```
mkdir components/AddProductModal
touch components/AddProductModal/index.js
```

3 – Inclua a estrutura principal do component no arquivo criado:

Default

```
1   import { Modal } from 'react-bootstrap';
2
3   export default function AddProductModal(props) {
4
5    return (
6      <Modal
7        show={props.show}
8        size="sm"
9        aria-labelledby="contained-modal-title-vcenter"
10       centered
11       keyboard={false}
12       onHide={() => props.onHide()}
13     >
14       <Modal.Header closeButton>
15         <h5 className='fw-bold mt-2'>Adicionar produto</h5>
16       </Modal.Header>
17       <Modal.Body>
18        Hello
19       </Modal.Body>
20     </Modal>
21    )
22  }
```

4 – Inclua ele no component CategoryProducts do DetailsRestaurant:

Default

```
1   import { useState } from 'react';
2   ...
3   import AddProductModal from '../../AddProductModal';
4
5   export default function CategoryProducts(props) {
6    const [productSelected, setProductSelected] = useState(null);
7
8    return(
9      <>
10       <AddProductModal
11         show={productSelected != null}
12         onHide={() => setProductSelected(null)}
13         product={productSelected}
14         restaurant={props.restaurant}
15       />
16       <h5 className='fw-bold'>{props.title}</h5>
17       <Row>
18        {props.products.map((product, i) =>
19          <Col md={4} sm={12} key={i}>
20            <Card className="mb-4 clickable_effect" onClick={() =>
    setProductSelected(product)}>
21
              ...
22
    </>
23
  )
24
}
```

5 – No AddModalProduct coloque a lógica:

Default

```
1   import { useState } from 'react';
2   import { Modal, Row, Col, Form, Button } from 'react-bootstrap';
```

```
3   import Image from 'next/image'
4   import toCurrency from '../../services/toCurrency';
5   import truncateString from '../../services/truncateString';
6
7   import { useRecoilState } from 'recoil';
8   import cartState from '../../store/atoms/cartAtom';
9
10
11  export default function AddProductModal(props) {
12   const [quantity, setQuantity] = useState(1);
13   const [cart, setCart] = useRecoilState(cartState);
14
15   const addProduct = (e) => {
16     e.preventDefault();
17
18     const product = {...props.product, ...{'quantity': quantity}}
19
20     if(cart.restaurant.id != props.restaurant.id)
21       setCart({ restaurant: props.restaurant, products: [product] })
22     else
23       setCart({ restaurant: props.restaurant, products: [...cart.products, product] })
24
25     setQuantity(1);
26     props.onHide();
27  }
28
29   if (!props.product) return null;
30
31   return (
32     <Modal
```

```
33      show={props.show}
34      size="sm"
35      aria-labelledby="contained-modal-title-vcenter"
36      centered
37      keyboard={false}
38      onHide={() => props.onHide()}
39    >
40      <Modal.Header>
41        <h5 className='fw-bold mt-2'>Adicionar produto</h5>
42      </Modal.Header>
43      <Modal.Body>
44        <Row>
45          <Col>
46            <Image
47              src={props.product.image_url}
48              alt={props.product.name}
49              width={300}
50              height={200}
51            />
52          </Col>
53        </Row>
54        <Row className="pb-0">
55          <Col md={8}>
56            <p className='fw-bold mb-0'>{props.product.name}</p>
57          </Col>
58          <Col>
59            <small className='border px-1 border-custom-gray fw-bold'>
60              {toCurrency(props.product.price)}
61            </small>
62          </Col>
```

```
63          </Row>
64          <Row>
65            <Col>
66              <p><small>{truncateString(props.product.description, 60)}</small></p>
67            </Col>
68          </Row>
69          <Form onSubmit={addProduct} className='d-flex'>
70            <Form.Group>
71              <Form.Control
72                required
73                type="number"
74                placeholder="quantidade"
75                min="1" step="1"
76                name="quantidade"
77                value={quantity}
78                onChange={(e) => setQuantity(e.target.value)}
79              />
80            </Form.Group>
81            <Button variant="custom-red"
82              type="submit"
83              className="text-white ms-6">
84             Adicionar
85            </Button>
86
87          </Form>
88
89        </Modal.Body>
90      </Modal>
91    )
92  }
```

6 – No component CategoryProducts na primeira coluna aumente o tamanho dela (md={4}) para 6 (md={6})

7 – Teste subindo o projeto (e observando o localstorage no application das ferramentas de depuração do browser):

Default

```
1  yarn dev
```

3 – Desafio Adicionar Produto repetidoa

No Modal de adicionar produto, na hora da adição verifique se o produto já foi previamente adicionado ao carrinho, se sim, ao invés de adiciona-lo novamente aumente a sua quantidade no pedido.

4 – Criando o Carrinho

1 – Gere os components necessários para o carrinho rodando:

Default

```
1  mkdir components/Cart
2  touch components/Cart/index.js
3  mkdir components/CartModal
4  touch components/CartModal/index.js
```

2 – No component Cart criei a estrutura do modal:

Default

```
import Modal from 'react-bootstrap/Modal';

export default function CartModal(props) {

 return (
   <Modal
     show={props.show}
     size="sm"
     aria-labelledby="contained-modal-title-vcenter"
     centered
     keyboard={false}
     onHide={() => props.onHide()}
   >
     <Modal.Header>
       <h5 className='fw-bold mt-2'>Carrinho</h5>
     </Modal.Header>
     <Modal.Body>
       Hello
     </Modal.Body>
   </Modal>
  )
}
```

3 – Adicione a chamada dele no Header:

Default

```
...
import CartModal from '../CartModal';
import { FaCrosshairs, FaShoppingBag } from 'react-icons/fa';

```

```
5  export default function Header() {
6   ...
7   const [cartModalShow, setCartModalShow] = useState(false);
8
9   return (
10    <Navbar bg='white' expand='lg' className='border-bottom border-custom-
gray'>
11      <Navbar.Brand className='mx-3'>
12        ...
13      </Navbar.Brand>
14
15      <Navbar.Toggle aria-controls='responsive-navbar-nav' />
16
17      <Navbar.Collapse id='responsive-navbar-nav' className='justify-content-end'>
18        <Nav className="me-lg-4 me-sm-0 text-center pt-2 pb-2">
19          <span className="clickable_effect" onClick={() =>
20  setCartModalShow(true)}>
21            <FaShoppingBag/> Carrinho
22          </span>
23          <CartModal
24            show={cartModalShow}
25            onHide={() => setCartModalShow(false)}
26            onShow={() => setCartModalShow(true)}
27          />
28        </Nav>
29
30        <Nav className="py-2 text-center">
31          ...
32        </Nav>
33        <SearchBox/>
34      </Navbar.Collapse>
```

```
35      </Navbar>
36     )
       }
```

4 – No component Cart inclua a estrutura do component:

Default

```
1   export default function Cart(props) {
2
3    return (
4      <>
5
6      </>
7    )
8   }
```

5 – Nesse mesmo component: a – Inclua o atom cart:

Default

```
1   ...
2
3   import { useRecoilState } from 'recoil';
4   import cartState from '../../store/atoms/cartAtom';
5
6
7   export default function Cart() {
8    const [cart, setCart] = useRecoilState(cartState);
9   ...
```

b – Inclua os components bootstrap e os services de formatação:

Default

```
import { Row, Col, Button }  from 'react-bootstrap';
import toCurrency from '../../services/toCurrency';
import truncateString from '../../services/truncateString';
```

c – Coloque o métodos para calcular o subtotal e o total do carrinho e a lógica para avisar quando o carrinho está vázio:

Default

```
export default function Cart() {
 ...

 const subTotal = () => cart.products.reduce(
  (a, b) => a + (parseFloat(b['price']) * parseFloat(b['quantity']) || 0), 0
 );

 const total = () => cart.restaurant.delivery_tax + subTotal();

 if (cart.products.length <= 0)
   return <p>Carrinho vazio</p>;
```

d – Inclua o método para remover produtos do carrinho:

Default

```
export default function Cart() {
 ...

 const removeProduct = (product) => {
  const new_products = cart.products.filter((p) => p.id != product.id);
  setCart({ restaurant: { ...cart.restaurant }, products: new_products });
}
```

e – Agora inclua a parte visual:

Default

```
...
return (
 <>
  <h5 className='fw-bolder'>{cart.restaurant.name}</h5>
  <hr />
 {cart.products.map((product, i) =>
   <div key={product.id}  className="mb-4" key={i}>
    <Row>
     <Col md={8} xs={8}>
      <small className='fw-bolder'>{product.quantity}x {product.name}</small>
     </Col>
     <Col md={4} xs={4} className="text-right">
      <small >
       {toCurrency(product.price)}
      </small>
     </Col>
    </Row>
    <Row className="mt-2">
     <Col md={8} xs={8}>
      <p><small>{truncateString(product.description, 40)}</small></p>
     </Col>
     <Col md={4} xs={4} className="text-right">
      <Button
       size="sm"
       variant="outline-dark"
       onClick={() => removeProduct(product)}
       className='border px-1 border-custom-gray'
      >
```

```
29              Remover
30             </Button>
31           </Col>
32         </Row>
33       </div>
34     )}
35     <hr />
36     <Row className="mt-4">
37       <Col md={8} xs={8}>
38         <p>Subototal</p>
39       </Col>
40       <Col md={4} xs={4} className="text-right">
41         <p>{toCurrency(subTotal())}</p>
42       </Col>
43     </Row>
44     <Row className="mt-n2">
45       <Col md={8} xs={8}>
46         <p>Taxa de entrega</p>
47       </Col>
48       <Col md={4} xs={4} className="text-right">
49         <p>{toCurrency(cart.restaurant.delivery_tax)}</p>
50       </Col>
51       <hr />
52     </Row>
53     <Row className="mb-4">
54       <Col md={8} xs={8}>
55         <p className='fw-bolder'>Total</p>
56       </Col>
57       <Col md={4} xs={4} className="text-right">
58         <p className='fw-bolder'>{toCurrency(total())}</p>
```

```
      </Col>
    </Row>
  </>
)
...
```

6 – No component CartModal: a – Inclua o Cart:

Default

```
...
import Cart from '../Cart';

...
<Modal.Body>
  <Cart show={props.show} />
</Modal.Body>
...
```

b – Inclua o atom cart também:

Default

```
...
import { useRecoilState } from 'recoil';
import cartState from '../../store/atoms/cartAtom';

export default function CartModal(props) {
  const [cart] = useRecoilState(cartState);
```

c – Para podermos ir para a página de finalização inclua:

Default

```
...
import Link from 'next/link';
import Button from 'react-bootstrap/Button';
...

<Modal.Body>
  <Cart show={props.show} />
  {cart.products.length > 0 &&
    <div className="text-center pt-2">
      <Link href='/orders/new'>
        <Button variant="custom-red text-white">Finalizar pedido</Button>
      </Link>
    </div>
  }
</Modal.Body>
...
```

7 – Suba o projeto e veja como ficou:

Default

```
yarn dev
```

Para referência: 1 – CartModal completo:

Default

```
import Modal from 'react-bootstrap/Modal';
import Cart from '../Cart';
import { useRecoilState } from 'recoil';
import cartState from '../../store/atoms/cartAtom';
import Link from 'next/link';
import Button from 'react-bootstrap/Button';
```

```
7
8   export default function CartModal(props) {
9     const [cart] = useRecoilState(cartState);
10
11    return (
12      <Modal
13        show={props.show}
14        size="sm"
15        aria-labelledby="contained-modal-title-vcenter"
16        centered
17        keyboard={false}
18        onHide={() => props.onHide()}
19      >
20        <Modal.Header>
21          <h5 className='fw-bold mt-2'>Carrinho</h5>
22        </Modal.Header>
23        <Modal.Body>
24          <Cart show={props.show} />
25          {cart.products.length > 0 &&
26            <div className="text-center pt-2">
27              <Link href='/orders/new'>
28                <Button variant="custom-red text-white" onClick={props.onHide}>
29                  Finalizar pedido
30                </Button>
31              </Link>
32            </div>
33          }
34        </Modal.Body>
35      </Modal>
36    )
```

```
37  }
```

2 – Cart completo:

Default

```
1   import React from 'react';
2   import { useRecoilState } from 'recoil';
3   import cartState from '../../store/atoms/cartAtom';
4   import { Row, Col, Button }  from 'react-bootstrap';
5   import toCurrency from '../../services/toCurrency';
6   import truncateString from '../../services/truncateString';
7
8
9   export default function Cart() {
10    const [cart, setCart] = useRecoilState(cartState);
11
12    const removeProduct = (product) => {
13      const new_products = cart.products.filter((p) => p.id != product.id);
14      setCart({ restaurant: { ...cart.restaurant }, products: new_products });
15    }
16
17    const subTotal = () => cart.products.reduce(
18      (a, b) => a + (parseFloat(b['price']) * parseFloat(b['quantity']) || 0), 0
19    );
20
21    const total = () => cart.restaurant.delivery_tax + subTotal();
22
23    if (cart.products.length <= 0)
24      return <p>Carrinho vazio</p>;
25
26    return (
```

```
27      <>
28        <h5 className='fw-bolder'>{cart.restaurant.name}</h5>
29        <hr />
30        {cart.products.map((product, i) =>
31          <div key={product.id}  className="mb-4" key={i}>
32            <Row>
33              <Col md={8} xs={8}>
34                <small className='fw-bolder'>{product.quantity}x {product.name}
</small>
35              </Col>
36              <Col md={4} xs={4} className="text-right">
37                <small >
38                  {toCurrency(product.price)}
39                </small>
40              </Col>
41            </Row>
42            <Row className="mt-2">
43              <Col md={8} xs={8}>
44                <p><small>{truncateString(product.description, 40)}</small></p>
45              </Col>
46              <Col md={4} xs={4} className="text-right">
47                <Button
48                  size="sm"
49                  variant="outline-dark"
50                  onClick={() => removeProduct(product)}
51                  className='border px-1 border-custom-gray'
52                >
53                  Remover
54                </Button>
55              </Col>
56
```

```
57            </Row>
58          </div>
59        )}
60        <hr />
61        <Row className="mt-4">
62          <Col md={8} xs={8}>
63            <p>Subototal</p>
64          </Col>
65          <Col md={4} xs={4} className="text-right">
66            <p>{toCurrency(subTotal())}</p>
67          </Col>
68        </Row>
69        <Row className="mt-n2">
70          <Col md={8} xs={8}>
71            <p>Taxa de entrega</p>
72          </Col>
73          <Col md={4} xs={4} className="text-right">
74            <p>{toCurrency(cart.restaurant.delivery_tax)}</p>
75          </Col>
76          <hr />
77        </Row>
78        <Row className="mb-4">
79          <Col md={8} xs={8}>
80            <p className='fw-bolder'>Total</p>
81          </Col>
82          <Col md={4} xs={4} className="text-right">
83            <p className='fw-bolder'>{toCurrency(total())}</p>
84          </Col>
85        </Row>
86      </>
```

```
87      )
      }
```

3 – Header completo:

Default

```
1   import { useState } from 'react';
2   import { Navbar, Nav } from 'react-bootstrap';
3   import Image from 'next/image';
4   import Link from 'next/link';
5   import SearchBox from '../SearchBox';
6   import AddressModal from '../AddressModal';
7   import CartModal from '../CartModal';
8   import { FaCrosshairs, FaShoppingBag } from 'react-icons/fa';
9
10  export default function Header() {
11    const [addressModalShow, setAddressModalShow] = useState(false);
12    const [cartModalShow, setCartModalShow] = useState(false);
13
14    return (
15      <Navbar bg='white' expand='lg' className='border-bottom border-custom-
    gray'>
16
        <Navbar.Brand className='mx-3'>
17
          <Link href='/restaurants'>
18
            <a>
19
              <Image
20
                src='/logo.png'
21
                alt="OneBitFood"
22
                width={200}
23
                height={44}
24
                className='clickable_effect'
25
```

```
26              />
27            </a>
28          </Link>
29        </Navbar.Brand>
30
31        <Navbar.Toggle aria-controls='responsive-navbar-nav' />
32
33        <Navbar.Collapse id='responsive-navbar-nav' className='justify-content-
end'>
34          <Nav className="me-lg-4 me-sm-0 text-center pt-2 pb-2">
35            <span className="clickable_effect" onClick={() =>
36  setCartModalShow(true)}>
37              <FaShoppingBag/> Carrinho
38            </span>
39            <CartModal
40              show={cartModalShow}
41              onHide={() => setCartModalShow(false)}
42              onShow={() => setCartModalShow(true)}
43            />
44          </Nav>
45
46          <Nav className="py-2 text-center">
47            <span className="clickable_effect" onClick={() =>
48  setAddressModalShow(true)}>
49              <FaCrosshairs className='mb-1'/> Endereço
50            </span>
51            <AddressModal
52              show={addressModalShow}
53              onHide={() => setAddressModalShow(false)}
54              onShow={() => setAddressModalShow(true)}
55            />
```

```
56          </Nav>
57          <SearchBox/>
58        </Navbar.Collapse>
        </Navbar>
      )
    }
```

5 – Finalização do pedido

1 – Crie a page Orders:

Default

```
1  mkdir pages/orders
2  touch pages/orders/new.js
```

2 – Crie os components base:

Default

```
1  mkdir components/NewOrder
2  touch components/NewOrder/index.js
3  mkdir components/NewOrder/OrderForm
4  touch components/NewOrder/OrderForm/index.js
```

3 – Crie a base dos components: a – NewOrder:

Default

```
1  export default function NewOrder() {
2   return (
3     <>
4     </>
5    )
6  }
```

b – OrderForm:

Default

```
export default function OrderForm() {
 return (
   <>
   </>
)
}
```

4 – Coloque na página criada:

Default

```
import NewOrder from "../../components/NewOrder";

export default function New() {
 return <NewOrder />
}
```

5 – Vamos criar a base do component NewOrder:

Default

```
import React from 'react';
import { Row, Col, Card } from 'react-bootstrap';

export default function NewOrder() {
 return (
   <>
     <Row className='justify-content-md-center'>
       <Col md={{span: 4}}>
         <Card className='p-5 mb-4'>
           <Cart/>
         </Card>
       </Col>
       <Col md={{span: 4}}>
         <Card className='p-5 mb-4'>
           <OrderForm/>
         </Card>
       </Col>
     </Row>
   </>
)
}
```

6 – Vamos importar e adicionar o Cart nele:

Default

```
...
import Cart from '../Cart';

...
<Row>
 <Col md={{span: 4, offset: 1}}>
   <Card className='p-5 mb-4'>
     <Cart/>
   </Card>
...
```

7 – Vamos criar um service para enviar a order para o backend, crie um service chamado createOrder.js e coloque nele:

Default

```
export default async function createOrder(order) {
 const response = await fetch(`${process.env.apiUrl}/api/orders`, {
   method: 'POST',
   headers: {
     'Accept': 'application/json',
     'Content-Type': 'application/json'
   },
   body: JSON.stringify({order: order}),
  });
 return response;
}
```

8 – No component OrderForm: a – Importe o service criado:

Default

```
import createOrder from '../../../services/createOrder';
```

b – Inclua também os atoms cart e address:

Default

```
import { useRecoilState, useResetRecoilState } from 'recoil';
import addressState from '../../../store/atoms/addressAtom';
import cartState from '../../../store/atoms/cartAtom';

...

export default function OrderForm() {
  const [address] = useRecoilState(addressState);
  const [cart] = useRecoilState(cartState);
  const resetCart = useResetRecoilState(cartState);
  ...
```

c – Crie um state para mostrar uma mensagem de erro quando a chamada falhar:

Default

```
import { useState } from 'react';
...

export default function OrderForm() {
  const [error, setError] = useState(null)
  ...
```

d – Importe também o router:

Default

```
import { useRouter } from 'next/router';
...

export default function OrderForm() {
...
const router = useRouter();
...
```

e – Importe os components do Bootstrap que vamos usar:

Default

```
import { Form, Alert, Button } from 'react-bootstrap';
```

f – Crie um state order:

Default

```
  export default function OrderForm() {
    ...
    const [order, setOrder] = useState({
      name: "",
      phone_number: "",
      ...address,
      order_products_attributes: cart.products.map(p => (
        {'product_id': p.id, 'quantity': p.quantity}
      )),
      restaurant_id: cart.restaurant.id
    })
```

g – Inclua um método para atualizar o estado order:

Default

```
const updateOrderState = (e) => {
  setOrder({ ...order, [e.target.name]: e.target.value });
}
```

h – Agora crie um método para chamar nosso serviço de criação de order:

Default

```
  const submitOrder = async (e) => {
    e.preventDefault();
    try {
      await createOrder(order);
      router.push('/orders/success');
      resetCart();
    } catch(err) {
      setError(true);
    }
  }
```

i – Por fim inclua a parte visual do component:

Default

```
    return (
      <Form onSubmit={e => submitOrder(e)}>
        <h4 className='fw-bold mb-5'>Detalhes finais</h4>
        <Form.Group>
          <Form.Label>Nome completo</Form.Label>
          <Form.Control
            required
            type="text"
            placeholder="Dennis Ritchie..."
            onChange={updateOrderState}
```

```
          value={order.name}
          name="name"
        />
      </Form.Group>
      <Form.Group className='mt-3'>
        <Form.Label>Número de telefone</Form.Label>
        <Form.Control
          required
          type="text"
          placeholder="(00) 00000-0000"
          onChange={updateOrderState}
          value={order.phone_number}
          name="phone_number"
        />
      </Form.Group>

      <div className="mt-5">
        <p className='fw-bolder'>Entregar em:</p>
        <p><small>{address.street} {address.number} {address.neighborhood}, {address.city}</small></p>
      </div>
    {cart.products.length > 0 &&
        <div className="text-center">
          <Button variant="custom-red" type="submit" size="lg" className="mt-4 text-white">
            Finalizar Pedido
          </Button>
        </div>
    }
```

```
41      {error && <Alert variant='custom-red' className="mt-4"> Erro no pedido!
      </Alert> }

        </Form>

        )
```

8 – Atualize o component OrderNew para incluir o component criado anteriormente:

Default

```
1     import React from 'react';
2     import { Row, Col, Card } from 'react-bootstrap';
3     import OrderForm from './OrderForm';
4     import Cart from '../Cart';
5
6     export default function NewOrder() {
7      return (
8        <>
9          <Row>
10           <Col md={{span: 4, offset: 1}}>
11             ...
12           </Col>
13           <Col md={{span: 4, offset: 2}}>
14             <Card className='p-5 mb-4'>
15               <OrderForm/>
16             </Card>
17           </Col>
18         </Row>
19       </>
20    )
21    }
```

9 – Teste finalizar um pedido:

Default

```
1  yarn dev
```

6 – Página de sucesso

1 – Dentro de pages/orders crie a página success.js e coloque nela:

Default

```
import { Row, Col, Card } from 'react-bootstrap';
import Image from 'next/image'

export default function Success() {
 return (
   <Row className="mt-4 justify-content-md-center">
     <Col md={{ span: 4 }}>
       <Card className="mt-4 px-4 py-4">
         <h4 className='fw-bold'>Pedido a caminho</h4>
         <p className="mt-2">Em breve você receberá sua comida em casa!</p>

         <Row className="my-4 justify-content-md-center">
           <Col md={{ span: 10 }}>
             <Image
               src='/status-ok.png'
               alt='Sucesso no pedido'
               width={100}
               height={100}
               layout="responsive"
             />
           </Col>
         </Row>
       </Card>
     </Col>
   </Row>
)
}
```

2 – Suba o projeto e teste todo o processo de escolha, seleção do produto e realização do pedido:

Default

```
1 yarn dev
```

7 – Desafio página 404

Crie uma página 404 bonita e faça com que o next exiba ela sempre que ele não encontrar uma URL pedida (inclusive quando você solicitar um restaurante que não existe).

8 – SSR, SSG e Production

Renderização do lado do cliente (CSR)

Prós

- Rápido no servidor – Como você está apenas renderizando uma página em branco, é muito rápido renderizar.
- Pode ser estática – a página em branco pode ser gerada estaticamente e servida a partir de algo como S3, o que a torna ainda mais rápida.
- Oferece suporte a aplicativos de página única – a renderização do lado do cliente é o único modelo que oferece suporte a aplicativos de página única ou SPAs.

Contras

Sem renderização inicial – você está enviando uma página em branco para o cliente. Portanto, se seu aplicativo for grande ou o cliente estiver em uma conexão lenta, isso não será o ideal.

Renderização do lado do servidor (SSR)

Prós

- Conteúdo imediatamente disponível – como você está enviando HTML renderizado para o cliente, o cliente começará a ver o conteúdo quase imediatamente.
- Nenhuma busca adicional do cliente – Idealmente, o processo de renderização do servidor fará todas as chamadas necessárias para obter os dados, portanto, você não fará nenhuma chamada de serviço adicional do cliente. Pelo menos até que o usuário comece a brincar um pouco com a página.
- Ótimo para SEO

Contras

- Mais lento no servidor – você está renderizando a página duas vezes, uma no servidor e outra no cliente. Você provavelmente também está fazendo chamadas de serviço do servidor para processar a página. Tudo isso leva tempo, então o envio inicial do HTML ao cliente pode ser atrasado.
- Incompatível com algumas bibliotecas de IU

Geração de site estático (SSG)

Prós

- Conteúdo imediatamente disponível – como você está enviando HTML renderizado para o cliente, o cliente começará a ver o conteúdo quase imediatamente.
- Nenhuma busca adicional do cliente – Idealmente, o processo de renderização do servidor fará todas as chamadas necessárias para obter os dados, portanto, você não fará nenhuma chamada de serviço adicional do cliente. Pelo menos até que o usuário comece a brincar um pouco com a página.
- Ótimo para SEO – motores de busca como conteúdo HTML. Se você está usando apenas a renderização do lado do cliente, então está contando com os motores de busca rodando seu Javascript, o que pode ou não acontecer.
- Rápido de servir incrível – O conteúdo estático é muito rápido de servir. Você também pode armazená-lo em cache de forma limpa.
- Sem servidor – você não precisa executar um servidor. Portanto, você não precisa monitorar esse servidor e obterá muito menos pings de dever de pager.

Contras

- Pode ser lento para reconstruir sites grandes – Se você tem dezenas de milhares de rotas, deve esperar lentidão. Embora a equipe do NextJS esteja trabalhando em reconstruções incrementais.
- Incompatível com algumas bibliotecas de IU

Usando SSR no projeto

1 – Atualize a page restaurants/[id].js colocando: a – A função getServerSideProps para baixar as informações do servidor assim que a página for chamada:

Default

```
export async function getServerSideProps(context) {
 const { id } = context.query;
 try {
  const res = await fetch(`${process.env.apiUrl}/api/restaurants/${id}`);
  const restaurant = await res.json();
  const isError = res.ok ? false : true
  return { props: { restaurant, isError: isError }, }
 } catch {
  return { props: { isError: true }, }
 }
}
```

b – Atualize a função Restaurant:

Default

```
export default function Restaurant({restaurant, isError = false}) {
 return <DetailsRestaurant restaurant={restaurant} isError={isError}/>
}
```

2 – Atualize o component DetailsRestaurant colocando:

Default

```
import Details from './Details';
import CategoryProducts from './CategoryProducts';
import { Alert } from 'react-bootstrap';

export default function DetailsRestaurant(props) {

 if(props.isError)
   return <Alert variant='custom-red'>Erro ao carregar</Alert>

 return(
   <>
    <Details {...props.restaurant} />
   {props.restaurant.product_categories.map((product_category, i) =>
      <CategoryProducts restaurant={props.restaurant} {...product_category} key={i} />
   )}
   </>
 )
}
```

3 – Suba o projeto para ver como ficou:

Default

```
yarn dev
```

Usando SSG

1 – Atualize a page restaurants/[id].js colocando: a – A função getStaticPaths para pegar todos os ids dos restaurantes:

Default

```
export async function getStaticPaths() {
 const res = await fetch(`${process.env.apiUrl}/api/restaurants`);
 const restaurants = await res.json();

 const paths = restaurants.map((restaurant) => ({
   params: { id: restaurant.id.toString() }
 }))

 return { paths, fallback: false }
}
```

b – A função getStaticProps para baixar cada um dos restaurantes disponíveis:

Default

```
export async function getStaticProps({ params }) {
 const res = await fetch(`${process.env.apiUrl}/api/restaurants/${params.id}`);
 const restaurant = await res.json();
 return {
   props: { restaurant },
   revalidate: 120
 }
}
```

c – Atualize a função Restaurante colocando:

Default

```
export default function Restaurant({restaurant}) {
 return <DetailsRestaurant restaurant={restaurant} />
}
```

2 – Atualize o component DetailsRestaurant colocando:

Default

```
import Details from './Details';
import CategoryProducts from './CategoryProducts';

export default function DetailsRestaurant(props) {

 return(
   <>
    <Details {...props.restaurant} />
   {props.restaurant.product_categories.map((product_category, i) =>
     <CategoryProducts restaurant={props.restaurant} {...product_category} key={i} />
   )}
   </>
  )
}
```

Colocando o projeto em modo Production

1 – Gere a versão para production do seu projeto rodando:

Default

```
yarn build
```

2 – Em package.json atualize a linha “start”: “next start” para:

Default

```
"start": "next start -p 3001"
```

3 – Rode seu projeto como production:

Default

1 yarn start