Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное

учреждение высшего образования

**«Владимирский государственный университет**

**имени Александра Григорьевича и Николая Григорьевича**

**Столетовых»**

**(ВлГУ)**

**Колледж инновационных технологий и предпринимательства**

Отчет по производственной практике

Выполнил:

ст. гр. ИСПсп-118

Максимова А.С.

Принял:

Шамышева О.Н.

Владимир, 2022

Содержание

[Проектировка дизайн-макеты приложения «Цветочный магазин» 4](#_Toc100604865)

[Верстать дизайн-макеты приложения «Цветочный магазин» с применением фреймворка ReactJs 10](#_Toc100604866)

[1. Главная станица 10](#_Toc100604867)

[2. Страница « Где нас найти?» 11](#_Toc100604868)

[3. Страница « О нас» 11](#_Toc100604869)

[4. Страница « Каталог» 14](#_Toc100604870)

[5. Страница « Товары» 15](#_Toc100604871)

[Разработать формы регистрации, авторизации и соответствующие процедуры работы с базой данных (бэкенд) 17](#_Toc100604872)

[1. Форма регистриции 17](#_Toc100604873)

[2. Форма авторизации 20](#_Toc100604874)

[Разработать корзину заказов приложения 21](#_Toc100604875)

[1. Страница « Корзина » 21](#_Toc100604876)

[2. Страница « Личный кабинет » 24](#_Toc100604877)

[3. Хранение данных на клиенте 26](#_Toc100604878)

[4. Добавление заказа 27](#_Toc100604879)

[Разработать административный интерфейс приложения 27](#_Toc100604880)

[1. Страница « Администратор» 27](#_Toc100604881)

[2. Форма добавление категории 32](#_Toc100604882)

[3. Форма добавление товара 34](#_Toc100604883)

[4. Добавление и вывода файла(фото) на сервере 35](#_Toc100604884)

[5. Вывод каких-либо данных с сервера 36](#_Toc100604885)

[Список литературы 37](#_Toc100604886)

[ПРИЛОЖЕНИЕ А 38](#_Toc100604887)

# Проектировка дизайн-макеты приложения «Цветочный магазин»

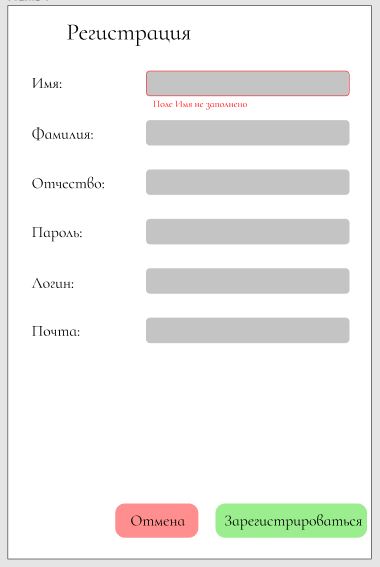


Рис.1. – Макет формы регистрации

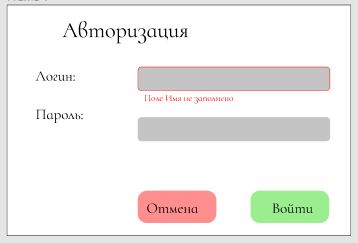


Рис.2. – Макет формы авторизации

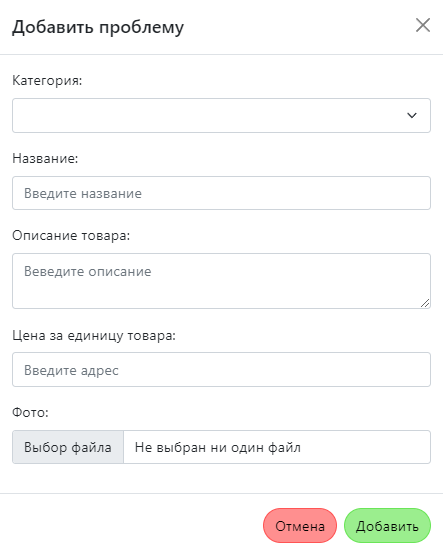


Рис.3. – Макет формы добавление товара

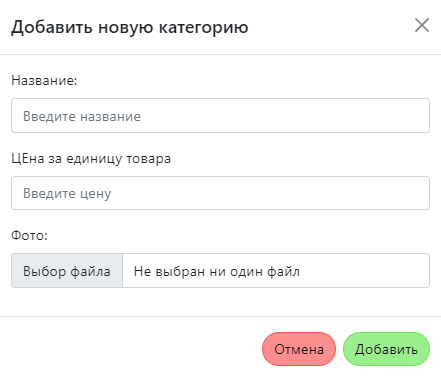


Рис.4. – Макет формы добавление категории

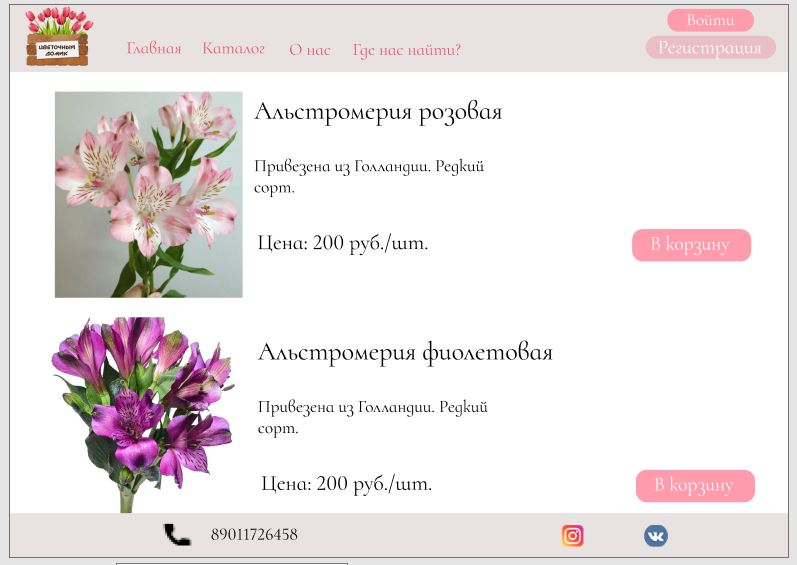


Рис.5. – Макет страницы «Товары»

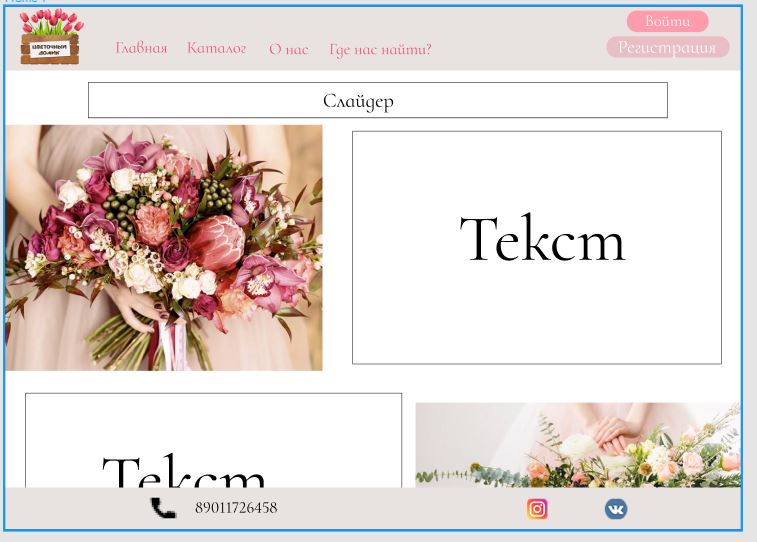


Рис.6. – Макет страницы «О нас»

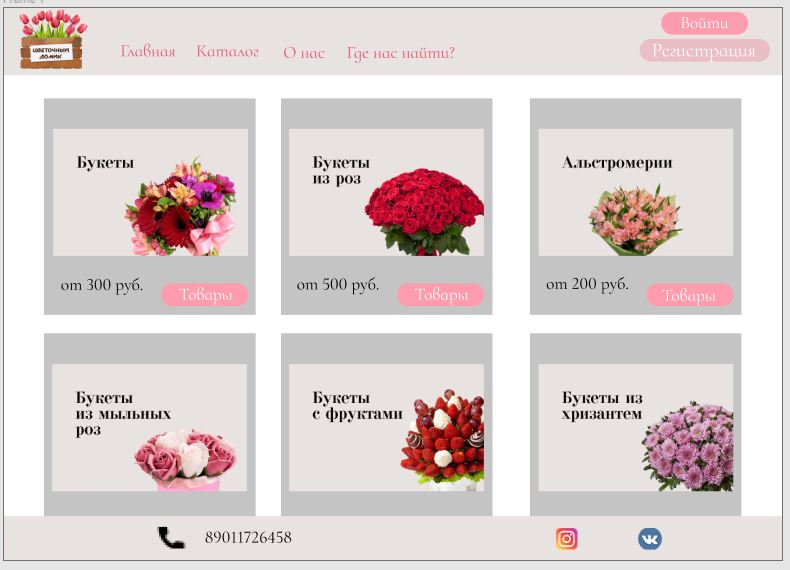


Рис.7. – Макет страницы «Каталог»

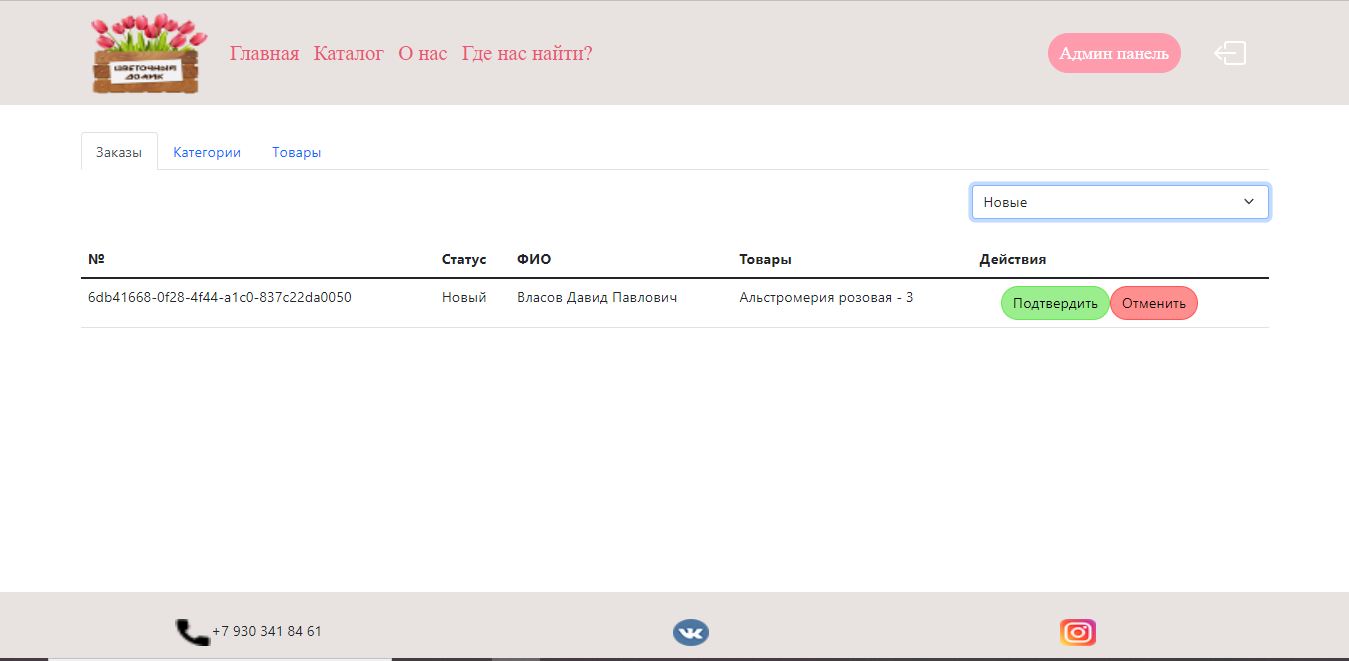


Рис.8. – Макет страницы «Администратор. Заказы»

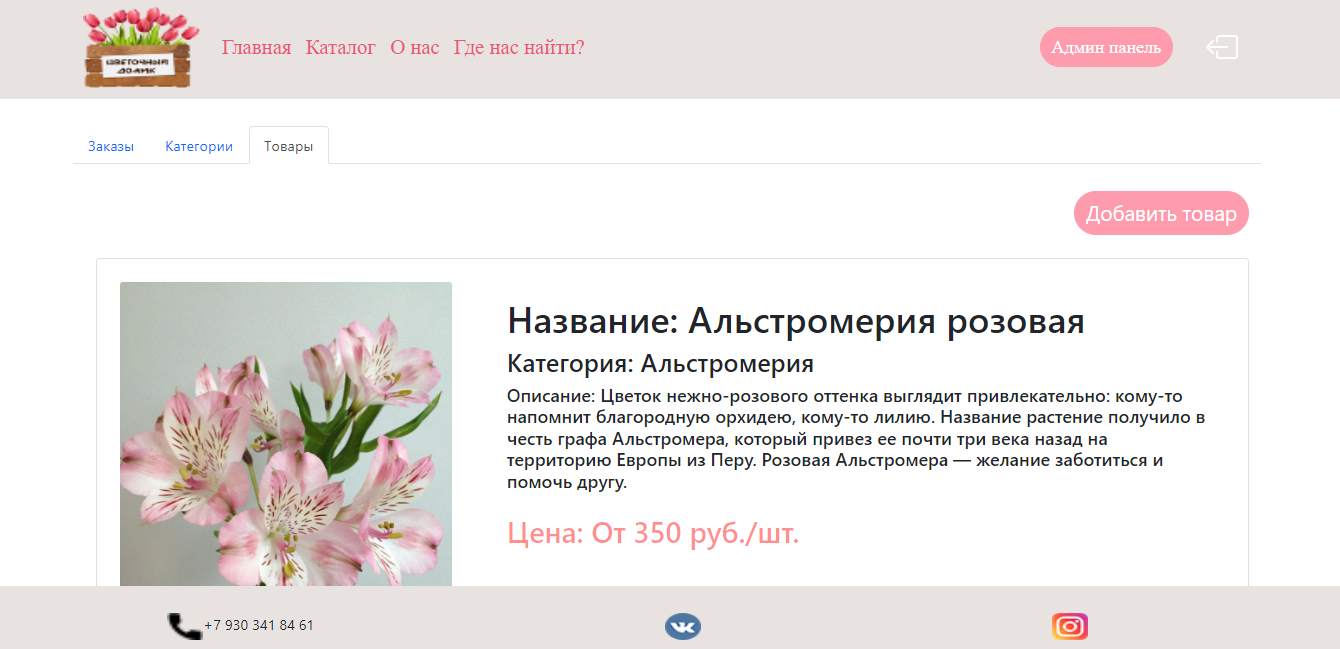


Рис.9. – Макет страницы «Администратор. Товары»

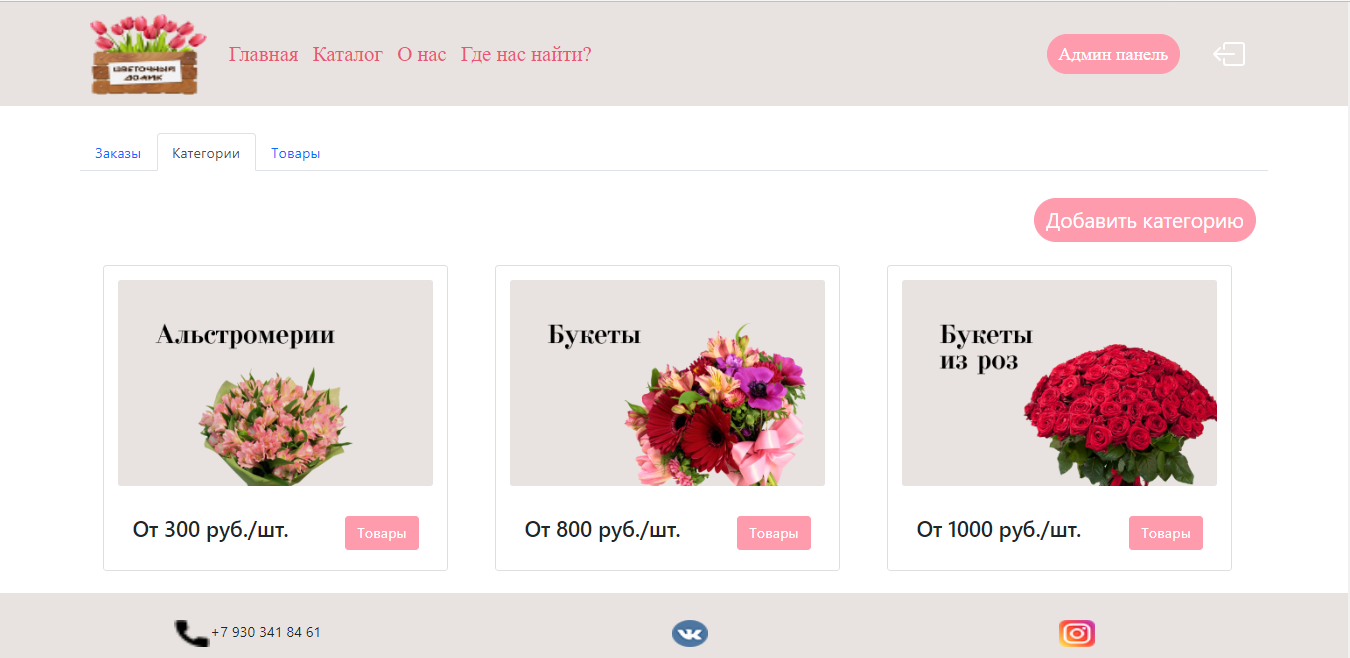


Рис.10. – Макет страницы «Администратор. Категории»

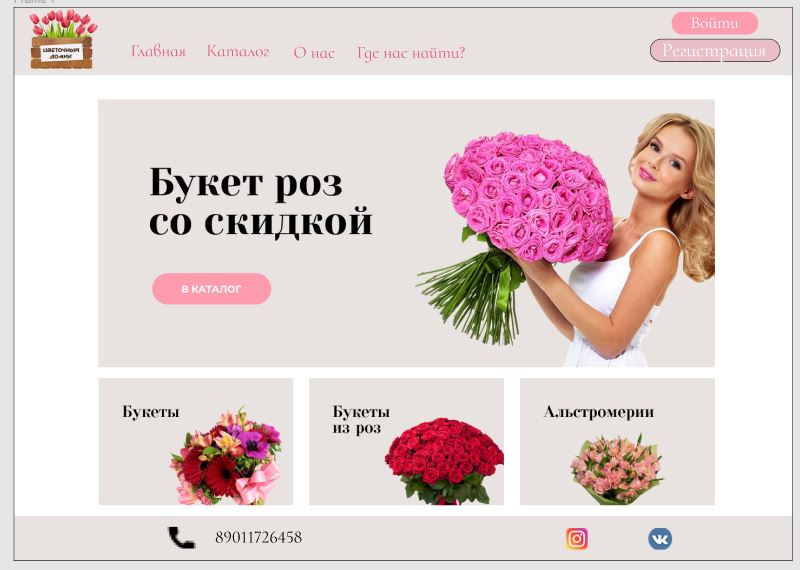


Рис.11. – Макет страницы «Главная»

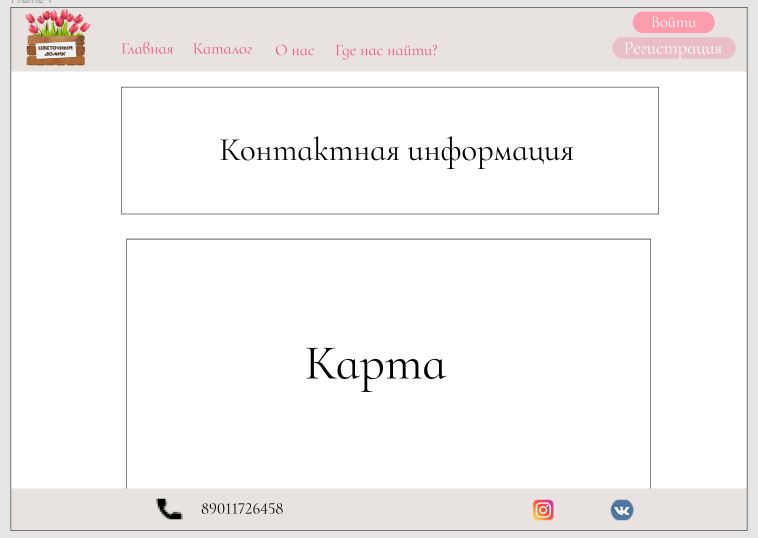


Рис.12. – Макет страницы «Где нас найти?»

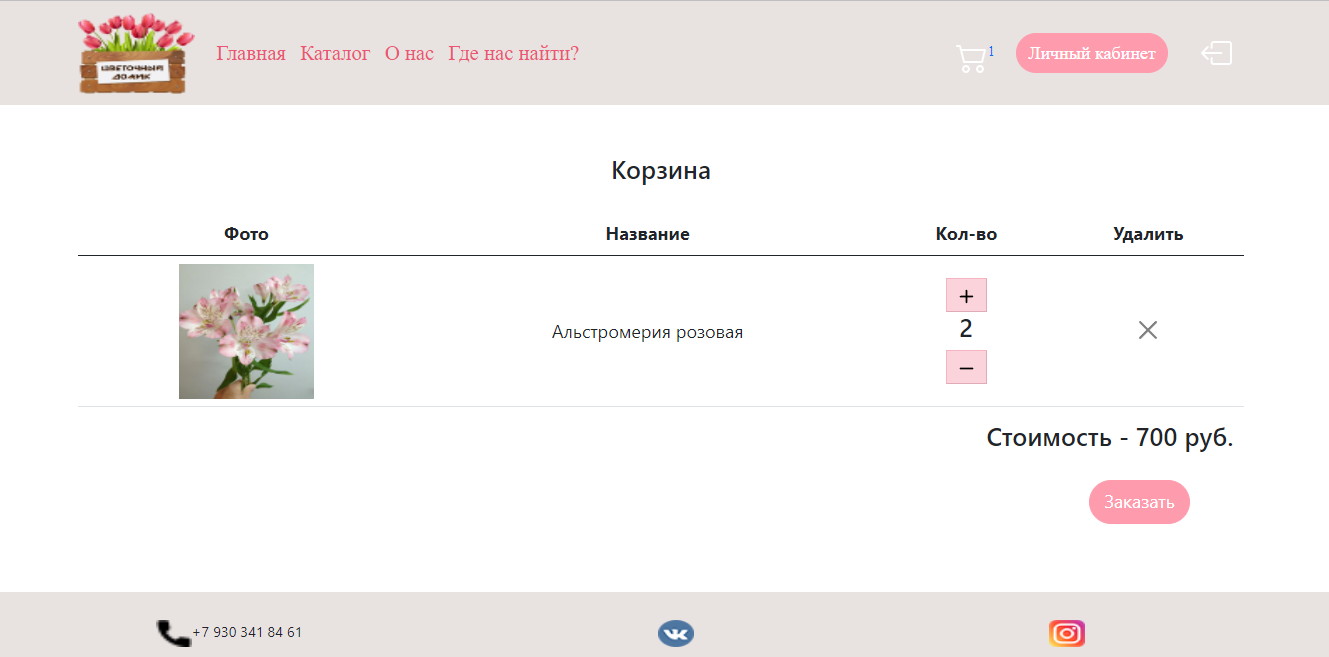


Рис.13. – Макет страницы «Корзина»

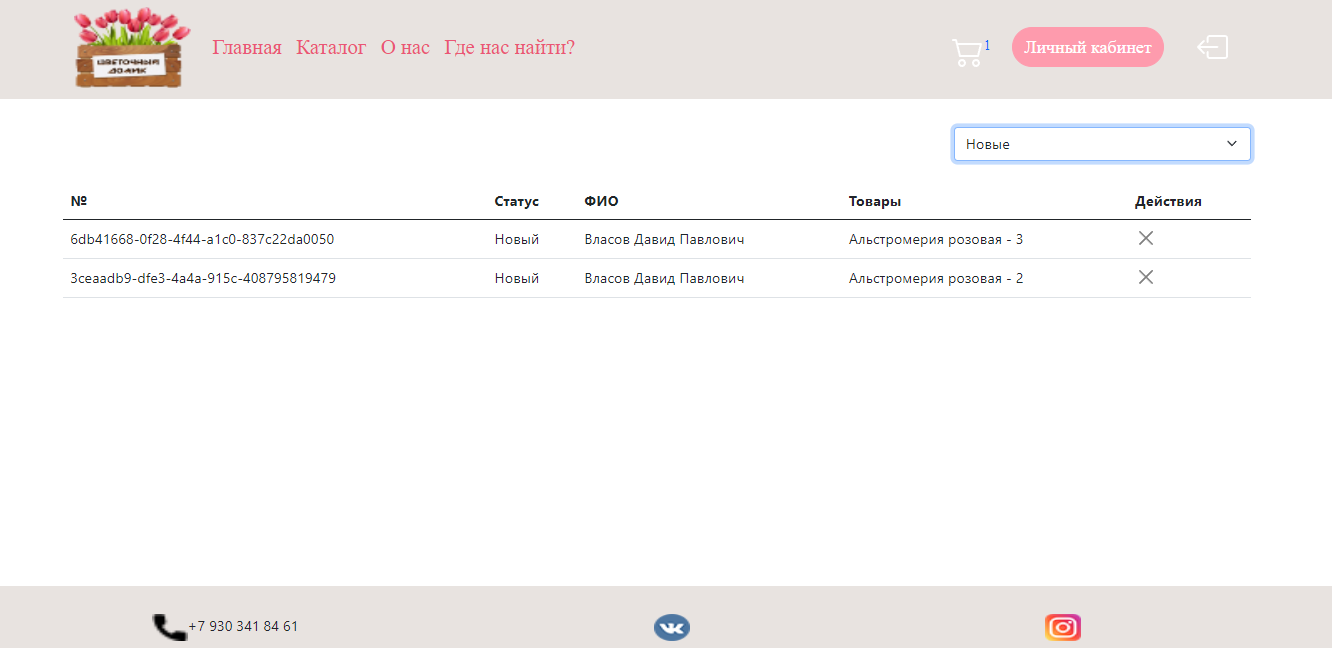


Рис.14. – Макет страницы «Личный кабинет»

# Верстать дизайн-макеты приложения «Цветочный магазин» с применением фреймворка ReactJs

## Главная станица

Листинг:

import React, { Component } from 'react'

import 'bootstrap'

import Header from '../components/Header';

import Slider from '../components/Slider';

import { Container } from 'react-bootstrap';

import main1 from '../img/main1.png'

import main2 from '../img/main2.png'

import main3 from '../img/main3.png'

import main4 from '../img/main4.png'

import main5 from '../img/main5.png'

import main6 from '../img/main6.png'

import main7 from '../img/main7.png'

import main8 from '../img/main8.png'

import main9 from '../img/main9.png'

import main10 from '../img/main10.png'

import main11 from '../img/main11.png'

import main12 from '../img/main12.png'

import main13 from '../img/main13.png'

import main14 from '../img/main14.png'

import main15 from '../img/main15.png'

const Home =()=> {

      return(

          <div>

              <Container style={{ padding:" 0% 0% 20% 0% "}}>

              <Slider/>

              <div style={{padding:" 0 0 7% 0", float:"right", textAlign:"center"}}>

                <img  style={{padding:"1%"}} src={main1}/>

                <img  style={{padding:"1%"}} src={main2}/>

                <img style={{padding:"1%"}}  src={main3}/>

                <img style={{padding:"1%"}} src={main4}/>

                <img  style={{padding:"1%"}} src={main5}/>

                <img  style={{padding:"1%"}} src={main6}/>

                <img  style={{padding:"1%"}} src={main7}/>

                <img style={{padding:"1%"}} src={main8}/>

                <img style={{padding:"1%"}}  src={main9}/>

                <img  style={{padding:"1%"}}src={main10}/>

                <img  style={{padding:"1%"}} src={main11}/>

                <img  style={{padding:"1%"}} src={main12}/>

                <img  style={{padding:"1%"}} src={main13}/>

                <img  style={{padding:"1%"}} src={main14}/>

                <img style={{padding:"1%"}} src={main15}/>

                </div>

              </Container>

             </div>

      );

}

export default Home

## Страница « Где нас найти?»

Листинг:

import React, { Component } from 'react'

import 'bootstrap'

import Header from '../components/Header';

import Slider from '../components/Slider';

import { Container } from 'react-bootstrap';

import vk from '../img/vc.png'

import insta from '../img/insta.png'

import ad from '../img/adress.png'

import { YMaps, Map, Placemark } from 'react-yandex-maps';

const Where =()=> {

      return(

          <div>

              <Container style={{ padding:" 0% 0% 20% 0% ", textAlign:"center"}}>

              <h2>Вы можете нас найти по адресу: г. Иваново, ул. Шевченко, д. 2.</h2>

<h2>Для связи можно использовать телефон: +7 930 341 84 61</h2>

<h2>Или по почте: Marya.ivanova.17@inbox.ru</h2>

<h2>Наши социальные сети:</h2>

 <a href='https://vk.com/roza5080' style={{ padding:"5%"}}><img  height="30"

                                width="40"  src={vk}

            /></a>

              <a href='https://www.instagram.com/cvetochnidomik/' style={{ padding:"5%"}}><img  height="30"

                                width="40" src={insta}/></a>

                                <h1> Вы найдете нас здесь))</h1>

<YMaps >

  <Map

width={1300} height={500}

    defaultState={{

      center: [57.000358, 40.930927],

      zoom: 15,

    }}

  >

    <Placemark geometry={[57.000358, 40.930927]} />

  </Map>

</YMaps>

              </Container>

             </div>

      );

}

export default Where

## Страница « О нас»

Листинг:

import React, { Component } from 'react'

import 'bootstrap'

import Header from '../components/Header';

import Slider from '../components/Slider';

import { Container, Breadcrumb } from 'react-bootstrap';

import cvet1 from '../img/cvet1.jpg'

import cvet2 from '../img/cvet2.jpg'

import cvet3 from '../img/cvet3.jpg'

import cvet4 from '../img/cvet4.jpg'

import cvet5 from '../img/cvet5.jpg'

const Info =()=> {

      return(

          <div>

              <Container style={{ padding:" 0% 0% 20% 0% "}}>

              <Breadcrumb>

              <Breadcrumb.Item href="/">Главная</Breadcrumb.Item>

  <Breadcrumb.Item active> О нас</Breadcrumb.Item>

</Breadcrumb>

              <Slider/>

              <div style={{padding:" 2% 2% 10% 2%", float:"right", textAlign:"left"}}>

              <img  style={{padding:"1%", float:"right", width:"40%", height:"40%"}} src={cvet1}/>

             <h1> Интернет магазин цветов Цветочный домик</h1>

<p>Интернет магазин цветов «Цветочный домик» предлагает жителям Иваново и других регионов приобретать свежие цветы и букеты в ассортименте с доставкой на лучших условиях. Закажите у наших флористов свою уникальную композицию, которая соответствует созданию торжественной атмосферы, наполняя любые помещения неповторимыми и уникальными ароматами. Обращайтесь к менеджерам компании, если хотите приобрести понравившийся вам букет в Иваново. Мы с удовольствием обсудим детали поставки, вашу оригинальную идею, способствующую воплощению самых смелых мечтаний.

</p>

<img  style={{padding:"1%", float:"left", width:"38%", height:"38%"}} src={cvet2}/>

<h1>Букеты для любых ситуаций</h1>

<p>Современный ритм жизни не обходится без выразительного «языка цветов», сохранившего свою актуальность. Цветы являются ассоциацией с нежностью и страстью, виной и смирением, благодарностью и любовью. Вы не знакомы с понятием цветовой гаммы? Это не проблема. Обратившись к нашим специалистам, вы получите услугу по индивидуальному созданию букета, с учетом вкусов и возраста получателя, в соответствии с конкретным поводом для презента.

</p>

<p>Мы поможем реализовать даже спонтанно пришедшую мысль об отправлении цветочной композиции. Наш коллектив состоит из опытных водителей и курьеров, которые помогут передать вашей любимой, подруге сюрприз, который поднимет настроение. Заказ цветов онлайн возможно в любых стилях, от романтического до эпатажного, от оригинального до классического. Мы позаботимся о реализации всех фантазий, о максимальном удовольствии и комфорте.

</p>

<img  style={{padding:"1%", float:"right", width:"40%", height:"35%"}} src={cvet3}/>

<h1>Выбирайте упаковку

</h1>

<p>В нашем магазине цветов большой выбор упаковок, коробок, корзин, поэтому мы поможем найти решение, соответствующее любым ситуациям. Заказывайте композицию в коробке, для свадьбы, годовщины, дня рождения или другого торжества, собирающего много гостей. Благодаря коробке композиция будет дольше оставаться свежей, даже без вазы. Эта проблема нередко вызывает затруднения, не только в месте, выбранном для мероприятия, но и в домашних условиях. Коробка является универсальным вариантом, одинаково хорошо выглядящим на полке или столе, иных поверхностях. Букет будет хорошо заметен, в сочетании с другими яркими красками, иными цветочными композициями. Он сохранит собственную микрофлору, продлевающую жизнь цветов.

</p>

<p>Корзинки также выделяются нарядным, торжественным, интересным видом. Они удобны при носке, идеально сочетаются с любыми соцветиями. Корзинки укомплектованы, аналогично коробкам, губкой – оазисом, напитывающимся влагой и сохраняющим жизнь растений длительней, в сравнении с обычной водой. Выбирая упаковку для букета, следует подумать о сохранении свежести, удобстве носки, чтобы спокойно прислонять к себе и не намокнуть. Цветы отличаются нежностью, трепетностью, подходят для свиданий и комплиментов.

</p>

<h6>Наша специализация заключается в отличном настроении!

</h6>

<p>Человек, который дарит цветы, должен поделиться положительными эмоциями, обусловленным внешней красотой и неповторимым ароматом. Розы ассоциируются с приятными мыслями, ромашки помогают успокоиться, вспомнить о счастливых моментах жизни. Яркость ароматных ноток, присущая пионам, наполняет хорошим настроением. Цветов много, но нельзя забывать о правильной презентации. Оформление букета напрямую влияет на ощущения, свойственные получателю.

</p>

<p>Наш магазин цветов обеспечивает помощью компетентных флористов, сопроводительными открытками и прочими приятными аксессуарами, своевременной доставкой, тщательно продуманным сервисом.

</p>

<img  style={{padding:"1%", float:"left", width:"35%", height:"30%"}} src={cvet4}/>

<h2>Преимущества интернет магазина «Цветочный домик»

</h2>

<h6>Наш магазин цветов предоставляет гарантию качества реализуемой нами продукции, подтверждающегося множеством благодарных отзывов. Доставьте радость своим близким благодаря букету, созданному руками умельца флористики, оригинальной и классической цветочной композиции. Наши преимущества обусловлены:

</h6>

<h6>Возможностью заказа оптовых и розничных партий цветов;

</h6>

<h6>Дружной и умелой командой опытных профессионалов;

</h6>

<h6>Обширным выбором цветов, соответствующим любым вкусам и предпочтениям;

</h6>

<h6>Возможностью оплаты удобными покупателю способами;

</h6>

<h6>Постоянными скидками, акциями, предназначенными постоянным и новым клиентам;

</h6>

<h6>Оперативной доставкой, соответствующей пожеланиям заказчиков.

</h6>

<p>Наша компания обеспечивает взаимовыгодную основу сотрудничества, важную для постоянных заказчиков и благодарных покупателей. В основе нашей деятельности лежит ответственный подход к делу, внимательное отношение к клиентам. Современные тенденции флористики соответствуют неугасающей моде живописных букетов, отличающихся оформлением. Мы порадуем вас изумительным цветочным набором, чтобы восхитить свою возлюбленную, потрясающим букетом, соответствующим юбилею или именинам.

</p><img  style={{padding:"1%", float:"right", width:"40%", height:"35%"}} src={cvet5}/>

<p>Мы подберем цветочный букет вашим друзьям или коллегам, вашей возлюбленной или маме. Если у вас не хватает времени, чтобы сделать выбор подарка, воспользуйтесь доставкой цветов, организованной магазином цветов «Цветочный домик» в Иваново. Цветы с доставкой являются универсальным способом, чтобы продемонстрировать свои чувства. Обращайтесь к нам, чтобы порадовать именинника цветочной композицией, которая станет воплощением вашего искреннего отношения.

</p>

                </div>

              </Container>

             </div>

      );

}

export default Info

## Страница « Каталог»

Листинг станицы:

import React, { Component } from 'react'

import 'bootstrap'

import Header from '../components/Header';

import Slider from '../components/Slider';

import { Container, Breadcrumb } from 'react-bootstrap';

import CardCate from '../components/CardCate';

const Catalog =()=> {

      return(

          <div>

              <Container style={{ padding:" 0% 0% 20% 0% "}}>

              <Breadcrumb>

              <Breadcrumb.Item href="/">Главная</Breadcrumb.Item>

  <Breadcrumb.Item active> Каталог</Breadcrumb.Item>

</Breadcrumb>

              <div style={{textAlign:"center"}}> <CardCate/></div>

              <div  style={{padding:" 0% 0% 1% 0% ", color:"white"}}>  dfvdfgdtd</div>

              </Container>

             </div>

      );

}

export default Catalog

Листинг компонента карточка:

import React, { Component, useState,useEffect } from 'react';

import { Card,  CardTitle, CardText, CardHeader, CardBody, CardImg, CardGroup, CardDeck, Button} from 'reactstrap'

const CardCate = () =>{

  const [category, setCategory]= useState([]);

const fetchData = async ()=>{

    const data = await fetch("http://localhost:8080/category/all");

    const dataJSON = await data.json();

    setCategory( dataJSON.category);

  }

  useEffect(()=>{

    fetchData();

  }, [])

  return (

      <div style={{ textAlign:"center"}}>

            {category.map(category => (

          <CardGroup  style={{width:"33%", float:"left", padding:"1% 2% 1% 2%"}}>

          <Card style={{display: "inline-block", verticalAlign: "middle",  padding:"4%"}}>

          <CardImg

                style={{height:"100", margin:"0 0 5% 0"}}

                src={`/file/${category.img}`}/>

            <CardBody>

            <CardTitle style={{ float:"left"}} ><h4> От {category.price} руб./шт.</h4> </CardTitle>

            <Button style={{ float:"right", background:"#FE9BAD", color:"white",  borderColor:" #FE9BAD" }}  href={`/catalog/${category.name}`}>Товары</Button>

            </CardBody>

        </Card>

        </CardGroup>

        ))}

    </div>

  );

}

export default CardCate

## Страница « Товары»

Листинг станицы:

import React, { Component } from 'react'

import 'bootstrap'

import { useParams } from 'react-router-dom'

import Header from '../components/Header'

import { Container, Breadcrumb } from 'react-bootstrap'

import CardProd from '../components/CardProd';

const Cvet =()=> {

  const name = useParams();

  console.log(name.id)

      return

          <div>

              <Container style={{ padding:" 0% 0% 20% 0% "}}>

              <Breadcrumb>

              <Breadcrumb.Item href="/">Главная</Breadcrumb.Item>

  <Breadcrumb.Item href="/catalog">

    Каталог

  </Breadcrumb.Item>

  <Breadcrumb.Item active> {name.id} </Breadcrumb.Item>

</Breadcrumb>

              <div style={{textAlign:"center"}}> <CardProd data = {name.id} /></div>

              <div  style={{padding:" 0% 0% 1% 0% ", color:"white"}}>  dfvdfgdtd</div>

              </Container>

             </div>

      );

}

export default Cvet

Листинг компонента карточка:

import React, { Component, useState,useEffect } from 'react';

import { Card,  CardTitle, CardText, CardHeader, CardBody, CardImg, CardGroup, CardDeck, Button} from 'reactstrap'

import user from "../store/user";

import cart from "../store/cart";

const CardProd = (nameC) =>{

  const [product, setProduct]= useState([]);

  const name = nameC.data;

  // console.log(id)

const fetchData = async ()=>{

    const data = await fetch(`http://localhost:8080/product/all/${name}`);

    const dataJSON = await data.json();

    console.log(dataJSON)

    setProduct(dataJSON.products);

  }

  useEffect(()=>{

    fetchData();

  }, [])

  return (

      <div >

            {product.map(product => (

     <div>

         <CardGroup  style={{ float:"left", padding:"1% 1% 1% 2%"}}>

          <Card style={{display: "inline-block", verticalAlign: "middle",  padding:"2%", height:"50%", width:"50%"}}>

          <CardImg

                style={{height:"30%", width:"30%", margin:"0 5% 0% 0", float:"left"}}

                src={`/file/${product.img}`}/>

            <CardBody style={{ textAlign:"left" }}>

            <CardTitle> <h1> Название: {product.name} </h1></CardTitle>

            <CardTitle> <h3>Категория: {product.category} </h3></CardTitle>

              <CardTitle> <h5> Описание: {product.description} </h5></CardTitle>

            <CardTitle style={{color:"rgba(255, 21, 21, 0.48)", margin:"2%"}}><h2> Цена: От {product.price} руб./шт.</h2> </CardTitle>

            {user.auth && <Button style={{ float:"right", background:"#FE9BAD", color:"white", bottom:"0",  borderColor:" #FE9BAD" }} onClick={()=>cart.addProduct(product)}>В корзину</Button>}

            </CardBody>

        </Card>

        </CardGroup>

      </div>

        ))}

    </div>

  );

}

export default CardProd

# Разработать формы регистрации, авторизации и соответствующие процедуры работы с базой данных (бэкенд)

## Форма регистриции

Листинг кода на клиенте:

import React, { useState } from 'react'

import { Alert } from 'react-bootstrap'

import {Modal, ModalHeader, ModalBody, ModalFooter, Label, FormFeedback, Form, FormGroup, Button, Input} from 'reactstrap'

const Signup = () =>  {

    const [opened, setOpened] = useState();

    const [decision, setDecision] = useState(false);

    const [name, setName] = useState('');

    const [surname, setSurname] = useState('');

    const [patronymic, setPatronymic] = useState('');

    const [login, setLogin] = useState('');

    const [password, setPassword] = useState('');

    const [email, setEmail] = useState('');

    const [nameInvalid, setNameInvalid]= useState(false);

    const [surnameInvalid, setSurnameInvalid]= useState(false);

    const [patronymicInvalid, setPatronymicInvalid]= useState(false);

    const [loginInvalid, setLoginInvalid]= useState(false);

    const [emailInvalid, setEmailInvalid]= useState(false);

    const [passwordInvalid, setPasswordInvalid] = useState(false);

    const toggle = ()=> setOpened((value)=> !value);

    const onSignup = async ()=>{

        setNameInvalid(false);

        setSurnameInvalid(false);

        setPatronymicInvalid(false);

        setLoginInvalid(false);

        setEmailInvalid(false);

        setPasswordInvalid(false);

        if ( !name || !surname || !patronymic || !login || !email || !password){

           if (!name) setNameInvalid(true);

           if (!surname) setSurnameInvalid(true);

           if (!patronymic) setPatronymicInvalid(true);

           if (!login) setLoginInvalid(true);

           if (!email) setEmailInvalid(true);

           if (!password) setPasswordInvalid(true);

           alert("Ошибка ввода");

           return;

        };

        console.log(name, surname, patronymic, login, email, password)

        const data = await fetch("/users/signup",{

            method:"POST",

            headers:{"content-Type": "application/json"},

            body: JSON.stringify({ name, surname, patronymic, login, email, password})

        });

        const dataJSON = await data.json();

        console.log(dataJSON)

         if (dataJSON.success === false){

             alert(dataJSON.code)

         }else{

             setDecision(true)};

             alert("Вы зарегистрировались. Можете войти в систему под своим логином и паролем!");

    }

    const handleOnChange = (event) =>{

        setName(event.target.value)

    }

    const handleOnChange1 = (event) =>{

        setSurname(event.target.value)

    }

    const handleOnChange2 = (event) =>{

        setPatronymic(event.target.value)

    }

     const handleOnChange3 = (event) =>{

        setPassword(event.target.value)

    }

    const handleOnChange4 = (event) =>{

        setLogin(event.target.value)

    }

    const handleOnChange5 = (event) =>{

         setEmail(event.target.value)

     }

        return (

            <>

                    <Button  onClick={toggle}   style={{ margin:"1%", fontSize: "23px", textAlign:"center" ,

                    borderRadius:"30px", background:"rgba(234, 86, 117, 0.26)", borderColor:"rgba(234, 86, 117, 0.26)"}}>

                    Регистрация

                    </Button>

                    <Modal isOpen={opened} toggle={toggle}>

            <ModalHeader toggle={toggle}>Регистрация</ModalHeader>

            <ModalBody>

                <Form>

                <FormGroup>

                    <Label >Фамилия:</Label>

                    <Input

                    type="text"

                    placeholder="Введите фамилию"

                    value={surname}

                    onChange={(event)=>handleOnChange1(event)}

                    invalid={surnameInvalid}

                     />

                    <FormFeedback>Обязательное поле</FormFeedback>

                </FormGroup>

                <FormGroup >

                    <Label>Имя:</Label>

                    <Input

                     type="text"

                     placeholder="Введите имя"

                     value={name}

                     onChange={(event)=>handleOnChange(event)}

                     invalid={nameInvalid}

                       />

                    <FormFeedback>Обязательное поле</FormFeedback>

                </FormGroup>

                <FormGroup>

                    <Label >Отчество:</Label>

                    <Input

                    type="text"

                    placeholder="Введите отчество"

                    value={patronymic}

                    onChange={(event)=>handleOnChange2(event)}

                    invalid={patronymicInvalid}

                     />

                    <FormFeedback>Обязательное поле</FormFeedback>

                </FormGroup>

                <FormGroup >

                    <Label>Почта:</Label>

                    <Input

                     type='email'

                     placeholder="Введите почту"

                     value={email}

                onChange={(event)=>handleOnChange5(event)}

                invalid={emailInvalid}

                       />

                    <FormFeedback>Обязательное поле</FormFeedback>

                </FormGroup>

                <FormGroup >

                    <Label>Логин:</Label>

                    <Input

                     type="text"

                     placeholder="Введите логин"

                     value={login}

                onChange={(event)=>handleOnChange4(event)}

                invalid={loginInvalid}

                       />

                    <FormFeedback>Обязательное поле</FormFeedback>

                </FormGroup>

                <FormGroup >

                    <Label>Пароль:</Label>

                    <Input

                     type="password"

                     placeholder="Введите пароль"

                     value={password}

                     onChange={(event)=>handleOnChange3(event)}

                     invalid={passwordInvalid}

                       />

                    <FormFeedback>Обязательное поле</FormFeedback>

                </FormGroup>

                </Form>

                </ModalBody>

                <ModalFooter>

                <Button style={{ borderRadius:"30px", background:"rgba(255, 21, 21, 0.48)", color:"black", borderColor:"rgba(255, 21, 21, 0.48)"}}  type="submit" onClick={toggle}>Отмена</Button>

                <Button style={{ borderRadius:"30px", background:"rgba(66, 222, 41, 0.53)", color:"black", borderColor:"rgba(66, 222, 41, 0.53)"}}  type="submit" onClick={ onSignup}>Зарегистрироваться</Button>

                               </ModalFooter>

                </Modal>

            </>

        )

        }

export default Signup

Листинг кода на сервере:

router.post("/signup",(req, res) => {

    const {name, surname, patronymic, login, email, password} = req.body;

    if(!name || !surname || !patronymic || !login || !email || !password){

        res.send({success: false,code:"wrong"});

        return;

    }

    let users=db.getData("/users");

    const currentUser=users.filter(el => el.email === email)[0];

    if (currentUser){

        res.send({success: false,code:"Пользователь с такой почтой уже существует."});

        return;

    }

    users = users.filter(el => el.email !== email);

    users.push({

        id:uuid(),

        name,

        surname,

        patronymic,

        login,

         email,

         password,

        role: "user"

    });

    db.push("/users",users);

    res.send({success:true});

});

## Форма авторизации

Листин кода будет почти такимже как и в форме регистрации только с полями «Лигн», «Пароль» и «Почта».

Листинг кода на сервере:

router.post("/login", (req, res) =>{

    const { login, email,   password} = req.body;

    if (!login   || !password || !email  ){

        if(!email){

        res.send({ success: false, code:"no email"})};

        if(!password){

            res.send({ success: false, code:" no passw"})};

        return;

        if(!login){

            res.send({ success: false, code:"no login"})};

        return;

    }

    let users = db.getData("/users")

    const currentUser = users.filter(el=> el.email === email)[0];

    if (!currentUser || currentUser.password !== password ){

        if (!currentUser){

            res.send({success: false, code: " Такого пользователя нет" });

            return;}

        if (currentUser.password !== password){

            res.send({success: false, code: "Пароль неверен" });

            return;}

    }

    const sessionId = uuid();

    db.push(`/session/${sessionId}`, currentUser.id);

    res.send({ success: true, token: sessionId, id: currentUser.id, role: currentUser.role});

}

);

# Разработать корзину заказов приложения

## Страница « Корзина »

Добавление заказа на сервер происходит так же как в добавление товара или добавление категории.

Листинг страницы «Корзина»:

import React from "react";

import {Container} from "react-bootstrap";

import { observer } from "mobx-react-lite";

import cart from "../store/cart";

import OrderButton from "../components/OrderBut";

import OrderTable from "../components/OrderTab";

const Cart = () => {

  return (

    <Container className="text-center min\_height">

      <h3 className="pt-4">Корзина</h3>

      <div className="pt-4">

        {cart.cart.length > 0 ? (

          <>

            <OrderTable/>

          <Container className="text-end mb-4">

                <h3>Стоимость - {cart.price} руб.</h3>

                <OrderButton/>

          </Container>

          </>

        ) : (

          <h4>Вы не добавили товар...</h4>

        )}

      </div>

    </Container>

  );

};

export default observer(Cart);

Листинг компанента «Таблица»:

import React from "react";

import { CloseButton, Image, Table } from "react-bootstrap";

import { Button } from "reactstrap"

import cart from "../store/cart";

const OrderTable = () => {

  console.log(cart.cart)

  return (

    <Table className="fs-5 align-middle " responsive="sm" >

      <thead>

        <tr>

          <th>Фото</th>

          <th>Название</th>

          <th>Кол-во</th>

          <th>Удалить</th>

        </tr>

      </thead>

      <tbody>

        {cart.cart.map((item) => (

          <tr key={item.id}>

            <td>

              <img

                width="150"

                height="150"

                src={`/file/${item.img}`}

              />

            </td>

            <td>{item.name}</td>

            <td>

            <button onClick={() => cart.addQuantity(item.id)} style={{ background:"rgba(234, 86, 117, 0.26)", borderColor:"rgba(234, 86, 117, 0.26)" } }>

                <svg

                  xmlns="http://www.w3.org/2000/svg"

                  width="30"

                  height="30"

                  className="bi-plus"

                  viewBox="0 0 16 16"

                >

                  <path d="M8 4a.5.5 0 0 1 .5.5v3h3a.5.5 0 0 1 0 1h-3v3a.5.5 0 0 1-1 0v-3h-3a.5.5 0 0 1 0-1h3v-3A.5.5 0 0 1 8 4z" />

                </svg>

              </button>

             <h3>{item.quantity}</h3>

             <button onClick={() => cart.delQuantity(item.id)} style={{ background:"rgba(234, 86, 117, 0.26)", borderColor:"rgba(234, 86, 117, 0.26)" } }>

                <svg

                  xmlns="http://www.w3.org/2000/svg"

                  width="30"

                  height="30"

                  fill="currentColor"

                  className="bi-dash"

                  viewBox="0 0 16 16"

                >

                  <path d="M4 8a.5.5 0 0 1 .5-.5h7a.5.5 0 0 1 0 1h-7A.5.5 0 0 1 4 8z" />

                </svg>

              </button>

            </td>

            <td>

              <CloseButton

                onClick={() => cart.delProduct(item.id)}

              ></CloseButton>

            </td>

          </tr>

        ))}

      </tbody>

    </Table>

  );

};

export default OrderTable;

Листинг компанента «Кнопка»:

import React, {  useState } from 'react'

import { Button, Form, Modal } from 'react-bootstrap'

import { observer } from 'mobx-react-lite'

import cart from '../store/cart'

import axios from 'axios';

const OrderButton = () => {

  const [show,setShow] = useState(false)

  const [password,setPassword] = useState('')

  const id = localStorage.getItem('id')

  const fetchOrder = () => {

    let data = JSON.stringify({id, password, products:cart.cart, price: cart.price})

    axios.post(`/order/add`,data,{

      headers:{

        "Content-Type": "application/json",

      }

    }).then(()=>{

      alert('Ваш заказ оформлен')

      cart.addOrder()

      setShow(false)

    }).catch(err=>{

      alert(err.response.data.values.message)

    })

    console.log(data)

  }

  return (

    <>

      <Button onClick={()=>setShow(true)} size="lg" style={{  fontSize: "20px", textAlign:"center" ,

                    borderRadius:"30px", background:"#FE9BAD",  borderColor:" #FE9BAD"}} className="mt-4 me-5">Заказать</Button>

      <Modal centered show={show} onHide={()=>setShow(false)}>

        <Modal.Header closeButton>Заказ</Modal.Header>

        <Modal.Body>

        <Form.Group className="mb-3" controlId="exampleForm.ControlInput1">

              <Form.Label>Введите пароль для подтверждения</Form.Label>

              <Form.Control

                value={password}

                onChange={(e)=>setPassword(e.target.value)}

                type="password"

                autoFocus

              />

            </Form.Group>

        </Modal.Body>

        <Modal.Footer>

          <Button onClick={fetchOrder} style={{ borderRadius:"30px", background:"rgba(66, 222, 41, 0.53)", color:"black",  borderColor:"rgba(66, 222, 41, 0.53)"}}>Подтвердить</Button>

        </Modal.Footer>

      </Modal>

    </>

  )

}

export default observer(OrderButton)

## Страница « Личный кабинет »

Листинг страница анологичен листингу страница «Товары» и сраницы «Категории» толькомпонент которыймы будем вызывать называется таблица заказов тользователя.

Листиг компанента « Таблица»:

import React, { useEffect, useState } from "react";

import {

  Button,

  Form,

  FormGroup,

  Modal,

  Spinner,

  Table,

  CloseButton

} from "react-bootstrap";

import axios from 'axios';

const TablePr = () => {

  const [data, setData] = useState([]);

  const [loading, setLoading] = useState(true);

   const id = localStorage.getItem('id')

  const fetchGetOrder = async () => {

    axios

      .get(`http://localhost:8080/order/all/${id}`, {

        headers: {

          "Content-Type": "application/json",

        },

      })

      .then((response) => {

        // console.log(response.data.order)

        setData(response.data.order);

        setSortData(response.data.order);

      })

      .catch((err) => {

        console.log(err);

      });

  };

  useEffect(() => {

    fetchGetOrder();

    // eslint-disable-next-line react-hooks/exhaustive-deps

  }, []);

  const [sortType, setSortType] = useState("all");

  const [sortData, setSortData] = useState([]);

  const sortArray = (type) => {

    const types = {

      all: "Все",

      new: "Новый"

    };

    const sortProperty = types[type];

    let sorted = [];

    sortProperty === "Все"

      ? (sorted = [...data])

      : (sorted = [...data].filter((a) => a.status === sortProperty));

    setSortData(sorted);

    setLoading(false);

  };

  useEffect(() => {

    sortArray(sortType);

    // eslint-disable-next-line react-hooks/exhaustive-deps

  }, [sortType]);

  const fetchDelete = async (id) => {

    await axios

      .post(`http://localhost:8080/order/delete`, JSON.stringify({ id: id}), {

        headers: {

          "Content-Type": "application/json"

        },

      })

      .then((response) => {

        alert('Вы удалили заказ');

        setData(response.data.values)

      })

      .catch((err) => console.log(err));

  };

console.log(sortData)

  return (

    <>

      {!loading ? (

        <>

          {sortData.length > 0 ? (

            <>

              <div className="d-flex justify-content-end">

                <Form.Select

                  onChange={(e) => setSortType(e.target.value)}

                  style={{ minWidth: "150px" }}

                  className="w-25"

                >

                  <option value="all">Все заказы</option>

                  <option value="new">Новые</option>

                </Form.Select>

              </div>

              <Table className="mt-4">

                <thead>

                  <tr>

                    <th>№</th>

                    <th>Статус</th>

                    <th>ФИО</th>

                    <th>Товары</th>

                    {sortType === "new" ? <th>Действия</th> : <></>}

                  </tr>

                </thead>

                <tbody>

                  {sortData.map((item) => (

                    <tr key={item.id}>

                      <td>{item.id}</td>

                      <td>{item.status}</td>

                      <td>{item.FIO || ""}</td>

                      <td>

                        {item.products.map((i) => (

                          <div key={i.id}>

                            {i.name} - {i.quantity}

                          </div>

                        ))}

                      </td>

                      {sortType === "new" ? (

                        <td>

                          <CloseButton

                       onClick={() => fetchDelete(item.id)}

                          ></CloseButton>

                        </td>

                      ) : (

                        <></>

                      )}

                    </tr>

                  ))}

                </tbody>

              </Table>

            </>

          ) : (

            <h3>Нет заказов</h3>

          )}

        </>

      ) : (

        <>

          <Spinner animation="border" role="status" />

        </>

      )}

    </>

  );

};

export default TablePr;

## Хранение данных на клиенте

Листинг хранение данных в localStorage и взаимодействие с ними:

import { makeAutoObservable } from "mobx";

class Cart {

  cart = JSON.parse(localStorage.getItem("cart")) || [];

  length = this.cart.length;

  price = this.cart.reduce((a, b) => a + b.quantity \* b.price, 0);

  constructor() {

    makeAutoObservable(this);

  }

  addProduct(item) {

    if (!this.cart.some((el) => el.id === item.id) || this.cart.length === 0) {

      item.quantity = 1;

      this.cart.push(item);

    } else {

      this.cart.forEach((i) => (i.id === item.id ? (i.quantity += 1) : null));

    }

    localStorage.setItem("cart", JSON.stringify(this.cart));

    this.length = this.cart.length;

    this.price = this.cart.reduce((a, b) => a + b.quantity \* b.price, 0);

  }

  delProduct(id) {

    this.cart = this.cart.filter((item) => item.id !== id);

    localStorage.setItem("cart", JSON.stringify(this.cart));

    this.length = this.cart.length;

    this.price = this.cart.reduce((a, b) => a + b.quantity \* b.price, 0);

  }

  addQuantity(id) {

    this.cart.forEach((item) => {

      if (item.id === id) {

        return (item.quantity += 1);

      }

    });

    localStorage.setItem("cart", JSON.stringify(this.cart));

    this.price = this.cart.reduce((a, b) => a + b.quantity \* b.price, 0);

  }

  delQuantity(id) {

    this.cart.forEach((item) => {

      if (item.id === id) {

        if (item.quantity <= 1) {

          return null;

        }

        item.quantity -= 1;

      }

    });

    localStorage.setItem("cart", JSON.stringify(this.cart));

    this.price = this.cart.reduce((a, b) => a + b.quantity \* b.price, 0);

  }

  addOrder() {

    localStorage.removeItem('cart')

    this.cart = [];

    this.length = this.cart.length;

    this.price = 0;

  }

}

export default new Cart();

## Добавление заказа

Добавление заказа происходит анологичным образом с добавление товара или добавлением категории или добавление нового пользователя. Аргументы которые приходят это список товаров, id пользователя , колличество каждого товаратоваров и общая цена.

# Разработать административный интерфейс приложения

## Страница « Администратор»

Листинг:

import React, { Component, useEffect } from 'react'

import { useParams } from 'react-router-dom';

import 'bootstrap'

import Header from '../components/Header';

import Slider from '../components/Slider';

import { Container } from 'react-bootstrap';

import AddCat from '../components/addCat';

import { Tab, Tabs } from 'react-bootstrap';

import CardCate from '../components/CardCate';

import AddPr from '../components/addPr';

import CardPrAll from '../components/CardPrAll';

import TableOrd from '../components/TadleOr';

const Admin =()=> {

      return(

          <div>

              <Container style={{  padding:" 0% 0% 20% 0% "}}>

                <Tabs

                defaultActiveKey="home"

                transition={false}

                id="noanim-tab-example"

                className="mb-3"

                >

                <Tab eventKey="home" title="Заказы">

                  <TableOrd/>

                </Tab>

                <Tab eventKey="profile" title="Категории">

                <div style={{ textAlign:"right"}}>  <AddCat/></div>

                <CardCate/>

                </Tab>

                <Tab eventKey="contact" title="Товары" >

                <div style={{ textAlign:"right"}}>  <AddPr/></div>

                <CardPrAll/>

                </Tab>

                </Tabs>

                <div  style={{padding:" 0% 0% 10% 0% ", color:"white"}}>  dfvdfgdtd</div>

              </Container>

             </div>

      );

}

export default Admin

Листинг отображение заказов:

import React, { useEffect, useState } from "react";

import {

  Form,

  FormGroup,

  Modal,

  Spinner,

  Table,

} from "react-bootstrap";

import {Button} from "reactstrap"

import axios from 'axios';

const TableOrd = () => {

  const [data, setData] = useState([]);

  const [loading, setLoading] = useState(true);

  const fetchGetOrder = async () => {

    axios

      .get(`http://localhost:8080/order/admin`, {

        headers: {

          "Content-Type": "application/json",

        },

      })

      .then((response) => {

        console.log(response.data.order)

        setData(response.data.order);

        setSortData(response.data.order);

      })

      .catch((err) => {

        console.log(err);

      });

  };

  useEffect(() => {

    fetchGetOrder();

    // eslint-disable-next-line react-hooks/exhaustive-deps

  }, []);

  const [sortType, setSortType] = useState("all");

  const [sortData, setSortData] = useState([]);

  const sortArray = (type) => {

    const types = {

      all: "Все",

      new: "Новый",

      success: "Подтверждено",

      cancel: "Отменено",

    };

    const sortProperty = types[type];

    let sorted = [];

    sortProperty === "Все"

      ? (sorted = [...data])

      : (sorted = [...data].filter((a) => a.status === sortProperty));

    setSortData(sorted);

    setLoading(false);

  };

  useEffect(() => {

    sortArray(sortType);

    // eslint-disable-next-line react-hooks/exhaustive-deps

  }, [sortType]);

  //Отказ

  const [show, setShow] = useState(false);

  const [comment, setComment] = useState("");

  const fetchCanOrder = async (id) => {

    let body\_data = JSON.stringify({ id: id, comment });

    await axios

      .post(`http://localhost:8080/order/update/cancel`, body\_data, {

        headers: {

          "Content-Type": "application/json",

        },

      })

      .then((response) => {

        alert('Вы отменили заказ');

        setData(response.data.values)

        setComment("");

        setShow(false)

      })

      .catch((err) => console.log(err));

  };

  //Подтверждение

  const fetchSucOrder = async (id) => {

    await axios

      .post(`http://localhost:8080/order/update/confirm`, JSON.stringify({ id: id}), {

        headers: {

          "Content-Type": "application/json"

        },

      })

      .then((response) => {

        alert('Вы подтвердили заказ');

        setData(response.data.values)

      })

      .catch((err) => console.log(err));

  };

console.log(sortData.length)

  return (

    <>

      {!loading ? (

        <>

          {sortData.length > 0 ? (

            <>

              <div className="d-flex justify-content-end">

                <Form.Select

                  onChange={(e) => setSortType(e.target.value)}

                  style={{ minWidth: "150px" }}

                  className="w-25"

                >

                  <option value="all">Все заказы</option>

                  <option value="new">Новые</option>

                  <option value="success">Подтвержденные</option>

                  <option value="cancel">Отмененные</option>

                </Form.Select>

              </div>

              <Table className="mt-4">

                <thead>

                  <tr>

                    <th>№</th>

                    <th>Статус</th>

                    <th>ФИО</th>

                    <th>Товары</th>

                    {sortType === "new" ? <th>Действия</th> : <></>}

                  </tr>

                </thead>

                <tbody>

                  {sortData.map((item) => (

                    <tr key={item.id}>

                      <td>{item.id}</td>

                      <td>{item.status}</td>

                      <td>{item.FIO || ""}</td>

                      <td>

                        {item.products.map((i) => (

                          <div key={i.id}>

                            {i.name} - {i.quantity}

                          </div>

                        ))}

                      </td>

                      {sortType === "new" ? (

                        <td>

                          <Button className="ms-4"

                            style={{ borderRadius:"30px", background:"rgba(66, 222, 41, 0.53)", color:"black", borderColor:"rgba(66, 222, 41, 0.53)"}}

                            onClick={() => fetchSucOrder(item.id)}

                          >

                            Подтвердить

                          </Button>

                          <Button

                          style={{ borderRadius:"30px", background:"rgba(255, 21, 21, 0.48)", color:"black", borderColor:"rgba(255, 21, 21, 0.48)"}}

                            onClick={() => setShow(true)}

                          >

                            Отменить

                          </Button>

                        </td>

                      ) : (

                        <></>

                      )}

                      <Modal centered show={show} onHide={() => setShow(false)}>

                        <Modal.Header closeButton>Отменить заказ</Modal.Header>

                        <Modal.Body>

                          <FormGroup>

                            <Form.Label>Введить причину отказа</Form.Label>

                            <Form.Control

                              value={comment}

                              onChange={(e) => setComment(e.target.value)}

                              as="textarea"

                            ></Form.Control>

                          </FormGroup>

                        </Modal.Body>

                        <Modal.Footer>

                          <div className="text-end">

                            <Button

                             style={{ borderRadius:"30px", background:"rgba(255, 21, 21, 0.48)", color:"black",  borderColor:"rgba(255, 21, 21, 0.48)"}}

                              onClick= { () =>setShow(false)}

                            >

                              Отменить действие

                            </Button>

                            <Button

                           style={{ borderRadius:"30px", background:"rgba(66, 222, 41, 0.53)", color:"black",  borderColor:"rgba(66, 222, 41, 0.53)"}}

                              onClick={() => fetchCanOrder(item.id)}

                            >

                              Отменить заказ

                            </Button>

                          </div>

                        </Modal.Footer>

                      </Modal>

                    </tr>

                  ))}

                </tbody>

              </Table>

            </>

          ) : (

            <h3>Нет заказов</h3>

          )}

        </>

      ) : (

        <>

          <Spinner animation="border" role="status" />

        </>

      )}

    </>

  );

};

export default TableOrd;

Листинг отображение категорий аналогичен листингу станицы «Каталог» в еомпаненте карточка.

Листинг отображение товары аналогичен листингу станицы «Товары» в компаненте карточка.

## Форма добавление категории

Листинг кода на клиенте :

import React, { useState }  from 'react';

import { Alert } from 'react-bootstrap';

import {Modal, ModalHeader, ModalBody, ModalFooter, Label, FormFeedback, Form, FormGroup, Button, Input} from 'reactstrap'

const AddCat =  ( idUser) =>  {

    const [opened, setOpened] = useState();

    const [name, setName]= useState("");

    const [price, setPrice]= useState("");

    const [file, setFile]= useState(null);

    const [nameInvalid, setNameInvalid]= useState(false);

    const [priceInvalid, setPriceInvalid]= useState(false);

    const [fileInvalid, setFileInvalid] = useState(false);

    const [error, setError]= useState(null);

    const toggle = ()=> setOpened((value)=> !value);

    const id = idUser.data;

    const onAdd = async () => {

        setNameInvalid(false);

        setPriceInvalid(false);

        setFileInvalid(false);

        setError(null);

        if( !name || !price || !file){

            if(!name) setNameInvalid(true);

            if(!price) setPriceInvalid(true);

            if(!file) setFileInvalid(true);

            alert("ошибка ввода" );

            return;

        }

       const formData = new FormData();

       formData.append("file", file);

 console.log(file)

       const result = await fetch("http://localhost:8080/file/upload",{

           method:"POST",

           body: formData

       });

       const resultJSON = await result.json();

             console.log(resultJSON);

       const fileId = resultJSON.fileId;

        await fetch(`http://localhost:8080/category/add`,{

        method:"POST",

        headers:{"content-Type": "application/json"},

        body: JSON.stringify({ name, price,  img: fileId }),

        });

        toggle();

        window.location.reload();

    }

    const handleOnChange = (event) =>{

        setName(event.target.value)

    }

    const handleOnChange1 = (event) =>{

        setPrice(event.target.value)

     }

    const handleOnChange2 = (event) =>{

        setFile(event.target.files[0])

    }

      return (

            <>

            {error &&(

                <div>

                    <Alert color='danger' toggle={()=> setError(null)}>

                        {error}

                    </Alert>

                    </div>

                    )}

                 <Button  onClick={toggle}   style={{ margin:"1%", fontSize: "23px",  borderColor:" #FE9BAD", textAlign:"right" ,

                    borderRadius:"30px", background:"#FE9BAD"}}>

                    Добавить категорию

                    </Button>

                    <Modal isOpen={opened} toggle={toggle}>

                    <ModalHeader toggle={toggle}>Добавить новую категорию</ModalHeader>

                    <ModalBody>

                        <Form>

                        <FormGroup >

                            <Label>Название:</Label>

                            <Input

                             type="text"

                             placeholder="Введите название"

                             value={name}

                            onChange={(event)=>handleOnChange(event)}

                            invalid={nameInvalid}

                               />

                            <FormFeedback>Обязательное поле</FormFeedback>

                        </FormGroup>

                        <FormGroup >

                            <Label for="exampleEmail">ЦЕна за единицу товара</Label>

                            <Input

                            type="text"

                            placeholder="Введите цену"

                            value={price}

                            onChange={(event)=>handleOnChange1(event)}

                            invalid={priceInvalid}

                             />

                            <FormFeedback>Обязательное поле</FormFeedback>

                        </FormGroup>

                        <FormGroup >

                            <Label for="examplePassword">Фото:</Label>

                            <Input

                            type="file"

                            placeholder="Загрузите файл"

                            id = "file"

                            onChange={(event)=>handleOnChange2(event)}

                            invalid={fileInvalid}  />

                        <FormFeedback>Обязательное поле</FormFeedback>

                        </FormGroup>

                        </Form>

                        </ModalBody>

                        <ModalFooter>

                        <Button style={{ borderRadius:"30px", background:"rgba(255, 21, 21, 0.48)", color:"black",  borderColor:"rgba(255, 21, 21, 0.48)"}}  type="submit" onClick={toggle}>Отмена</Button>

                       <Button style={{ borderRadius:"30px", background:"rgba(66, 222, 41, 0.53)", color:"black",  borderColor:"rgba(66, 222, 41, 0.53)"}}  type="submit" onClick={ onAdd}>Добавить</Button>

                        </ModalFooter>

                        </Modal>

            </>

        )

}

export default AddCat

Листинг кода на сервере:

router.post("/add", (req, res)=>{

    // console.log(req.files.img)

    const {name, price, img} = req.body;

    if (!name || !price || !img){

        console.log(177)

        res.send({

            success: false

        });

        return;

    }

    let category=db.getData("/category");

    if(!db.getData(`/files/${img}`)){

        res.send({ success:false});

        return;

    }

    category.push({

        id: uuid(),

        name,

        price,

        img

    });

    db.push("/category", category)

    res.send({success: true, category });

});

## Форма добавление товара

Листин кода будет почти таким же как и в форме добавление категории только ещё дабавится поле «Описание».

Листинг кода на сервере:

router.post("/add", (req, res)=>{

    const{name, category, price, description, img} = req.body;

    console.log(name, category, price, description, img)

//  const {img} = req.files

    if (!name || !category || !description || !price  || !img){

        console.log(177)

        res.send({

            success: false

        });

        return;

    }

    let products=db.getData("/product");

    if(!db.getData(`/files/${img}`)){

        res.send({ success:false});

        return;

    }

    products.push({

        id: uuid(),

        name,

        category,

        price,

        description,

        img

    });

    db.push("/product", products)

    res.send({success: true, products });

});

## Добавление и вывода файла(фото) на сервере

Листинг:

const path = require("path");

const express = require("express");

const fileUploads = require("express-fileupload");

const uuid = require("uuid").v4

const db = require("../db");

const router = express.Router();

router.post(

    "/upload",

    fileUploads({tempFileDir: true}),

    (req, res) => {

        if(!req.files.file){

            res.send({success: false});

            return;

        }

        const fileId=uuid()+ ".jpg";

        const filePath = path.join(

            \_\_dirname,

            "..",

            "/static",

            fileId

        );

        req.files.file.mv(filePath);

        const mimetype = req.files.file.mimetype;

        db.push(`/files/${fileId}`, {mimetype, filePath});

        res.send({success: true, fileId})

    }

);

router.get("/:fileId", (req,res) =>{

    const fileId = req.params.fileId;

    const file=  db.getData(`/files/${fileId}`);

    if (!file){

        res.sendStatus(404);

        return;

    }

    res.setHeader("Content-Type", file.mimetype)

    res.sendFile(file.filePath);

})

module.exports = router;

## Вывод каких-либо данных с сервера

Листинг- пример вывода всех категорий:

router.get("/all", (req, res) =>{

    let { limit, offset} = req.query;

    if (limit) limit=+limit; else limit = 20;

    if (offset) offset = +offset; else offset=0;

   let category=db.getData("/category");

        res.send({

            success: true,

            category: category.slice(offset, offset+limit),

            limit,

            offset,

            count: category.length

        });

});

Листинг- пример вывода всех товар с условием категория:

router.get("/all/:name", (req, res) =>{

    const nameCategory = req.params.name;

    // console.log(nameCategory);

   let products=db.getData("/product");

//    console.log(products)

    products = products.filter(el=> el.category == nameCategory);

        // console.log(products);

        res.send({

            success: true,

            products: products,

            count: products.length

        });

})

Результат работы всех форм представлен в Приложение А.

# Список литературы

1. Node.js в действии / Кантелон М., Хартер М., Головайчук T., Райлих Н., - 2-е изд. - СПб.: Питер, 2018
2. Стоян Стефанов React.js. Быстрый старт. Питер, 2017.
3. Браун Итан Веб-разработка с применением Node и Express. Полноценное использование стека JavaScript. — СПб.: Питер, 2017.
4. React и Redux: функциональная веб-разработка/Алекс Бэнкс, Ева Порселло. Питер, 2018.
5. Используем localStorage в React [Электронный ресурс] – Режим доступа: https://bookflow.ru/ispolzuem-localstorage-v-react/
6. Использование localStorageс React Hooks [Электронный ресурс] – Режим доступа: https://blog.logrocket.com/using-localstorage-react-hooks/

# ПРИЛОЖЕНИЕ А

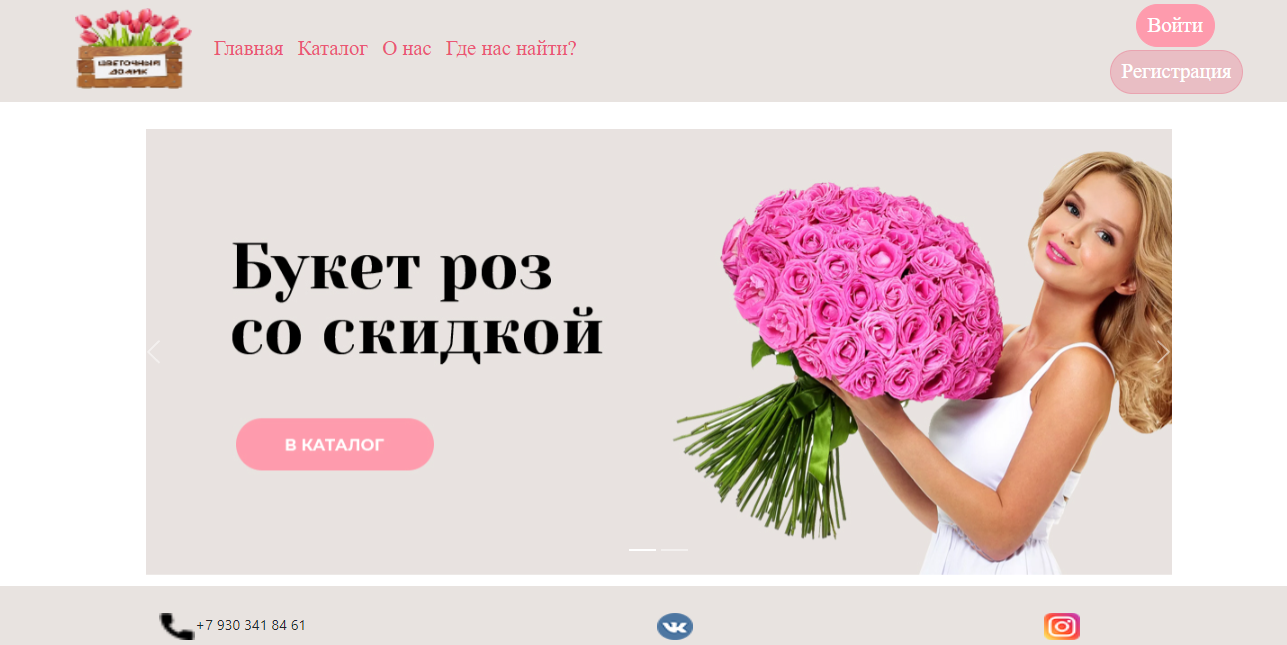


Рис.1. – Страница до авторизации

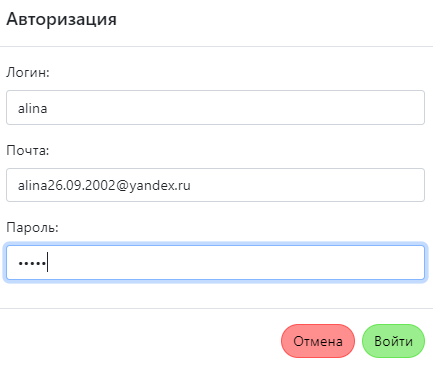


Рис.2. – Форма авторизации

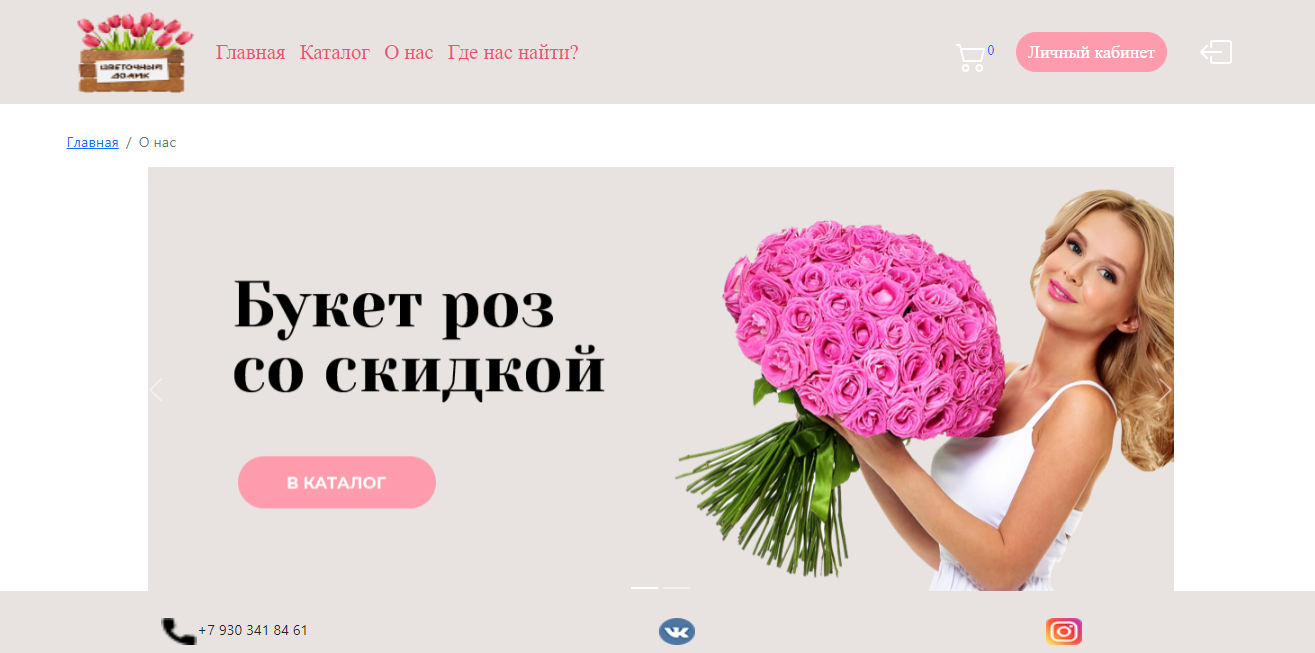


Рис.3. – Страница после аторизации обычного пользователя (user)

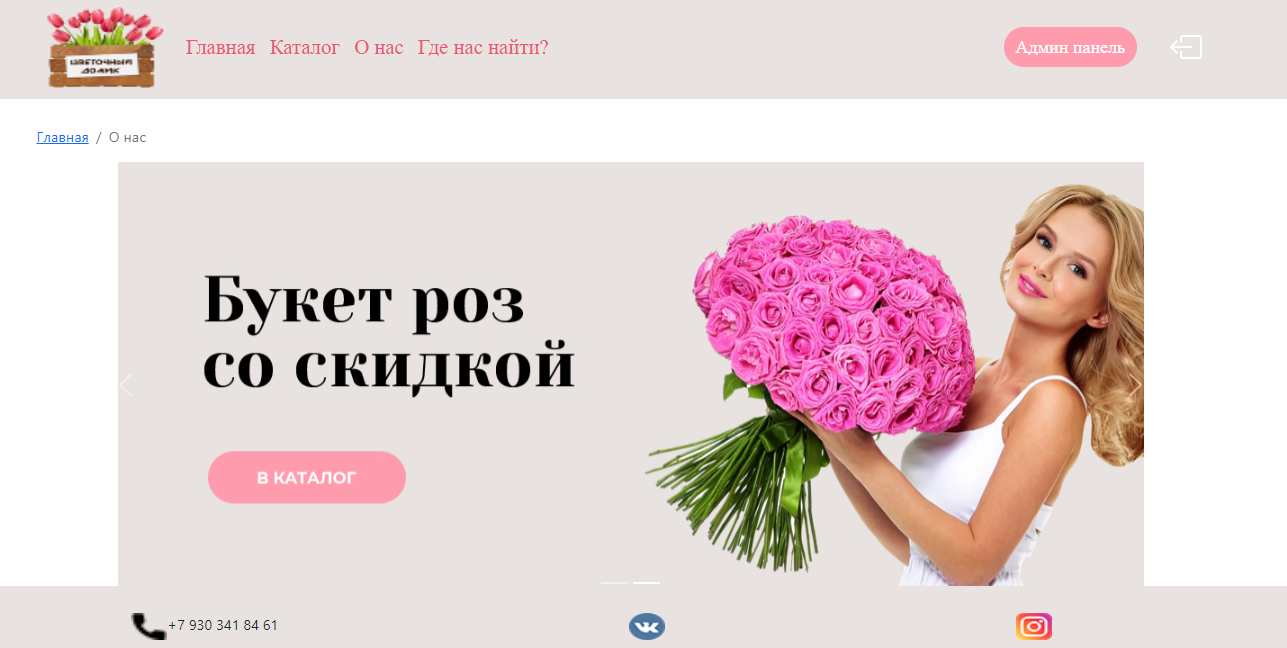


Рис.4. – Страница после аторизации администратора (admin)

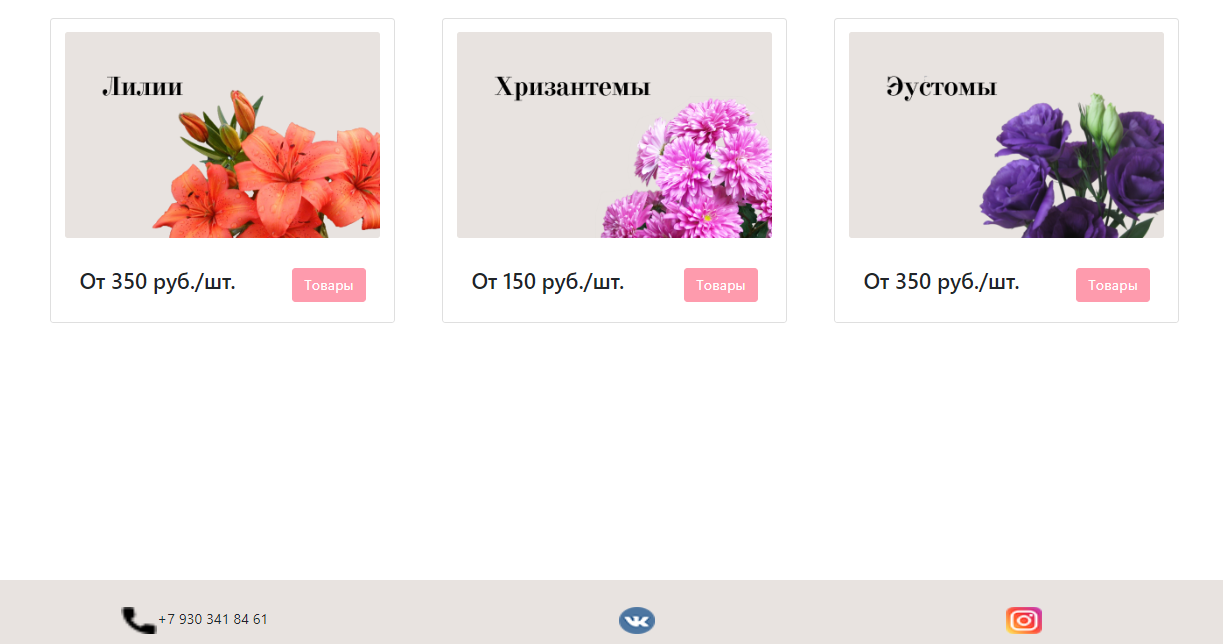


Рис.5. – Страница до добавление категории

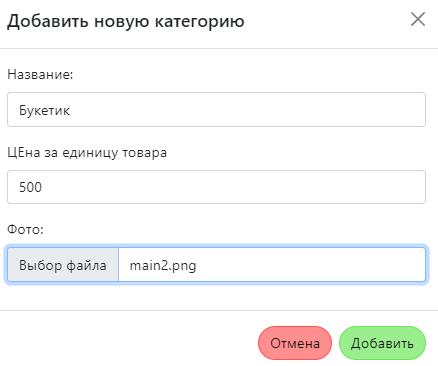


Рис.6. – Форма добавления категории

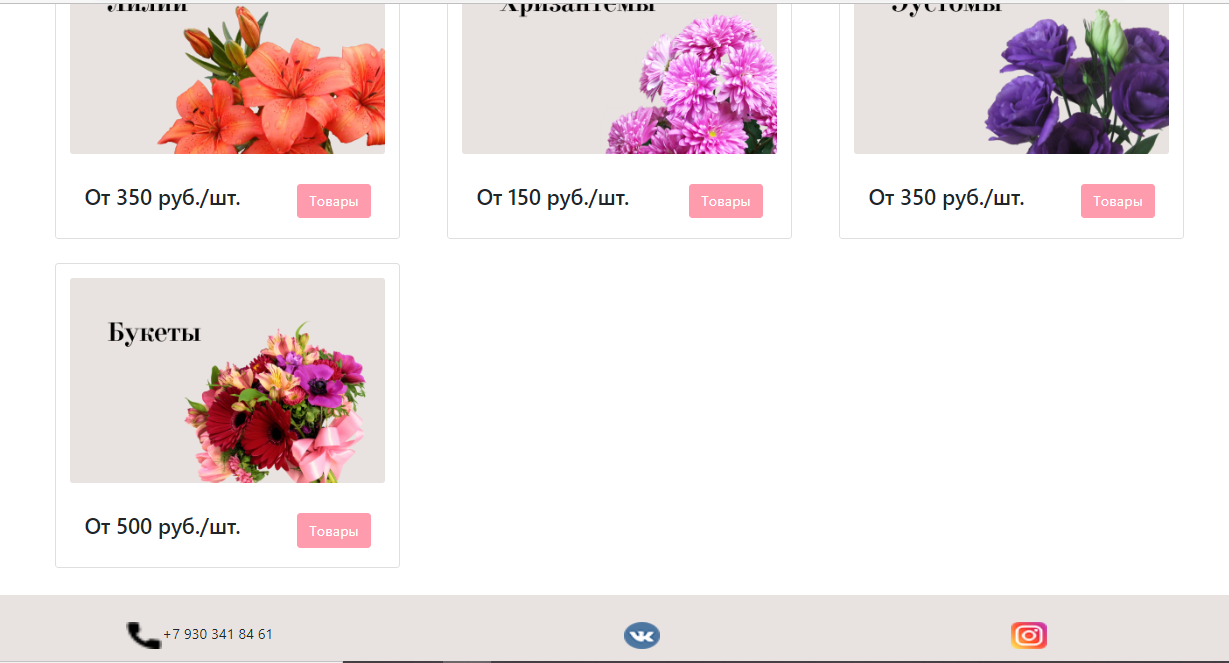


Рис.7. – Страница после добавление категории

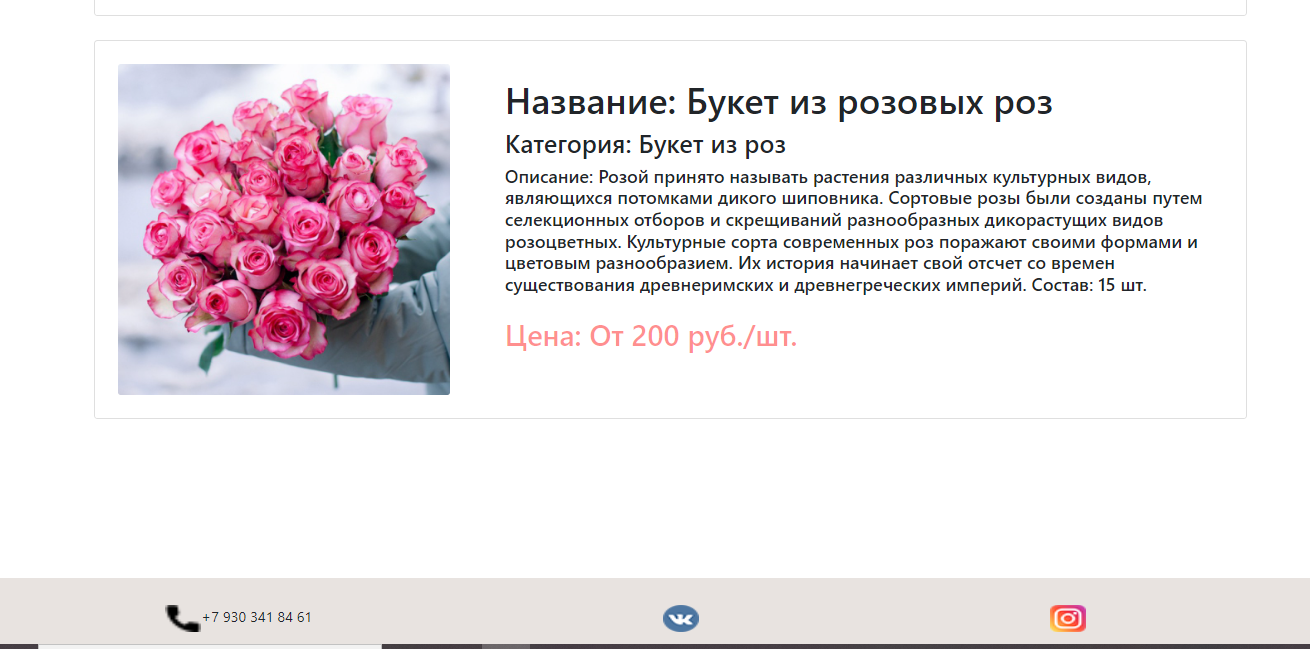


Рис.8. – Страница до добавление товара

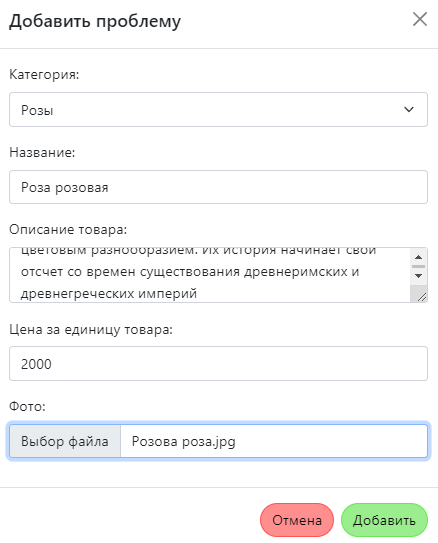


Рис.9. – Форма добавления товара

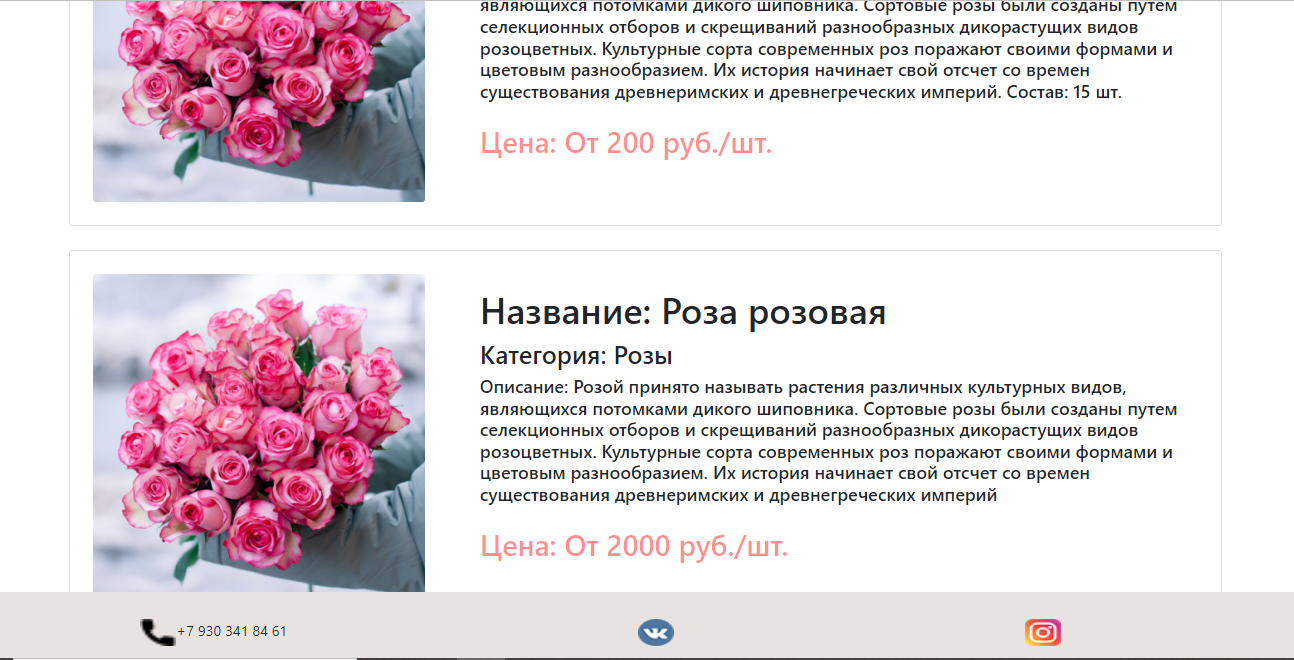


Рис.10. – Страница после добавление товара