**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук  
Департамент программной инженерии

|  |  |
| --- | --- |
| СОГЛАСОВАНО  Заместитель декана по учебно-методической работе  доцент департамента больших данных и информационного поиска факультета компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И. Ю. Самоненко «\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2020 г. | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия» профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. В. Шилов «\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2020 г. |

|  |  |
| --- | --- |
| ***Подп. и дата*** |  |
| ***Инв. № дубл.*** |  |
| ***Взам. инв. №*** |  |
| ***Подп. и дата*** |  |
| ***Инв. № подл*** | RU.17701729.04.01-01 12 01-1-ЛУ |

**WEB ПРИЛОЖЕНИЕ ДЛЯ СОЗДАНИЯ ГЕНЕАЛОГИЧЕСКОГО ДРЕВА**

**Текст программы**

**ЛИСТ УТВЕРЖДЕНИЯ**

**RU.17701729.04.01-01 12 01-1-ЛУ**

Исполнитель  
студент группы БПИ173  
\_\_\_\_\_\_\_\_\_\_ / Переплетчиков А. И. /  
«\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2020 г.

**Москва 2020**

УТВЕРЖДЕНRU.17701729.04.01-01 12 01-1-ЛУ

|  |  |
| --- | --- |
| Подп. и дата |  |
| Инв. № дубл. |  |
| Взам. инв. № |  |
| Подп. и дата |  |
| Инв. № подл | RU.17701729.04.01-01 12 01-1 |

**WEB ПРИЛОЖЕНИЕ ДЛЯ СОЗДАНИЯ ГЕНЕАЛОГИЧЕСКОГО ДРЕВА**

**Текст программы**

**RU.17701729.04.01-01 12 01-1**

**Листов 28**

**Москва 2020**

**Содержание**

**[1. Текст программы](#_Toc37605255)** [3](#_Toc37605255)

[**1.1. Компонент AuthProvider** 3](#_Toc37605256)

[**1.2. Компонент Routes** 3](#_Toc37605257)

[**1.3. Компонент Add** 4](#_Toc37605258)

[**1.4. Компонент Info** 10](#_Toc37605259)

[**1.5. Компонент Login** 11](#_Toc37605260)

[**1.6. Компонент Main** 12](#_Toc37605261)

[**1.7. Компонент SignUp** 13](#_Toc37605262)

[**1.8. Компонент Loading** 19](#_Toc37605263)

[**1.9. Компонент Menu** 19](#_Toc37605264)

[**1.10. Компонент Navbar** 20](#_Toc37605265)

[**1.11. Компонент EditPerson** 20](#_Toc37605266)

[**1.12. Компонент PersonPage** 26](#_Toc37605267)

[**1.13. Компонент PersonCard** 28](#_Toc37605268)

[**1.14. Компонент SelectDate** 32](#_Toc37605269)

[**1.15. Компонент Tree** 35](#_Toc37605270)

# **1. Текст программы**

## **1.1. Компонент AuthProvider**

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

import app from '../base';

export const AuthContext = React.createContext();

export const AuthProvider = ({ children }) => {

  const [currentUser, setCurrentUser] = useState(null);

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

  useEffect(() => {

    app.auth().onAuthStateChanged(user => {

      setCurrentUser(user);

      setLoading(false);

    });

  }, []);

  return (

    <AuthContext.Provider value={{ currentUser, loading }}>

      {children}

    </AuthContext.Provider>

  );

};

## **1.2. Компонент Routes**

import React, { useContext } from 'react';

import { Switch, Route, Redirect } from 'react-router-dom';

import { AuthContext } from './context/Auth';

import { Navbar } from './components/Navbar';

import { Menu } from './components/Menu';

import { Login } from './pages/Login';

import { SignUp } from './pages/SignUp';

import { Main } from './pages/Main';

import { Add } from './pages/Add';

import { Loading } from './pages/Loading';

import { Info } from './pages/Info';

export const Routes = () => {

  const { currentUser, loading } = useContext(AuthContext);

  if (loading) {

    return (

      <div>

        <Loading />

      </div>

    );

  }

  if (!currentUser) {

    return (

      <div>

        <Switch>

          <Route exact path="/login" component={Login} />

          <Route exact path="/signup" component={SignUp} />

          <Route exact path="/info" component={Info} />

          <Redirect to="/login" />

        </Switch>

        <Navbar />

      </div>

    );

  }

  return (

    <div>

      <Switch>

        <Route exact path="/">

          <Menu currentPage={'/'} />

          <Main />

        </Route>

        <Route exact path="/add">

          <Menu currentPage={'/add'} />

          <Add />

        </Route>

        <Route exact path="/info">

          <Menu currentPage={'/info'} />

          <Info />

        </Route>

        <Redirect to="/" />

      </Switch>

    </div>

  );

};

## **1.3. Компонент Add**

import React, { useState } from 'react';

import { SelectDate } from '../components/SelectDate';

import { Main } from './Main';

export const Add = ({ person }) => {

  const [candidate, setCandidate] = useState(null);

  const [firstName, setFirstName] = useState(person ? person.firstName : '');

  const [secondName, setSecondName] = useState(person ? person.secondName : '');

  const [lastName, setLastName] = useState(person ? person.lastName : '');

  const [birthday, setBirthday] = useState(person ? person.birthday : '');

  const [birthmonth, setBirthmonth] = useState(person ? person.birthmonth : '');

  const [birthyear, setBirthyear] = useState(person ? person.birthyear : '');

  const [deathday, setDeathday] = useState(person ? person.deathday : '');

  const [deathmonth, setDeathmonth] = useState(person ? person.deathmonth : '');

  const [deathyear, setDeathyear] = useState(person ? person.deathyear : '');

  const [nativeCity, setNativeCity] = useState(person ? person.nativeCity : '');

  const [sex, setSex] = useState(person ? person.sex : '');

  const [bio, setBio] = useState(person ? person.bio : '');

  const handleInput = (event) => {

    const target = event.target;

    const value = target.value;

    const name = target.name;

    switch (name) {

      case 'firstName':

        setFirstName(value);

        break;

      case 'secondName':

        setSecondName(value);

        break;

      case 'lastName':

        setLastName(value);

        break;

      case 'birthday':

        setBirthday(value);

        break;

      case 'birthmonth':

        setBirthmonth(value);

        break;

      case 'birthyear':

        setBirthyear(value);

        break;

      case 'deathday':

        setDeathday(value);

        break;

      case 'deathmonth':

        setDeathmonth(value);

        break;

      case 'deathyear':

        setDeathyear(value);

        break;

      case 'nativeCity':

        setNativeCity(value);

        break;

      case 'sex':

        setSex(value);

        break;

      case 'bio':

        setBio(value);

        break;

      default:

        console.log('Strange feature detected');

        break;

    }

  };

  const handleAdd = (event) => {

    event.preventDefault();

    const {

      firstName,

      secondName,

      lastName,

      birthday,

      birthmonth,

      birthyear,

      deathday,

      deathmonth,

      deathyear,

      nativeCity,

      sex,

      bio,

    } = event.target.elements;

    setCandidate({

      firstName: firstName.value,

      secondName: secondName.value,

      lastName: lastName.value,

      birthDate: {

        day: birthday.value,

        month: birthmonth.value,

        year: birthyear.value,

      },

      deathDate: {

        day: deathday.value,

        month: deathmonth.value,

        year: deathyear.value,

      },

      nativeCity: nativeCity.value,

      sex: sex.value,

      bio: bio.value,

    });

  };

  if (candidate) {

    return (

      <Main returnFunction={() => setCandidate(null)} newPerson={candidate} />

    );

  }

  return (

    <div className="form-wrapper">

      <form className="add" onSubmit={handleAdd} autoComplete={false}>

        <div className="form-row">

          <div className="col form-group">

            <label htmlFor="lastName">Фамилия</label>

            <input

              type="text"

              name="lastName"

              id="lastName"

              className="form-control"

              placeholder="Иванов"

              value={lastName}

              onChange={handleInput}

              required

            />

          </div>

          <div className="col form-group">

            <label htmlFor="firstName">Имя</label>

            <input

              type="text"

              name="firstName"

              id="firstName"

              className="form-control"

              placeholder="Иван"

              value={firstName}

              onChange={handleInput}

              required

            />

          </div>

          <div className="col form-group">

            <label htmlFor="secondName">Отчество \*</label>

            <input

              type="text"

              name="secondName"

              id="secondName"

              className="form-control"

              placeholder="Иванович"

              value={secondName}

              onChange={handleInput}

            />

          </div>

        </div>

        <div className="form-row">

          <div className="col-6 form-group">

            <label htmlFor="birthday">Дата рождения \*</label>

            <SelectDate

              personDay={birthday}

              personMonth={birthmonth}

              personYear={birthyear}

              required={false}

              typeOfDate="birth"

            />

          </div>

          <div className="col-6 form-group">

            <label htmlFor="deathday">Дата смерти \*</label>

            <SelectDate

              personDay={deathday}

              personMonth={deathmonth}

              personYear={deathyear}

              required={false}

              typeOfDate="death"

            />

          </div>

        </div>

        <div className="form-row">

          <div className="col-6 form-group">

            <label htmlFor="nativeCity">Место рождения \*</label>

            <input

              type="text"

              className="form-control"

              name="nativeCity"

              id="nativeCity"

              placeholder="Город"

              value={nativeCity}

              onChange={handleInput}

            />

          </div>

          <div className="col-1 form-group">

            <label htmlFor="sex">Пол</label>

            <ul className="sex-choose">

              <li>

                <input

                  type="radio"

                  value="Мужской"

                  name="sex"

                  id="male"

                  onChange={handleInput}

                  defaultChecked

                />

                <label className="form-check-label" htmlFor="male">

                  <i className="fa fa-mars" aria-hidden="true"></i>

                </label>

              </li>

              <li>

                <input

                  type="radio"

                  value="Женский"

                  name="sex"

                  id="female"

                  onChange={handleInput}

                />

                <label className="form-check-label" htmlFor="female">

                  <i className="fa fa-venus" aria-hidden="true"></i>

                </label>

              </li>

            </ul>

          </div>

          <div className="col-5 form-group">

            <label htmlFor="photo" style={{ color: 'grey' }}>

              Фотография (временно не работает)

            </label>

            <label className="form-control" style={{ color: 'grey' }}>

              Выбрать файл

              <input name="photo" type="file" hidden disabled />

            </label>

          </div>

        </div>

        <div className="form-group">

          <label htmlFor="bio">Биография \*</label>

          <textarea

            className="form-control"

            name="bio"

            placeholder="Места обучения, профессиональные интересы, интересные факты из жизни и т.д."

            value={bio}

            onChange={handleInput}

          ></textarea>

        </div>

        <div className="mt-4">

          <button type="submit" className="btn btn-submit">

            Добавить

          </button>

          <small className="ml-3 hint">

            \* Данные поля необязательны к заполнению

          </small>

        </div>

      </form>

    </div>

  );

};

## **1.4. Компонент Info**

import React, { useContext } from 'react';

import image from '../img/info-tree.png';

import { AuthContext } from '../context/Auth';

import '../style/forms.scss';

export const Info = () => {

  const { currentUser } = useContext(AuthContext);

  return (

    <div className="position-absolute h-100 w-100 d-flex justify-content-center align-items-center">

      <div className="container">

        <div className="row">

          <div className="col-7 d-flex align-items-center">

            <div>

              <p>

                <span style={{ fontWeight: 900, color: '#9733be' }}>

                  Predki.io

                </span>{' '}

                - это удобный онлайн-инструмент для создания и управления

                родословной. Добавьте своих родственников на генеалогическое

                древо, заполните информацию о них и напишите историю своей

                семьи!

              </p>

              {!currentUser ? (

                <div className="mt-3">

                  <a href="/login" className="btn btn-submit">

                    Войти

                  </a>

                  <small className="ml-3">

                    <a className="hint" href="/signup">

                      Зарегистрироваться

                    </a>

                  </small>

                </div>

              ) : null}

            </div>

          </div>

          <div className="col-5 d-flex align-items-center">

            <img className="w-75" src={image} alt="" />

          </div>

        </div>

      </div>

    </div>

  );

};

## **1.5. Компонент Login**

import React, { useCallback } from 'react';

import '../style/forms.scss';

import app from '../base';

export const Login = ({ history }) => {

  const handleLogin = useCallback(

    async event => {

      event.preventDefault();

      const { email, password } = event.target.elements;

      try {

        await app

          .auth()

          .signInWithEmailAndPassword(email.value, password.value);

        history.push('/');

      } catch (error) {

        alert(error);

      }

    },

    [history]

  );

  return (

    <div className="form-wrapper">

      <form className="login" onSubmit={handleLogin}>

        <div className="form-group">

          <label htmlFor="email">Email</label>

          <input

            type="email"

            name="email"

            className="form-control"

            placeholder="you@email.com"

            required

          />

        </div>

        <div className="form-group">

          <label htmlFor="password">Пароль</label>

          <input

            type="password"

            name="password"

            className="form-control"

            placeholder="\*\*\*\*\*\*\*\*"

            required

          />

        </div>

        <div className="mt-4">

          <button type="submit" className="btn btn-submit">

            Войти

          </button>

          <small className="ml-3">

            <a className="hint" href="/signup">

              Еще нет аккаунта?

            </a>

          </small>

        </div>

      </form>

    </div>

  );

};

## **1.6. Компонент Main**

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

import { Tree } from '../components/Tree';

import app from '../base';

import { AuthContext } from '../context/Auth';

import { Loading } from '../pages/Loading';

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

import add from '../addMethod';

export const Main = ({ returnFunction, newPerson }) => {

  const [people, setPeople] = useState(null);

  const [families, setFamilies] = useState(null);

  const history = useHistory();

  const { currentUser } = useContext(AuthContext);

  // Fetching data

  useEffect(() => {

    const fetchData = async () => {

      const db = app.firestore();

      const peopleDocs = await db

        .collection('users')

        .doc(currentUser.uid)

        .collection('people')

        .get();

      const familiesDocs = await db

        .collection('users')

        .doc(currentUser.uid)

        .collection('families')

        .get();

      setPeople(peopleDocs);

      setFamilies(familiesDocs);

    };

    fetchData();

  }, [currentUser.uid]);

  // Adding to tree

  const addToTree = newPerson

    ? async (id, relationship) => {

        const db = app.firestore();

        await add(

          newPerson,

          id,

          relationship,

          db.collection('users').doc(currentUser.uid).collection('people'),

          db.collection('users').doc(currentUser.uid).collection('families')

        );

        history.push('/');

      }

    : null;

  if (people && families) {

    return (

      <Tree

        people={people}

        families={families}

        candidate={newPerson}

        sendRelative={addToTree}

        returnFunction={returnFunction}

      />

    );

  } else {

    return <Loading />;

  }

};

## **1.7. Компонент SignUp**

import React, { useCallback } from 'react';

import app from '../base';

import '../style/forms.scss';

import { SelectDate } from '../components/SelectDate';

export const SignUp = ({ history }) => {

  const handleSignUp = useCallback(

    async (event) => {

      event.preventDefault();

      const {

        email,

        password,

        confirmPassword,

        firstName,

        secondName,

        lastName,

        birthday,

        birthmonth,

        birthyear,

        nativeCity,

        sex,

      } = event.target.elements;

      if (password.value !== confirmPassword.value) {

        document

          .getElementById('confirmPassword')

          .setCustomValidity('Пароли не совпадают');

      } else {

        try {

          await app

            .auth()

            .createUserWithEmailAndPassword(email.value, password.value)

            .then((resp) => {

              return app

                .firestore()

                .collection('users')

                .doc(resp.user.uid)

                .set({

                  firstName: firstName.value,

                  secondName: secondName.value,

                  lastName: lastName.value,

                  birthDate: {

                    day: birthday.value,

                    month: birthmonth.value,

                    year: birthyear.value,

                  },

                  nativeCity: nativeCity.value,

                  sex: sex.value,

                })

                .then(() => {

                  const newUserId = app

                    .firestore()

                    .collection('users')

                    .doc(resp.user.uid)

                    .collection('people')

                    .doc().id;

                  app

                    .firestore()

                    .collection('users')

                    .doc(resp.user.uid)

                    .collection('people')

                    .doc(newUserId)

                    .set({

                      firstName: firstName.value,

                      secondName: secondName.value,

                      lastName: lastName.value,

                      birthDate: {

                        day: birthday.value,

                        month: birthmonth.value,

                        year: birthyear.value,

                      },

                      deathDate: {

                        day: '',

                        month: '',

                        year: ''

                      },

                      nativeCity: nativeCity.value,

                      sex: sex.value,

                    });

                  const newFamilyId =

                    '\_' +

                    app

                      .firestore()

                      .collection('users')

                      .doc(resp.user.uid)

                      .collection('families')

                      .doc().id;

                  app

                    .firestore()

                    .collection('users')

                    .doc(resp.user.uid)

                    .collection('families')

                    .doc(newFamilyId)

                    .set({

                      [sex.value === 'Мужской' ? 'husband' : 'wife']: newUserId,

                      children: [],

                    });

                });

            });

          history.push('/');

        } catch (error) {

          alert(error);

        }

      }

    },

    [history]

  );

  return (

    <div className="form-wrapper">

      <form className="signup" autoComplete="off" onSubmit={handleSignUp}>

        <div className="form-row">

          <div className="col form-group">

            <label htmlFor="inputEmail">Email</label>

            <input

              type="email"

              name="email"

              className="form-control"

              id="inputEmail"

              placeholder="you@email.com"

              required

            />

          </div>

          <div className="col form-group">

            <label htmlFor="inputPassword">Пароль</label>

            <input

              type="password"

              name="password"

              className="form-control"

              id="inputPassword"

              placeholder="\*\*\*\*\*\*\*\*"

              required

            />

          </div>

          <div className="col form-group">

            <label htmlFor="confirmPassword">Подтвердите пароль</label>

            <input

              type="password"

              name="confirmPassword"

              className="form-control"

              id="confirmPassword"

              placeholder="\*\*\*\*\*\*\*\*"

              required

            />

          </div>

        </div>

        <div className="form-row">

          <div className="col form-group">

            <label htmlFor="lastName">Фамилия</label>

            <input

              type="text"

              name="lastName"

              id="lastName"

              className="form-control"

              placeholder="Иванов"

              required

            />

          </div>

          <div className="col form-group">

            <label htmlFor="firstName">Имя</label>

            <input

              type="text"

              name="firstName"

              id="firstName"

              className="form-control"

              placeholder="Иван"

              required

            />

          </div>

          <div className="col form-group">

            <label htmlFor="secondName">Отчество</label>

            <input

              type="text"

              name="secondName"

              id="secondName"

              className="form-control"

              placeholder="Иванович"

              required

            />

          </div>

        </div>

        <div className="form-row">

          <div className="col-7 form-group">

            <label htmlFor="birthday">Дата рождения</label>

            <SelectDate

              personDay={''}

              personMonth={''}

              personYear={''}

              required={true}

              typeOfDate="birth"

            />

          </div>

          <div className="col-4 form-group">

            <label htmlFor="nativeCity">Место рождения</label>

            <input

              type="text"

              className="form-control"

              id="nativeCity"

              placeholder="Город"

              required

            />

          </div>

          <div className="col-1 form-group">

            <label htmlFor="sex">Пол</label>

            <ul className="sex-choose">

              <li>

                <input

                  type="radio"

                  value="Мужской"

                  name="sex"

                  id="male"

                  defaultChecked

                />

                <label className="form-check-label" htmlFor="male">

                  <i className="fa fa-mars" aria-hidden="true"></i>

                </label>

              </li>

              <li>

                <input type="radio" value="Женский" name="sex" id="female" />

                <label className="form-check-label" htmlFor="female">

                  <i className="fa fa-venus" aria-hidden="true"></i>

                </label>

              </li>

            </ul>

          </div>

        </div>

        <div className="mt-3">

          <button type="submit" className="btn btn-submit">

            Регистрация

          </button>

          <small className="ml-3">

            <a className="hint" href="/login">

              Уже есть аккаунт?

            </a>

          </small>

        </div>

      </form>

    </div>

  );

};

## **1.8. Компонент Loading**

import React from 'react';

export const Loading = () => {

  return (

    <div className="position-absolute h-100 w-100 d-flex justify-content-center align-items-center">

      <div className="spinner-border" role="status">

        <span className="sr-only">Loading...</span>

      </div>

    </div>

  );

};

## **1.9. Компонент Menu**

import React from 'react';

import app from '../base';

import '../style/menu.scss';

export const Menu = ({ currentPage }) => {

  const checkPage = (page) => currentPage === page;

  return (

    <div>

      <div className="menu">

        <a href={checkPage('/add') ? '/' : '/add'}>

          <div className="fixed-part">

            <ion-icon

              name={

                checkPage('/add') ? 'arrow-back-circle-sharp' : 'add-circle'

              }

            ></ion-icon>

          </div>

          <div className="slide-out-part">

            {checkPage('/add') ? 'Назад' : 'Добавить человека'}

          </div>

        </a>

        {/\*<a href="/share">

          <div className="fixed-part">

            <ion-icon name="share-social-sharp"></ion-icon>

          </div>

          <div className="slide-out-part">Поделиться</div>

        </a>

        <a href="/settings">

          <div className="fixed-part">

            <ion-icon name="cog"></ion-icon>

          </div>

          <div className="slide-out-part">Настройки</div>

            </a>\*/}

        <a href="/login" onClick={() => app.auth().signOut()}>

          <div className="fixed-part">

            <ion-icon name="exit"></ion-icon>

          </div>

          <div className="slide-out-part">Выход</div>

        </a>

        <a href={checkPage('/info') ? '/' : '/info'}>

          <div className="fixed-part">

            <ion-icon

              name={

                checkPage('/info')

                  ? 'arrow-back-circle-sharp'

                  : 'information-circle-sharp'

              }

            ></ion-icon>

          </div>

          <div className="slide-out-part">

            {checkPage('/info') ? 'Назад' : 'Информация'}

          </div>

        </a>

        <div className="brand" href="/">

          PREDKI.IO

        </div>

      </div>

      <div className="darker"></div>

    </div>

  );

};

## **1.10. Компонент Navbar**

import React from 'react';

export const Navbar = () => {

  return (

    <nav className="navbar fixed-bottom justify-content-center">

      <a className="navbar-brand" href="/info">

        PREDKI.IO

      </a>

    </nav>

  );

};

## **1.11. Компонент EditPerson**

import React, { useState, useContext } from 'react';

import { SelectDate } from './SelectDate';

import app from '../base';

export const EditPerson = ({ userId, person, handleClose }) => {

  const [firstName, setFirstName] = useState(

    person ? person.data().firstName : ''

  );

  const [secondName, setSecondName] = useState(

    person ? person.data().secondName : ''

  );

  const [lastName, setLastName] = useState(

    person ? person.data().lastName : ''

  );

  const [birthday, setBirthday] = useState(

    person ? person.data().birthDate.day : ''

  );

  const [birthmonth, setBirthmonth] = useState(

    person ? person.data().birthDate.month : ''

  );

  const [birthyear, setBirthyear] = useState(

    person ? person.data().birthDate.year : ''

  );

  const [deathday, setDeathday] = useState(

    person ? person.data().deathDate.day : ''

  );

  const [deathmonth, setDeathmonth] = useState(

    person ? person.data().deathDate.month : ''

  );

  const [deathyear, setDeathyear] = useState(

    person ? person.data().deathDate.year : ''

  );

  const [nativeCity, setNativeCity] = useState(

    person ? person.data().nativeCity : ''

  );

  const [sex, setSex] = useState(person ? person.data().sex : '');

  const [bio, setBio] = useState(person ? person.data().bio : '');

  const handleInput = (event) => {

    const target = event.target;

    const value = target.value;

    const name = target.name;

    switch (name) {

      case 'firstName':

        setFirstName(value);

        break;

      case 'secondName':

        setSecondName(value);

        break;

      case 'lastName':

        setLastName(value);

        break;

      case 'birthday':

        setBirthday(value);

        break;

      case 'birthmonth':

        setBirthmonth(value);

        break;

      case 'birthyear':

        setBirthyear(value);

        break;

      case 'deathday':

        setDeathday(value);

        break;

      case 'deathmonth':

        setDeathmonth(value);

        break;

      case 'deathyear':

        setDeathyear(value);

        break;

      case 'nativeCity':

        setNativeCity(value);

        break;

      case 'sex':

        setSex(value);

        break;

      case 'bio':

        setBio(value);

        break;

      default:

        console.log('Strange feature detected');

        break;

    }

  };

  const handleEdit = async (event) => {

    event.preventDefault();

    const db = app.firestore();

    try {

      await db

        .collection('users')

        .doc(userId)

        .collection('people')

        .doc(person.id)

        .update({

          firstName,

          secondName,

          lastName,

          birthDate: {

            day: event.target.elements.birthday.value,

            month: event.target.elements.birthmonth.value,

            year: event.target.elements.birthyear.value,

          },

          deathDate: {

            day: event.target.elements.deathday.value,

            month: event.target.elements.deathmonth.value,

            year: event.target.elements.deathyear.value,

          },

          sex,

          nativeCity,

          bio,

        });

        window.location.reload();

    } catch (error) {

      alert('Изменение не удалось');

    }

  };

  return (

    <div className="info-page-container">

      <form className="edit" autoComplete={false} onSubmit={handleEdit}>

        <div className="form-row">

          <div className="col d-flex align-items-center">

            <h4 className="mb-0">Редактирование профиля</h4>

          </div>

          <div className="col">

            <button

              type="reset"

              className="btn btn-close"

              aria-label="Close"

              onClick={handleClose}

            >

              &times;

            </button>

          </div>

        </div>

        <hr className="heading" />

        <div className="form-row">

          <div className="col form-group">

            <label htmlFor="lastName">Фамилия</label>

            <input

              type="text"

              name="lastName"

              id="lastName"

              className="form-control"

              placeholder="Иванов"

              value={lastName}

              onChange={handleInput}

              required

            />

          </div>

          <div className="col form-group">

            <label htmlFor="firstName">Имя</label>

            <input

              type="text"

              name="firstName"

              id="firstName"

              className="form-control"

              placeholder="Иван"

              value={firstName}

              onChange={handleInput}

              required

            />

          </div>

          <div className="col form-group">

            <label htmlFor="secondName">Отчество</label>

            <input

              type="text"

              name="secondName"

              id="secondName"

              className="form-control"

              placeholder="Иванович"

              value={secondName}

              onChange={handleInput}

            />

          </div>

        </div>

        <div className="form-row">

          <div className="col-6 form-group">

            <label htmlFor="birthday">Дата рождения</label>

            <SelectDate

              personDay={birthday}

              personMonth={birthmonth}

              personYear={birthyear}

              required={false}

              typeOfDate="birth"

            />

          </div>

          <div className="col-6 form-group">

            <label htmlFor="deathday">Дата смерти</label>

            <SelectDate

              personDay={deathday}

              personMonth={deathmonth}

              personYear={deathyear}

              required={false}

              typeOfDate="death"

            />

          </div>

        </div>

        <div className="form-row">

          <div className="col-6 form-group">

            <label htmlFor="nativeCity">Место рождения</label>

            <input

              type="text"

              className="form-control"

              name="nativeCity"

              id="nativeCity"

              placeholder="Город"

              value={nativeCity}

              onChange={handleInput}

            />

          </div>

          <div className="col-1 form-group">

            <label htmlFor="sex">Пол</label>

            <ul className="sex-choose">

              <li>

                <input

                  type="radio"

                  value="Мужской"

                  name="sex"

                  id="male"

                  onChange={handleInput}

                  checked={sex === 'Мужской'}

                />

                <label className="form-check-label" htmlFor="male">

                  <i className="fa fa-mars" aria-hidden="true"></i>

                </label>

              </li>

              <li>

                <input

                  type="radio"

                  value="Женский"

                  name="sex"

                  id="female"

                  checked={sex === 'Женский'}

                  onChange={handleInput}

                />

                <label className="form-check-label" htmlFor="female">

                  <i className="fa fa-venus" aria-hidden="true"></i>

                </label>

              </li>

            </ul>

          </div>

          <div className="col-5 form-group">

            <label htmlFor="photo" style={{color: 'grey'}}>Фотография (временно не работает)</label>

            <label className="form-control" style={{color: 'grey'}}>

              Выбрать файл

              <input name="photo" type="file" hidden disabled />

            </label>

          </div>

        </div>

        <div className="form-group">

          <label htmlFor="bio">Биография</label>

          <textarea

            className="form-control"

            name="bio"

            placeholder="Места обучения, профессиональные интересы, интересные факты из жизни и т.д."

            value={bio}

            onChange={handleInput}

          ></textarea>

        </div>

        <div className="mt-4">

          <button type="submit" className="btn btn-submit">

            Изменить

          </button>

        </div>

      </form>

    </div>

  );

};

## **1.12. Компонент PersonPage**

import React from 'react';

import nouser from '../img/nouser.png';

export const PersonPage = ({ person, clickOutside }) => {

  console.log(person.id);

  const personData = person.data();

  const name =

    personData.firstName +

    ' ' +

    personData.secondName +

    ' ' +

    personData.lastName;

  const birthDate = personData.birthDate

    ? personData.birthDate.day +

      ' ' +

      personData.birthDate.month +

      ' ' +

      personData.birthDate.year

    : '  ';

  const deathDate = personData.deathDate

    ? personData.deathDate.day +

      ' ' +

      personData.deathDate.month +

      ' ' +

      personData.deathDate.year

    : '  ';

  return (

    <div class="info-page-container">

      <div

        className="position-absolute w-100 h-100"

        onClick={clickOutside}

      ></div>

      <div className="person-page container">

        <div className="row">

          <div className="col-3">

            <img src={personData.photo ? personData.photo : nouser} />

          </div>

          <div className="col-9">

            <ul className="person-features">

              <li>

                <strong>ФИО:</strong>

                {name}

              </li>

              {birthDate === '  ' ? null : (

                <li>

                  <strong>

                    Родил{personData.sex === 'Мужской' ? 'ся' : 'ась'}:

                  </strong>

                  {birthDate}

                </li>

              )}

              {deathDate === '  ' ? null : (

                <li>

                  <strong>

                    Умер{personData.sex === 'Мужской' ? '' : 'ла'}:

                  </strong>

                  {deathDate}

                </li>

              )}

              {personData.nativeCity === '' ? null : (

                <li>

                  <strong>Родной город:</strong>

                  {personData.nativeCity}

                </li>

              )}

              <li>

                <strong>Биография:</strong>

                {personData.bio ? personData.bio : <i>Биография отсутствует</i>}

              </li>

            </ul>

          </div>

        </div>

      </div>

    </div>

  );

};

## **1.13. Компонент PersonCard**

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

import '../style/person.scss';

import nouser from '../img/nouser.png';

import { PersonPage } from './PersonPage';

import ReactDOM from 'react-dom';

import { EditPerson } from './EditPerson';

import { AuthContext } from '../context/Auth';

import remove from '../removeMethod';

export const PersonCard = ({

  person,

  handlePersonSelection,

  selected = false,

  bannedRoles,

}) => {

  const { currentUser } = useContext(AuthContext);

  const personData = person.data();

  const birthDate = personData.birthDate

    ? personData.birthDate.day +

      ' ' +

      personData.birthDate.month +

      ' ' +

      personData.birthDate.year

    : '';

  const deathDate = personData.deathDate

    ? personData.deathDate.day +

      ' ' +

      personData.deathDate.month +

      ' ' +

      personData.deathDate.year

    : '';

  const [relationship, setRelationship] = useState('parent');

  const [avatarHovered, setAvatarHovered] = useState(false);

  const [bodyHovered, setBodyHovered] = useState(false);

  const [showFull, setShowFull] = useState(false);

  const [showEdit, setShowEdit] = useState(false);

  const style = handlePersonSelection ? { border: 'solid 1px #d048b6' } : null;

  const borderClassName = personData.sex === 'Мужской' ? 'male' : 'female';

  const handleBodyClick = () => {

    if (handlePersonSelection) {

      handlePersonSelection(person.id, relationship);

    } else {

      setShowFull(true);

    }

  };

  const handleRelationshipClick = (msg) => {

    if (

      (msg === 'parent' && bannedRoles[0]) ||

      (msg === 'spouse' && bannedRoles[1]) ||

      (msg === 'child' && bannedRoles[2])

    ) {

      setRelationship(msg);

      if (handlePersonSelection) {

        handlePersonSelection(person.id, msg);

      }

    }

  };

  useEffect(() => {

    ReactDOM.render(

      showEdit ? (

        <EditPerson

          userId={currentUser.uid}

          person={person}

          handleClose={() => setShowEdit(false)}

        />

      ) : null,

      document.getElementById('info-container')

    );

  }, [showEdit]);

  useEffect(() => {

    ReactDOM.render(

      showFull ? (

        <PersonPage person={person} clickOutside={() => setShowFull(false)} />

      ) : null,

      document.getElementById('info-container')

    );

  }, [showFull]);

  return (

    <>

      <div className={person.id + ' person'}>

        <div

          className="avatar"

          onMouseEnter={() => setAvatarHovered(true)}

          onMouseLeave={() => setAvatarHovered(false)}

        >

          <img

            src={personData.photo ? personData.photo : nouser}

            style={

              handlePersonSelection

                ? null

                : avatarHovered

                ? { top: -35 + 'px' }

                : null

            }

            onClick={handleBodyClick}

          />

          {handlePersonSelection ? null : (

            <div className="buttons">

              <button

                className="btn btn-edit"

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

              >

                <i className="fa fa-pencil" aria-hidden="true"></i>

              </button>

              <button

                className="btn btn-trash"

                onClick={async () => {

                  await remove(person.id, currentUser.uid);

                }}

              >

                <i className="fa fa-trash" aria-hidden="true"></i>

              </button>

            </div>

          )}

        </div>

        <div

          className="person-body"

          style={selected || bodyHovered ? style : null}

          onClick={handleBodyClick}

          onMouseEnter={() => setBodyHovered(true)}

          onMouseLeave={() => setBodyHovered(false)}

        >

          <h6 className="name">

            {personData.firstName +

              ' ' +

              personData.secondName +

              ' ' +

              personData.lastName}

          </h6>

          <div className="info">

            {birthDate}

            {deathDate !== '  ' ? ' - ' + deathDate : deathDate}

          </div>

          {personData.nativeCity ? (

            <div className="info">

              <a

                href={

                  'https://www.google.ru/maps/search/' + personData.nativeCity

                }

                target="\_blank"

              >

                <i className="fa fa-map-marker" aria-hidden="true"></i>

                {' ' + personData.nativeCity}

              </a>

            </div>

          ) : null}

          <hr className={borderClassName} />

        </div>

        {selected ? (

          <ul className="list-group list-group-horizontal choose-relative">

            <li

              className={

                'list-group-item ' +

                (relationship === 'parent' ? 'selected ' : ' ') +

                (bannedRoles[0] ? null : 'banned')

              }

              onClick={() => handleRelationshipClick('parent')}

            >

              Родитель

            </li>

            <li

              className={

                'list-group-item ' +

                (relationship === 'spouse' ? 'selected ' : ' ') +

                (bannedRoles[1] ? null : 'banned')

              }

              onClick={() => handleRelationshipClick('spouse')}

            >

              Супруг

            </li>

            <li

              className={

                'list-group-item ' +

                (relationship === 'child' ? 'selected ' : ' ') +

                (bannedRoles[2] ? null : 'banned')

              }

              onClick={() => handleRelationshipClick('child')}

            >

              Ребенок

            </li>

          </ul>

        ) : null}

      </div>

    </>

  );

};

## **1.14. Компонент SelectDate**

import React, { useState } from 'react';

const oddMonths = [

  'Января',

  'Марта',

  'Мая',

  'Июля',

  'Августа',

  'Октября',

  'Декабря',

];

const evenMonths = ['Апреля', 'Июня', 'Сентября', 'Ноября'];

export const SelectDate = ({

  personDay,

  personMonth,

  personYear,

  required,

  typeOfDate,

}) => {

  const [day, setDay] = useState(personDay);

  const [month, setMonth] = useState(personMonth);

  const [year, setYear] = useState(personYear);

  const [dayOptions, setDayOptions] = useState(

    [...Array(31).keys()].map((i) => <option key={i + 1}>{i + 1}</option>)

  );

  const [daySelectStyle, setDaySelectStyle] = useState({

    color: personDay ? '#d0d7e1' : '#6c757d',

  });

  const [monthSelectStyle, setMonthSelectStyle] = useState({

    color: personMonth ? '#d0d7e1' : '#6c757d',

  });

  const handleChange = (event) => {

    switch (event.target.name) {

      case typeOfDate + 'day':

        setDay(event.target.value);

        setDaySelectStyle({

          color: '#d0d7e1',

        });

        break;

      case typeOfDate + 'month':

        setMonth(event.target.value);

        setMonthSelectStyle({

          color: '#d0d7e1',

        });

        if (event.target.value === 'Февраля') {

          setDayOptions(

            [...Array(29).keys()].map((i) => (

              <option key={i + 1}>{i + 1}</option>

            ))

          );

          if (day > 29) {

            setDay(1);

          }

        } else if (oddMonths.includes(event.target.value)) {

          setDayOptions(

            [...Array(31).keys()].map((i) => (

              <option key={i + 1}>{i + 1}</option>

            ))

          );

        } else if (evenMonths.includes(event.target.value)) {

          setDayOptions(

            [...Array(30).keys()].map((i) => (

              <option key={i + 1}>{i + 1}</option>

            ))

          );

          if (day > 30) {

            setDay(1);

          }

        }

        break;

      case typeOfDate + 'year':

        const re = /^[0-9\b]+$/;

        if (

          (event.target.value === '' || re.test(event.target.value)) &&

          event.target.value >= 0 &&

          event.target.value <= 2020

        ) {

          setYear(event.target.value);

        }

        break;

      default:

        throw Error;

    }

  };

  return (

    <div className="form-row">

      <div className="col">

        <select

          className="form-control"

          onChange={handleChange}

          name={typeOfDate + 'day'}

          style={daySelectStyle}

          value={day}

          required={required}

        >

          <option value="" hidden disabled>

            День

          </option>

          {dayOptions}

        </select>

      </div>

      <div className="col">

        <select

          className="form-control"

          onChange={handleChange}

          name={typeOfDate + 'month'}

          style={monthSelectStyle}

          value={month}

          required={required}

        >

          <option value="" disabled>

            Месяц

          </option>

          <option>Января</option>

          <option>Февраля</option>

          <option>Марта</option>

          <option>Апреля</option>

          <option>Мая</option>

          <option>Июня</option>

          <option>Июля</option>

          <option>Августа</option>

          <option>Сентября</option>

          <option>Октября</option>

          <option>Ноября</option>

          <option>Декабря</option>

        </select>

      </div>

      <div className="col">

        <input

          type="text"

          className="form-control"

          placeholder="Год"

          name={typeOfDate + 'year'}

          id="year"

          value={year}

          onChange={handleChange}

          required={required}

        />

      </div>

    </div>

  );

};

## **1.15. Компонент Tree**

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

import ScrollBooster from 'scrollbooster';

import { PersonCard } from './PersonCard';

import '../style/tree.scss';

import { SteppedLineTo } from 'react-lineto';

export const Tree = ({

  people,

  families,

  candidate,

  sendRelative,

  returnFunction,

}) => {

  const edges = () => {

    return families.docs.map((fam) => (

      <>

        <SteppedLineTo

          from={fam.data().husband}

          to={fam.data().husbandFamily}

          borderColor="#d048b6"

          within="tree"

          delay={true}

          zIndex={-50}

        />

        <SteppedLineTo

          from={fam.data().wife}

          to={fam.data().wifeFamily}

          fromAnchor={fam.data().husbandFamily ? '50 33' : '50 50'}

          borderColor="#d048b6"

          within="tree"

          delay={true}

          zIndex={-50}

        />

        <SteppedLineTo

          from={fam.data().husband}

          to={fam.data().wife}

          borderColor="#d048b6"

          within="tree"

          delay={true}

          zIndex={0}

        />

        {fam

          .data()

          .children.map((child) =>

            child.startsWith('\_') ? null : (

              <SteppedLineTo

                from={fam.id}

                to={child}

                borderColor="#d048b6"

                within="tree"

                delay={true}

                zIndex={0}

              />

            )

          )}

      </>

    ));

  };

  const Family = (family, depth) => {

    const familyData = family.data();

    if (candidate) {

      return (

        <div id={family.id} className="family">

          <div className="tree-row">

            <div className={family.id + ' pair'}>

              {familyData.husband ? (

                <PersonCard

                  person={findById(familyData.husband)}

                  handlePersonSelection={handlePersonSelection}

                  selected={selectedPersonId === familyData.husband}

                  bannedRoles={[

                    true,

                    !familyData.wife && candidate.sex === 'Женский',

                    familyData.husbandFamily

                      ? candidate.sex === 'Мужской'

                        ? !findById(familyData.husbandFamily).data().husband

                        : !findById(familyData.husbandFamily).data().wife

                      : true,

                  ]}

                />

              ) : null}

              {familyData.wife ? (

                <PersonCard

                  person={findById(familyData.wife)}

                  handlePersonSelection={handlePersonSelection}

                  selected={selectedPersonId === familyData.wife}

                  bannedRoles={[

                    true,

                    !familyData.husband && candidate.sex === 'Мужской',

                    familyData.wifeFamily

                      ? candidate.sex === 'Мужской'

                        ? !findById(familyData.wifeFamily).data().husband

                        : !findById(familyData.wifeFamily).data().wife

                      : true,

                  ]}

                />

              ) : null}

            </div>

          </div>

          <div className="tree-row">

            {familyData.children.map((child) =>

              child.startsWith('\_') ? (

                familyLayout(findById(child), depth + 1, family.id)

              ) : (

                <PersonCard

                  person={findById(child)}

                  handlePersonSelection={handlePersonSelection}

                  selected={selectedPersonId === child}

                  bannedRoles={[

                    true,

                    candidate.sex !== findById(child).data().sex,

                    candidate.sex === 'Мужской'

                      ? !familyData.husband

                      : !familyData.wife,

                  ]}

                />

              )

            )}

          </div>

        </div>

      );

    } else {

      return (

        <div className="family">

          <div className="tree-row">

            <div className={family.id + ' pair'}>

              {familyData.husband ? (

                <PersonCard person={findById(familyData.husband)} />

              ) : null}

              {familyData.wife ? (

                <PersonCard person={findById(familyData.wife)} />

              ) : null}

            </div>

          </div>

          <div className="tree-row">

            {familyData.children.map((child) =>

              child.startsWith('\_') ? (

                familyLayout(findById(child), depth + 1, family.id)

              ) : (

                <PersonCard person={findById(child)} />

              )

            )}

          </div>

        </div>

      );

    }

  };

  const [selectedPersonId, setSelectedPersonId] = useState(null);

  const [selectedRelationship, setSelectedRelationship] = useState(null);

  const drawenFamilies = [];

  const familiesToRender = [];

  const depths = [];

  // Running ScrollBooster

  useEffect(() => {

    const viewport = document.querySelector('.tree-container');

    const content = document.querySelector('.tree');

    const sb = new ScrollBooster({

      viewport,

      content,

      emulateScroll: true,

      onUpdate: (state) => {

        content.style.transform = `translate(

            ${-state.position.x}px,

            ${-state.position.y}px

          )`;

      },

    });

  }, []);

  const handlePersonSelection = sendRelative

    ? (id, relationship) => {

        if (id !== selectedPersonId) {

          setSelectedPersonId(id);

        }

        if (relationship !== selectedRelationship) {

          setSelectedRelationship(relationship);

        }

      }

    : null;

  const handleClick = sendRelative

    ? () => {

        if (selectedPersonId) {

          sendRelative(selectedPersonId, selectedRelationship);

        }

      }

    : null;

  const findById = (id) => {

    if (id.startsWith('\_')) {

      return families.docs.find((element, index, array) => element.id === id);

    } else {

      return people.docs.find((element, index, array) => element.id === id);

    }

  };

  const familyLayout = (family, depth, parentsId = null) => {

    const familyData = family.data();

    drawenFamilies.push(family.id);

    // Если родителей нет, мы добрались до вершины ветки - сохраняем глубину

    if (!familyData.husbandFamily && !familyData.wifeFamily) {

      depths.push(depth);

    }

    // Если обеих родительских веток нет или есть одна и она отрисована

    if (

      (!familyData.husbandFamily && !familyData.wifeFamily) ||

      (familyData.husbandFamily &&

        !familyData.wifeFamily &&

        drawenFamilies.indexOf(familyData.husbandFamily) !== -1) ||

      (familyData.wifeFamily &&

        !familyData.husbandFamily &&

        drawenFamilies.indexOf(familyData.wifeFamily) !== -1)

    ) {

      return Family(family, depth);

    }

    // Есть только мужская родительская ветка и она еще не отрисована

    if (

      familyData.husbandFamily &&

      !familyData.wifeFamily &&

      drawenFamilies.indexOf(familyData.husbandFamily) === -1

    ) {

      return familyLayout(findById(familyData.husbandFamily), depth - 1);

    }

    // Есть только женская родительская ветка и она еще не отрисована

    if (

      familyData.wifeFamily &&

      !familyData.husbandFamily &&

      drawenFamilies.indexOf(familyData.wifeFamily) === -1

    ) {

      return familyLayout(findById(familyData.wifeFamily), depth - 1);

    }

    // Есть обе родительские ветки и они обе не отрисованы

    if (

      familyData.husbandFamily &&

      familyData.wifeFamily &&

      drawenFamilies.indexOf(familyData.husbandFamily) === -1 &&

      drawenFamilies.indexOf(familyData.wifeFamily) === -1

    ) {

      return familyLayout(findById(familyData.husbandFamily), depth - 1);

    }

    // Есть обе родительские ветки и мужская уже отрисована

    if (

      familyData.husbandFamily &&

      familyData.wifeFamily &&

      drawenFamilies.indexOf(familyData.husbandFamily) !== -1 &&

      drawenFamilies.indexOf(familyData.wifeFamily) === -1

    ) {

      familiesToRender.push([familyData.wifeFamily, depth - 1]);

      return Family(family, depth);

    }

    // Есть обе родительские ветки и женская уже отрисована

    if (

      familyData.husbandFamily &&

      familyData.wifeFamily &&

      drawenFamilies.indexOf(familyData.husbandFamily) === -1 &&

      drawenFamilies.indexOf(familyData.wifeFamily) !== -1

    ) {

      familiesToRender.push([familyData.husbandFamily, depth - 1]);

      return null;

    }

    // Есть обе родительские ветки и они уже отрисованы

    if (

      familyData.husbandFamily &&

      familyData.wifeFamily &&

      drawenFamilies.indexOf(familyData.husbandFamily) !== -1 &&

      drawenFamilies.indexOf(familyData.wifeFamily) !== -1

    ) {

      return familyData.husbandFamily === parentsId

        ? Family(family, depth)

        : null;

    }

  };

  const treeLayout = () => {

    const renderedFamilies = [];

    renderedFamilies.push(familyLayout(families.docs[0], 0));

    while (familiesToRender.length !== 0) {

      renderedFamilies.push(

        familyLayout(findById(familiesToRender[0][0]), familiesToRender[0][1])

      );

      familiesToRender.shift();

    }

    const maxDepth = Math.min(...depths);

    return renderedFamilies.map((fam, i) => (

      <div style={{ marginTop: 370 \* (depths[i] - maxDepth) + 'px' }}>

        {fam}

      </div>

    ));

  };

  return (

    <>

      {candidate ? (

        <div className="position-absolute w-100 mt-3 d-flex justify-content-center">

          <div className="top-container">

            <button className="btn mr-3 btn-back" onClick={returnFunction}>

              <ion-icon name="arrow-back-circle"></ion-icon>

            </button>

            <div className="message">

              <div>Отметьте ближайшего родственника</div>

              <button

                className="btn ml-3 btn-submit"

                onClick={handleClick}

                disabled={!selectedPersonId}

              >

                Добавить

              </button>

            </div>

          </div>

        </div>

      ) : null}

      <div id="info-container"></div>

      <div className="tree-container">

        <div className="tree">

          {treeLayout()}

          {edges()}

        </div>

      </div>

    </>

  );

};

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Изм. | Номера листов | | | | Всего листов (страниц) в документе | № документа | Входя- щий № сопро- водит. докум. и дата | Под- пись | Дата |
| изменен- ных | заменен- ных | новых | аннулиро- ванных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |