***«Разработка веб-приложения для командной работы над проектами»***

Автор:

*Магомаева Лейла Румановна,*

*Садаханов Сулиман Салманович,*

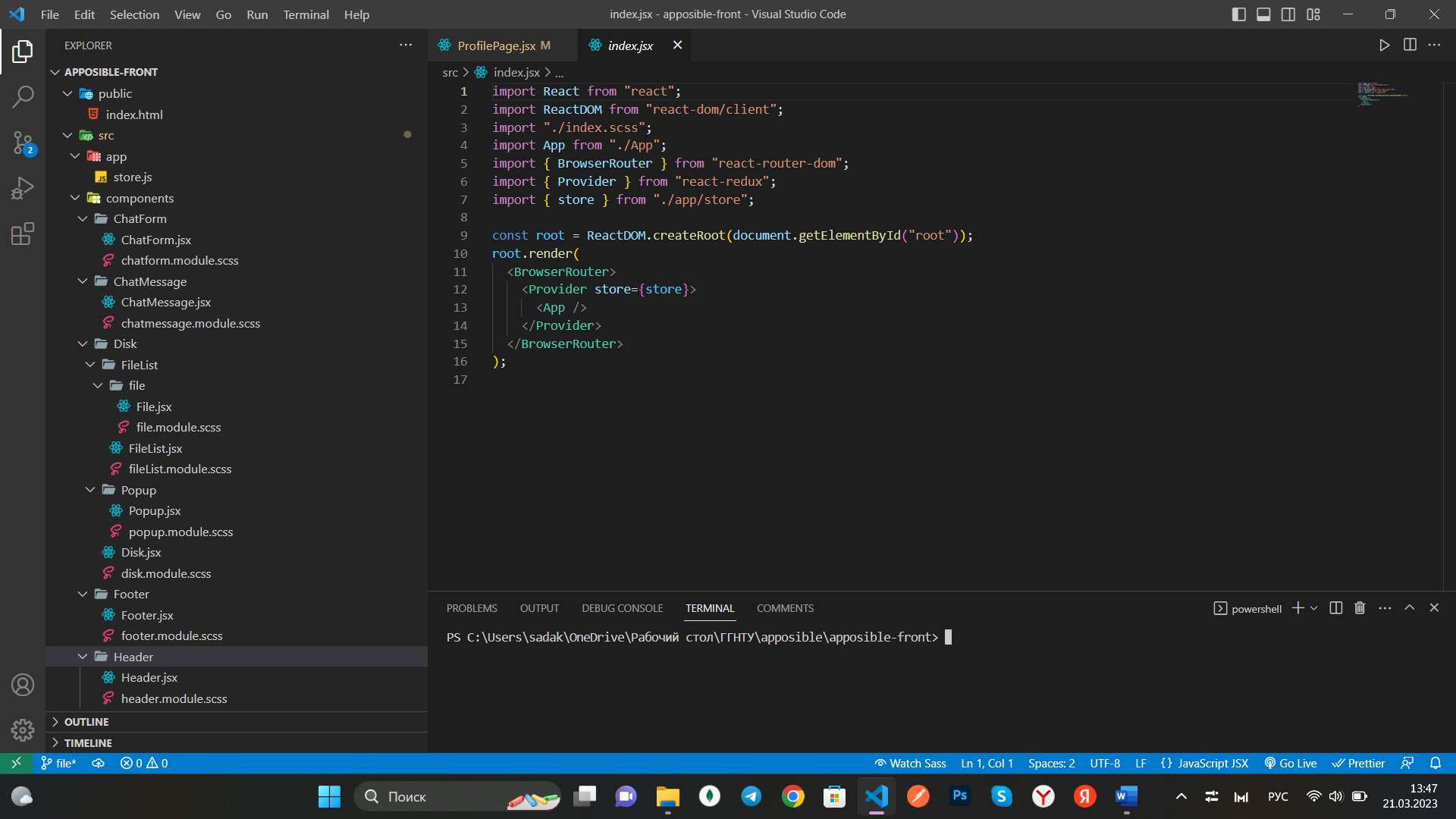
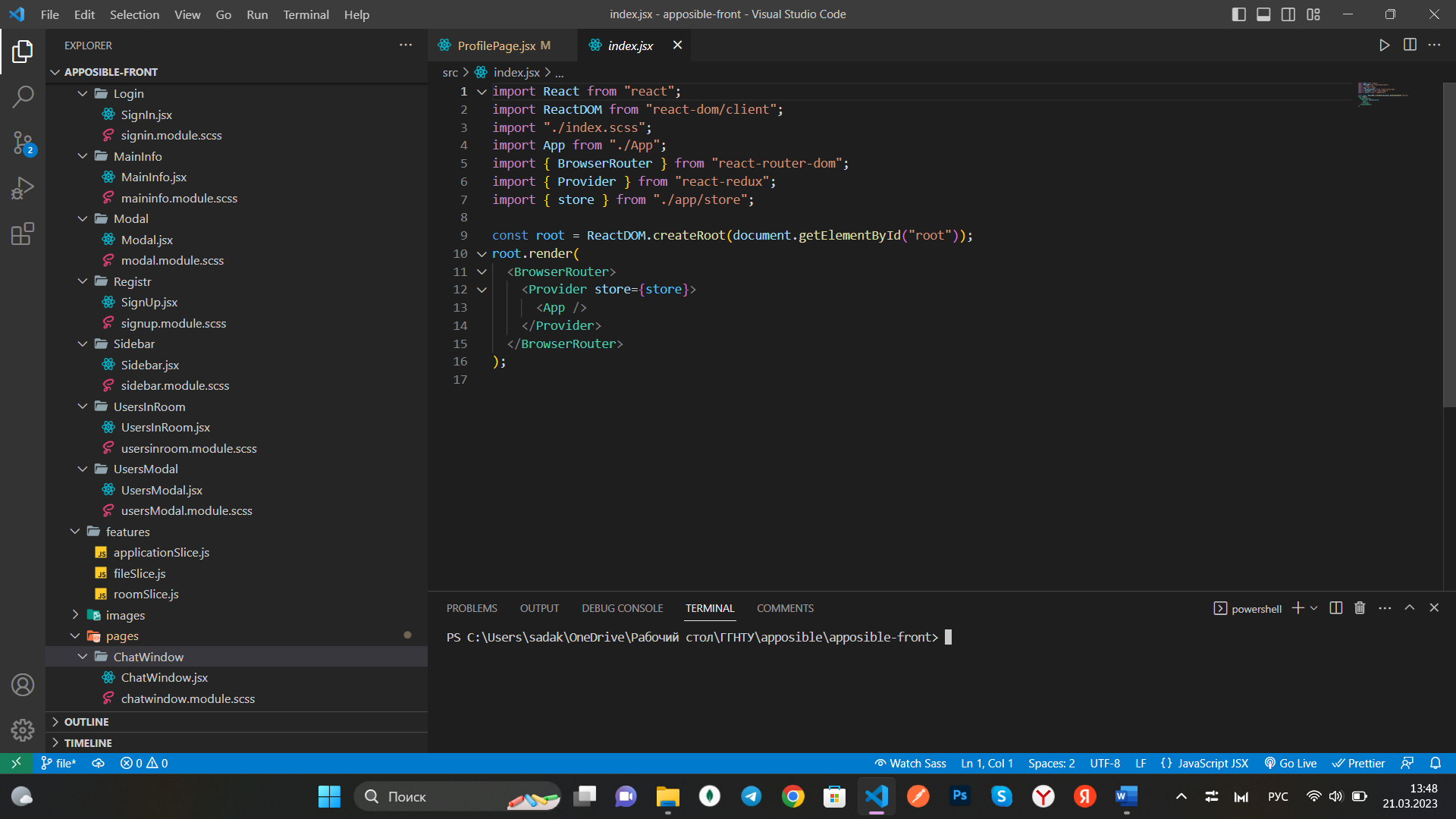
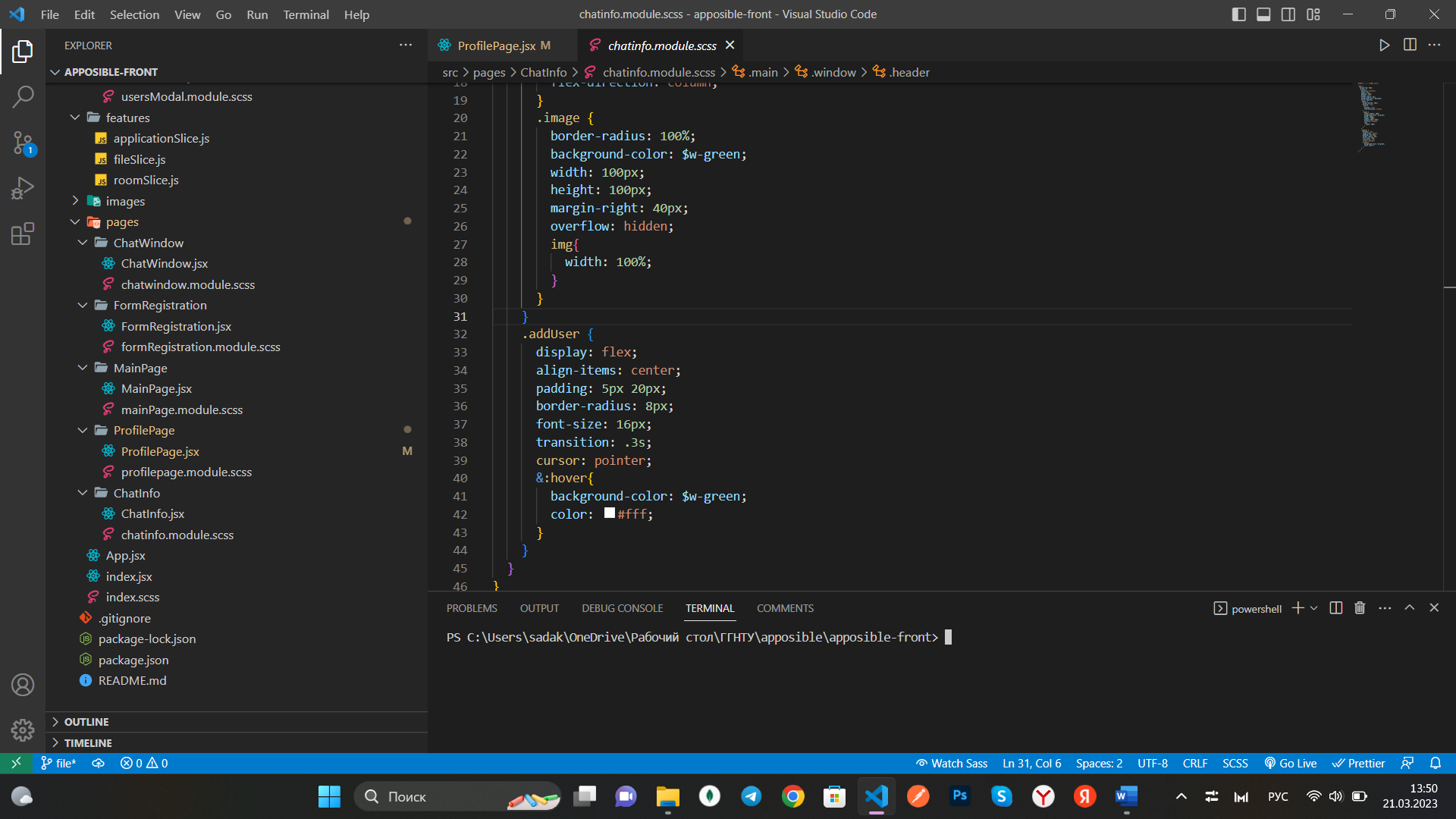
*Матаева Лиза Султановна,*

*Гапаева Хава Алиевна*

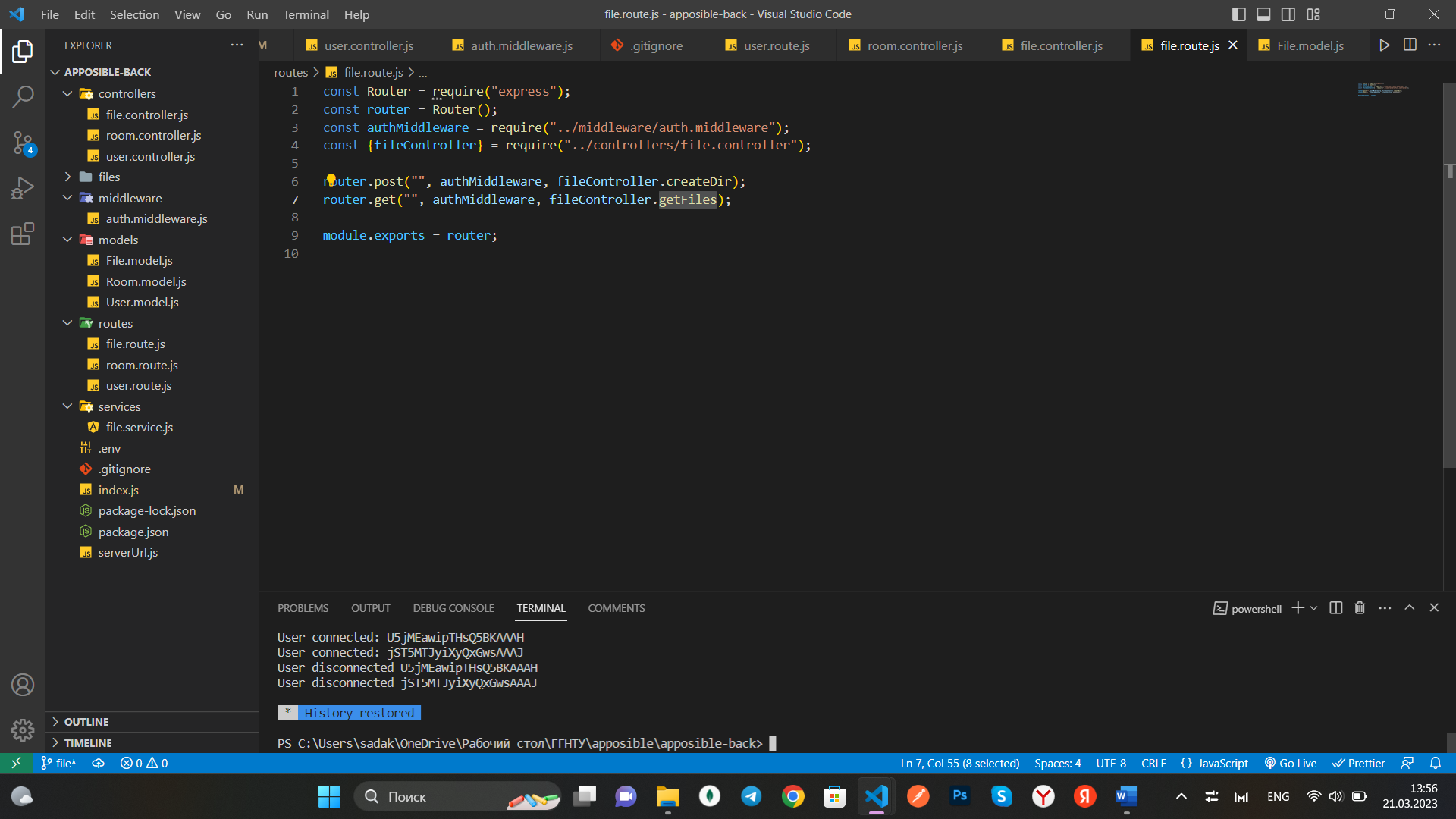
# Правообладатель:

Федеральное государственное бюджетное образовательное учреждение высшего образования "Грозненский государственный нефтяной технический университет имени академика М.Д. Миллионщикова"

**Структура папок на Front-end:**

**Структура папок на Back-end:**



**Содержание файлов Front-end:**

## **Index.html**

<!DOCTYPE html>

<html lang="ru">

  <head>

    <meta charset="utf-8" />

    <link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

    <meta name="viewport" content="width=device-width, initial-scale=1" />

    <meta name="theme-color" content="#000000" />

    <meta

      name="description"

      content="Web site created using create-react-app"

    />

    <link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

    <title>Apposible</title>

  </head>

  <body>

    <noscript>You need to enable JavaScript to run this app.</noscript>

    <div id="root"></div>

  </body>

</html>

## **store.js**

import { configureStore } from "@reduxjs/toolkit";

import application from "../features/applicationSlice";

import room from "../features/roomSlice";

import file from "../features/fileSlice";

export const store = configureStore({

  reducer: {

    application,

    room,

    file,

  },

});

## **ChatForm.jsx**

import React from "react";

import io from "socket.io-client";

import styles from "./chatform.module.scss";

import { useDispatch, useSelector } from "react-redux";

import { useNavigate } from "react-router-dom";

import { deleteRoom } from "../../features/roomSlice";

export const socket = io.connect(`http://localhost:3001`);

const ChatForm = ({ name, roomID, access }) => {

  const username = useSelector((state) => state.application.login);

  const dispath = useDispatch();

  const navigate = useNavigate();

  const joinRoom = () => {

    if (username !== "" && roomID !== "") {

      socket.emit("join\_room", roomID);

      navigate(`/room/${roomID}`);

    }

  };

  const handleDelete = (e) => {

    e.stopPropagation();

    dispath(deleteRoom({ id: roomID }));

  };

  return (

    <div className={styles.window} onClick={joinRoom}>

      <h1>

        Комната: <span>{name}</span>

      </h1>

      <button onClick={(e) => handleDelete(e)}>Удалить комнату</button>

    </div>

  );

};

export default ChatForm;

## **ChatMessage.jsx**

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

import { useDispatch, useSelector } from "react-redux";

import { useNavigate, useParams } from "react-router-dom";

import { addComment, fetchRoom } from "../../features/roomSlice";

import styles from "./chatmessage.module.scss";

import { socket } from "../ChatForm/ChatForm";

import { fetchUsers } from "../../features/applicationSlice";

import Disk from "../Disk/Disk";

const Chat = () => {

  const { id } = useParams();

  const dispath = useDispatch();

  const navigate = useNavigate();

  const messagesEndRef = useRef(null);

  const [currentMessage, setCurrentMessage] = useState("");

  const [messageList, setMessageList] = useState([]);

  const scrollToBottom = () => {

    messagesEndRef.current?.scrollIntoView({ behavior: "smooth" });

  };

  useEffect(() => {

    scrollToBottom();

  }, [messageList]);

  const messages = useSelector((state) =>

    state.room.room.find((message) => {

      return message.\_id === id;

    })

  );

  const userID = useSelector((state) => state.application.userId);

  const author = useSelector((state) =>

    state.application.users.find((user) => user.\_id === userID)

  );

  const minutes =

    new Date(Date.now()).getMinutes() < 10

      ? "0" + new Date(Date.now()).getMinutes()

      : new Date(Date.now()).getMinutes();

  const hours = new Date(Date.now()).getHours();

  const sendMessage = () => {

    dispath(

      addComment({

        id,

        user: userID,

        comment: currentMessage,

        time: hours + ":" + minutes,

      })

    );

    if (currentMessage !== "") {

      const messageData = {

        room: id,

        author: userID,

        authorName: author.name,

        message: currentMessage,

        time: hours + ":" + minutes,

      };

      socket.emit("send\_message", messageData);

      setMessageList((list) => [...list, messageData]);

      setCurrentMessage("");

    }

  };

  const handleRouteRoom = () => {

    navigate(`/room/info/${id}`);

  };

  useEffect(() => {

    socket.on("receive\_message", (data) => {

      setMessageList((list) => [...list, data]);

    });

  }, []);

  useEffect(() => {

    dispath(fetchUsers());

    dispath(fetchRoom());

  }, [dispath]);

  return (

    <div className={styles.bg}>

      <div className={styles.chatWindow}>

        <div className={styles.main}>

          <div className={styles.header}>

            <img

              src={require("../../images/logo2.png")}

              alt="logo"

              onClick={handleRouteRoom}

            />

          </div>

          <div className={styles.chatForm}>

            <div className={styles.body}>

              {messages?.users.map((item) => {

                return (

                  <div

                    className={

                      userID === item.user.\_id ? styles.you : styles.outher

                    }

                    key={item.\_id}

                  >

                    <div>

                      <div className={styles.message}>

                        <span>{item.comment}</span>

                      </div>

                      <div className={styles.comment\_footer}>

                        <span className={styles.time}>{item.time}</span>

                        <span className={styles.author}>{item.user.name}</span>

                      </div>

                    </div>

                  </div>

                );

              })}

            </div>

            <div className={styles.body}>

              {messageList.map((messageContent, index) => {

                return (

                  <div

                    className={

                      userID === messageContent.author

                        ? styles.you

                        : styles.outher

                    }

                    key={index}

                  >

                    <div>

                      <div className={styles.message}>

                        <span>{messageContent.message}</span>

                      </div>

                      <div className={styles.comment\_footer}>

                        <span className={styles.time}>

                          {messageContent.time}

                        </span>

                        <span className={styles.author}>

                          {messageContent.authorName}

                        </span>

                      </div>

                    </div>

                  </div>

                );

              })}

            </div>

            <div ref={messagesEndRef} />

          </div>

          <div className={styles.footer}>

            <input

              type="text"

              value={currentMessage}

              placeholder="Hey..."

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

              onKeyDown={(e) => {

                e.key === "Enter" && sendMessage();

              }}

            />

          </div>

        </div>

        <Disk />

      </div>

    </div>

  );

};

export default Chat;

## **File.jsx**

import React from "react";

import styles from "./file.module.scss";

import { BsFiletypeHtml } from "react-icons/bs";

import { BsFiletypeScss } from "react-icons/bs";

import { SiJavascript } from "react-icons/si";

import { AiTwotoneFolderOpen } from "react-icons/ai";

import { AiFillFileImage } from "react-icons/ai";

import { AiFillFileWord } from "react-icons/ai";

import { FaFilePowerpoint } from "react-icons/fa";

import { useDispatch } from "react-redux";

import { fetchFile } from "../../../../features/fileSlice";

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

const File = ({ file }) => {

  const { id } = useParams();

  const image = ["img", "jpeg", "jpg", "svg", "png"];

  const dispatch = useDispatch();

  const handleOpenDir = () => {

    dispatch(fetchFile({ dirId: file?.parent, room: id }));

  };

  return (

    <div

      className={styles.file}

      onClick={file.type === "dir" ? () => handleOpenDir() : ""}

    >

      {file.type === "dir" && <AiTwotoneFolderOpen size={22} />}

      {file.type === "html" && <BsFiletypeHtml size={22} />}

      {file.type === "css" && <BsFiletypeScss size={22} />}

      {file.type === "js" && <SiJavascript size={22} />}

      {file.type === "pptx" && <FaFilePowerpoint size={22} />}

      {file.type === "docx" && <AiFillFileWord size={22} />}

      {image.includes(file.type) && <AiFillFileImage size={22} />}

      <div className={styles.name}>{file.name}</div>

      <div className={styles.date}>{file.date?.slice(0, 10)}</div>

      <div className={styles.size}>{file.size}</div>

    </div>

  );

};

export default File;

## **FileList.jsx**

import React, { useEffect } from "react";

import { useDispatch, useSelector } from "react-redux";

import styles from "./fileList.module.scss";

import File from "./file/File";

import { fetchDir, fetchFile } from "../../../features/fileSlice";

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

const FileList = () => {

  const { id } = useParams();

  const dispath = useDispatch();

  const files = useSelector((state) => state.file.currentDir);

  useEffect(() => {

    dispath(fetchDir({ dirId: files?.\_id, room: id }));

  }, [dispath, id]);

  return (

    <div className={styles.fileList}>

      <div className={styles.header}>

        <div className={styles.name}>Название</div>

        <div className={styles.date}>Время</div>

        <div className={styles.size}>Размер</div>

      </div>

      {files?.map((file) => {

        return <File key={file.\_id} file={file} />;

      })}

    </div>

  );

};

export default FileList;

## **Popup.jsx**

import React, { useState } from "react";

import { useDispatch, useSelector } from "react-redux";

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

import { addDir } from "../../../features/fileSlice";

import styles from "./popup.module.scss";

const Popup = ({ st }) => {

  const { id } = useParams();

  const dispatch = useDispatch();

  const [dirName, setDirName] = useState("");

  const currentDir = useSelector((state) => state.file.currentDir);

  const handleChange = (e) => {

    setDirName(e.target.value);

  };

  const handleAddFile = () => {

    const type = dirName.includes(".")

      ? dirName.match(/\.(\w+)/g)[0].slice(1)

      : "dir";

    dispatch(

      addDir({

        name: dirName,

        parent: currentDir.\_id,

        type,

        room: id,

      })

    );

    setDirName("");

  };

  return (

    <div className={styles.popup}>

      <div className={styles.content}>

        <div className={styles.header}>

          <div className={styles.title}>

            <span>Для создания файлы указывайте соответсвующее расширение</span>

            <span>

              Для создания папок указывать ничего кроме названия не нужно

            </span>

          </div>

          <button className={styles.close} onClick={() => st(false)}>

            X

          </button>

        </div>

        <input

          type="text"

          placeholder="Введите название ..."

          value={dirName}

          onChange={handleChange}

        />

        <button className={styles.create} onClick={handleAddFile}>

          Создать

        </button>

      </div>

    </div>

  );

};

export default Popup;

## **Disk.jsx**

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

import { useDispatch } from "react-redux";

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

import { fetchFile } from "../../features/fileSlice";

import FileList from "./FileList/FileList";

import styles from "./disk.module.scss";

import Popup from "./Popup/Popup";

const Disk = () => {

  const { id } = useParams();

  const dispath = useDispatch();

  const [modal, setModal] = useState(false);

  const handleAddFile = () => {

    setModal(true);

  };

  useEffect(() => {

    dispath(fetchFile({ room: id }));

  }, [dispath, id]);

  return (

    <div className={styles.disk}>

      {modal ? <Popup st={setModal} /> : null}

      <div className={styles.buttons}>

        <button className={styles.back}>Назад</button>

        <button className={styles.create} onClick={handleAddFile}>

          Создать

        </button>

      </div>

      <FileList />

    </div>

  );

};

export default Disk;

## **Footer.jsx**

import React from "react";

import styles from "./footer.module.scss";

const Footer = () => {

  return (

    <footer>

      <ul className={styles.trend}>

        <h2>Trending Links</h2>

        <li>Class Shedules</li>

        <li>Apply Online</li>

        <li>Programs</li>

      </ul>

      <ul className={styles.contact}>

        <h2>Contact Us</h2>

        <li>Community College</li>

        <li>Springfield</li>

        <li>+7(906) 525-34-97</li>

      </ul>

      <div className={styles.follow}>

        Follow Us

        <ul>

          <li>

            <a href="#">

              <img src={require("../../images/facebook.png")} alt="" />

            </a>

          </li>

          <li>

            <a href="#">

              <img src={require("../../images/linkedin.png")} alt="" />

            </a>

          </li>

          <li>

            <a href="https://telegram.me/frontend\_develope" target="\_blank">

              <img src={require("../../images/telegram.png")} alt="" />

            </a>

          </li>

          <li>

            <a href="#">

              <img src={require("../../images/twitter.png")} alt="" />

            </a>

          </li>

          <li>

            <a

              href="https://api.whatsapp.com/send?phone={{89065253497}}"

              target="\_blank"

            >

              <img src={require("../../images/whatsapp.png")} alt="" />

            </a>

          </li>

        </ul>

      </div>

    </footer>

  );

};

export default Footer;

## **Header.jsx**

import React from "react";

import styles from "./header.module.scss";

import { Link, useLocation } from "react-router-dom";

import { useSelector } from "react-redux";

import logo from "../../images/logo2.png";

const Header = () => {

  const params = useLocation();

  const token = useSelector((state) => state.application.token);

  const id = localStorage.getItem("id");

  const chat = params.pathname.includes("chat");

  const profile = params.pathname.includes("profile");

  return (

    <header className={styles.header}>

      <div className={styles.logo}>

        <Link to="/">

          <img src={logo} alt="logo" />

        </Link>

      </div>

      <div className={styles.nav\_top}>

        <ul>

          {token ? (

            <Link

              to="/chat"

              className={chat ? `${styles.link} ${styles.active}` : styles.link}

            >

              <li>Chat</li>

            </Link>

          ) : (

            <li className={styles.link\_off}>Chat</li>

          )}

          {token ? (

            <Link

              to={`profile/${id}`}

              className={

                profile ? `${styles.link} ${styles.active}` : styles.link

              }

            >

              <li>profile</li>

            </Link>

          ) : (

            <li className={styles.link\_off}>profile</li>

          )}

          <Link to="/login" className={styles.link}>

            <li className={styles.link\_mg}>sign in / sign up</li>

          </Link>

        </ul>

      </div>

    </header>

  );

};

export default Header;

## **SignIn.jsx**

import React from "react";

import { useState } from "react";

import { useDispatch } from "react-redux";

import { Link, useNavigate } from "react-router-dom";

import { authSignIn } from "../../features/applicationSlice";

import styles from "./signin.module.scss";

import logo from "../../images/logo2.png";

import arrow from "../../images/arrow-left.png";

import { motion } from "framer-motion";

const SignIn = () => {

  const navigate = useNavigate();

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

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

  const [text, setText] = useState(true);

  const [iconPassword, setIconPassword] = useState(true);

  const dispatch = useDispatch();

  const handleLogin = (e) => {

    setLogin(e.target.value);

  };

  const handlePassword = (e) => {

    setPassword(e.target.value);

  };

  const handleSignUp = (e) => {

    e.preventDefault();

    setPassword("");

    setLogin("");

    dispatch(authSignIn({ login, password }));

    navigate("/");

  };

  const handleClick = () => {

    setText(!text);

    setIconPassword(!iconPassword);

  };

  return (

    <div className={styles.container}>

      <div className={styles.block\_signIn}>

        <motion.div

          className={styles.block\_two}

          initial={{ y: -500, opacity: 0 }}

          animate={{ y: 0, opacity: 1 }}

          transition={{ duration: 1.5 }}

        >

          <img src={logo} alt="logo" />

          <Link to="/login" className={styles.link}>

            <img src={arrow} alt="return" />

          </Link>

          <h1>Войти</h1>

          <form onSubmit={handleSignUp} className={styles.form\_signIn}>

            <input

              autoComplete="on"

              type="text"

              value={login}

              placeholder="Логин"

              onChange={(e) => handleLogin(e)}

              className={styles.input4}

            />

            <input

              autoComplete="on"

              type={text ? "password" : "text"}

              value={password}

              placeholder="Пароль"

              onChange={(e) => handlePassword(e)}

              className={styles.input5}

            />

            <img

              src={

                iconPassword

                  ? "https://cdn-icons-png.flaticon.com/512/159/159604.png"

                  : "https://cdn-icons-png.flaticon.com/512/7615/7615155.png"

              }

              alt="#"

              className={styles.eyeForPasswor}

              onClick={handleClick}

            />

            <button type="submit" className={styles.btn}>

              ВХОД

            </button>

          </form>

          <div className={styles.link}>

            <Link to="/signup" className={styles.l}>

              Нет аккаунта ?

            </Link>

          </div>

        </motion.div>

      </div>

    </div>

  );

};

export default SignIn;

## **MainInfo.jsx**

import React from "react";

import styles from "./maininfo.module.scss";

import { motion } from "framer-motion";

import word from "../../images/word.svg";

import excel from "../../images/excel.svg";

import powerpoint from "../../images/powerpoint.svg";

const MainInfo = () => {

  return (

    <>

      <div className={styles.container}>

        <div className={styles.img\_block\_1}>

          <motion.img

            initial={{ opacity: 0, transform: "scale(0)" }}

            whileInView={{ opacity: 1, transform: "scale(1)" }}

            transition={{ duration: 1.5 }}

            src={word}

            alt="#"

          />

          <motion.img

            initial={{ opacity: 0, transform: "scale(0)" }}

            whileInView={{ opacity: 1, transform: "scale(1)" }}

            transition={{ duration: 1.5, delay: 0.5 }}

            src={excel}

            alt="#"

          />

          <motion.img

            initial={{ opacity: 0, transform: "scale(0)" }}

            whileInView={{ opacity: 1, transform: "scale(1)" }}

            transition={{ duration: 1.5, delay: 1 }}

            src={powerpoint}

            alt="#"

          />

        </div>

        <motion.p

          initial={{ opacity: 0, x: -400 }}

          whileInView={{ opacity: 1, x: 0 }}

          transition={{ duration: 2, delay: 2 }}

        >

          Возможность работать и сохранять файлы различных форматов, в числе

          которых самые популярные приложения как Word, Excel, PowerPoint и

          другие.

        </motion.p>

      </div>

      <div className={styles.container}>

        <motion.p

          initial={{ opacity: 0, x: -400 }}

          whileInView={{ opacity: 1, x: 0 }}

          transition={{ duration: 2 }}

        >

          Хотите работать в команде и всегда оставаться на связи?

          <br />

          <br />

          Не проблема! Вы можете комментировать свои действия, общаться с

          командой, делиться ходом работы и просто общаться

        </motion.p>

        <div className={styles.img\_block\_2}>

          <motion.img

            initial={{ opacity: 0 }}

            whileInView={{ opacity: 1 }}

            transition={{ duration: 1.8 }}

            src={require("../../images/team\_image.png")}

            alt="#"

          />

        </div>

      </div>

    </>

  );

};

export default MainInfo;

## **Modal.jsx**

import React, { useState } from "react";

import { useDispatch, useSelector } from "react-redux";

import { createRoom } from "../../features/roomSlice";

import styles from "./modal.module.scss";

const Modal = () => {

  const user = useSelector((state) => state.application.userId);

  const [roomName, setRoomName] = useState("");

  const dispath = useDispatch();

  const handleChange = (e) => {

    setRoomName(e.target.value);

  };

  const handleAddRoom = () => {

    if (roomName.length > 0) {

      dispath(createRoom({ id: user, name: roomName, admin: user }));

    }

  };

  return (

    <div className={styles.modal} onClick={(e) => e.stopPropagation()}>

      <div className={styles.modalName}>Введите название группы : </div>

      <div className={styles.inputBlock}>

        <input

          type="text"

          value={roomName}

          onChange={(e) => handleChange(e)}

          onKeyDown={(e) => {

            e.key === "Enter" && handleAddRoom();

          }}

        />

        <button onClick={() => handleAddRoom()}>Создать</button>

      </div>

    </div>

  );

};

export default Modal;

## **SignUp.jsx**

import React from "react";

import { useState } from "react";

import { useDispatch } from "react-redux";

import { authSignUp } from "../../features/applicationSlice";

import styles from "./signup.module.scss";

import logo from "../../images/logo2.png";

import { Link, useNavigate } from "react-router-dom";

import arrow from "../../images/arrow-left.png";

import { motion } from "framer-motion";

const SignUp = () => {

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

  const [mail, setMail] = useState("");

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

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

  const [text, setText] = useState(true);

  const [iconPassword, setIconPassword] = useState(true);

  const navigate = useNavigate();

  const dispatch = useDispatch();

  const handleName = (e) => {

    setName(e.target.value);

  };

  const handleMail = (e) => {

    setMail(e.target.value);

  };

  const handleLogin = (e) => {

    setLogin(e.target.value);

  };

  const handlePassword = (e) => {

    setPassword(e.target.value);

  };

  const handleSignUp = (e) => {

    dispatch(authSignUp({ name, email: mail, login, password }));

    e.preventDefault();

    setPassword("");

    setName("");

    setMail("");

    setLogin("");

    navigate("/signin");

  };

  const handleClick = () => {

    setText(!text);

    setIconPassword(!iconPassword);

  };

  return (

    <div className={styles.container}>

      <div className={styles.block\_signUp}>

        <motion.div

          className={styles.block\_two}

          initial={{ y: -500, opacity: 0 }}

          animate={{ y: 0, opacity: 1 }}

          transition={{ duration: 1.5 }}

        >

          <img src={logo} alt="logo" />

          <Link to="/login" className={styles.link}>

            <img src={arrow} alt="return" />

          </Link>

          <h1>Авторизоваться</h1>

          <form onSubmit={handleSignUp} className={styles.form\_signUp}>

            <input

              autoComplete="on"

              type="text"

              value={name}

              placeholder="Имя"

              onChange={(e) => handleName(e)}

            />

            <input

              autoComplete="on"

              type="email"

              value={mail}

              placeholder="Mail"

              onChange={(e) => handleMail(e)}

            />

            <input

              autoComplete="on"

              type="text"

              value={login}

              placeholder="Логин"

              onChange={(e) => handleLogin(e)}

            />

            <input

              autoComplete="on"

              type={text ? "password" : "text"}

              value={password}

              placeholder="Пароль"

              onChange={(e) => handlePassword(e)}

            />

            <img

              src={

                iconPassword

                  ? "https://cdn-icons-png.flaticon.com/512/159/159604.png"

                  : "https://cdn-icons-png.flaticon.com/512/7615/7615155.png"

              }

              alt="#"

              className={styles.eyeForPassword}

              onClick={handleClick}

            />

            <button type="submit" className={styles.btn}>

              РЕГИСТРАЦИЯ

            </button>

          </form>

          <div className={styles.link}>

            <Link to="/signin" className={styles.l}>

              Есть аккаунт ?

            </Link>

          </div>

        </motion.div>

      </div>

    </div>

  );

};

export default SignUp;

## **UsersInRoom.jsx**

import { useDispatch, useSelector } from "react-redux";

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

import { deleteUser } from "../../features/roomSlice";

import styles from "./usersinroom.module.scss";

const UsersInRoom = () => {

  const { id } = useParams();

  const dispath = useDispatch();

  const room = useSelector((state) =>

    state.room.room.find((item) => item.\_id === id)

  );

  const handleDelete = (user) => {

    dispath(deleteUser({ id: id, user: user }));

  };

  return (

    <div className={styles.users}>

      {room?.access.map((item) => {

        return (

          <div className={styles.body} key={item.\_id}>

            <div

              className={

                item.\_id === localStorage.getItem("id")

                  ? styles.you

                  : styles.outher

              }

            >

              {item.name}

            </div>

            {item.\_id !== localStorage.getItem("id") ? (

              <img

                src={require("../../images/delete.png")}

                onClick={() => handleDelete(item.\_id)}

                alt="delete"

              />

            ) : null}

          </div>

        );

      })}

    </div>

  );

};

export default UsersInRoom;

## **UsersModal.jsx**

import { useDispatch, useSelector } from "react-redux";

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

import { addUserRoom } from "../../features/roomSlice";

import styles from "./usersModal.module.scss";

const UsersModal = () => {

  const dispath = useDispatch();

  const { id } = useParams();

  const usersAll = useSelector((state) =>

    state.application.users.filter(

      (item) => item.\_id !== localStorage.getItem("id")

    )

  );

  const handleAdd = (e, user) => {

    dispath(addUserRoom({ id: id, user: user.\_id }));

  };

  return (

    <div className={styles.modal} onClick={(e) => e.stopPropagation()}>

      {usersAll?.map((user) => {

        return (

          <div

            key={user.\_id}

            className={styles.name}

            onClick={(e) => handleAdd(e, user)}

          >

            {user.name}

          </div>

        );

      })}

    </div>

  );

};

export default UsersModal;

## **ChatWindow.jsx**

import React, { useState } from "react";

import { useEffect } from "react";

import { useDispatch, useSelector } from "react-redux";

import ChatForm from "../../components/ChatForm/ChatForm";

import Modal from "../../components/Modal/Modal";

import { fetchRoom } from "../../features/roomSlice";

import styles from "./chatwindow.module.scss";

const ChatWindow = () => {

  const [modal, setModal] = useState(false);

  const room = useSelector((state) => state.room.room);

  const sortRoom = room.filter((item) =>

    item.access.find((room) => room.\_id === localStorage.getItem("id"))

  );

  const dispath = useDispatch();

  const handleAdd = (e) => {

    e.stopPropagation();

    setModal(true);

  };

  const handleClose = (e) => {

    if (e.currentTarget) {

      setModal(false);

    }

  };

  useEffect(() => {

    dispath(fetchRoom());

  }, [dispath]);

  return (

    <div onClick={(e) => handleClose(e)} className={styles.main}>

      {modal ? <Modal /> : null}

      <div className={styles.header}>

        <button onClick={(e) => handleAdd(e)}>Добавить комнату</button>

      </div>

      {sortRoom?.map((item) => {

        return (

          <div key={item.\_id}>

            <ChatForm

              name={item.name}

              users={item.users}

              access={item.access}

              roomID={item.\_id}

            />

          </div>

        );

      })}

    </div>

  );

};

export default ChatWindow;

## **FormRegistration.jsx**

import React from "react";

import { Link } from "react-router-dom";

import logo from "../../images/logo2.png"

import styles from "./formRegistration.module.scss";

const FormRegistration = () => {

  return (

    <div className={styles.main}>

      <img src={logo} alt="logo" />

      <h1>Apposible</h1>

      <Link to="/signin">

        <button className={styles.signin}>Войти</button>

      </Link>

      <Link to="/signup">

        <button className={styles.signup}>Зарегистрироваться</button>

      </Link>

    </div>

  );

};

export default FormRegistration;

## **MainPage.jsx**

import React, { useLayoutEffect, useRef } from "react";

import styles from "./mainPage.module.scss";

import logo from "../../images/logo.png";

import { gsap } from "gsap";

import { motion } from "framer-motion";

import ScrollTrigger from "gsap/ScrollTrigger";

import MainInfo from "../../components/MainInfo/MainInfo";

import Footer from "../../components/Footer/Footer";

gsap.registerPlugin(ScrollTrigger);

const MainPage = () => {

  const titleRef = useRef();

  const logoRef = useRef();

  const headRef = useRef();

  const logoBlock = useRef();

  const containerRef = useRef();

  useLayoutEffect(() => {

    gsap.from(logoRef.current, {

      transform: "scale(0.75)",

    });

    gsap.to(logoRef.current, {

      transform: "scale(1.2)",

      repeat: false,

      duration: 2,

      delay: 1,

    });

    gsap.from(headRef.current, {

      transform: "scale(0.85)",

    });

    gsap.to(headRef.current, {

      transform: "scale(1)",

      repeat: false,

      duration: 2,

      delay: 2,

    });

    gsap.to(logoBlock.current, {

      position: "sticky",

      scrollTrigger: {

        trigger: containerRef.current,

        start: "center center",

        end: "center center",

        scrub: true,

      },

    });

    gsap.from(logoBlock.current, {

      transform: "scale(0.9)",

    });

    gsap.to(logoBlock.current, {

      transform: "scale(1)",

      repeat: false,

      duration: 3,

      delay: 2,

    });

    gsap.to(titleRef.current, {

      scrollTrigger: {

        trigger: containerRef.current,

        start: "center center",

        end: "center 200px",

        scrub: true,

      },

      opacity: 1,

      transform: "scale(1)",

      duration: 3,

    });

  }, []);

  return (

    <>

      <div className={styles.main}></div>

      <div className={styles.container\_main} ref={containerRef}>

        <div className={styles.logo} ref={logoBlock}>

          <motion.img

            initial={{ opacity: 0 }}

            animate={{ opacity: 1 }}

            transition={{ delay: 0.5, duration: 2 }}

            ref={logoRef}

            src={logo}

            id={styles.logo}

            alt="marine logo"

          />

          <motion.p

            initial={{ opacity: 0 }}

            animate={{ opacity: 1 }}

            transition={{ delay: 2, duration: 2 }}

            ref={headRef}

            id={styles.logo\_text}

            alt="marine breeze"

          >

            Apposible

          </motion.p>

          <p className={styles.title} ref={titleRef}>

            Apposible - сервис на котором вы можете работать с файлами в команде

            и не переживать за их сохранность, вы спокойно можете связываться с

            участниками команды как по видео звонкам, так и воспользовавшись

            чатом

          </p>

        </div>

      </div>

      <MainInfo />

      <Footer />

    </>

  );

};

export default MainPage;

## **ProfilePage.jsx**

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

import { useDispatch, useSelector } from "react-redux";

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

import { changeUser, fetchUsers } from "../../features/applicationSlice";

import styles from "./profilepage.module.scss";

import avatar from "../../images/avatar.svg";

const ProfilePage = () => {

  const { id } = useParams();

  const dispatch = useDispatch();

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

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

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

  const user = useSelector((state) =>

    state.application.users.filter(

      (item) => item.\_id === localStorage.getItem("id") || item.\_id === id

    )

  );

  const handleChangeUser = () => {

    dispatch(changeUser({ id, name, email, login }));

  };

  useEffect(() => {

    dispatch(fetchUsers());

  }, [dispatch]);

  return (

    <div className={styles.container}>

      <div className={styles.main}>

        {user.map((item) => {

          return (

            <React.Fragment key={item.\_id}>

              <div className={styles.avatar}>

                {item.avatar ? (

                  <img src={item.avatar} alt="avatar" />

                ) : (

                  <img src={avatar} alt="avatar" />

                )}

                <div className={styles.buttons}>

                  <button className={styles.add}>Добавить фото</button>

                  {!item.avatar ? null : (

                    <button className={styles.delete}> Удалить</button>

                  )}

                </div>

              </div>

              <div className={styles.info}>

                <input

                  type="text"

                  placeholder={item.name}

                  value={name}

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

                />

                <input

                  type="text"

                  placeholder={item.email}

                  value={email}

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

                />

                <input

                  type="text"

                  placeholder={item.login}

                  value={login}

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

                />

                <button onClick={handleChangeUser}>Сохранить</button>

              </div>

            </React.Fragment>

          );

        })}

      </div>

    </div>

  );

};

export default ProfilePage;

## **ChatInfo.jsx**

import styles from "./chatinfo.module.scss";

import UsersInRoom from "../../components/UsersInRoom/UsersInRoom";

import { useEffect, useState } from "react";

import UsersModal from "../../components/UsersModal/UsersModal";

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

import { useDispatch, useSelector } from "react-redux";

import { fetchRoom } from "../../features/roomSlice";

const ChatInfo = () => {

  const { id } = useParams();

  const dispatch = useDispatch();

  const [modal, setModal] = useState(false);

  const room = useSelector((state) =>

    state.room.room.find((item) => item.\_id === id)

  );

  const handleOpen = (e) => {

    e.stopPropagation();

    setModal(true);

  };

  const handleClose = (e) => {

    if (e.currentTarget) {

      setModal(false);

    }

  };

  useEffect(() => {

    dispatch(fetchRoom());

  }, [dispatch]);

  return (

    <div className={styles.main} onClick={(e) => handleClose(e)}>

      <div className={styles.window} onClick={(e) => handleClose(e)}>

        {modal ? <UsersModal /> : null}

        <div className={styles.header}>

          <div className={styles.image}>

            {room?.image ? (

              <img src={room.image} alt="#" />

            ) : (

              <img src={require("../../images/room\_logo.png")} alt="#" />

            )}

          </div>

          <div className={styles.info}>

            <h2>{room?.name}</h2>

            <span>Участников: {room?.access.length}</span>

          </div>

        </div>

        <button className={styles.addUser} onClick={handleOpen}>

          Добавить участников

        </button>

        <UsersInRoom />

      </div>

    </div>

  );

};

export default ChatInfo;

## **App.jsx**

import { Route, Routes } from "react-router-dom";

import Header from "./components/Header/Header";

import ChatWindow from "./pages/ChatWindow/ChatWindow";

import ChatMessage from "./components/ChatMessage/ChatMessage";

import FormRegistration from "./pages/FormRegistration/FormRegistration";

import { useLocation } from "react-router-dom";

import MainPage from "./pages/MainPage/MainPage";

import SignIn from "./components/Login/SignIn";

import SignUp from "./components/Registr/SignUp";

import ProfilePage from "./pages/ProfilePage/ProfilePage";

import ChatInfo from "./pages/СhatInfo/ChatInfo";

function App() {

  const location = useLocation();

  return (

    <div className="App">

      {location.pathname === "/signin" ||

      location.pathname === "/signup" ? null : (

        <Header />

      )}

      <Routes>

        <Route path="/" element={<MainPage />} />

        <Route path="/chat" element={<ChatWindow />} />

        <Route path="/profile/:id" element={<ProfilePage />} />

        <Route path="/room/:id" element={<ChatMessage />} />

        <Route path="/room/info/:id" element={<ChatInfo />} />

        <Route path="/login" element={<FormRegistration />} />

        <Route path="/signin" element={<SignIn />} />

        <Route path="/signup" element={<SignUp />} />

      </Routes>

    </div>

  );

}

export default App;

## **index.jsx**

import React from "react";

import ReactDOM from "react-dom/client";

import "./index.scss";

import App from "./App";

import { BrowserRouter } from "react-router-dom";

import { Provider } from "react-redux";

import { store } from "./app/store";

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(

  <BrowserRouter>

    <Provider store={store}>

      <App />

    </Provider>

  </BrowserRouter>

);

## **applicationSlice.js**

import { createAsyncThunk, createSlice } from "@reduxjs/toolkit";

const initialState = {

  users: [],

  token: localStorage.getItem("token"),

  userId: localStorage.getItem("id"),

};

export const fetchUsers = createAsyncThunk(

  "fetch/users",

  async (\_, thunkAPI) => {

    try {

      const res = await fetch(`http://localhost:3001/api/auth/users`, {

        method: "GET",

      });

      const data = await res.json();

      return data;

    } catch (error) {

      return thunkAPI.rejectWithValue(error);

    }

  }

);

export const authSignIn = createAsyncThunk(

  "auth/signin",

  async ({ login, password }, thunkAPI) => {

    try {

      const res = await fetch(`http://localhost:3001/api/auth/login`, {

        method: "POST",

        headers: {

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

        },

        body: JSON.stringify({ login, password }),

      });

      const data = await res.json();

      if (data.error) {

        return thunkAPI.rejectWithValue(data.error);

      }

      localStorage.setItem("token", data.token);

      localStorage.setItem("id", data.user.\_id);

      return data;

    } catch (error) {

      return thunkAPI.rejectWithValue(error);

    }

  }

);

export const authSignUp = createAsyncThunk(

  "auth/signUp",

  async ({ login, password, name, email, avatar }, thunkAPI) => {

    try {

      const res = await fetch(`http://localhost:3001/api/auth/registr`, {

        method: "POST",

        headers: {

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

        },

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

      });

      const json = await res.json();

      if (json.error) {

        return thunkAPI.rejectWithValue(json.error);

      }

      return json;

    } catch (e) {

      thunkAPI.rejectWithValue(e);

    }

  }

);

export const changeUser = createAsyncThunk(

  "change/user",

  async ({ id, name, email, login }, thunkAPI) => {

    const token = localStorage.getItem("token");

    try {

      const res = await fetch(`http://localhost:3001/api/auth/user/${id}`, {

        method: "PATCH",

        headers: {

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

          Authorization: "Bearer " + token,

        },

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

      });

      return res.json();

    } catch (error) {}

  }

);

const applicationSlice = createSlice({

  name: "application",

  initialState,

  reducers: {},

  extraReducers: (builder) => {

    builder

      .addCase(authSignIn.fulfilled, (state, action) => {

        state.token = action.payload.token;

      })

      .addCase(authSignUp.fulfilled, (state, action) => {

        state.token = action.payload;

      })

      .addCase(fetchUsers.fulfilled, (state, action) => {

        state.users = action.payload;

      })

      .addCase(changeUser.fulfilled, (state, action) => {

        state.users = state.users.filter((item) => {

          if (item.\_id === action.payload.\_id) {

            return action.payload;

          }

          return item;

        });

      });

  },

});

export default applicationSlice.reducer;

## **fileSlice.js**

import { createAsyncThunk, createSlice } from "@reduxjs/toolkit";

const initialState = {

  files: [],

  currentDir: [],

};

export const fetchFile = createAsyncThunk(

  "fetch/files",

  async ({ dirId, room }, thunkAPI) => {

    try {

      const token = localStorage.getItem("token");

      const res = await fetch(

        `http://localhost:3001/api/files?room=${room}

            ${dirId ? "&parent=" + dirId : ""}`,

        {

          method: "GET",

          headers: {

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

            Authorization: "Bearer " + token,

          },

        }

      );

      const file = await res.json();

      return { file, room };

    } catch (error) {

      return thunkAPI.rejectWithValue(error);

    }

  }

);

export const fetchDir = createAsyncThunk(

  "fetch/dir",

  async ({ dirId, room }, thunkAPI) => {

    try {

      const token = localStorage.getItem("token");

      const res = await fetch(

        `http://localhost:3001/api/files?room=${room}

            ${dirId ? "&parent=" + dirId : ""}`,

        {

          method: "GET",

          headers: {

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

            Authorization: "Bearer " + token,

          },

        }

      );

      const file = await res.json();

      return file;

    } catch (error) {

      return thunkAPI.rejectWithValue(error);

    }

  }

);

export const addDir = createAsyncThunk(

  "add/files",

  async ({ name, type, parent, room }, thunkAPI) => {

    try {

      const token = localStorage.getItem("token");

      const res = await fetch(`http://localhost:3001/api/files`, {

        method: "POST",

        headers: {

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

          Authorization: "Bearer " + token,

        },

        body: JSON.stringify({ name, type, parent, room }),

      });

      const data = await res.json();

      return data;

    } catch (error) {

      return thunkAPI.rejectWithValue(error);

    }

  }

);

const fileSlice = createSlice({

  name: "file",

  initialState,

  reducers: {},

  extraReducers: (builder) => {

    builder

      .addCase(fetchFile.fulfilled, (state, action) => {

        state.currentDir = action.payload.file;

        state.files = action.payload;

      })

      .addCase(fetchDir.fulfilled, (state, action) => {

        state.currentDir = action.payload;

      })

      .addCase(addDir.fulfilled, (state, action) => {

        state.files.file.push(action.payload);

        state.currentDir.push(action.payload);

      });

  },

});

export default fileSlice.reducer;

## **roomSlice.js**

import { createAsyncThunk, createSlice } from "@reduxjs/toolkit";

const initialState = {

  room: [],

  loader: false,

};

export const fetchRoom = createAsyncThunk("fetch/room", async (\_, thunkAPI) => {

  try {

    const token = localStorage.getItem("token");

    const res = await fetch(`http://localhost:3001/room`, {

      method: "GET",

      headers: {

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

        Authorization: "Bearer " + token,

      },

    });

    const room = await res.json();

    return room;

  } catch (error) {

    thunkAPI.rejectWithValue(error);

  }

});

export const createRoom = createAsyncThunk(

  "create/room",

  async ({ id, name }, thunkAPI) => {

    try {

      const token = localStorage.getItem("token");

      const res = await fetch(`http://localhost:3001/room/create/${id}`, {

        method: "POST",

        headers: {

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

          Authorization: "Bearer " + token,

        },

        body: JSON.stringify({ name, access: id, admin: name }),

      });

      const room = await res.json();

      return room;

    } catch (error) {

      thunkAPI.rejectWithValue(error);

    }

  }

);

export const deleteRoom = createAsyncThunk(

  "delete/room",

  async ({ id }, thunkAPI) => {

    try {

      const token = localStorage.getItem("token");

      const res = await fetch(`http://localhost:3001/room/delete/${id}`, {

        method: "DELETE",

        headers: {

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

          Authorization: "Bearer " + token,

        },

      });

      const delRoom = await res.json();

      return id;

    } catch (error) {

      thunkAPI.rejectWithValue(error);

    }

  }

);

export const addUserRoom = createAsyncThunk(

  "addUser/room",

  async ({ id, user }, thunkAPI) => {

    try {

      const token = localStorage.getItem("token");

      const res = await fetch(`http://localhost:3001/room/addUser/${id}`, {

        method: "PATCH",

        headers: {

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

          Authorization: "Bearer " + token,

        },

        body: JSON.stringify({ user }),

      });

      const addUser = await res.json();

      return addUser;

    } catch (error) {

      thunkAPI.rejectWithValue(error);

    }

  }

);

export const deleteUser = createAsyncThunk(

  "deleteUser/room",

  async ({ id, user }, thunkAPI) => {

    try {

      const token = localStorage.getItem("token");

      const res = await fetch(`http://localhost:3001/room/deleteUser/${id}`, {

        method: "PATCH",

        headers: {

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

          Authorization: "Bearer " + token,

        },

        body: JSON.stringify({ id, user }),

      });

      const delUser = await res.json();

      return delUser;

    } catch (error) {

      thunkAPI.rejectWithValue(error);

    }

  }

);

//////ОБЯЗАТЕЛЬНО ПЕРЕДЕЛАТЬ ЭТОТ THUNK///////////

export const addComment = createAsyncThunk(

  "addComment/room",

  async ({ id, user, comment, time }, thunkAPI) => {

    try {

      const token = localStorage.getItem("token");

      const res = await fetch(`http://localhost:3001/room/addComment/${id}`, {

        method: "PATCH",

        headers: {

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

          Authorization: "Bearer " + token,

        },

        body: JSON.stringify({ user, comment, time }),

      });

      const addComment = await res.json();

      return { \_id: id, user, comment };

    } catch (error) {

      thunkAPI.rejectWithValue(error);

    }

  }

);

const roomSlice = createSlice({

  name: "room",

  initialState,

  reducers: {},

  extraReducers: (builder) => {

    builder

      //////////FETCH-ROOM///////////

      .addCase(fetchRoom.fulfilled, (state, action) => {

        state.room = action.payload;

        state.loader = false;

      })

      .addCase(fetchRoom.pending, (state, action) => {

        state.loader = true;

      })

      .addCase(fetchRoom.rejected, (state, action) => {

        state.loader = false;

        console.log("slice rejected");

      })

      //////////DELETE-ROOM///////////

      .addCase(deleteRoom.fulfilled, (state, action) => {

        state.room = state.room.filter((item) => item.\_id !== action.payload);

        state.loader = false;

      })

      .addCase(deleteRoom.pending, (state, action) => {

        state.loader = true;

      })

      .addCase(deleteRoom.rejected, (state, action) => {

        state.loader = false;

        console.log("slice rejected");

      })

      //////////ADD-USER-ROOM///////////

      .addCase(addUserRoom.fulfilled, (state, action) => {

        state.room = state.room.map((item) => {

          if (item.\_id === action.payload.\_id) {

            return (item = action.payload);

          }

          return item;

        });

        state.loader = false;

      })

      .addCase(addUserRoom.pending, (state, action) => {

        state.loader = true;

      })

      .addCase(addUserRoom.rejected, (state, action) => {

        state.loader = false;

        console.log("slice rejected");

      })

      //////////DELETE-USER-ROOM///////////

      .addCase(deleteUser.fulfilled, (state, action) => {

        state.loader = false;

        state.room = state.room.map((item) => {

          if (item.\_id === action.payload.\_id) {

            return (item = action.payload);

          }

          return item;

        });

      })

      .addCase(deleteUser.pending, (state) => {

        state.loader = true;

      })

      .addCase(deleteUser.rejected, (state) => {

        state.loader = false;

        console.log("slice rejected");

      })

      //////////ADD-ROOM///////////

      .addCase(createRoom.fulfilled, (state, action) => {

        state.room.push(action.payload);

        state.loader = false;

      })

      .addCase(createRoom.pending, (state) => {

        state.loader = true;

      })

      .addCase(createRoom.rejected, (state) => {

        state.loader = false;

        console.log("slice rejected");

      });

    //////////ADD-COMMENT///////////

  },

});

export default roomSlice.reducer;

## **chatform.module.scss**

@import "../../index.scss";

.window {

  margin: auto;

  width: 70%;

  background-color: $b-green;

  border-radius: 10px;

  padding: 1px 20px 10px;

  box-sizing: border-box;

  margin-bottom: 20px;

  color: #fff;

  cursor: pointer;

  transition: .8s;

  &:hover {

    background-color: rgba(12, 59, 46, 0.8);

  }

  h1 {

    span {

      font-size: 25px;

      font-weight: 300;

      font-style: normal;

    }

  }

  button {

    background-color: $yellow;

    padding: 6px 20px;

    border-radius: 5px;

    cursor: pointer;

    transition: .5s;

    &:hover {

      background-color: red;

      color: #fff;

    }

  }

}

## **chatmessage.module.scss**

@import "../../index.scss";

.bg {

  min-height: 100vh;

  background: linear-gradient(to right, rgb(0 0 0 / 0.8), rgb(0 0 0 / 0)),

    url("../../images/chat-bg.jpg");

  background-repeat: no-repeat;

  background-position: center;

  background-attachment: fixed;

  background-size: cover;

  padding: 100px 0 0 50px;

  @media screen and (max-width: 500px) {

    padding-left: 0;

  }

}

.chatWindow {

  display: flex;

  width: 100%;

  @media screen and (max-width: 800px) {

    flex-direction: column;

    width: 100%;

    align-items: center;

    justify-content: center;

  }

}

.main {

  width: 40%;

  border-radius: 8px;

  background-color: $b-green;

  height: 80vh;

  position: relative;

  display: flex;

  flex-direction: column;

  @media screen and (max-width: 800px) {

    flex-direction: column;

    width: 90%;

    // align-items: center;

    // justify-content: center;

    // margin: auto;

  }

  .header {

    position: relative;

    padding: 5px 20px;

    display: flex;

    justify-content: space-between;

    align-items: center;

    color: #fff;

    img {

      filter: invert(100%);

      width: 40px;

      height: 40px;

    }

    span {

      cursor: pointer;

    }

  }

  .chatForm {

    height: 82%;

    overflow: auto;

    padding: 10px 15px;

    box-sizing: border-box;

    &::-webkit-scrollbar {

      width: 0;

    }

    .scroll {

      height: 600px;

      &::-webkit-scrollbar {

        width: 0;

      }

    }

    .body {

      display: flex;

      flex-direction: column;

      box-sizing: border-box;

      .you {

        min-width: 20%;

        max-width: 48%;

        background-color: $w-green;

        align-self: flex-end;

        @media screen and (max-width: 800px) {

          width: 50%;

        }

        .comment\_footer {

          flex-direction: row-reverse;

        }

      }

      .outher {

        min-width: 20%;

        max-width: 40%;

        background-color: $orange;

        align-self: flex-start;

        @media screen and (max-width: 800px) {

          width: 50%;

        }

      }

    }

  }

}

.message {

  word-break: break-all;

}

.body div {

  color: #fff;

  position: relative;

  padding: 5px 5px 0 5px;

  margin-bottom: 20px;

  border-radius: 10px;

}

.comment\_footer {

  width: 100%;

  position: absolute;

  bottom: -15px;

  display: flex;

  justify-content: space-between;

  color: #fff;

}

.footer {

  width: 90%;

  height: 10%;

  position: absolute;

  bottom: 0;

  display: flex;

  justify-content: center;

  align-self: center;

  input {

    width: 100%;

    height: 80%;

    padding: 2px 15px;

    border: 2px solid transparent;

    box-sizing: border-box;

    border-radius: 5px;

    outline: none;

  }

}

## **file.module.scss**

.file {

  display: grid;

  grid-template-columns: 1fr 4fr repeat(4, 1fr);

  align-items: center;

  margin-top: 10px;

  &:hover {

    transform: scale(1.01);

    cursor: pointer;

  }

  .name {

    grid-column-start: 2;

  }

  .date {

    grid-column-start: 3;

    justify-self: center;

    white-space: nowrap;

  }

  .size {

    grid-column-start: 5;

    justify-self: center;

  }

}

## **fileList.module.scss**

@import "../../../index.scss";

.fileList {

  margin: 20px 0;

  padding: 10px;

  background-color: $w-green;

  border-radius: 8px;

  .header {

    color: $yellow;

    display: grid;

    grid-template-columns: 1fr 4fr repeat(4, 1fr);

    div {

      font-size: 20px;

      font-weight: 700;

      letter-spacing: 2px;

    }

    .name {

      grid-column-start: 2;

    }

    .date {

      grid-column-start: 3;

      justify-self: center;

    }

    .size {

      grid-column-start: 5;

      justify-self: center;

    }

  }

}

## **popup.module.scss**

@import "../../../index.scss";

.popup {

  position: absolute;

  display: flex;

  justify-content: center;

  align-items: center;

  width: 100%;

  height: 100vh;

  background-color: rgba(0, 0, 0, 0.5);

  right: 0;

  left: 0;

  top: 0;

  bottom: 0;

  margin-top: 20px;

  .content {

    width: 400px;

    background: #fff;

    padding: 20px;

    border-radius: 8px;

    .header {

      margin-bottom: 20px;

      display: flex;

      justify-content: space-between;

      align-items: center;

      .title {

        font-size: 18px;

        display: flex;

        flex-direction: column;

        span:nth-child(2) {

          font-size: 12px;

          color: $orange;

        }

      }

      button {

        background-color: unset;

        align-self: flex-start;

        transition: 0.4s;

        &:hover {

          color: $b-green;

          font-weight: 700;

        }

      }

    }

    .create {

      align-self: flex-end;

      height: 100%;

      padding: 5px 10px;

      border-radius: 5px;

      background-color: $w-green;

      transition: 0.3s;

      &:hover {

        color: $yellow;

        background-color: $b-green;

      }

    }

    input {

      border-radius: 5px;

      padding: 5px 12px;

      outline: none;

      margin-right: 8px;

    }

  }

}

button {

  cursor: pointer;

}

## **disk.module.scss**

.disk {

  margin: 20px;

  .buttons {

    display: flex;

    button {

      height: min-content;

      font-size: 16px;

      padding: 4px 16px;

      border-radius: 8px;

      border: none;

      white-space: nowrap;

    }

    .create {

      margin-left: 20px;

    }

  }

}

## **footer.module.scss**

@import "../../index.scss";

footer {

  width: 100%;

  height: 200px;

  padding: 20px 40px;

  color: #fff;

  border-top: 2px solid #fff;

  display: flex;

  justify-content: space-around;

  align-items: center;

  background-color: $b-blue;

  ul {

    list-style: none;

  }

  .trend,

  .contact {

    li {

      line-height: 30px;

      text-transform: uppercase;

      font-weight: 500;

      color: #fff;

      border-bottom: 2px solid transparent;

      overflow: visible;

      white-space: nowrap;

      cursor: pointer;

      &:hover {

        animation-duration: 2s;

        animation-name: li\_underline;

      }

    }

  }

  .follow {

    margin: 0 auto;

    text-align: center;

    font-size: 32px;

    display: flex;

    flex-direction: column;

    align-items: center;

    align-content: center;

    ul {

      display: flex;

      margin: 0;

      padding: 0;

    }

  }

}

@keyframes li\_underline {

  0% {

    border-bottom: 2px solid #01c38d;

    width: 0%;

  }

  100% {

    border-bottom: 2px solid #01c38d;

    width: 100%;

  }

}

## **header.module.scss**

@import "../../index.scss";

.header {

  width: 100%;

  display: flex;

  align-items: center;

  justify-content: space-between;

  position: absolute;

  top: 0;

  left: 0;

  z-index: 2;

  .logo {

    width: 100px;

    height: 60px;

    margin-left: 50px;

    display: flex;

    align-items: center;

    object-fit: cover;

    img {

      filter: invert(100%);

      height: 44px;

      transition: 0.4s;

      cursor: pointer;

      &:hover {

        scale: 1.1;

        filter: invert(70%);

      }

    }

  }

  .nav\_top {

    width: 80%;

    display: flex;

    justify-content: flex-end;

    ul {

      display: flex;

      align-items: center;

      width: 70%;

      list-style-type: none;

      justify-content: space-between;

      margin-right: 5%;

      .link {

        text-decoration: none;

        li {

          text-transform: uppercase;

          font-weight: 500;

          color: #20c83e;

          transition: 0.2s;

          border-bottom: 2px solid transparent;

          cursor: pointer;

          &:hover {

            margin-bottom: 5px;

            border-bottom: 2px solid #01c38d;

          }

        }

      }

      .active {

        margin-bottom: 5px;

        border-bottom: 2px solid #01c38d;

        li {

          &:hover {

            margin-bottom: 0;

            border-bottom: 2px solid transparent;

          }

        }

      }

      .link\_off {

        text-transform: uppercase;

        font-weight: 500;

        color: silver;

        border-bottom: none;

        cursor: default;

        &:hover {

          margin-bottom: 0;

          border-bottom: 0;

        }

      }

    }

  }

}

.link\_mg {

  margin-left: 100px;

}

## **signin.module.scss**

.container {

  background-color: #1f2029;

  height: 100vh;

  display: flex;

  justify-content: center;

  position: relative;

  z-index: 1;

  h1 {

    font-size: 20px;

    text-align: center;

    color: #fff;

  }

  .block\_signIn {

    display: flex;

    flex-direction: column;

    align-items: center;

    justify-content: center;

    .block\_two {

      height: 500px;

      width: 380px;

      padding-top: 15px;

      background-color: #2a2b38;

      border-radius: 10px;

      position: relative;

      text-align: center;

      & > img {

        height: 35px;

        filter: invert(100%);

      }

      .link {

        margin-top: 40px;

        .l {

          text-decoration: none;

          color: #fff;

          transition: .5s;

          &:hover{

            color: #929292;

          }

        }

      }

      .form\_signIn {

        position: relative;

        display: flex;

        flex-direction: column;

        align-items: center;

        justify-content: center;

        margin-top: 10px;

        font-family: "Montserrat", sans-serif;

        input {

          height: 40px;

          width: 90%;

          margin-bottom: 5px;

          font-weight: 100;

          border-radius: 5px;

          background-color: #1f2029;

          border-right: none;

          border-bottom: none;

          color: bisque;

          padding-left: 20px;

          outline: 0;

        }

      }

      .link > img {

        width: 28px;

        height: 25px;

        position: absolute;

        filter: invert(100);

        top: 20px;

        left: 15px;

      }

    }

  }

}

.btn {

  width: 90%;

  cursor: pointer;

  margin-top: 20px;

  padding: 8px 0;

  font-weight: bold;

  font-size: 16px;

  border-radius: 10px;

  transition: 0.3s;

  &:hover {

    background-color: #359999;

    color: #fff;

  }

}

.eyeForPasswor {

  position: absolute;

  top: 56px;

  right: 30px;

  width: 20px;

  height: 20px;

  cursor: pointer;

  filter: invert(100%);

  z-index: 10;

}

.signup\_block\_signUp {

  width: 10px;

}

## **maininfo.module.scss**

@import "../../index.scss";

.container {

  background-color: $b-blue;

  width: 100%;

  padding: 0 100px;

  min-height: 100vh;

  color: #fff;

  display: flex;

  justify-content: flex-start;

  overflow: hidden;

  p {

    width: 40%;

    font-size: 36px;

    font-weight: 500;

    margin-left: 40px;

    align-self: center;

  }

  .img\_block\_1 {

    width: 40%;

    position: relative;

    img {

      position: absolute;

      top: 150px;

      left: 0;

      width: 250px;

      &:nth-child(2) {

        left: 80px;

        top: 250px;

      }

      &:nth-child(3) {

        left: 160px;

        top: 350px;

      }

    }

  }

  .img\_block\_2 {

    width: 40%;

    display: flex;

    align-items: center;

    img {

      width: 100%;

      margin-left: 100px;

      object-fit: cover;

    }

  }

}

## **modal.module.scss**

@import "../../index.scss";

.modal {

  position: fixed;

  top: 10%;

  left: 35%;

  background-color: $w-green;

  padding: 40px;

  width: 300px;

  border-radius: 10px;

  z-index: 1;

  .modalName {

    font-size: 18px;

    font-weight: 400;

    margin-bottom: 10px;

    color: $yellow;

  }

  .inputBlock {

    display: flex;

    justify-content: space-between;

    width: 100%;

    input {

      border-radius: 5px;

      border: none;

      outline: none;

      padding: 4px 8px;

      width: 70%;

    }

    button {

      border-radius: 5px;

      color: #fff;

      background-color: $b-green;

      cursor: pointer;

    }

  }

}

## **signup.module.scss**

.container {

  background-color: #1f2029;

  height: 100vh;

}

.block\_signUp {

  display: flex;

  flex-direction: column;

  align-items: center;

  justify-content: center;

  height: 100vh;

}

.block\_two {

  height: 500px;

  width: 380px;

  padding-top: 15px;

  color: #fff;

  background-color: #2a2b38;

  border-radius: 10px;

  position: relative;

  text-align: center;

  & > img {

    height: 35px;

    filter: invert(100%);

  }

  h1 {

    font-size: 20px;

    text-align: center;

    color: #fff;

  }

  .link {

    margin-top: 40px;

    .l {

      text-decoration: none;

      color: #fff;

      transition: 0.5s;

      &:hover {

        color: #929292;

      }

    }

  }

}

.form\_signUp {

  position: relative;

  display: flex;

  flex-direction: column;

  align-items: center;

  justify-content: center;

  width: 100%;

  margin-top: 10px;

  font-family: "Montserrat", sans-serif;

  input {

    height: 40px;

    width: 90%;

    margin-bottom: 5px;

    font-weight: 100;

    border-radius: 5px;

    background-color: #1f2029;

    border-right: none;

    border-bottom: none;

    color: bisque;

    padding-left: 20px;

    padding-left: 20px;

    outline: none;

  }

}

.eyeForPassword {

  position: absolute;

  top: 145px;

  right: 30px;

  width: 20px;

  height: 20px;

  cursor: pointer;

  filter: invert(100%);

}

.title {

  margin-top: 20px;

  margin-bottom: 0;

  text-align: center;

  font-weight: bold;

  color: #999;

}

.btn {

  cursor: pointer;

  margin-top: 20px;

  padding: 10px 40px;

  font-weight: bold;

  font-size: 20px;

  height: 40px;

  border-radius: 100px;

  transition: 0.3s;

  &:hover {

    background-color: #359999;

    color: #fff;

  }

}

.signup\_block\_signUp {

  width: 10px;

}

.link > img {

  width: 28px;

  height: 25px;

  position: absolute;

  filter: invert(100);

  top: 20px;

  left: 15px;

}

## **usersinroom.module.scss**

@import "../../index.scss";

.users {

  margin: 40px 0 0 10px;

  .you {

    margin-bottom: 5px;

    padding: 5px 20px;

    border-radius: 5px;

    background-color: $b-green;

    color: $yellow;

    cursor: default;

  }

  .body {

    display: flex;

    img {

      height: 24px;

      width: 24px;

      cursor: pointer;

      filter: invert(100%);

    }

    .outher {

      margin-bottom: 5px;

      padding: 5px 20px;

      border-radius: 5px;

      background-color: $b-green;

      color: #fff;

      cursor: default;

    }

  }

  .head {

    color: #fff;

    margin-bottom: 10px;

    font-size: 24px;

    font-weight: 700;

  }

}

## **usersModal.module.scss**

@import '../../index.scss';

.modal{

    position: absolute;

    background-color: #fff;

    border-radius: 15px;

    width: 70%;

    color: black;

    z-index: 100;

    padding: 5px 10px 0px 10px;

    top: 30px;

    right: 20px;

    .name{

        margin-bottom: 5px;

        border-radius: 10px;

        padding: 5px 10px;

        background-color: $b-green;

        color: $orange;

        cursor: pointer;

        transition: .4s;

        &:hover{

            color: $yellow;

        }

    }

    .close{

        text-align: end;

        margin-bottom: 10px;

    }

}

## **chatWindow.module.scss**

@import "../../index.scss";

.main {

  padding-top: 80px;

  min-height: 100vh;

  background: linear-gradient(

    to right,

    rgb(0 0 0 / 0.7),

    rgb(0 0 0 / 0)),

    url("../../images/room-bg.jpg");

  background-repeat: no-repeat;

  background-size: cover;

  .header {

    display: flex;

    justify-content: flex-end;

    margin-right: 15%;

    margin-bottom: 30px;

    button {

      font-size: 18px;

      padding: 5px 15px;

      border-radius: 5px;

      background-color: $b-green;

      color: $orange;

      cursor: pointer;

      &:hover {

        opacity: 0.9;

      }

    }

  }

}

## **formRegistration.module.scss**

\* {

  box-sizing: border-box;

}

.main {

  background: linear-gradient(rgba(255, 255, 255, 0.1), rgba(0, 0, 0, 0.9)),

    url(../../images/bg-img-2.jpg);

  background-repeat: no-repeat;

  background-attachment: fixed;

  min-height: 100vh;

  display: flex;

  color: #fff;

  flex-direction: column;

  align-items: center;

  justify-content: center;

  font-family: -apple-system, "Segoe UI", Roboto, Ubuntu, "SF Pro Text",

    system-ui, "Helvetica Neue", sans-serif;

  img {

    filter: invert(100%);

  }

  button {

    width: 250px;

    border-radius: 6px;

    font-size: 14px;

    font-weight: 400;

    text-transform: uppercase;

    text-align: center;

    padding: 12px;

    cursor: pointer;

  }

  h1 {

    font-size: 60px;

    font-weight: 500;

  }

  .signin {

    font-size: 16px;

    background-color: rgb(50, 58, 75);

    color: #fff;

    margin-bottom: 20px;

    border: 1px solid transparent;

    &:hover {

      background-color: inherit;

      border: 1px solid #fff;

    }

  }

  .signup {

    background-color: inherit;

    color: #fff;

    border: 1px solid #fff;

    &:hover {

      background-color: rgb(50, 58, 75);

      border: 1px solid transparent;

    }

  }

}

## **mainPage.module.scss**

.main {

  width: 100%;

  height: 100vh;

  background-image: url("../../images/bg-img-3.jpg");

  background-size: cover;

  background-repeat: no-repeat;

  background-position: bottom;

  background-origin: unset;

  overflow: hidden;

  position: fixed;

  z-index: -10;

}

.container\_main {

  position: relative;

  display: flex;

  justify-content: center;

  align-items: center;

  flex-direction: column;

  width: 90%;

  height: 150vh;

  margin: 0 auto;

  .logo {

    position: fixed;

    top: 0px;

    display: flex;

    flex-direction: column;

    justify-content: center;

    align-items: center;

    margin: 0 auto;

    padding-top: 70px;

    min-height: 100vh;

    max-height: 100vh;

    #logo {

      filter: invert(100%);

      width: 220px;

      object-fit: contain;

    }

    #logo\_text {

      margin-top: -50px;

      text-align: center;

      font-size: 46px;

      color: #fff;

      width: 260px;

      object-fit: contain;

    }

    .title {

      height: 100px;

      width: 75%;

      margin-top: -25px;

      opacity: 0;

      text-align: center;

      font-size: 28px;

      font-weight: 300;

      line-height: 35px;

      color: #fff;

      transform: scale(0.6);

    }

  }

}

## **profilepage.module.scss**

@import "../../index.scss";

.container {

  padding-top: 200px;

  background: linear-gradient(to right, rgb(0 0 0 / 0.7), rgb(0 0 0 / 0)),

    url("../../images/profile-bg.jpg");

  background-repeat: no-repeat;

  background-position: center;

  background-size: cover;

  min-height: 100vh;

  background-attachment: fixed;

  display: flex;

  padding-left: 160px;

  @media screen and (max-width: 800px) {

    padding-left: 0;

  }

  .main {

    height: 300px;

    color: red;

    display: flex;

    @media screen and (max-width: 800px) {

      flex-direction: column;

      margin: 0 auto;

    }

    .avatar {

      width: 300px;

      margin: auto;

      display: flex;

      flex-direction: column;

      background-color: rgb(222, 221, 221);

      border-radius: 15px;

      img {

        height: 200px;

        object-fit: cover;

      }

      .buttons {

        display: flex;

        flex-direction: column;

        align-items: center;

        justify-content: center;

        margin-bottom: 20px;

        button {

          padding: 5px 20px;

          border-radius: 5px;

          font-size: 16px;

          background-color: $orange;

          color: $b-green;

          cursor: pointer;

          transition: 0.6s;

          &:hover {

            background-color: $b-green;

            color: $yellow;

          }

        }

      }

    }

  }

  .info {

    margin: auto auto auto 50px;

    display: flex;

    flex-direction: column;

    @media screen and (max-width: 800px) {

      margin: 40px auto auto;

    }

    input {

      border: none;

      outline: none;

      font-size: 24px;

      margin-bottom: 14px;

      background-color: rgb(222, 221, 221);

      border-radius: 5px;

      padding: 4px 10px;

    }

    button {

      padding: 10px 15px;

      border-radius: 5px;

      font-size: 18px;

      background-color: $orange;

      color: $b-green;

      cursor: pointer;

      transition: 0.6s;

      &:hover {

        background-color: $b-green;

        color: $yellow;

      }

    }

  }

}

## **chatinfo.module.scss**

@import "../../index.scss";

.main {

  margin-top: 60px;

  .window {

    position: relative;

    width: 70%;

    padding: 20px;

    margin: auto;

    border-radius: 8px;

    background-color: $b-green;

    color: $yellow;

    .header {

      margin-bottom: 20px;

      display: flex;

      .info {

        display: flex;

        flex-direction: column;

      }

      .image {

        border-radius: 100%;

        background-color: $w-green;

        width: 100px;

        height: 100px;

        margin-right: 40px;

        overflow: hidden;

        img{

          width: 100%;

        }

      }

    }

    .addUser {

      display: flex;

      align-items: center;

      padding: 5px 20px;

      border-radius: 8px;

      font-size: 16px;

      transition: .3s;

      cursor: pointer;

      &:hover{

        background-color: $w-green;

        color: #fff;

      }

    }

  }

}

## **index.scss**

$b-green: rgb(12, 59, 46);

$w-green: #6d9773;

$b-blue: #001427;

$orange: #bb8a52;

$yellow: #ffba00;

body {

  margin: 0;

  padding: 0;

  width: 100%;

}

body::-webkit-scrollbar {

  width: 0px;

}

button {

  border: none;

}

li:hover,

span:hover,

p:hover,

h1:hover,

h2:hover,

h3:hover,

h4:hover,

h5:hover,

h6:hover {

  cursor: default;

}

## **package.json**

{

  "name": "apposible-front",

  "version": "0.1.0",

  "private": true,

  "dependencies": {

    "@reduxjs/toolkit": "^1.8.6",

    "@testing-library/jest-dom": "^5.16.5",

    "@testing-library/react": "^13.4.0",

    "@testing-library/user-event": "^13.5.0",

    "axios": "^1.1.3",

    "framer-motion": "^7.6.4",

    "gsap": "^3.11.4",

    "react": "^18.2.0",

    "react-dom": "^18.2.0",

    "react-icons": "^4.8.0",

    "react-redux": "^8.0.4",

    "react-router-dom": "^6.4.3",

    "react-scripts": "5.0.1",

    "react-scroll-to-bottom": "^4.2.0",

    "react-toastify": "^9.1.0",

    "sass": "^1.55.0",

    "socket.io-client": "^4.5.3",

    "web-vitals": "^2.1.4"

  },

  "scripts": {

    "start": "react-scripts start",

    "build": "react-scripts build",

    "test": "react-scripts test",

    "eject": "react-scripts eject"

  },

  "eslintConfig": {

    "extends": [

      "react-app",

      "react-app/jest"

    ]

  },

  "browserslist": {

    "production": [

      ">0.2%",

      "not dead",

      "not op\_mini all"

    ],

    "development": [

      "last 1 chrome version",

      "last 1 firefox version",

      "last 1 safari version"

    ]

  }

}

**Содержание файлов Back-end:**

## **file.controller.js**

const fileService = require("../services/file.service");

const File = require("../models/File.model");

module.exports.fileController = {

  createDir: async (req, res) => {

    try {

      const { name, type, parent, room } = req.body;

      const file = await File.create({

        name,

        type,

        parent,

        room,

      });

      const parentFile = await File.findOne({ \_id: parent });

      if (!parentFile) {

        file.path = name;

        await fileService.createDir(file);

      } else {

        file.path = `${parentFile.path}\\${file.name}`;

        await fileService.createDir(file);

        parentFile.childs.push(file.\_id);

        await parentFile.save();

      }

      await file.save();

      return res.json(file);

    } catch (error) {

      return res.status(400).json(error);

    }

  },

  getFiles: async (req, res) => {

    try {

      const files = await File.find({

        room: req.query.room,

        parent: req.query.parent,

      });

      return res.json(files);

    } catch (error) {

      res.status(500).json({ message: "can not get files" });

    }

  },

};

## **room.controller.js**

const Room = require("../models/Room.model");

const fileService = require("../services/file.service");

const File = require("../models/File.model");

module.exports.roomController = {

  createRoom: async (req, res) => {

    try {

      const { name } = req.body;

      const data = new Room({

        name,

        access: req.params.id,

        admin: req.params.id,

      });

      data.save();

      await fileService.createDir(new File({ room: data.\_id, name: "" }));

      const result = await data.populate("access admin");

      res.json(result);

    } catch (error) {

      res.json(error);

    }

  },

  getRoom: async (req, res) => {

    try {

      const data = await Room.find().populate("users.user access admin");

      res.json(data);

    } catch (error) {

      res.json(error);

    }

  },

  deleteRoom: async (req, res) => {

    try {

      const data = await Room.findByIdAndDelete(req.params.id);

      res.json(data);

    } catch (error) {

      res.json(error);

    }

  },

  addComment: async (req, res) => {

    try {

      const { user, comment, time } = req.body;

      const room = await Room.findById(req.params.id);

      const result = room.access.filter((item) => item.toString() === user);

      if (result.toString()) {

        const data = await Room.findByIdAndUpdate(req.params.id, {

          $push: { users: { user: user, comment: comment, time: time } },

        });

        res.json(data.users);

      } else {

        res.json("Такой пользователь не найден");

      }

    } catch (error) {

      res.json(error);

    }

  },

  addUserToRoom: async (req, res) => {

    try {

      const data = await Room.findByIdAndUpdate(

        req.params.id,

        {

          $addToSet: { access: req.body.user },

        },

        { new: true }

      ).populate("access");

      return res.json(data);

    } catch (error) {

      res.json(error);

    }

  },

  deleteUserRoom: async (req, res) => {

    try {

      const data = await Room.findByIdAndUpdate(

        req.params.id,

        {

          $pull: { access: req.body.user },

        },

        { new: true }

      ).populate("access");

      return res.json(data);

    } catch (error) {

      res.json(error);

    }

  },

};

## **user.controller.js**

const User = require("../models/User.model");

const bcrypt = require("bcrypt");

const jwt = require("jsonwebtoken");

module.exports.userController = {

  registration: async (req, res) => {

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

    try {

      const hash = await bcrypt.hash(password, +process.env.BCRYPT\_ROUNDS);

      const user = new User({

        password: hash,

        login,

        name,

        email,

        avatar,

      });

      await user.save();

      res.json(user);

    } catch (error) {

      res.json(error);

    }

  },

  login: async (req, res) => {

    const { login, password } = req.body;

    const candidate = await User.findOne({ login });

    if (!candidate) {

      return res.status(401).json("Неверный логин");

    }

    const valid = await bcrypt.compare(password, candidate.password);

    if (!valid) {

      return res.status(401).json("Неверный пароль");

    }

    const payload = {

      id: candidate.\_id,

      login: candidate.login,

    };

    const user = candidate;

    const token = await jwt.sign(payload, process.env.SECRET\_JWT\_KEY, {

      expiresIn: "7d",

    });

    return res.json({ token, user });

  },

  getUser: async (req, res) => {

    const data = await User.find();

    res.json(data);

  },

  deleteUser: async (req, res) => {

    const data = await User.findByIdAndDelete(req.params.id);

    res.json(data);

  },

  changeUser: async (req, res) => {

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

    const data = await User.findByIdAndUpdate(

      req.params.id,

      {

        name,

        email,

        avatar,

      },

      { new: true }

    );

    res.json(data);

  },

};

## **auth.middleware.js**

const jwt = require("jsonwebtoken");

module.exports = async (req, res, next) => {

  const { authorization } = req.headers;

  const [type, token] = authorization.split(" ");

  // проверка authorization в header

  if (!authorization) {

    return res.status(401).json("Ошибка авторизации");

  }

  // проверка type token'a

  if (type !== "Bearer") {

    return res.status(401).json("Неверный тип токена");

  }

  try {

    const decoder = await jwt.verify(token, process.env.SECRET\_JWT\_KEY);

    req.user = decoder;

    next();

  } catch (error) {

    return res.status(401).json(error.toString());

  }

};

## **File.model.js**

const mongoose = require("mongoose");

const fileSchema = mongoose.Schema({

  name: {

    type: String,

    required: true,

  },

  type: {

    type: String,

    required: true,

  },

  accessLink: String,

  size: {

    type: Number,

    default: 0,

  },

  path: {

    type: String,

    default: "",

  },

  date: {

    type: Date,

    default: Date.now(),

  },

  user: {

    type: mongoose.SchemaTypes.ObjectId,

    ref: "User",

  },

  room: {

    type: mongoose.SchemaTypes.ObjectId,

    ref: "Room",

  },

  parent: {

    type: mongoose.SchemaTypes.ObjectId,

    ref: "File",

  },

  childs: [

    {

      type: mongoose.SchemaTypes.ObjectId,

      ref: "File",

    },

  ],

});

const File = mongoose.model("File", fileSchema);

module.exports = File;

## **Room.model.js**

const mongoose = require("mongoose");

const roomSchema = mongoose.Schema({

  name: String,

  users: [

    {

      user: {

        type: mongoose.SchemaTypes.ObjectId,

        ref: "User",

      },

      comment: {

        type: String,

      },

      time: {

        type: String,

      },

    },

  ],

  access: [

    {

      type: mongoose.SchemaTypes.ObjectId,

      ref: "User",

    },

  ],

  admin: {

    type: mongoose.SchemaTypes.ObjectId,

    ref: "User",

  },

});

const Room = mongoose.model("Room", roomSchema);

module.exports = Room;

## **User.model.js**

const mongoose = require("mongoose");

const userSchema = mongoose.Schema({

  name: String,

  avatar: String,

  email: {

    type: String,

    required: true,

  },

  login: {

    type: String,

    required: true,

  },

  password: {

    type: String,

    required: true,

  },

});

const User = mongoose.model("User", userSchema);

module.exports = User;

## **file.route.js**

const Router = require("express");

const router = Router();

const authMiddleware = require("../middleware/auth.middleware");

const {fileController} = require("../controllers/file.controller");

router.post("", authMiddleware, fileController.createDir);

router.get("", authMiddleware, fileController.getFiles);

module.exports = router;

## **room.route.js**

const Router = require("express");

const router = Router();

const authMiddleware = require("../middleware/auth.middleware");

const { roomController } = require("../controllers/room.controller");

router.get("", authMiddleware, roomController.getRoom);

router.post("/create/:id", authMiddleware, roomController.createRoom);

router.delete("/delete/:id", authMiddleware, roomController.deleteRoom);

router.patch("/addComment/:id", authMiddleware, roomController.addComment);

router.patch("/addUser/:id", roomController.addUserToRoom);

router.patch("/deleteUser/:id", authMiddleware, roomController.deleteUserRoom);

module.exports = router;

## **user.route.js**

const { Router } = require("express");

const router = Router();

const { userController } = require("../controllers/user.controller");

const authMiddleware = require("../middleware/auth.middleware");

router.get("/users", userController.getUser);

router.post("/registr", userController.registration);

router.post("/login", userController.login);

router.delete("/user/:id", authMiddleware, userController.deleteUser);

router.patch("/user/:id", authMiddleware, userController.changeUser);

module.exports = router;

## **file.service.js**

require("dotenv").config();

const fs = require("fs");

const File = require("../models/File.model");

class FileService {

  createDir(file) {

    const filePath = `${process.env.FILE\_PATH}\\${file.room}\\${file.path}`;

    return new Promise((resolve, reject) => {

      try {

        if (!fs.existsSync(filePath)) {

          fs.mkdirSync(filePath);

          return resolve({ message: "File was created" });

        } else {

          return reject({ message: "File already exist" });

        }

      } catch (error) {

        return reject({ message: "File error", error });

      }

    });

  }

}

module.exports = new FileService();

## **index.js**

require("dotenv").config();

const express = require("express");

const app = express();

const path = require("path");

const cors = require("cors");

const morgan = require("morgan");

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

const mongoose = require("mongoose");

const http = require("http");

const { Server } = require("socket.io");

const userRouter = require("./routes/user.route");

const fileRouter = require("./routes/file.route");

const roomRouter = require("./routes/room.route");

app.use(fileUpload({}));

app.use(cors());

app.use(express.json());

app.use(morgan("dev"));

app.use(express.static(path.join(\_\_dirname, "public")));

app.use("/public", express.static(path.resolve(\_\_dirname, "public")));

//Routes

app.use("/api/auth", userRouter);

app.use("/api/files", fileRouter);

app.use("/room", roomRouter);

//MongooseConnect

mongoose

  .connect(process.env.MONGO\_SERVER)

  .then(() => console.log("mongoose connect"))

  .catch(() => console.log("mongoose warning"));

//File

//Chat

const server = http.createServer(app);

const io = new Server(server, {

  cors: {

    origin: `http://localhost:3000`,

    methods: ["GET", "POST"],

  },

});

io.on("connection", (socket) => {

  console.log(`User connected: ${socket.id}`);

  socket.on("join\_room", (data) => {

    socket.join(data);

    console.log(`User with ID: ${socket.id} joined room: ${data}`);

  });

  socket.on("send\_message", (data) => {

    socket.to(data.room).emit("receive\_message", data);

  });

  socket.on("disconnect", () => {

    console.log("User disconnected", socket.id);

  });

});

server.listen(process.env.PORT, () => {

  console.log(`Сервер запущен на порте ${process.env.PORT}`);

});

## **serverUrl.js**

const { NODE\_ENV } = process.env;

export let serverUrl;

if (NODE\_ENV === "development") {

  serverUrl = "http://localhost:3001"; // адрес сервера на локалке

} else {

  serverUrl = ""; // адрес сервера после выгрузки

}

## **package.json**

{

  "name": "apposible-back",

  "version": "1.0.0",

  "description": "",

  "main": "index.js",

  "scripts": {

    "start": "nodemon index.js",

    "test": "echo \"Error: no test specified\" && exit 1"

  },

  "keywords": [],

  "author": "",

  "license": "ISC",

  "dependencies": {

    "bcrypt": "^5.1.0",

    "config": "^3.3.8",

    "cors": "^2.8.5",

    "dotenv": "^16.0.3",

    "express": "^4.18.2",

    "express-fileupload": "^1.4.0",

    "express-validator": "^6.14.2",

    "file-type": "^18.0.0",

    "fs": "^0.0.1-security",

    "jsonwebtoken": "^8.5.1",

    "mongoose": "^6.7.1",

    "morgan": "^1.10.0",

    "multer": "^1.4.5-lts.1",

    "socket.io": "^4.5.3"

  },

  "devDependencies": {

    "nodemon": "^2.0.20"

  }

}