# САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №2

Выполнил:

Жаров Александр Павлович

Группа: К33402

Проверил: Добряков Д. И.

Санкт-Петербург

#### Задача

Расширить функционал на сайте к первой лабораторной. Связать все данные с бд и добавить запросы к API.

### Ход работы

Для начала был создан сервер на Node js на котором написан функционал для авторизации и собственное арі, с помощью которого сервер общается с MongoDB.

```
You, 6 days ago | 1 author (You)
    const express = require("express");
const cors = require("cors"); 4.5k (gzipped: 1.9k)
3 const mongoose = require("mongoose"); 872.8k (gzipped: 235.7k)
4 const apiRouter = require("./routers/apiRouter");
    const authRouter = require("./routers/authRouter");
    const app = express();
    const PORT = 3030;
    const db = "mongodb://127.0.0.1:27017/portfolio";
11
12
    mongoose
      .connect(db)
       .then((res) => console.log("Connect to DB"))
       .catch((error) => console.log(error));
17
    app.use(cors());
19 \times app.listen(PORT, (error) => {
      error ? console.log(error) : console.log(`listening port ${PORT}`);
21
    });
    app.use(express.urlencoded({ extended: false }));
    app.use(express.json());
    app.use("/", authRouter);
    app.use("/api", apiRouter);
```

```
const Router = require("express");
const authRouter = new Router();
const authController = require("../controllers/authController");

authRouter.post("/registration", authController.registration);

authRouter.post("/login", authController.login);

module.exports = authRouter;
```

Процесс авторизации выглядит следующим образом: сначала пользователь заполняет форму

## Создание аккаунта

Имя		
имя		
Почта		
email@example.com		
Пароль		
Пароль		
	Уже есть аккаунт	Создать аккаунт

При нажатии на кнопку создать аккаунт – отправляется post запрос на сервер с введенными данными.

```
const onRegistration = async () => {
 const res = await fetch("http://localhost:3030/registration", {
   method: "POST",
   headers: {
      "Content-Type": "application/json",
   body: JSON.stringify(formState),
    .then((response) => response.json())
    .then((data) => {
     if (data.token) {
        setCookie("token", JSON.stringify(data.token));
        fetchUser(data.token).then((user) => {
          setUser(user);
          setAuth(true);
         navigate("/");
        });
      } else {
        console.log(data.message);
    })
    .catch((error) => console.log(error));
 return res;
};
```

На сервере происходит обработка запроса, в ходе которой в бд создается пользователь, и генерируется jwt token, который отправляется в ответе на клиент.

```
const genereteAccessToken = (id, email) => {
      const payload = {
        id,
        email,
     };
2
     return jwt.sign(payload, secret, { expiresIn: "12h" }); You, 5 da
    You, 5 days ago | 1 author (You)
    class authController {
      async registration(req, res) {
        try {
          const { email, name, password } = req.body;
          const candidate = await User.findOne({ email });
          if (candidate) {
21
            return res.status(400).json({
              message: "Пользователь с этой почтой уже существует",
            });
          const hashPassword = bcrypt.hashSync(password, 7);
          const user = new User({ email, name, password: hashPassword });
29
          await user.save();
          const token = genereteAccessToken(user._id, user.email);
          return res.json({ token });
34
        } catch (error) {
          console.error(error);
36
          res.status(500).send("Internal Server Error");
```

На клиенте полученный токен сохраняется в куки и отправляется следующий запрос, уже к Арі.

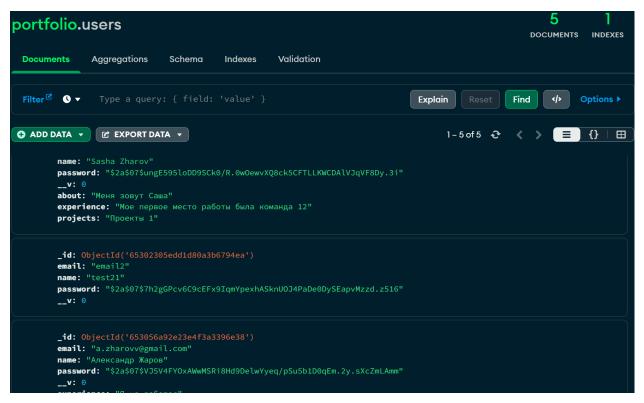
```
useEffect(() => {
 const token = getCookie("token");
 console.log(token);
 if (token && !isAuthenticated) {
   fetchUserCallback(token).then((user) => {
     if (user.message) {
       setAuth(false);
       setUser(null);
       deleteCookie("token");
      } else {
       setAuth(true);
       setUser(user);
       navigate("/");
       console.log(user);
You, 11 hours ago • Uncommit
     }
   });
 // eslint-disable-next-line react-hooks/exhaustive-deps
}, []);
```

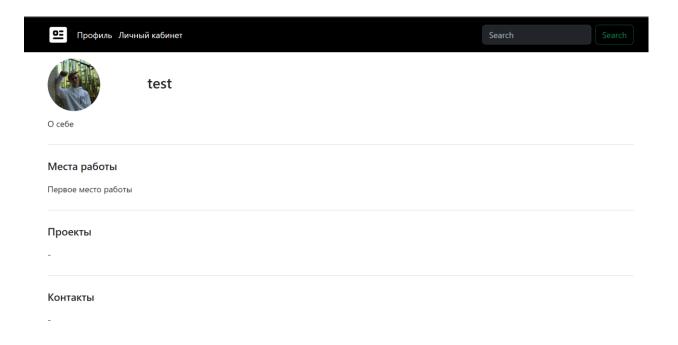
Токен отправляется в хедере запроса, т.к без него арі не вернет результат.

```
async getUserById(req, res) {
   try {
      const user = await User.findById(req.user.id);
      res.send(user);
   } catch (error) {
      console.error(error);
      res.status(500).send("Internal Server Error");
   }
}
```

```
const jwt = require("jsonwebtoken"); 60.8k (gzipped: 18k)
   const { secret } = require("../config");
 v module.exports = function (req, res, next) {
     if (req.method === "OPTIONS") {
       next();
     try {
       const token = req.headers.authorization.split(" ")[1];
       if (!token) {
         return res.status(403).json({ message: "Пользователь не авторизован" });
       const decodeData = jwt.verify(token, secret);
       req.user = decodeData;
8
       next();
      } catch (error) {
       console.log(error);
       return res
          .status(403)
          .json({ message: "Пользователь не авторизован ошибка" });
   };
```

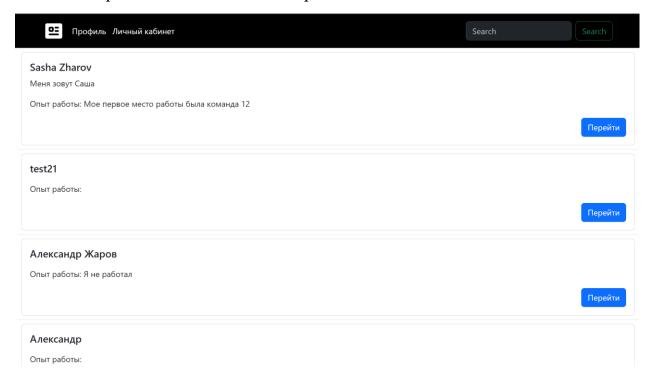
### Информация о пользователе сохранена в БД и выведена на клиенте





При выходе или окончании срока жизни jwt, он удаляется из cookie и приложение запрашивает авторизоваться заново.

Также был реализован поиск по юзерам.



При поиске на сервер отправляется запрос на получение юзеров и данные из строки поиска.

Сервер возвращает результат поиска

```
export async function fetchUsers(token, search) {
  const res = await fetch("http://localhost:3030/api/get-users", {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
     Authorization: `Bearer ${token}`,
    },
    body: JSON.stringify({ search }),
    .then((response) => response.json())
    .then((responseData) => {
      if (responseData) {
      return responseData;
      } else {
       console.log(responseData.message);
    .catch((error) => console.log(error));
  return res;
```

## Вывод:

В ходе выполнения работы я создал свой сервер и связал его с бд, а также связал фронт с беком через получение данных с помощью запросов к АРІ.