

Welcome page.



Hi, J1JJJ3! Log in or Sing up :)

First of all sign up. Let's do it!

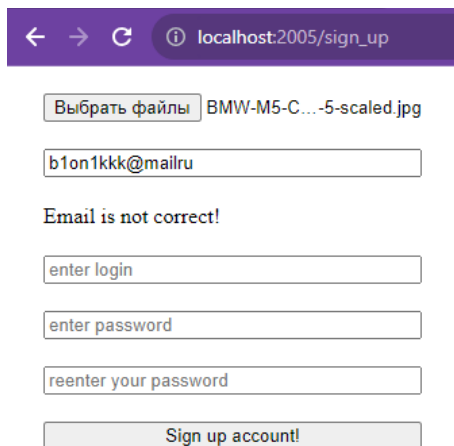
Sign up page:

A screenshot of a web browser window showing a sign-up page. The address bar shows 'localhost:2005/sign_up'. The page has a white background. At the top, there is a button labeled 'Выбрать файлы' and text 'Файл не выбран'. Below this are four input fields: 'enter email', 'enter login', 'enter password', and 'reenter your password'. At the bottom, there is a button labeled 'Sign up account!'.

First input is input for downloading your account avatar. To download user's avatar photo I used multer library. On the screenshot below you can see, that user can download only images.

```
24
25   const fileFilter = (
26     request: Request,
27     file: Express.Multer.File,
28     callback: FileFilterCallback
29   ): void => {
30     if (
31       file.mimetype === "image/png" ||
32       file.mimetype === "image/jpg" ||
33       file.mimetype === "image/jpeg"
34     ) {
35       callback(null, true);
36     } else {
37       callback(null, false);
38     }
39   };
40
```

Next step is user's email. I created primitive custom handler that show error if user's input do not contain "@" and "." somewhere in input.

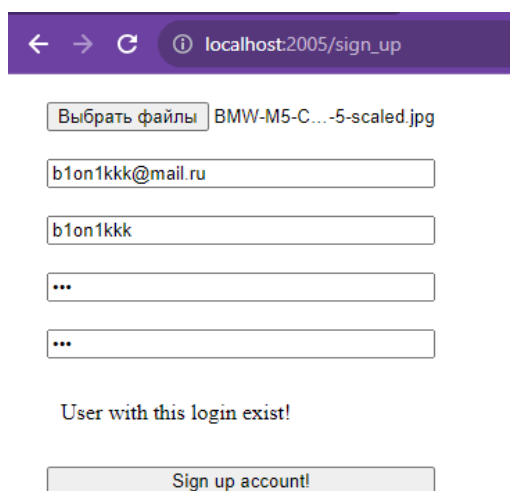


The screenshot shows a web browser window with the address bar displaying "localhost:2005/sign_up". The page contains a sign-up form with the following elements:

- A file selection button labeled "Выбрать файлы" followed by the text "BMW-M5-C...-5-scaled.jpg".
- An email input field containing the text "b1on1kkk@mailru".
- A red error message below the email field: "Email is not correct!".
- A login input field with the placeholder text "enter login".
- A password input field with the placeholder text "enter password".
- A re-password input field with the placeholder text "reenter your password".
- A "Sign up account!" button at the bottom.

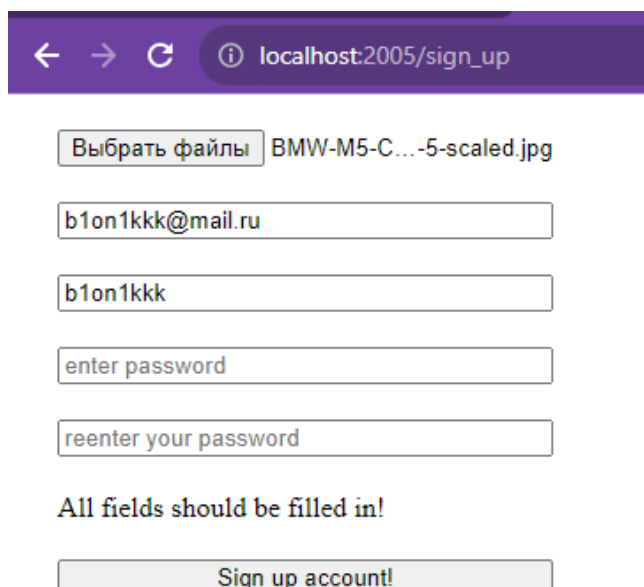
Login checking when user click sign up. After clicking this button we send request to database and get all users information. Next step is checking if this login is unique, if its true set it up if not – send error.

```
3
4 export default function SignUpUser(req: any, res: Response) {
5   // создаю массив для будущих ошибок
6   const errors: object[] = [];
7
8   // проверяю, имеется ли человек под таким логином
9   const query = `SELECT * FROM registration WHERE login = '${req.body.login}'`;
10
11   db.query(query, (error: Error, results: any) => {
12     if (error) throw error;
13
14     // если имеется, тогда кидаем ошибку и пишем, что такой пользователь имеется
15     if (results.length > 0)
16       errors.push({ error: "User with this login exist!" });
17   });
18 }
```



A screenshot of a web browser window with the address bar showing 'localhost:2005/sign_up'. The page contains a sign-up form with the following elements: a file upload button labeled 'Выбрать файлы' next to the filename 'BMW-M5-C...-5-scaled.jpg'; an email input field containing 'b1on1kkk@mail.ru'; a login input field containing 'b1on1kkk'; two empty password input fields, each preceded by three dots '...'; an error message 'User with this login exist!' displayed in red; and a 'Sign up account!' button.

If form have empty fields and user click sign up button – show error.



A screenshot of a web browser window with the address bar showing 'localhost:2005/sign_up'. The page contains a sign-up form with the following elements: a file upload button labeled 'Выбрать файлы' next to the filename 'BMW-M5-C...-5-scaled.jpg'; an email input field containing 'b1on1kkk@mail.ru'; a login input field containing 'b1on1kkk'; two empty password input fields with placeholder text 'enter password' and 'reenter your password'; a validation error message 'All fields should be filled in!' displayed in red; and a 'Sign up account!' button.

Error if passwords are not match

← → ↻ ⓘ localhost:2005/sign_up

Выбрать файлы BMW-M5-C...-5-scaled.jpg

b1on1kkk@mail.ru

b1on1kkk1

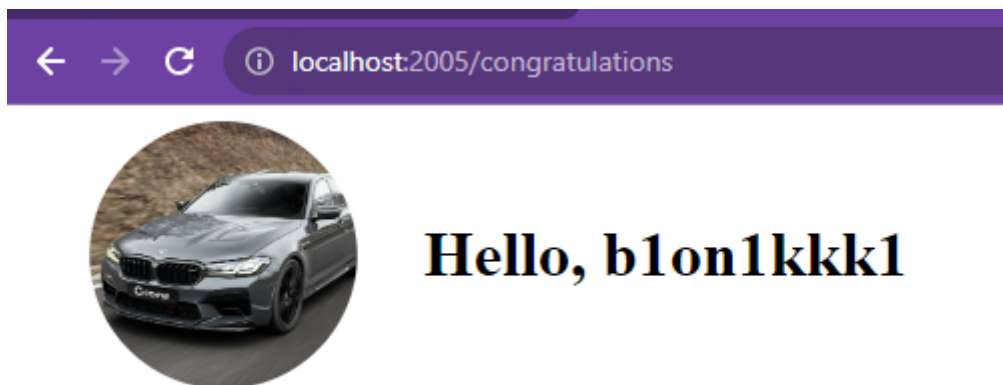
...

....

Passwords are not match!

Sign up account!

If everything is good and user created account – congratulate him/her with successful operation and show avatar and greet user by his/her nickname.

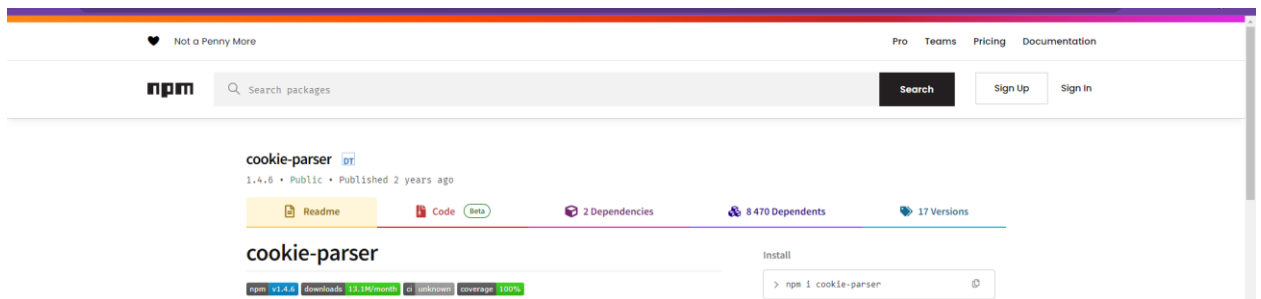


To save that user is logged in the system we use cookies.

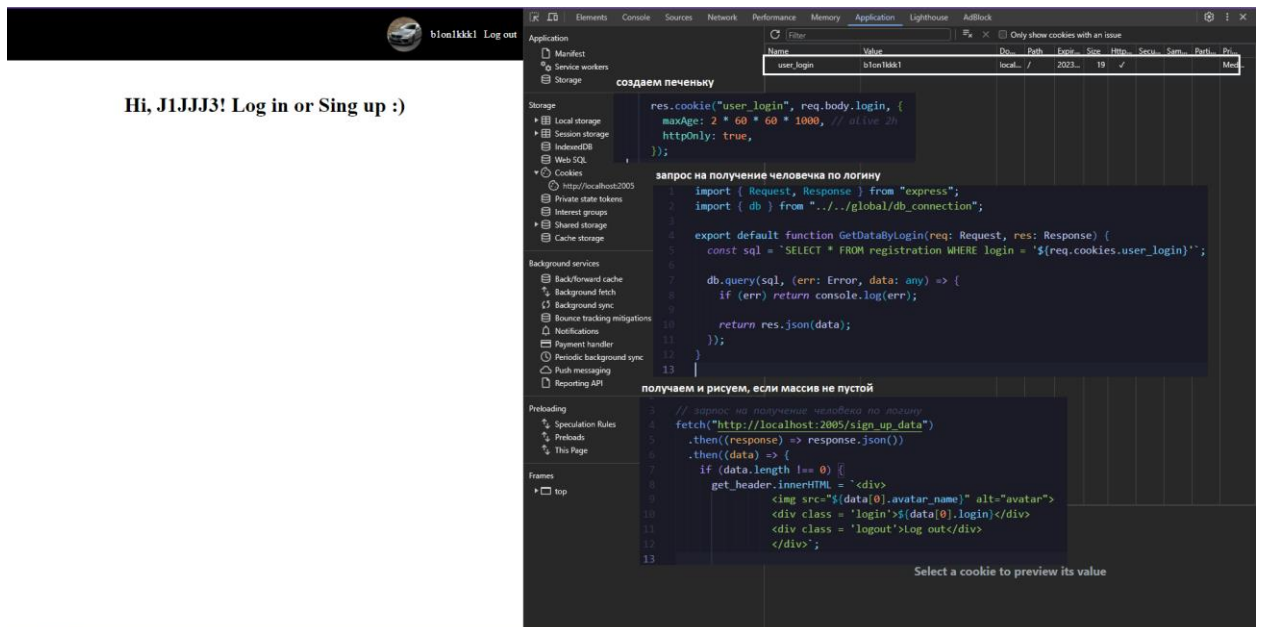


Hi, J1JJJ3! Log in or Sing up :)

I used cookie-parser to make it alive.



In cookie we save user's nickname, which in future using as a request to database



To log out from account we just delete cookie.

front

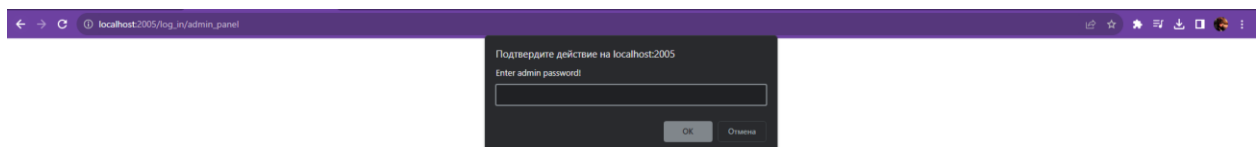
```
17 // запрос на удаление печеньки
18 logOutButton.addEventListener("click", () => {
19   fetch("http://localhost:2005/log_out", {
20     method: "POST",
21   });
22
23   location.reload();
24 });
```

back

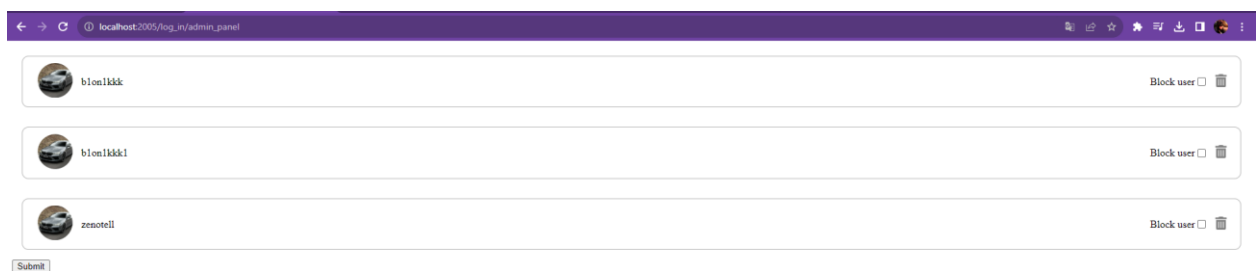
```
2
3 export default function logOutUser(req: Request, res: Response) {
4   res.clearCookie("user_login");
5
6   res.send("User logged out successfully");
7 }
8
```

Let's talk about admin panel that I made here.

Before entering admin panel you have to input special password.



If you entered seccesfully list of users will be shown



Here you can delete, block and play with them as you want 😊

For example you can block two random users

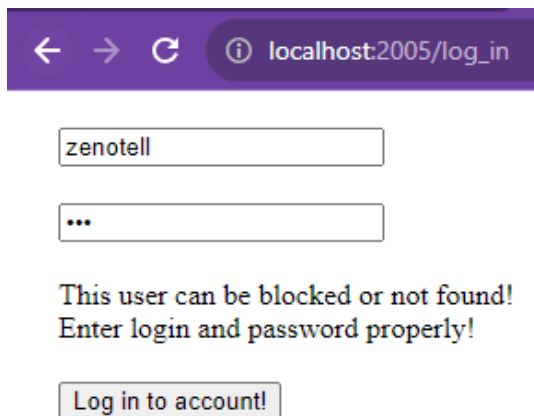


(choose users and push submit button)

In database status was changed from false (0 – not blocked) to true (1 – blocked)

				id	email	login	password	avatar_name	status
<input type="checkbox"/>	Edit	Copy	Delete	1	b1on1kkk@mail.ru	b1on1kkk	123	1697145221975-BMW-M5-CS-G-Power-tuning-5-scaled.jp...	0
<input type="checkbox"/>	Edit	Copy	Delete	2	b1on1kkk@mail.ru	b1on1kkk1	123	1697146509277-BMW-M5-CS-G-Power-tuning-5-scaled.jp...	1
<input type="checkbox"/>	Edit	Copy	Delete	3	just96.96@mail.ru	zenotell	123	1697147551891-BMW-M5-CS-G-Power-tuning-5-scaled.jp...	1

If you block someone this user/users do not have any ability to enter



Error appears 😊

Here I show how delete button is working

```
52  delete_button.forEach((e, idx) => {
53    e.addEventListener("click", () => {
54      PostFetchRequest(
55        { login: data[idx].login },
56        "delete_user_by_login"
57      );
58
59      location.reload();
60    });
61  });
62
```

Getting user log in and pushig it into the request (fetch request I made as module, because I use the same code structure in more than one place)

Here is the fetch module

```
1  export default function PostFetchRequest(data, link) {
2    fetch(`http://localhost:2005/${link}`, {
3      method: "POST",
4      headers: {
5        "Content-Type": "application/json",
6      },
7      body: JSON.stringify(data),
8    });
9  }
10
```

And request to database.

```
1  import { Request, Response } from "express";
2  import { db } from "../global/db_connection";
3
4  export default function DeleteUserByLogin(req: Request, res: Response) {
5    db.query(`DELETE FROM registration WHERE login = '${req.body.login}'`);
6
7    res.send("Person deleted successfully");
8  }
9
```