МИНОБРНАУКИ РОССИИ

–––––––——————————–––––––

Санкт-Петербургский государственный  
электротехнический университет «ЛЭТИ»

————————————————————

**Web-программирование**

**Лабораторная № 7**

**Тема: ОРГАНИЗАЦИЯ ПЕРЕДАЧИ ДАННЫХ МЕЖДУ ЗАПРОСАМИ ПОЛЬЗОВАТЕЛЕЙ**

Выполнил: студент группы 3311 Шарпинский Денис

Проверил: Калмыков Михаил Александрович

Санкт-Петербург

Издательство СПбГЭТУ «ЛЭТИ»

2025

Цель работы: освоить использование cookie для передачи данных между запросами пользователей, обеспечивая сохранение информации о состоянии сеанса (авторизация) при повторных обращениях.

Cookie – это небольшие текстовые файлы, хранящиеся на стороне клиента и используемые для передачи состояния между запросами HTTP.

Они позволяют идентифицировать пользователя и сохранять данные между сессиями.

Отличие от хранения в памяти сервера в том, что данные сохраняются у клиента и отправляются при каждом запросе.

Для того, чтобы пользователю не приходилось авторизовываться каждый раз, я добавлю на сервере логику, отвечающую за работу с cookie файлами. В них я буду записывать id, name и color пользователя. Айди пользователя будет работать как аналог токена в современных приложениях. Имя и цвет пользователя, поскольку являются простой и легковесной информацией можно также положить в cookie файлы, что избавит от потребности делать запросы на сервер, чтобы получить информацию о клиенте, вместо этого клиент сразу сможет прочесть данные о себе.

Для начала я подключаю middleware cookie-parser, о котором я упоминал во второй лабораторной. Он позволит удобно работать с cookie файлами. Затем в /api/auth роуте при успешной авторизации я добавляю строки, записывающие в cookie информацию о пользователе.

Также я написал собственное middlware, которое проверяет наличие id в cookie и выполняет необходимые редиректы (с сохранением query параметров).

const checkAuthMiddleware: express.RequestHandler = (req, res, next) => {

  if (req.path.startsWith("/api")) {

    return next();

  }

  const username = req.cookies.username;

  console.log("запрос на ", req.path, "от ", username);

  const doesUserExist = Object.values(users).some(user => user.name === username);

  const lang = getLanguage(req);

  const queryParams = `?lang=${lang}`;

  if (req.path === "/admin-panel") {

    return next();

  }

  // я исключаю здесь роуты /profile и /auth, т.к. если юзер уже на них, то нет смысла редиректить

  // иначе был бы бесконечный цикл, т к эта middleware используется во всех запросах

  if (doesUserExist && req.path !== "/profile" && req.path !== "/auth") {

    console.log("пользователь уже был авторизован ранее, перенаправление на /profile");

    return res.redirect(`/profile${queryParams}`);

  } else if (!doesUserExist && req.path !== "/auth") {

    console.log("пользователь не был авторизован ранее, перенаправление на /auth");

    return res.redirect(`/auth${queryParams}`);

  } else {

    return next();

  }

};

Теперь при повторном открытии вкладки пользователю не нужно будет снова авторизовываться.

Выводы:

В ходе выполнения лабораторной работы была добавлена новая функциональность – сохранение информации о пользователе в cookie файлы, что позволило избежать необходимости в повторной авторизации.