

МИНОБРНАУКИ РОССИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**

УТВЕРЖДАЮ

Зав.кафедрой,

к. ф.-м. н.

\_\_\_\_\_ М. В. Огнева

**ОТЧЕТ О ПРАКТИКЕ**

студента 2 курса 273 группы факультета КНиИТ  
Кулакова Максима Сергеевича

вид практики: производственная (научно-исследовательская работа)

кафедра: Информатики и программирования

курс: 2

семестр: 1

продолжительность: 18 нед., с 01.09.2023 г. по 14.01.2024 г.

Руководитель практики от университета,

к. э. н., доцент

\_\_\_\_\_

Л. В. Кабанова

Руководитель практики от организации (учреждения, предприятия),

к. э. н., доцент

\_\_\_\_\_

Л. В. Кабанова

Тема практики: «Разработка платформы единого резюме»

## СОДЕРЖАНИЕ

|   |    |
|---|----|
| ВВЕДЕНИЕ .....  | 4  |
| 1 Анализ научной литературы .....                         | 5  |
| 2 Реализация платформы единого резюме .....               | 6  |
| 2.1 Общие настройки динамического приложения .....        | 6  |
| 2.2 Написание механизма авторизации.....                  | 7  |
| 2.3 Реализация страницы резюме .....                      | 9  |
| 2.4 Взаимодействие с сервисом hh.ru .....                 | 13 |
| 2.5 Реализация страниц сравнения и обновления резюме..... | 15 |
| ЗАКЛЮЧЕНИЕ .....  | 18 |
| СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....                    | 19 |

## **ВВЕДЕНИЕ**

Вопрос поиска работы всегда находился перед лицом человека, ведь работа должна приносить не только деньги, но и удовлетворение физических и психологических потребностей человека. Наиболее актуальной проблемой со стороны соискателя является то, где искать необходимого ему работодателя, а также с какой стороны преподнести свои навыки и умения, чтобы в ближайшие дни занимать рабочее место своей мечты.

Задачами работы являются следующие пункты:

1. Масштабирование платформы до клиент-серверного приложения;
2. Конфигурация основного функционала;
3. Реализация взаимодействия по API.
4. Обзор научной литературы (в том числе научно-технической) по теме «Разработка платформы единого резюме»;
5. Рассмотрение и анализ существующих платформ для создания резюме;
6. Формулировка собственных методов разработки единой платформы резюме;
7. Подведение итогов проведенной научно-исследовательской работы.

## **1 Анализ научной литературы**

Рассматриваемая литература будет затрагивать тему аспектов составления резюме, принципы их составления и критерии, по которым работодателю с наибольшей вероятностью понравится грамотно составленное резюме. После проведения анализа данной темы нам предоставится возможность выделить основные пункты, которые будут учитываться при разработке собственной единой платформы резюме.

Для начала стоит рассмотреть научные статьи, связанные с доказательством важности правильного составления резюме в настоящее время, и какие изменения оно претерпевает. В статье К.В. Косолаповой «Типологические особенности современного резюме на английском языке» автор выделяет основные пункты в резюме, которые было принято считать достаточными:

1. Полные ФИО;
2. Возраст;
3. Место проживания на текущий момент;
4. Место учёбы, уровень образования;
5. Список умений;
6. Опыт работы (при его наличии);
7. Контактные данные. [1]

## 2 Реализация платформы единого резюме

### 2.1 Общие настройки динамического приложения

Для компенсации недостатков статических сайтов и динамической генерации страниц из набора входящих параметров и данных инициализируем проект с NextJS и добавим интерфейсную библиотеку NextUI для удобства разработки следующими командами:

```
1 npx create-next-app@latest
2 npm i @nextui-org/react framer-motion
```

Для безопасной работы с проектом необходимо создать файл `process.env`, в котором будут храниться переменные окружения, например токен доступа к системе.

Дополнительно для каждой страницы необходимо задать общий шаблон в файле `layout.tsx`, находящимся в корневой папке приложения. Данная структура представляет из себя обёртку над стандартным представлением HTML файла, внутрь которого добавлено реактивное представление. Дополнительно внутри атрибутов указываются стили для блоков, такие как минимальная высота контейнера, цвет фона, размеры контейнеров, отступов и их поведение, выбор шрифта и текущая цветовая схема. В дальнейшем содержимое атрибутов `className` будет игнорироваться.

```
1 export default function RootLayout({ children, }: {
2     children: React.ReactNode;
3 }) { return (
4 <html lang="en" suppressHydrationWarning>
5     <head></head>
6     <body className={clsx( "min-h-screen bg-background font-sans antialiased",
7       ↪ fontSans.variable )}>
8     <Providers themeProps={{ attribute: "class", defaultTheme: "dark" }}>
9     <div className="relative flex flex-col h-screen">
10    <Navbar/>
11    <main className="container mx-auto max-w-7xl pt-16 px-6 flex-grow">
12    {children}
13    </main></div></Providers></body>
14 </html>
15 );}
```

## 2.2 Написание механизма авторизации

Для реализации авторизации создадим файл `auth.ts`, в котором укажем конфигурацию с сервисом Github посредством открытого протокола безопасности OAuth и с классической проверкой логина и пароля. Для этого создадим два соответствующих свойства и в первое передадим переменные приложения для авторизации через сторонний сервис, а в втором реализуем проверку введённых в поля данных и в случае соответствия вернём самого пользователя с его именем, фотографией профиля и ролью.

```
1 export const authConfig: AuthOptions = { providers: [  
2   GithubProvider({  
3     clientId: process.env.GITHUB_ID!,  
4     clientSecret: process.env.GITHUB_SECRET!,}),  
5   Credentials({  
6     credentials: {  
7       email: { label: 'email', type: 'email', required: true },  
8       password: { label: 'password', type: 'password', required: true }, },  
9     async authorize(credentials) {  
10      if (!credentials?.email || !credentials.password) return null;  
11      const currentUser = users.find(user => user.email ===  
12        ↪ credentials.email)  
13      if (currentUser && currentUser.password === credentials.password) {  
14        const { password, ...userWithoutPass } = currentUser;  
15        return userWithoutPass as User;  
16      }  
17      return null  
18    })  
19  ], pages: { signIn: '/signin' } }
```

Ограничение доступа неавторизованного пользователя к внутренним страницам достигается созданием файла `middleware.ts`, в котором через параметры указываются пути навигации и, в случае попадания в заданный путь, перевод клиента на страницу `signIn`.

```
1 export const config = { matcher: ['/profile/:path*', '/hh',  
2   '/resume', '/protected/:path*'] }
```

Дополнительно к предыдущему пункту контроля доступа опишем корневой файл `providers.tsx`, созданный для окружения корневого элемента страницы в её компоненты, свойства стилей и сессию пользователя.

```
1 export function Providers({ children, themeProps }: ProvidersProps) {  
2   return (  
3     <AuthProvider>  
4       <ThemeProvider>  
5         <SessionProvider>  
6           <AuthProvider>
```

```

3 <SessionProvider><NextUIProvider>
4   <NextThemesProvider {...themeProps}>
5     {children}
6   <NextThemesProvider>
7 </NextUIProvider></SessionProvider>;}

```

Процесс авторизации реализуется на клиентской и серверной части приложения, поэтому для корректного взаимодействия необходимо создать обработчик событий по API в динамическом маршруте [..auth]/route.tsx с следующими параметрами:

```

1 const handler = NextAuth(authConfig);
2 export { handler as GET, handler as POST }

```

Реализуем форму авторизации через компонент SignInForm с двумя текстовыми формами, для этого инициализируем роутер переходов, обратные вызовы для динамического контента и обработчик нажатия кнопки отправки формы, получающий данные полей и выполняющий вход в систему.

```

1 const SignInForm = () => {
2   const router = useRouter();
3   const [isVisible, setIsVisible] = React.useState(false);
4   const toggleVisibility = () => setIsVisible(!isVisible);
5   const handleSubmit: FormEventHandler<HTMLFormElement> = async (event) => {
6     event.preventDefault();
7     const formData = new FormData(event.currentTarget);
8     const res = await signIn("credentials", {
9       email: formData.get("email"),
10      password: formData.get("password"),
11      redirect: false

```

Кнопка авторизации через Github работает через параметры поисковой строки, получаемые в процессе возврата на страницу платформы после успешного входа на стороне внешнего сервиса.

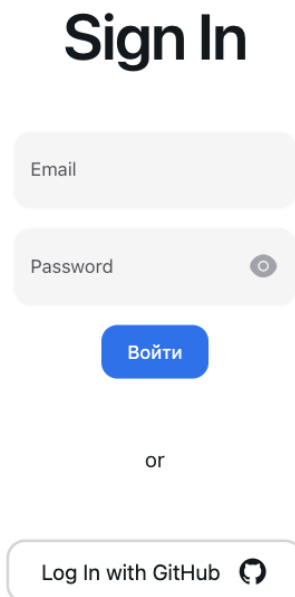
```

1 export const GithubButton = () => {
2   const searchParams = useSearchParams();
3   const callbackUrl = searchParams.get("callbackUrl") || "/profile";
4   return (
5     <Button onClick={() => signIn("github", { callbackUrl })}>
6       Log In with GitHub
7     </Button>
8   );}

```



После написания компонентов необходимо создать саму страницу авторизации, для этого внутри всей иерархической оболочки странички передадим заголовок вместе с компонентами `SignInForm` и `GithubButton`, после которых страница будет располагаться по маршруту `app/signIn`. Итоговый вид страницы авторизации отображён на рисунке 2.1.



The image shows a 'Sign In' form. At the top, the text 'Sign In' is displayed in a large, bold, black font. Below it, there are two input fields: 'Email' and 'Password'. The 'Email' field is a simple light gray box. The 'Password' field is a light gray box with a small eye icon on the right side to toggle visibility. Below these fields is a blue button with the text 'Войти' (Login) in white. Underneath the button is the word 'or' in a small, gray font. At the bottom, there is a rounded rectangular button with the text 'Log In with GitHub' and the GitHub logo icon.

Рисунок 2.1 – Страница авторизации

### 2.3 Реализация страницы резюме

Для написания модуля необходимо создать структуру данных резюме. Информация о профиле хранится в формате текстовом формате обмена данными JSON и содержит в себе следующие поля:

1. Id – генерируемый уникальный идентификатор резюме;
2. Name – фамилия и имя человека;
3. Description – краткое описание профиля;
4. Social links – ссылки на социальные сети;
5. Profile image – изображение работника для резюме;
6. About – текстовое поле с рассказом о себе, являющееся аналогом сопроводительного письма;
7. Education, experience – группы образования и опыта работы, включающие в себя следующие пункты:
  - а) Name – наименование места работы или учебного учреждения;
  - б) Description – описание чем занимался соискатель в данное время;

- в) Start date, end date – дата начала и окончания;
  - з) Current date – поле, показывающее что дата окончания отсутствует и при подсчёте её следует считать текущим днём;
8. Projects – поле проектов, следующее из опыта работы. Включает в себя:
- а) Name – наименование проекта;
  - б) Description – описание проекта и над чем происходила работа;
  - в) Image – изображение проекта, его логотип, интерфейс.

Последовательно реализуем компоненты, начиная с блока контактов и ссылок на социальные сети, получающий на вход список адресов и возвращающий горизонтальный список кнопок с переходом и миниатюрами соответствующих сервисов. Изображения предварительно описываются в формате svg в компоненте icons.tsx, после чего из строки выделяется имя хостинга с последующим добавлением в массив элементов для отображения на странице.

```

1 function IconVariant(service: string, icon_color:string, icon_size:number) {
2   switch(service) {
3     case "github.com":
4       { return <GithubIcon className={icon_color} size={icon_size}/>; }
5     default:
6       { return <InternetIcon className={icon_color} size={icon_size}/>; }
7   }
8 }
9
10 export const SocialLink = (props : SocialProps) => {
11   const icon_size, icon_color = 36, "text-default-500";
12   const social_icons = props.links.map((element, index) => {
13     var url = new URL(element.toString());
14     return
15       <Link isExternal href={element.toString()} key={index}>
16         {IconVariant(url.host.toString(), icon_color, icon_size)}
17       </Link>;
18   });
19 }

```

Итоговый вид компонента показан на рисунке 2.2.

Опишем компонент вывода образования и опыта работы. Для этого реализуем функционал подсчёта текущего затраченного времени через получение двух строковых и одной логической переменных из файла резюме, преобразования текста в объект даты и нахождения разницы между двумя величинами. После вычисления необходимо округлить полученное значение до целого числа и преобразовать в удобочитаемый внешний вид с помощью проверки количества

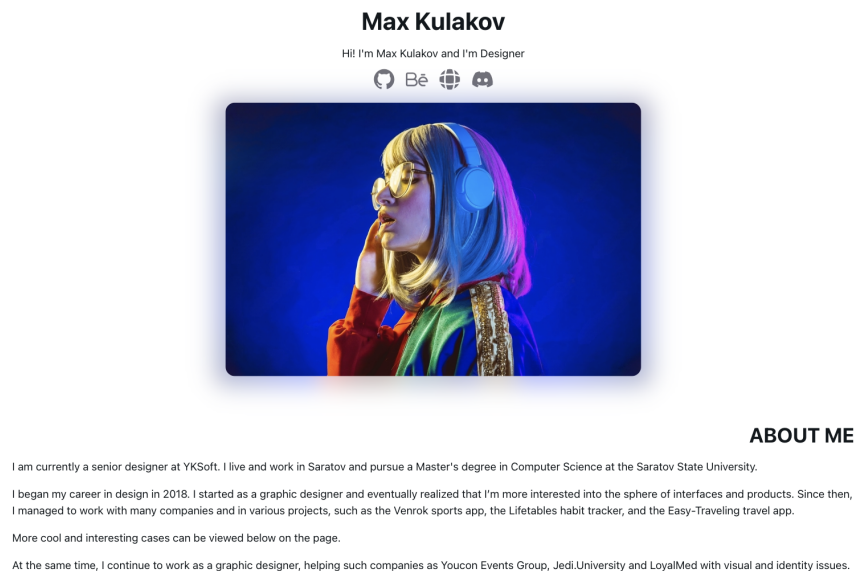


Рисунок 2.2 – Компонент отображения контактов и ссылок

месяцев и лет. Результатом работы функций станет вывод временного значения в строке компонента.

```

1 function diffDates(day_one:any, day_two:any) {
2     return (day_one - day_two) / (60 * 60 * 24 * 1000);
3 };
4 function getFormattedStringFromDays(numberOfDays:number) {
5     numberOfDays = Math.abs(numberOfDays)
6     var years = Math.floor(numberOfDays / 365);
7     var months = Math.floor(numberOfDays % 365 / 30);
8     var yearsDisplay = years > 0 ?
9         years + (years == 1 ? " year " : " years ") : "";
10    var monthsDisplay = months > 0 ?
11        months + (months == 1 ? " month " : " months ") : "";
12    return yearsDisplay + monthsDisplay;
13 };
14 export const Experience = (props : ExperienceProps) => {
15     const experience_item_list = props.experience_list.map((element, index)=>{
16         const date1 = new Date(element.start_date);
17         const date2 = element.current_date ?
18             new Date() : new Date(element.end_date)
19         return { getFormattedStringFromDays(diffDates(element.start_date,
20             ↪ element.end_date)) })
21     return <>{experience_item_list}</>
22 };

```

Итоговый вид компонента показан на рисунке 2.3.

| EDUCATION                                    |   |  |
|--|---|--|
| 2022-09-01 - 2023-10-30<br>1 year 4 months   | Master's degree, Saratov State University   | Computer Science, Mathematical Support and Administration of Information Systems   |
| 2017-09-01 - 2022-06-30<br>4 years 10 months | Bachelor's degree, Saratov State University | Computer Science, Mathematical Support and Administration of Information Systems. Graduate work: «Development of mobile application with implementation of accessibility»  |
| EXPERIENCE                                   |   |  |
| 2021-04-25 - 2023-10-30<br>2 years 8 months  | Senior Designer, YKSoft                     | Audit, development and implementation of solutions in the products of various companies. Implementation of new projects and bringing them to release. Support for existing sites and the application in order to refine the functionality. Interaction with customers and related teams in the process of working on projects. Managing a mini-team of designers |
| 2019-10-01 - 2021-02-01<br>1 year 4 months   | Interface Designer, Venrok                  | Launch and development of projects from an idea to a final solution with subsequent support  |
| 2019-02-01 - 2019-10-01<br>8 months          | Designer, Youcon Events Group               | Participation in the organization of events. Work on the website, printed and digital products   |

Рисунок 2.3 – Компонент отображения опыта работы и образования

Для реализации портфолио в шахматном представлении воспользуемся классовыми атрибутами сетки из библиотеки Tailwind. Получим порядковый индекс списка проектов, начинающийся с нуля, и дальше в зависимости от чётности отдельно и поочерёдно позиционируем текст с 1 по 6 колонку и изображение с 7 по 13.

```

1 export const Projects = (props : ProjectsProps) => {
2   const projects = props.projects_list.map((element, index) => {
3     var position_image = index % 2 == 0 ?
4       "col-start-7 col-end-13" : "col-start-1 col-span-6 ";
5     var position_text = index % 2 == 0 ?
6       "col-start-1 col-end-6" : "col-start-8 col-end-13 order-1";
7     return (
8       <div className={position_text}>
9         <h2>{element.name}</h2>
10        <p>{element.description}</p>
11      </div>
12      <Image className={position_image}/>
13    ))
14   return <>{projects}</>
15 };

```

Итоговый вид компонента списка проектов показан на рисунке 2.4.

После описания компонентов реализуем страницу резюме, для этого необходимо произвести запрос данных резюме и передать его поля в соответствующие теги и компоненты.

```

1 export default function Template1Page() {
2   const profile_data = require("@/data-template/template-1-data.json");
3   return (

```

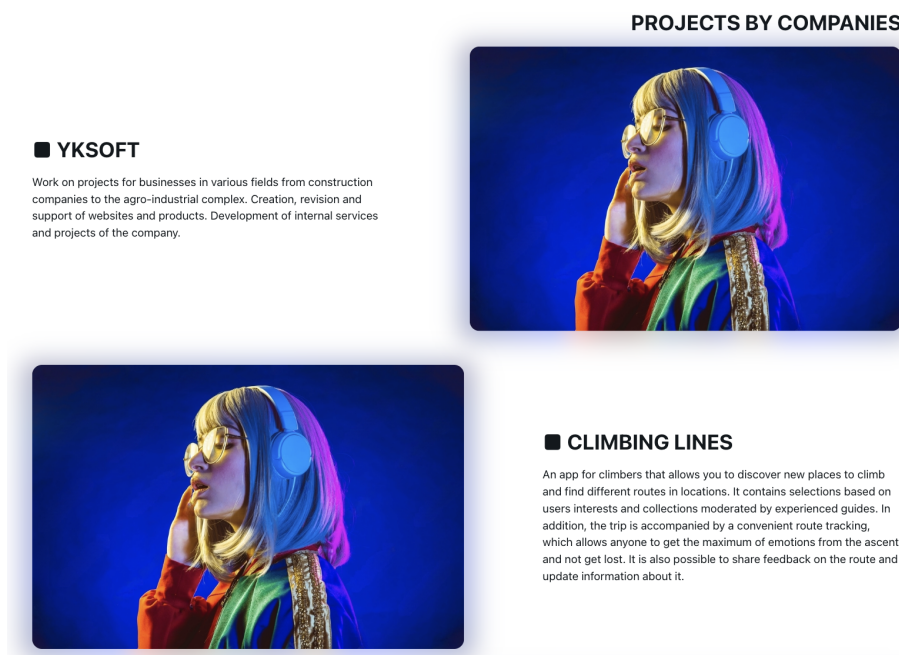


Рисунок 2.4 – Компонент отображения списка проектов

```

4 <SocialLink links = {profile_data.social_links}/>
5 ...
6 <div className="container pt-5 pb-5">
7   <h2 id="projects">projects by companies</h2>
8   <Projects projects_list={...profile_data.projects}/>
9 </div>
10 );}

```

## 2.4 Взаимодействие с сервисом hh.ru

Реализуем взаимодействие пользователя с сервисом hh.ru через API. Первоначальная конфигурация в виде авторизации состоит из 3 этапов:

1. Перенаправление пользователя по адресу сервиса с переданными параметрами строки;
2. Получение временного кода авторизации;
3. Обмен кода авторизации на долгосрочный токен доступа к профилю и кода обновления токена в случае необходимости.

Первый этап достигается с помощью передачи параметров строки запроса через вопросительный знак, имя параметра и значение. В ответ на полученную от сервиса hh.ru ссылку с переходом необходимо обработать ответный параметр временного кода авторизации и обмена на долгосрочный. Опишем данный функционал в `api/hh/route.tsx`, для этого необходимо реализовать GET запрос, предварительно считывающий строку запроса и параметр `code`, в заголовок ко-

торого добавляем желаемое действие, открытый и секретный идентификаторы приложения и полученный на предыдущем обработчике код. Так как функция запросов асинхронна – воспользуемся обработчиком событий и через ожидающий обратный вызов функции передадим запрос по адресу сервиса, получив в ответ необходимый набор данных: ответ об успешности запроса и токены.

```
1 export async function GET(req: Request) {
2   const { searchParams } = new URL(req.url)
3   const code = searchParams.get('code');
4   if (code != null) {
5     var myHeaders = new Headers();
6     myHeaders.append("Content-Type", "application/x-www-form-urlencoded");
7     var urlencoded = new URLSearchParams();
8     urlencoded.append("grant_type", "authorization_code");
9     urlencoded.append("client_id", process.env.HH_ID!);
10    urlencoded.append("client_secret", process.env.HH_SECRET!);
11    urlencoded.append("code", code!);
12    try {
13      const res = await fetch("https://hh.ru/oauth/token", {
14        method: 'POST', headers: myHeaders,
15        body: urlencoded, redirect: 'follow' });
16      const result = await res.json();
17      return NextResponse.json(result);
18    }
19  }
```

После получения долгосрочного токена для конкретного пользователя, у приложения появляется возможность воспользоваться доступом к сервису hh.ru. Получим все существующие резюме человека и выведем их на страницу. Для этого отправим GET запрос с токеном авторизации по заданному маршруту и вернём данные в формате JSON.

```
1 const token = searchParams.get('token');
2 if (token != null) {
3   var myHeaders = new Headers();
4   myHeaders.append("Authorization", `Bearer ${token}`);
5   try {
6     const res = await fetch("https://api.hh.ru/resumes/mine", {
7       method: 'GET', headers: myHeaders, redirect: 'follow' });
8     const result = await res.json();
9     return NextResponse.json(result);
10  }
```

## 2.5 Реализация страниц сравнения и обновления резюме

Страница сравнения навыков представляет из себя таблицу, содержащую наименование поля, значения совпадающих ключей для каждого из входных резюме и поле взаимодействие, содержащее возможность добавления собственного значения, а также удаления строки. Так как резюме содержит в себе значения разных типов, их отображение будет возможно только при сепарации. Для этого реализуем компонент сравнения, получающий значение свойства из JSON и далее в случае одиночной строки возвращающий тег с его содержимым, в случае списка строк обрабатывающий каждую из них, присваивающую уникальный индекс отображения, а в случае более глубокой вложенности добавляющей перед элементом его наименование.

```
1  if (typeof props.cv_editor == "string") {
2      cv_editor_item_list = <div>{props.cv_editor}</div>;
3  } else {
4      Object.keys(props.cv_editor).map((element, index) => {
5          let arrObj = props.cv_editor[element];
6          if (typeof arrObj == "object") {
7              cv_editor_item_list = props.cv_editor.map((element: any, index: any) => {
8                  return (
9                      {Object.keys(element).map((el, index) => {
10                         return( <div key={index}>{el}: {element[el]}</div> );}}}
11                  );});
12          } else {
13              cv_editor_item_list = props.cv_editor.map((element: any, index: any) =>
14                  { return {element}; })
15              });}
```

Итоговый вид страницы сравнения навыков показан на рисунке 2.5.

В серверной части платформы реализуем API метод для обновления данных резюме на сервисе hh.ru. Для этого в файле `api/hh/route.tsx` опишем PUT запрос, в заголовок которого передаётся токен пользователя и ссылка на идентификатор резюме, а в тело наименование поля данных с его содержимым. После выполнения запроса в ответ от внутреннего сервера вернётся JSON с кодом и сообщением об успешности выполненной операции.

```
1  export async function PUT(req: Request) {
2      if (resume != null) {
3          const body = await req.json()
```

## HH Dev Page

| Field Name   | CV Editor   | hh.ru  | Custom  |
|--------------|---|--|---|
| Full Name    | <input checked="" type="radio"/> Max Kulakov  | <input type="radio"/> Max Kulakov  | <input type="radio"/> <span>Add item</span> <span>🗑️</span> |
| Description  | <input checked="" type="radio"/> Hi! I'm Max Kulakov and I'm Designer   | <input type="radio"/> Hi! I'm Max Kulakov and I'm Designer   | <input type="radio"/> <span>Add item</span> <span>🗑️</span> |
| Social Links | <input checked="" type="radio"/> <a href="https://github.com/nesbox/TIC-80">https://github.com/nesbox/TIC-80</a><br><a href="https://www.behance.net/KulakovMax">https://www.behance.net/KulakovMax</a><br><a href="https://t.me/MaxKulakov">https://t.me/MaxKulakov</a><br><a href="https://discord.com/servers">https://discord.com/servers</a> | <input type="radio"/> <a href="https://github.com/nesbox/TIC-80">https://github.com/nesbox/TIC-80</a><br><a href="https://www.behance.net/KulakovMax">https://www.behance.net/KulakovMax</a><br><a href="https://t.me/MaxKulakov">https://t.me/MaxKulakov</a><br><a href="https://discord.com/servers">https://discord.com/servers</a> | <input type="radio"/> <span>Add item</span> <span>🗑️</span> |
| About        | <input checked="" type="radio"/> I am currently a senior designer at YKSoft. I live and work in Saratov and pursue a Master's degree in Computer Science at the Saratov State University.   | <input type="radio"/> I am currently a senior designer at YKSoft. I live and work in Saratov and pursue a Master's degree in Computer Science at the Saratov State University.   | <input type="radio"/> <span>Add item</span> <span>🗑️</span> |

Рисунок 2.5 – Страница сравнения навыков

```

4   var raw = JSON.stringify( body );
5   try { const res = await fetch(`https://api.hh.ru/resumes/${resume}`, {
6       method: 'PUT', headers: myHeaders,
7       body: raw, redirect: 'follow' })
8   return NextResponse.json({ message: "Resume has been updated" });
9   }}}

```

Реализуем страницу обновления навыков, для которой запросим и выведем данные из резюме hh.ru в виде списка тегов с возможностью их удаления из списка. Для этого создадим страницу и инициализируем начальное состояние значений и функцию удаления, проверяющую количество элементов и в случае их полного удаления возвращающего к исходному значению.

```

1  export default function ResumePage() {
2      const [skill_set, setSkill_set] = React.useState(initialSkills);
3      const handleClose = (skillToRemove:any) => {
4          setSkill_set(skill_set.filter((skill: any) =>
5              skill !== skillToRemove));
6          if (skill_set.length === 1) {setSkill_set(initialSkills);}
7      }};

```

Добавим кнопку обновления резюме, по нажатию на которую вызывается функция-обработчик массива строк с преобразованием их к объектному формату JSON и помещающая итоговое значение в тело PUT запроса, отправляющего сообщение через внутренний сервер на сервис hh.ru.

```

1  const skills_body = JSON.parse(JSON.stringify
2      (`{"skill_set": [${skill_set.map((x: any) => `"${x}"`)}]}`))

```



```

3  return (
4  skill_set.map((skill: any, index: any) => (
5      <Chip key={index} onClose={() => handleClose(skill)} variant="flat">
6          {skill}
7      </Chip>
8      <Button onClick={updateResume}>Обновить резюме</Button>
9  )))

```

Итоговый вид страницы обновления навыков, содержащей статус изменения списка, редактируемое поле ввода и теги представлен на рисунке 2.6.

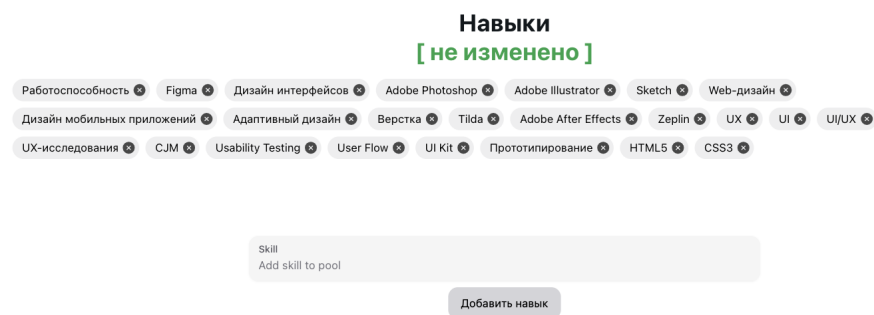


Рисунок 2.6 – Страница обновления навыков

## ЗАКЛЮЧЕНИЕ

В результате проведения исследовательской работы были приобретены навыки анализа качества и эффективности научной литературы в области разработки сервисов с автоматическим обновлением данных, достигнут навык анализа конкурентных платформ для создания резюме и сформулирован собственный метод разработки единой платформы резюме, тем самым было достигнуто полное выполнение поставленных задач.

В качестве объектов анализа научной литературы выступили статьи по темам составления резюме, их анализа со стороны социологии, а также статьи, в которых рассматриваются инструменты для веб-разработки. С учётом проведённого анализа научной литературы были составлены основные требования для разработки будущей платформы как с технической стороны, так и со стороны гуманитарно-социальных наук.

Анализ конкурентных платформ позволил выявить слабые стороны существующих сервисов, исправление которых возможно реализовать в разработке собственной единой платформы резюме при условии его дальнейшего масштабирования. [2]

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 *Гриднева, М. А.* Особенности работы с персоналом в it-сфере / М. А. Гриднева, М. Ю. Федосова // *Телескоп*. — 2021. — № 1. — С. 106–112.
- 2 *Архипов, Л. В.* Концепция применения контентно-адресуемых систем хранения для контроля версий / Л. В. Архипов // *Инновации в науке*. — 2015. — № 12 (49). — С. 42–46.