МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное

образовательное учреждение высшего образования

«Самарский национальный исследовательский университет

имени академика С.П. Королева»

(Самарский университет)

Институт дополнительного образования

Цифровая кафедра

ИТОГОВАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

«WEB-ПРИЛОЖЕНИЕ ДЛЯ ОБРАБОТКИ ПОЛЬЗОВАТЕЛЕЙ»

по программе подготовки «Web-разработка»

дополнительной профессиональной программы профессиональной переподготовки института дополнительного образования

очной формы обучения

Обучающийся Де Лео Дэвид Даниель

(подпись, дата)

Руководитель ИКР\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.С. Бельгер

(подпись, дата)

Самара 2024

Задание на итоговую квалификационную работу

слушателю дополнительной образовательной программы профессиональной переподготовки «Web-разработка»

Слушатель: Де Лео Дэвид Даниель

Тема работы: Web-приложение для обработки пользователей

Цель: реализовать веб-приложение для обработки пользователей

Задачи:

В рамках выбранной предметной области реализовать одностраничное веб-приложение, которое позволяет создавать\редактировать объекты предметной области. Список объектов должен быть отображен на странице в виде таблицы. Объекты предметной области должен содержать не менее 5 полей разных типов данных. Обязательно наличие стилизации приложения. Объектов предметной области должно быть не менее 3, под каждый выделена одна страница. В приложение обязательно наличие навигационной панели по страницам. Предметную область выбираете сами, они не должны повторяться в группе. Пример: магазин автомобильных товаров, объекты: запчасти, шины, автожидкости.

При желании можно сохранять массив объектов в localStorage\IndexedDB.

Выполненная итоговая квалификационная работа является формой итоговой аттестации по дополнительной образовательной программе профессиональной переподготовки «Web-разработка».

С заданием на выполнение итоговой квалификационной работы ознакомлен:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Де Лео Дэвид Даниель

СОДЕРЖАНИЕ

[Введение 4](#_Toc135381868)

[1 Описание предметной области 5](#_Toc135381869)

[1.1 Схема базы данных 5](#_Toc135381870)

[2 Реализация системы 6](#_Toc135381871)

[2.1 Выбор средств реализации 6](#_Toc135381872)

[2.2 Описание интерфейса пользователя 7](#_Toc135381873)

[Заключение 7](#_Toc135381874)

[Список использованной литературы 10](#_Toc135381875)

[ПРИЛОЖЕНИЕ А Код программы 11](#_Toc135381876)

Введение

Angular представляет фреймворк от компании Google для создания клиентских приложений. Прежде всего он нацелен на разработку SPA-решений (Single Page Application), то есть одностраничных приложений. В этом плане Angular является наследником другого фреймворка AngularJS. В то же время Angular это не новая версия AngularJS, а принципиально новый фреймворк.

Angular предоставляет такую функциональность, как двустороннее связывание, позволяющее динамически изменять данные в одном месте интерфейса при изменении данных модели в другом, шаблоны, маршрутизация и так далее.

Одной из ключевых особенностей Angular является то, что он использует в качестве языка программирования TypeScript. Поэтому перед началом работы рекомендуется ознакомиться с основами данного языка, про которые можно прочитать [здесь](https://metanit.com/web/typescript/).

Но мы не ограничены языком TypeScript. При желании можем писать приложения на Angular с помощью таких языков как Dart или JavaScript. Однако TypeScript все таки является основным языком для Angular.

Последняя версия Angular - Angular 17 вышла в ноябре 2023 года. Официальный репозиторий фреймворка на гитхабе: <https://github.com/angular/angular>. Там вы можете найти сами исходные файлы, а также некоторую дополнительную информацию.

### Начало работы c Angular

Для работы с Angular необходимо установить сервер Node.js и пакетный менеджер npm, если они отсутствуют на рабочей машине. При этом особого какого-то знания для работы с NodeJS и npm не требуется. Для установки можно использовать [программу установки node.js](https://nodejs.org/en/). Вместе с сервером она также установит и npm.

Следует учитывать, что Angular поддерживает те версии node.js, которые в текущий момент находятся в статусе "Active LTS" или "Maintenance LTS". На момент написания данной статьи это версии 18 и 20. Поэтому, если Node.js уже ранее был установлен, но имеет более старую или, наоборот новую, но еще не поддерживаемую версию, то лучше его обновить. Проверить совместимость версий node.js (а также версий typescript и библиотеки RxJS) для определенных версий Angular можно по адресу: <https://angular.dev/reference/versions>. Так, для последний на данный момент версии Angular 17 требования следующие:

* Версия Node.js должна быть либо 18 начиная с подверсии 18.13.0, либо 20 начиная с подверсии 20.9.0
* Версия TypeScript должна быть 4.9.3 и выше, но ниже 5.3.0 (нужная версия TypeScript устанавливается локально в проект)

То же самое касается и npm. Если версия слишком старая или, наоборот, одна из последних, то Angular может ее не поддерживать. При работе с Angular лучше полагаться на ту версию npm, которая устанавливается вместе с LTS-версией Node.js.

1. Описание предметной области

Добавление пользователей - это фундаментальная функция, необходимая для многих систем, будь то веб-сайты, приложения или локальные сети. Она позволяет администраторам создавать новые учетные записи, назначать им права доступа и управлять их профилями.

**Основные аспекты:**

* **Создание учетных записей:** Это включает в себя сбор необходимой информации от пользователя, например, имени, адреса электронной почты и пароля.
* **Назначение прав доступа:** Определение уровня доступа, который пользователь будет иметь к системе. Это может включать доступ к определенным функциям, данным или ресурсам.
* **Управление профилями:** Предоставление пользователям возможности редактировать свою информацию, менять пароли и настраивать параметры своей учетной записи.
* **Аутентификация:** Проверка личности пользователя при входе в систему.
* **Авторизация:** Определение того, к каким ресурсам и функциям пользователь имеет доступ после успешной аутентификации.

**Способы реализации:**

* **Локальные базы данных:** Хранение информации о пользователях в локальной базе данных на сервере.
* **Внешние службы аутентификации:** Использование сторонних служб, таких как Google Auth или Facebook Login, для аутентификации пользователей.
* **LDAP (Lightweight Directory Access Protocol):** Использование LDAP-сервера для централизованного управления учетными записями пользователей.

**Соображения безопасности:**

* **Защита паролей:** Хранение паролей в защищенном виде, например, с использованием хеширования и соли.
* **Предотвращение взлома:** Реализация мер защиты от взлома, таких как CAPTCHA и двухфакторная аутентификация.
* **Контроль доступа:** Ограничение доступа пользователей к конфиденциальным данным и функциям.

**В целом, добавление пользователей является важной функцией, которая должна быть реализована с учетом безопасности, удобства использования и масштабируемости.**

Выбор средств реализации

JavaScript, HTML и CSS - это три основополагающих языка, которые используются для создания веб-сайтов и веб-приложений. Каждый из них имеет свои уникальные преимущества, которые делают их незаменимыми инструментами для веб-разработчиков.

**HTML**

HTML (HyperText Markup Language) - это язык разметки, который используется для создания структуры веб-страниц. Он определяет элементы страницы, такие как заголовки, параграфы, изображения и ссылки. HTML - это простой язык, который легко изучить, и он является основой всех веб-страниц.

**Преимущества HTML:**

* **Простота:** HTML - это простой язык, который легко изучить и использовать.
* **Универсальность:** HTML используется для создания всех веб-страниц.
* **Совместимость:** HTML поддерживается всеми веб-браузерами.

**CSS**

CSS (Cascading Style Sheets) - это язык стилей, который используется для оформления веб-страниц. Он позволяет контролировать внешний вид элементов HTML, таких как цвет, шрифт, выравнивание и разметку. CSS - это мощный язык, который позволяет создавать красивые и привлекательные веб-сайты.

**Преимущества CSS:**

* **Контроль:** CSS обеспечивает разработчикам точный контроль над внешним видом веб-страниц.
* **Разделение:** CSS отделяет оформление от контента, что делает код более понятным и поддерживаемым.
* **Повторное использование:** CSS-стили можно повторно использовать на разных веб-страницах, что экономит время и усилия.

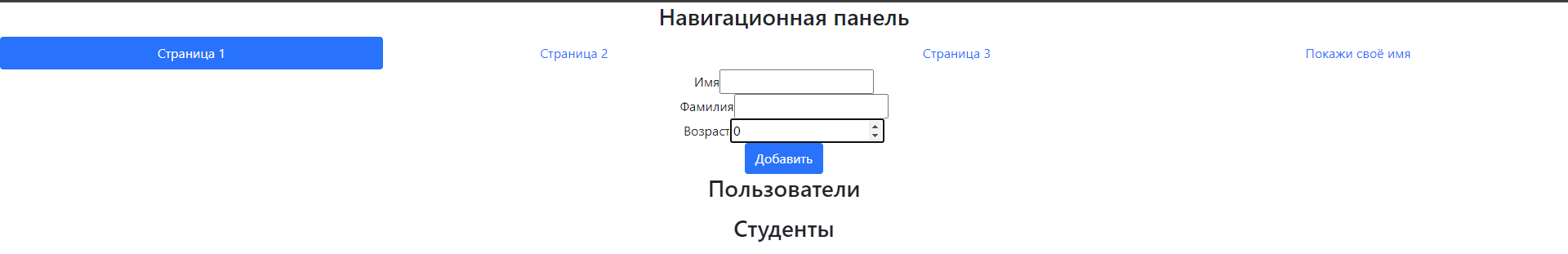
**JavaScript**

JavaScript - это язык программирования, который используется для добавления интерактивности веб-страницам. Он позволяет разработчикам создавать динамические веб-сайты, которые реагируют на действия пользователя. JavaScript - это мощный язык, который используется для создания широкого спектра веб-приложений, таких как игры, карты и онлайн-инструменты.

**Преимущества JavaScript:**

* **Интерактивность:** JavaScript позволяет создавать интерактивные веб-сайты, которые реагируют на действия пользователя.
* **Динамичность:** JavaScript позволяет создавать динамические веб-сайты, которые обновляются без перезагрузки страницы.
* **Функциональность:** JavaScript можно использовать для создания широкого спектра веб-приложений.
  1. Описание интерфейса пользователя

Оформление текста на 1 странице:



Код 1 страницы:

<head>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-+0n0xVW2eSR5OomGNYDnhzAbDsOXxcvSN1TPprVMTNDbiYZCxYbOOl7+AMvyTG2x" crossorigin="anonymous">

</head>

<div>

    <div class = "text-center">

        <label>Имя</label>

        <input name="name" [(ngModel)]="name" (ngModelChange)="onNameChange()"/>

    </div>

    <div class = "text-center">

        <label>Фамилия</label>

        <input name="surname" [(ngModel)]="surname"/>

    </div>

    <div class = "text-center">

        <label>Возраст</label>

        <input type="number" name="age" [(ngModel)]="age"/>

    </div>

    <div class = "text-center">

        <button type="button" class="btn btn-primary justify-content-center">Добавить</button>

    </div>

</div>

<div>

    <h3 class = "text-center">Пользователи</h3>

    <ul>

        <li \*ngFor="let user of users"> {{user.name}} {{user.surname}}  {{user.age}}</li>

    </ul>

</div>

<div>

    <h3 class = "text-center">Студенты</h3>

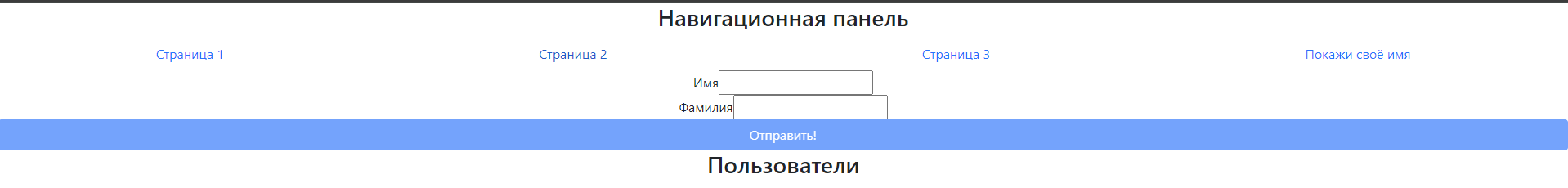
    <ul>

        <li \*ngFor="let student of students"> {{student.name}} {{student.surname}}  {{student.group}}</li>

    </ul>

</div>

Оформление текста на 2 странице:



Код 2 страницы:

<head>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-+0n0xVW2eSR5OomGNYDnhzAbDsOXxcvSN1TPprVMTNDbiYZCxYbOOl7+AMvyTG2x" crossorigin="anonymous">

</head>

<form [formGroup]="myForm" (ngSubmit)="onSubmit()">

    <div class = "text-center">

        <label>Имя</label>

        <input name="userName" formControlName="userName" />

    </div>

    <div \*ngIf="myForm.controls['userName'].invalid && myForm.controls['userName'].touched">

        Не указано Имя!

    </div>

    <div class = "text-center">

        <label>Фамилия</label>

        <input name="userSurname" formControlName="userSurname" />

    </div>

    <div class = "text-center" \*ngIf="myForm.controls['userSurname'].invalid && myForm.controls['userSurname'].touched">

        Не указана Фамилия!

    </div>

    <div class="d-grid gap-2">

        <button class="btn btn-primary justify-content-center" type="submit" [disabled]="myForm.invalid">Отправить!</button>

    </div>

</form>

<div>

    <h3 class = "text-center">Пользователи</h3>

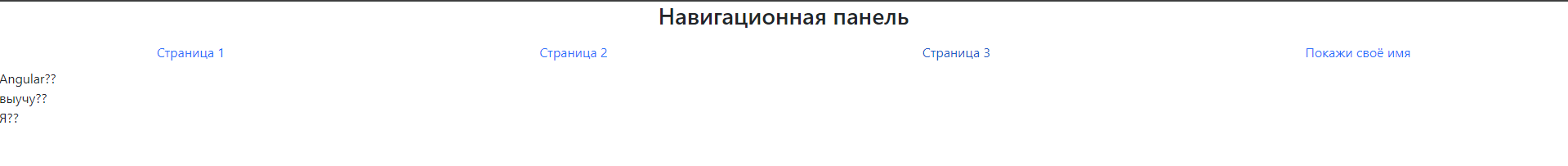
    <ul>

        <li class = "text-center" \*ngFor="let user of users"> {{user.userSurname}} {{user.userName}}</li>

    </ul>

</div>

Оформление текста на 3 странице:



Код 3 страницы:

<head>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-+0n0xVW2eSR5OomGNYDnhzAbDsOXxcvSN1TPprVMTNDbiYZCxYbOOl7+AMvyTG2x" crossorigin="anonymous">

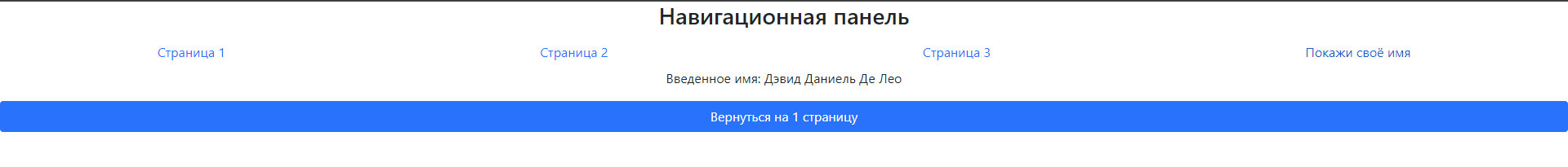
</head>

<div \*ngFor="let item of exampleArray | reverse: {isAddingPointInTheEnd: false, isAddingTwoPointsInTheEnd: true}">

    {{item}}

</div>

Оформление текста на 4 странице:



Код 4 страницы:

<head>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-+0n0xVW2eSR5OomGNYDnhzAbDsOXxcvSN1TPprVMTNDbiYZCxYbOOl7+AMvyTG2x" crossorigin="anonymous">

</head>

<p class = "text-center">Введенное имя: {{name}}</p>

<div class="d-grid gap-2">

    <button class="btn btn-primary justify-content-center" (click)="goMain()"  >Вернуться на 1 страницу</button>

</div>

## 

ЗАКЛЮЧЕНИЕ

В данной работе была разработана веб-приложение для обработки пользователей. Приложение позволяет:

* **Добавлять пользователей:**
  + Вводить имя, фамилию, возраст и группу (только на второй странице).
  + Нажимать кнопку "Добавить".
* **Просматривать пользователей:**
  + Список всех пользователей отображается на первой странице.
  + Список студентов отображается на второй странице.
* **Удалять пользователей:**
  + Нажмите кнопку "Удалить" рядом с пользователем, которого хотите удалить.

Приложение реализовано с использованием следующих технологий:

* HTML, CSS, JavaScript: для создания структуры и оформления интерфейса пользователя.
* Angular: для создания одностраничного приложения и динамического взаимодействия с данными.
* Bootstrap: для создания отзывчивого дизайна.

Приложение имеет простой и понятный интерфейс, а также удобную навигацию.

**В дальнейшем планируется:**

* Реализовать редактирование пользователей.
* Добавить фильтрацию и поиск пользователей.
* Реализовать экспорт данных в различные форматы (CSV, Excel).

**Приложение может быть использовано:**

* В образовательных учреждениях для ведения учета студентов.
* В компаниях для ведения учета сотрудников.
* В любых других организациях, где требуется вести учет пользователей.

**В целом, данное веб-приложение является полезным инструментом для обработки пользователей.**

**Благодарности:**

* Google за создание Angular.
* Bootstrap за CSS-фреймворк.
* Самарскому национальному исследовательскому университету имени академика С.П. Королева за предоставление возможности выполнить данную работу.

Список использованной литературы

x

|  |  |
| --- | --- |
| 1. | М.С. А. Разработка базы данных регистратуры ведомственной поликлиники "Эскулап", Краснодар, 2017. |
| 2. | Учимся проектированию Entity Relationship — диаграмм [Электронный ресурс] // Хабр: [сайт]. [2019]. URL: https:/​/​habr.com/​ru/​articles/​440556/ (дата обращения: 23.05.2023). |
| 3. | JavaScript: описание, преимущества и недостатки, использование [Электронный ресурс] // Otus Journal: [сайт]. [2022]. URL: https:/​/​otus.ru/​journal/​javascript-opisanie-preimushhestva-i-nedostatki-ispolzovanie/ (дата обращения: 23.05.2023). |
| 4. | Что такое JQuery и для чего это нужно, плюсы и минусы [Электронный ресурс] // Boodet.online: [сайт]. [2020]. URL: https:/​/​boodet.online/​blog/​chto-takoe-jquery-i-dlya-chego-eto-nuzhno-pljusy-i-minusy-boodet-online (дата обращения: 23.05.2023). |
| 5. | LocalStorage, sessionStorage [Электронный ресурс] // JavaScript.ru: [сайт]. [2022]. URL: https:/​/​learn.javascript.ru/​localstorage (дата обращения: 23.05.2023). |

x