Правительство Российской Федерации

Федеральное государственное автономное

образовательное учреждение высшего образования

«Национальный исследовательский университет «Высшая школа экономики»

Факультет компьютерных наук

Образовательная программа бакалавриата 09.03.04 «Программная инженерия»

**ОТЧЕТ**

**по производственной практике**

**в**  ООО "1С"

Выполнил студент

группы БПИ191

Царёв К.Д.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



**Проверили:**

**Руководитель практики от предприятия**

Старичков Никита Юрьевич

Заместитель директора по работе с научно-исследовательскими университетами, ООО "1С"

Дата 25.08.2022 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_      \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка)

**Руководитель практики от факультета компьютерных наук**

доцент Департамента программной инженерии ФКН С.А. Шершаков

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

Москва – 2022

СОДЕРЖАНИЕ

[АННОТАЦИЯ 2](#_Toc112341001)

[1. ЦЕЛЬ И ЗАДАЧИ ПРАКТИКИ 3](#_Toc112341002)

[2. ОПИСАНИЕ МЕСТА ПРОХОЖДЕНИЯ ПРАКТИКИ 3](#_Toc112341003)

[3. ОБЗОР ИЗУЧЕННЫХ ТЕХНОЛОГИЙ 3](#_Toc112341004)

[4. ОПИСАНИЕ АЛГОРИТМОВ, ТЕХНОЛОГИЙ, СРЕДСТВ РАЗРАБОТКИ, ИСПОЛЬЗОВАННЫХ ДЛЯ РЕШЕНИЯ ПОСТАВЛЕННЫХ ЗАДАЧ 5](#_Toc112341005)

[5. ЗАКЛЮЧЕНИЕ 8](#_Toc112341006)

[6. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 9](#_Toc112341007)

[7. РАБОЧИЙ ПЛАН-ГРАФИК ПРОХОЖДЕНИЯ ПРАКТИКИ 10](#_Toc112341014)

**АННОТАЦИЯ**

Данный отчет по производственной практике в ООО "1С" содержит следующие разделы: «Цель и задачи практики», «Описание места прохождения практики», «Обзор изученных технологий», «Описание алгоритмов, технологий, средств разработки, использованных для решения поставленных задач», «Заключение», «Список использованных источников», «Рабочий план-график прохождения практики».

В разделе «Цель и задачи практики» указаны поставленная на период практики цель и задачи, которые необходимо было выполнить в ходе прохождения практики.

В разделе «Описание места прохождения практики» указаны виды деятельности и основные задачи ООО "1С".

В разделе «Обзор изученных технологий» описаны изученные алгоритмы, технологии, сервисы.

Раздел «Описание алгоритмов, технологий, средств разработки, использованных для решения поставленных задач» содержит подробное описание проделанной за период прохождения практики работы.

В разделе «Заключение» указаны полученные во время практики результаты и выводы.

Раздел «Список использованных источников» содержит оформленный по ГОСТу список источников.

В разделе «Рабочий план-график прохождения практики» содержится рабочий план-график прохождения практики с отметками о выполнении.

1. **ЦЕЛЬ И ЗАДАЧИ ПРАКТИКИ**

**Цель прохождения практики:**

Реализовать клиентскую часть веб-приложения по планированию отпусков для сотрудников и подключить ее к уже готовой серверной части.

**Задачи практики:**

1. Изучить основы создания и подключения к бэкенду клиентской части веб-приложений;
2. Выбрать ЯП и фреймворки для реализации веб-интерфейса;
3. Реализовать веб-интерфейс;
4. Протестировать веб-интерфейс с использованием локального бэкенда;
5. Опубликовать серверную часть в облаке Microsoft Azure и подключить её к веб-интерфейсу.
6. **ОПИСАНИЕ МЕСТА ПРОХОЖДЕНИЯ ПРАКТИКИ**

Фирма "1С" специализируется на разработке, дистрибьюции, издании и поддержке компьютерных программ. Из разработок "1С" наиболее известна система программ "1С:Предприятие" – решения ERP-класса для повышения эффективности предприятий и учреждений, построенные на инновационной технологической платформе, которая обеспечивает высокую гибкость, масштабируемость и производительность корпоративных решений, работу в режиме "облачного" сервиса (SaaS) и на мобильных устройствах. "1С:Предприятие" успешно применяется для автоматизации управления и учета более чем в 1 500 000 организаций, включая крупнейшие корпорации и госструктуры. Система лидирует на рынке ERP России по количеству автоматизированных рабочих мест.

1. **ОБЗОР ИЗУЧЕННЫХ** **ТЕХНОЛОГИЙ**

В ходе прохождения практики первым делом было начато изучение основ создания клиентской части веб-приложений. В интернете было найдено множество статей и туториалов, связанных с разработкой фронтенда. Так как серверная часть была написана мной же в прошлом году на платформе ASP.Net (язык программирования C#), то было бы логичнее писать клиентскую часть на этой же платформе. Но даже в пределах этой одной платформы оказалось довольно много различных технологий для написания фронтенда. Поэтому остановимся только на вариантах, предлагаемых ASP.Net, и изучим их более детально:

1. Приложения, которые преобразовывают для просмотра пользовательский интерфейс с сервера.
   1. **Razor Pages** — это модель на основе страниц. Задачи, связанные с пользовательским интерфейсом и бизнес-логикой, хранятся отдельно, но на одной странице. Razor Pages — рекомендуемый способ создания приложений на основе страниц или форм для разработчиков, которые еще не работали с ASP.NET Core.
   2. **ASP.NET MVC** отрисовывает пользовательский интерфейс на сервере и использует шаблон архитектуры "Модель — представление — контроллер" (MVC). Шаблон MVC разделяет приложение на три основных группы компонентов: модели, представления и контроллеры. Запросы пользователей направляются в контроллер. Контроллер отвечает за работу с моделью для выполнения действий пользователей и получения результатов запросов. Контроллер выбирает представление для отображения пользователю со всеми необходимыми данными модели.
2. Приложения, которые преобразовывают для просмотра пользовательский интерфейс на клиенте в браузере.
   1. Приложения **Blazor** состоят из компонентов Razor: сегменты доступны для повторного использования, а пользовательский веб-интерфейс реализуется с помощью C#, HTML и CSS. Коды клиента и сервера написаны на языке C#, что позволяет использовать общий код и библиотеки. Компоненты Razor можно преобразовывать для просмотра или предварительно отрисовывать из представлений и страниц.
   2. Одностраничное приложение (SPA) в ASP.NET Core с платформами JavaScript, такими как **Angular** и **React**.
3. Гибридные приложения, использующие преимущества методов отрисовки серверных и клиентских пользовательских интерфейсов. Например, большинство пользовательских веб-интерфейсов преобразовываются для просмотра на сервере, а компоненты, отображаемые клиентом, добавляются по мере необходимости.
   1. **MVC, Razor Pages и Blazor** являются частью платформы ASP.NET Core и разработаны для совместного использования. Компоненты Razor можно интегрировать в приложения Razor Pages и MVC в размещенном решении Blazor WebAssembly или Blazor Server. Одновременно с отрисовкой страницы или представления можно выполнять предварительную обработку компонентов.

Так как в результате практики должно получиться одностраничное приложение по планированию отпусков, была выбрана такая технология как создание SPA (Single Page Application) с использованием фреймворка Angular. Из плюсов такого решения можно выделить:

1. Среда выполнения JavaScript уже предоставлена вместе с браузером.
2. Большое сообщество и продуманная экосистема. (Со многими проблемами при разработке удалось справиться путем изучения различных форумов)

Конечно, есть и недостатки такого решения. Например, на сайте Microsoft указан такой - “Требуются дополнительные языки программирования, платформы и средства.” Но, как мне кажется, практика – это одно из тех мест, где можно попробовать себя в чем-то новом.

Таким образом, был выбран этот фреймворк, и далее было начато изучение основ Angular и JavaScript.

1. **ОПИСАНИЕ АЛГОРИТМОВ, ТЕХНОЛОГИЙ, СРЕДСТВ РАЗРАБОТКИ, ИСПОЛЬЗОВАННЫХ ДЛЯ РЕШЕНИЯ ПОСТАВЛЕННЫХ ЗАДАЧ**

После изучения теории пришло время переходить к практике. Но для начала необходимо было определиться с тем, как должен выглядеть веб-интерфейс. После согласований с руководителем практики, был выбран следующий формат:



Одностраничный сайт со списком всех сотрудников в качестве таблицы и необходимый минимум кнопок.

Разработка фронтенда велась последовательно, то есть, сначала на веб-интерфейс была добавлена только таблица, затем кнопка добавления сотрудника, затем изменения и удаления, затем генерации отпусков и так далее. Поэтому, перед рассмотрением всех этих этапов хочется рассказать про модель сущности сотрудника, хранящуюся в БД.

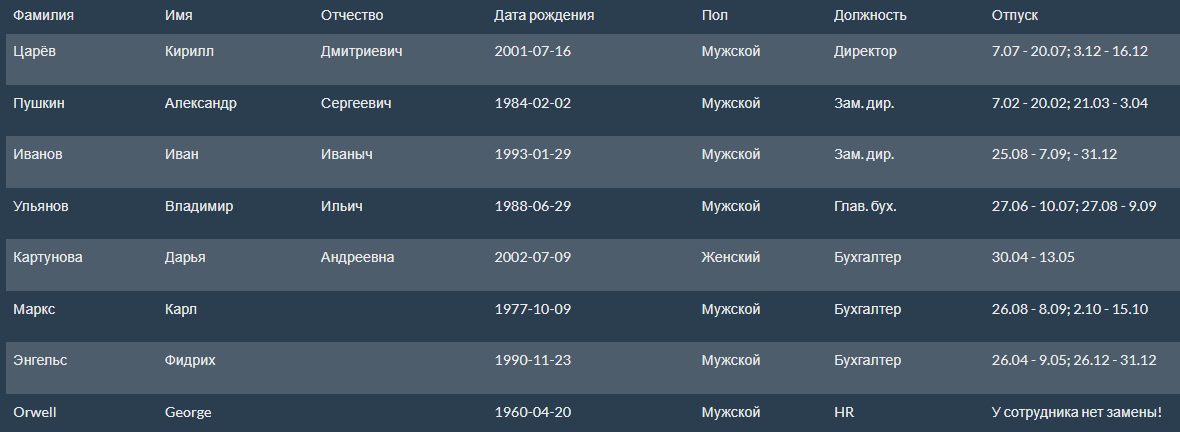
**Модель сотрудников**

Так как данное приложение является лишь прототипом, то в модели сотрудников присутствует минимальный набор полей, необходимый для тестирования: ID сотрудника, фамилия, имя, отчество, дата рождения, пол, должность, даты отпуска. ID сотрудника необходимо только для базы данных, оно представляет из себя целое число, автоматически присваиваемое сотруднику, при добавлении его в БД. Фамилия, имя и отчество представляют из себя строки, ограниченные по длине (от 2 до 40 символов), и по наличию в них только букв. Дата рождения представлена в формате даты, и ограничена возрастом сотрудника (от 16 до 99 лет). Возраст вычисляется исходя из даты рождения и текущей даты. Даты отпусков представлены в строковом формате. Пол представлен целым числом (0 – мужской, 1 - женский). Должность также представлена числом для простоты обращения с ней. Ниже представлены коды должностей и их названия (взаимозаменяемые должности отмечены одинаковым цветом).

1. Директор
2. Заместитель директора
3. Начальник IT-отдела
4. Заместитель начальника IT-отдела
5. Главный бухгалтер
6. Бухгалтер
7. C# разработчик
8. Java разработчик
9. C++ разработчик
10. Дизайнер
11. Сотрудник HR
12. Системный администратор

**Таблица со списком сотрудников**

После того как стало ясно, какие данные должны быть отображены в таблице, непосредственно сама таблица была добавлена в веб-интерфейс. Для начала в приложении не было предусмотрено никаких кнопок или полей для управления списком сотрудников (добавление, изменение, удаление). Данные только получались GET-ом при загрузке страницы, а добавлялись в БД вручную с помощью Postman.



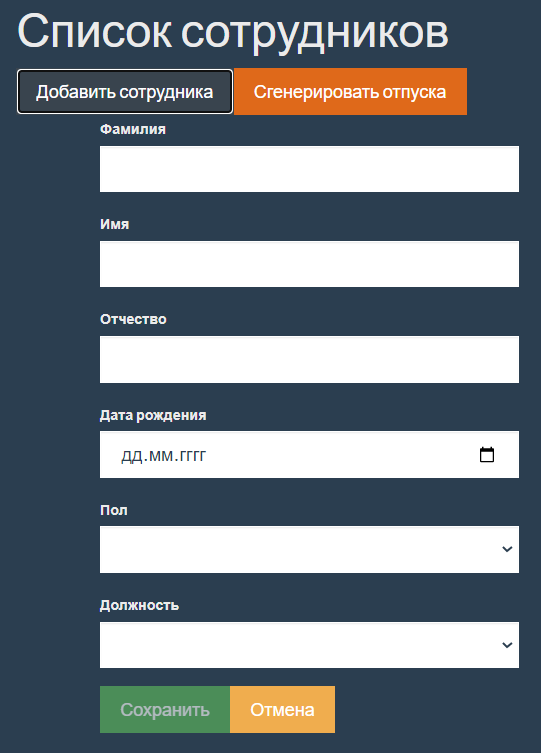
Итак, таблица для просмотра добавлена, но тут же выявились первые проблемы, которые необходимо было решить. Ниже приведен список проблем и краткое описание того, как удалось от них избавиться.

|  |  |
| --- | --- |
| Проблема | Решение |
| Поля “Пол” и “Должность” выводились своим числовым представлением, хранящимся на сервере. | В html разметке страницы были добавлены тернарные операторы при загрузке данных с сервера (если должность = 1, то директор, иначе если должность = 2, то …) |
| Поле “Дата рождения” выводилось не просто числом, но и временем. | В html разметке страницы был добавлен метод обрезки строки при загрузке данных с сервера (выводились только первые 10 символов) |

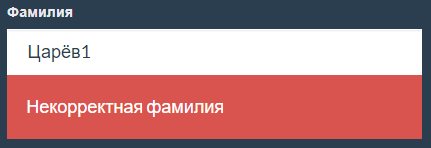
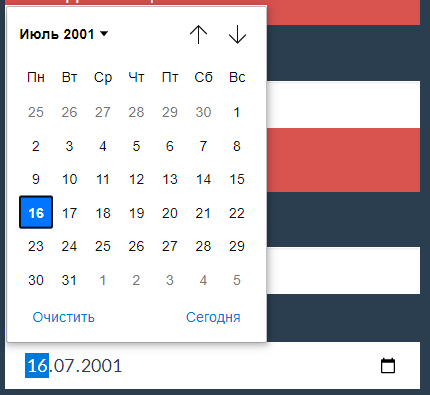
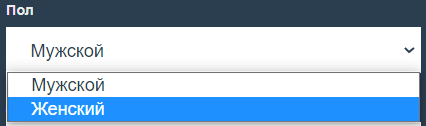
После исправления этих проблем таблица стала иметь вид как на скриншоте выше.

**Форма добавления сотрудника**

Затем была добавлена кнопка добавления нового сотрудника и форма для заполнения информации о нем.

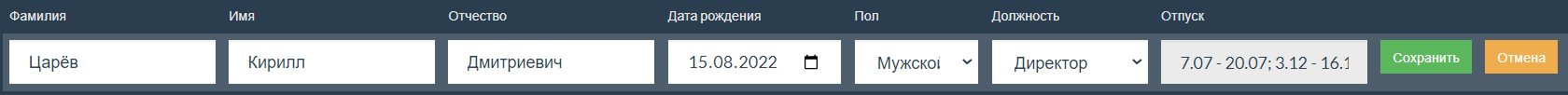


После опыта исправления ошибок в основной таблице, в данной форме их уже не возникло. Пол и должность выбираются из выпадающего списка, и сразу преобразовываются в числовой формат. Дата рождения выбирается в календарном формате. Также все поля сразу валидируются и выдают информативные сообщения об ошибке. К тому же, при ошибке валидации невозможно сохранить данные в БД.

**Изменение данных в таблице**

Изменение данных было реализовано прямо в таблице. При нажатии на кнопку “Изменить”, поля сотрудника становились активны для редактирования.



С точки зрения валидации всё устроено так же, как и при добавлении сотрудника. Единственное отличие – поле “Отпуск” недоступно для редактирования (оно помечено атрибутом “disabled” в html разметке страницы).

Удаление устроено примитивно (без каких-либо подтверждений удаления).

**Генерирование отпусков**

И последним этапом разработки стало добавление функциональности генерирования отпусков. Была добавлена соответствующая кнопка, при нажатии на которую у каждого сотрудника в БД добавлялись отпуска, а затем новые данные подгружались в таблицу веб-интерфейса.

Алгоритм генерации отпусков работает только при наличии как минимум двух взаимозаменяемых сотрудников, поэтому при невозможности запуска алгоритма, поле отпуска остается пустым, и для большей понятности, на стороне клиента в этом пустом поле пишется “У сотрудника нет замены!”

**Публикация**

После окончания разработки веб-приложение было опубликовано в облаке Microsoft Azure (такое решение было выбрано в связи с тем, что это довольно просто сделать прямо из Visual Studio, в которой и проходила разработка, а также потому, что студентам там предоставляется бесплатное обслуживание на 100$ в год для учебных проектов).

Посмотреть готовый проект можно по ссылке: <https://vacationplanning1c.azurewebsites.net/>

Также исходные коды и документацию можно посмотреть на github: <https://github.com/kdtsaryov/VacationPlanning>

1. **ЗАКЛЮЧЕНИЕ**

По окончании практики было реализовано рабочее веб-приложение по планированию отпусков, были получены теоретические знания в различных технологиях разработки веб-клиентов и практические навыки разработки фронтенда. В ходе практики также были изучены принципы работы в Angular, были получены навыки публикации веб-приложений, подключения фронтенда к бэкенду.

1. **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**
2. Выбор пользовательского веб-интерфейса ASP.NET Core [Электронный ресурс] // Microsoft Docs. [2022—2023]. Дата обновления: 05.06.2022. URL: <https://docs.microsoft.com/ru-ru/aspnet/core/tutorials/choose-web-ui?view=aspnetcore-6.0> (дата обращения: 02.07.2022)
3. Пример приложения Web API [Электронный ресурс] // metanit. [2012—2022]. Дата обновления: 25.12.2021. URL: <https://metanit.com/sharp/aspnet6/11.1.php> (дата обращения: 02.07.2022)
4. Учебник. Начало работы с Razor Pages в ASP.NET Core [Электронный ресурс] // Microsoft Docs. [2022—2023]. Дата обновления: 13.06.2022. URL: <https://docs.microsoft.com/ru-ru/aspnet/core/tutorials/razor-pages/razor-pages-start?view=aspnetcore-6.0&tabs=visual-studio> (дата обращения: 03.07.2022)
5. Введение в Angular [Электронный ресурс] // metanit. [2012—2022]. Дата обновления: 05.06.2022. URL: <https://metanit.com/web/angular2/1.1.php> (дата обращения: 04.07.2022)
6. Краткое руководство: связываем ASP.NET Core Web API + Angular 5 [Электронный ресурс] // Хабр. [2006—2022]. Дата обновления: 20.02.2018. URL: <https://habr.com/ru/post/349522/> (дата обращения: 05.07.2022)
7. Angular 11 CRUD with .Net 5 REST API Tutorial [Электронный ресурс] // medium. [2021—2022]. Дата обновления: 05.03.2021. URL: <https://medium.com/scrum-and-coke/angular-11-crud-100-000-rows-d1db21afc31c> (дата обращения: 07.07.2022)
8. Deploy an ASP.NET Core App with EF Core and SQL Server to Azure [Электронный ресурс] // medium. [2021-2022]. Дата обновления: 01.01.2020. URL: <https://medium.com/net-core/deploy-an-asp-net-core-app-with-ef-core-and-sql-server-to-azure-e11df41a4804> (дата обращения: 14.07.2022).
9. **РАБОЧИЙ ПЛАН-ГРАФИК ПРОХОЖДЕНИЯ ПРАКТИКИ**

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Сроки проведения | Планируемые работы | Отметка о выполнении |
| 1 | 01.07.2022 | Инструктаж по ознакомлению с требованиями охраны труда, техники безопасности, пожарной безопасности, а также правилами внутреннего трудового распорядка | ВЫПОЛНЕНО |
| 2 | 02.07.2022 – 05.07.2022 | Изучение основ создания и подключения к бэкенду клиентской части веб-приложений | ВЫПОЛНЕНО |
| 3 | 06.07.2022 | Консультация с руководителем практики по выбору ЯП и фреймворков для написания веб-интерфейса | ВЫПОЛНЕНО |
| 4 | 07.07.2022 – 13.07.2022 | Написание веб-интерфейса с использованием локального бэкенда для тестирования | ВЫПОЛНЕНО |
| 5 | 14.07.2022 | Публикация серверной части приложения в облаке Microsoft Azure и подключение её к веб-интерфейсу | ВЫПОЛНЕНО |
| 6 | 15.07.2022 | Предоставление руководителю практики финального решения | ВЫПОЛНЕНО |