Оглавление

[**1.** **Предметная область** 2](#_Toc103116636)

[**2.** **Диаграммы** 3](#_Toc103116637)

**[3.](#_Toc103116638)****[Требования к приложению](#_Toc103116638)** [10](#_Toc103116638)

**[4.](#_Toc103116639)****[Архитектура приложения](#_Toc103116639)** [11](#_Toc103116639)

[**5.** **Средства разработки** 12](#_Toc103116640)

[**6.** **Модель данных** 14](#_Toc103116641)

[**7.** **Прототип приложения** 16](#_Toc103116642)

[**8.** **Источники данных** 22](#_Toc103116643)

1. **Предметная область**

Управление договорными отношениями, подразумевает собой приложение для взаимодействия менеджера с договорами. В данном приложении должна быть возможность создать договор на оказание какой-либо услуги, которую заказчик заказал у компании.

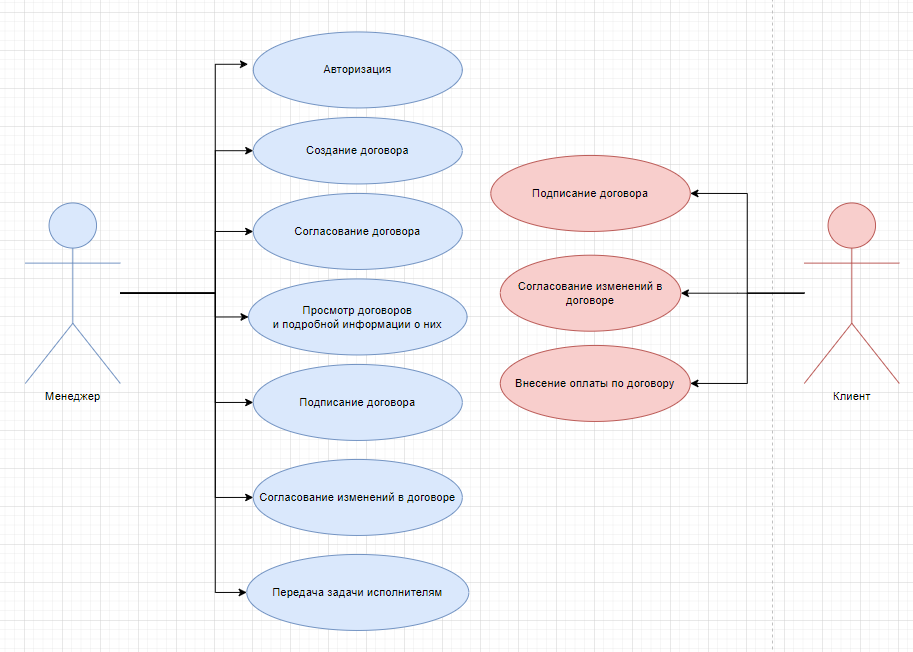
Менеджер имеет свой личный профиль, в котором указана основная информация о менеджере: имя, фамилия, номер телефона, электронная почта, город, дата рождения. Менеджер может просматривать оформленные им договора и информацию о них, например, ФИО клиента, с которым оформлен договор, номер телефона клиента, номер договора, содержание договора и т.д.

Менеджер может создать договор на оказание услуги, об оказании которой он договорился с клиентом. Для создания договора нужно выбрать тип договора в выпадающем списке. После ее выбора мы сможем увидеть первоначальный шаблон договора, примерные временные рамки на выполнение, назначить полную стоимость по договору, указать сумму аванса, разбить платежи на несколько частей и указать сроки оплаты для клиента. Менеджер обсуждает подробные условия с клиентом и вносит корректировки в договор, позже менеджер отправляет предварительный договор юристу компании, который просматривает его и после утверждает его или вносит контрольные корректировки.

Если клиента не устроят условия договора, он сможет связаться с менеджером для обсуждения внесения корректировок. После итогового утверждения менеджер и передает всю необходимую информацию исполнителям. Когда наступает платежный период по договору, клиенту приходит письмо на почту о необходимости погасить задолженность

1. **Диаграммы**

На рисунке 1 представлена диаграмма вариантов использования приложения, отражающая отношения между актерами (пользователями) и прецедентами системы (варианты использования разрабатываемой системы). Она поможет понять, как необходимо построить логику приложения.



На данной диаграмме представлены следующие категории пользователей:

-Клиент – это человек, который решил подписать договор с компанией на оказание услуг. Такой пользователь имеет возможность подписания договора, а также возможное внесение корректировок в договор и внесение оплаты по договору.

-Менеджер – сотрудник компании, который имеет права на просмотр своих договоров, а также, их подписание, согласование и внесение изменений. Менеджер может просматривать всю информацию о договоре.

Выполним описание основных вариантов использования приложения. Например, опишем функцию «Подписание договора». Результат описания представлен в таблице 1.

Таблица 1 Вариант использования "Создание договора"

|  |  |
| --- | --- |
| *Название прецедента* | Создание договора |
| *Цель* | Создать договор с клиентом компании |
| *Актеры* | Менеджер |
| *Тип* | Основной |
| *Стиль* | Базовый |
| *Ссылки (предусловия)* | Реализован прецеденты авторизация |
| *Результаты (постусловия)* | В случае успешного основного сценария передача данных о договоре в базу данных и появление договора в приложении. |

Основной успешный сценарий для данного варианта использования представлен в таблице 2.

Таблица 2 Основной успешный сценарий "Создание договора"

| *Действие актера* | *Отклик системы* |
| --- | --- |
| 1.Менеджер переходит на вкладку «Создание договора».  3. Менеджер выбирает тип договора, временные сроки, заполнить сам договор и выставить цену для клиента. | 2.Приложение переводит пользователя на страницу создания договора.  4.Сервер создает запись нового договора в базе данных и отображает информацию о ней в приложении. |

Далее опишем функцию «Согласование договора»

Таблица 3. Вариант использования «Согласования договора»

|  |  |
| --- | --- |
| *Название прецедента* | Согласование договора |
| *Цель* | Согласовать договор с клиентом компании |
| *Актеры* | Менеджер, клиент |
| *Тип* | Основной |
| *Стиль* | Базовый |
| *Ссылки (предусловия)* | Реализован прецеденты создание договора |
| *Результаты (постусловия)* | В случае успешного основного сценария статус договора измениться на согласован |

Основной успешный сценарий для данного варианта использования представлен в таблице 4.

Таблица 4. Основной успешный сценарий "Согласование договора"

| *Действие актера* | *Отклик системы* |
| --- | --- |
| 1.Менеджер отправляет договор на согласование юристом.  3. Менеджер выбирает нужный договор и изменяет его статус на согласованный. | 2.Юрист согласовывает или вносит корректировки в договор и подписывает договор.  4.Сервер изменяет статус договора на согласованный. |

Далее опишем функцию «Просмотр договоров и подробной информации о них».

Таблица 5. Вариант использования «Просмотр договоров и информации о них»

|  |  |
| --- | --- |
| *Название прецедента* | Просмотр договоров и подробной информации о них |
| *Цель* | Просмотреть информацию о договоре |
| *Актеры* | Менеджер |
| *Тип* | Основной |
| *Стиль* | Базовый |
| *Ссылки (предусловия)* | Реализован прецеденты авторизация |
| *Результаты (постусловия)* | В случае успешного основного сценария в приложении отобразиться вся информация о просматриваемом договоре. |

Основной успешный сценарий для данного варианта использования представлен в таблице 6.

Таблица 6 Основной успешный сценарий "Просмотр договоров и информации о них"

| *Действие актера* | *Отклик системы* |
| --- | --- |
| 1.Менеджер переходит на страницу «Договоры».  3. Менеджер выбирает нужный договор и нажимает на кнопку «подробнее». | 2. Приложение отображает все договоры менеджера.  4. Приложение отображает информацию о договоре. |

Далее опишем функцию «Подписание договора».

Таблица 7 Вариант использования «Подписание договора»

|  |  |
| --- | --- |
| *Название прецедента* | Подписание договора |
| *Цель* | Подписать договор с клиентом компании |
| *Актеры* | Менеджер, клиент |
| *Тип* | Основной |
| *Стиль* | Базовый |
| *Ссылки (предусловия)* | Реализован прецеденты согласование договора |
| *Результаты (постусловия)* | В случае успешного основного сценария статус договора измениться на подписан. |

Основной успешный сценарий для данного варианта использования представлен в таблице 8.

Таблица 8. Основной успешный сценарий «Подписание договора»

| *Действие актера* | *Отклик системы* |
| --- | --- |
| 1.Менеджер подписывает письменный договор с клиентом.  2. Менеджер изменяет статус договора в приложении на «подписан». | 4. Сервер изменяет статус договора на подписан. |

1. **Требования к приложению**

Требования:

1. Разработать возможность авторизации менеджера под личной учетной записью
2. Разработать возможность изменение менеджером статуса договора.
3. Разработать возможность внесения изменения в договор.
4. Разработать возможность создания менеджером нового договра.
5. Реализовать возможность просмотра менеджером всех его договоров.
6. **Архитектура приложения**

Наше приложение будет построено на клиент-сервер архитектуре. Под клиент-серверным приложением мы будем понимать информационную систему, основанную на использовании серверов баз данных.

На стороне клиента выполняется код приложения, в который обязательно входят компоненты, поддерживающие интерфейс с конечным пользователем, производящие отчеты, выполняющие другие специфичные для приложения функции

Клиентская часть приложения взаимодействует с клиентской частью программного обеспечения управления базами данных, которая, фактически, является индивидуальным представителем СУБД для приложения.

К преимуществам архитектуры «клиент-сервер» можно отнести:

* Централизованность, поскольку все данные и управление сосредоточены в центральном сервере;
* Информационная безопасность, поскольку ресурсы общего пользования администрируются централизованно;
* Производительность, использование выделенного сервера повышает скорость работы ресурсов общего пользования;
* Масштабируемость, количество клиентов и серверов можно увеличивать независимо друг от друга.

1. **Средства разработки**

Для разработки нашего приложения можно использовать язык программирования C# и платформу пользовательского интерфейса для создания клиентских приложений для настольных систем WPF в среде разработки Visual Studio.

Рассмотрим преимущества и недостатки использования данных средств разработки:

Преимущества C#:

* C# – это объектно-ориентированный, простой и в то же время мощный язык программирования, который позволяет разработчикам создавать многофункциональные приложения.
* C# относиться к языкам компилируемого типа, поэтому он обладает всеми преимуществами таких языков.
* Мощная библиотека классов. .NET представляет единую для всех поддерживаемых языков библиотеку классов. Какое бы приложение мы ни собирались писать на C# — текстовый редактор, чат или сложный веб-сайт — так или иначе мы задействуем библиотеку классов .NET.
* C# объединяет лучшие идеи современных языков программирования Java, C++, Visual Basic и т.д.
* Из-за большого разнообразия синтаксических конструкций и возможности работать с платформой .Net, C# позволяет быстрее, чем любой другой язык, разрабатывать программные решения.
* Разнообразие технологий. Общеязыковая среда исполнения CLR и базовая библиотека классов — это основа для целого стека технологий, которые разработчики могут задействовать при создании разных приложений. Например, для баз данных в этом стеке имеется технология ADO.NET и Entity Framework Core. Для графических приложений с насыщенным интерфейсом — технологии WPF и UWP. Для более простых графических приложений — Windows Forms. Для разработки мобильных приложений — Xamarin. Для создания веб-сайтов - ASP.NET и т.д.

Но есть у C# и некоторые недостатки:

* Приоритетная ориентированность на платформу Windows;

Преимущества использования WPF:

* **Независимость графики от разрешения и устройства.** Основной единицей измерения в графической системе WPF является аппаратно-независимый пиксель, размер которого составляет 1/96 дюйма. Это создает основу для независимой от разрешения и аппаратной платформы отрисовки. Каждый аппаратно-независимый пиксель автоматически масштабируется в соответствии с заданным в системе количеством точек на дюйм (DPI).
* **Повышение точности.** Система координат WPF основана на числах двойной точности с плавающей запятой, а не числах одинарной точности. Значения преобразования и прозрачности также выражаются числами двойной точности. Платформа WPF также поддерживает широкую цветовую палитру (scRGB) и имеет встроенную поддержку управления входными данными из разных цветовых схем.
* **Расширенная поддержка графики и анимации.** Платформа WPF упрощает программирование графики, автоматически управляя анимированными сценами. Вам не нужно беспокоиться об обработке сцен, циклах отрисовки и билинейной интерполяции. Кроме того, WPF обеспечивает поддержку проверки попадания и полную поддержку альфа-версии компоновки.
* **Аппаратное ускорение.** Система графики WPF использует возможности графического оборудования, чтобы снизить нагрузку на ЦП.

Недостатки использования WPF:

* Плохая документированность.
* Чувствительность к драйверам видеокарты.

Исходя из преимуществ и недостатков было принято решение использовать данные средства разработки для реализации требований нашего приложения. С помощью них мы можем полностью реализовать функционал и визуальную составляющую приложения, на языке C# и XAML, с которыми, как раз, мы уже знакомы, так как изучали его в нашем колледже.

1. **Модель данных**

На этапе построения концептуальной модели и базы данных было выявлено 4 сущности: Клиент, Договор, Типы договоров, Менеджер.

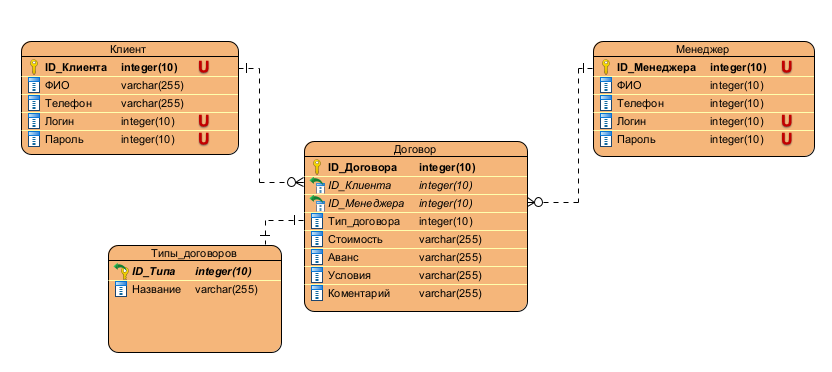


Рисунок 3 - ER-диаграмма

На основе ЕР диаграммы будут построены таблицы: Клиент, Договор, Типы\_договоров, Менеджер.

Таблица «Клиент» - в данной таблице хранятся данные о клиенте. Здесь имеются поля: ID\_Клиента, ФИО, Телефон, Логин, Пароль. Имеет связи с сущностями «Договор».

Таблица 1 описание полей таблицы «Клиент»

|  |  |
| --- | --- |
| ID\_Клиента (PK) | Integer |
| ФИО | nvarchar |
| Телефон | nvarchar |
| Логин | nvarchar |
| Пароль | nvarchar |

Таблица «Менеджер» - в данной таблице хранятся данные о менеджере. Здесь имеются поля: ID\_Менеджера, ФИО, Телефон, Логин, пароль.

|  |  |
| --- | --- |
| ID\_Менеджера | Integer |
| ФИО | nvarchar |
| Телефон | nvarchar |
| Логин | nvarchar |
| Пароль | nvarchar |

Таблица «Договор» - в данной таблице храниться информация о договорах. Здесь имеются поля: ID\_Договора, ID\_Клиента, ID\_Менеджера, Тип\_Договора, Стоимость, Аванс, Условия, комментарий. Имеет связи с сущностями «Клиент», «Типы\_договоров», «Менеджер».

|  |  |
| --- | --- |
| ID\_Договора (РК) | Integer |
| ID\_Клиента (FK) | Integer |
| ID\_Менеджера (FK) | Integer |
| Тип\_Договора | Integer |
| Стоимость | money |
| Аванс | money |
| Условия | nvarchar |
| комментарий | nvarchar |

Таблица «Тип договоров» - здесь имеются поля: ID\_Типа, Название.

|  |  |
| --- | --- |
| ID\_Типа (PK) | Integer |
| Название | nvarchar |

1. **Прототип приложения**



Рисунок 2 – Окно авторизации

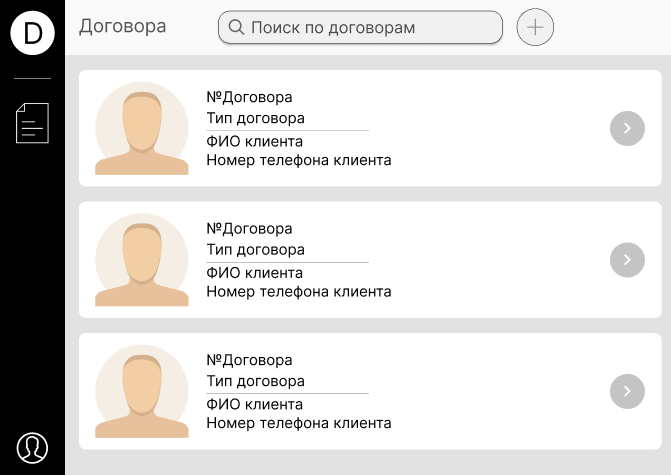


Рисунок 2 – Окно договоров

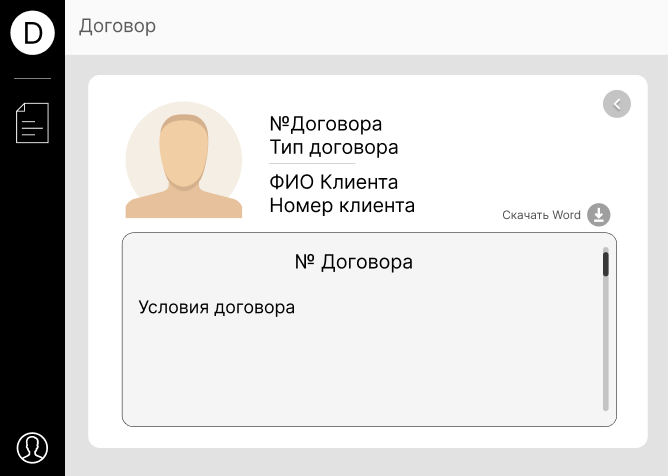


Рисунок 3 – Окно подробной информации о договоре

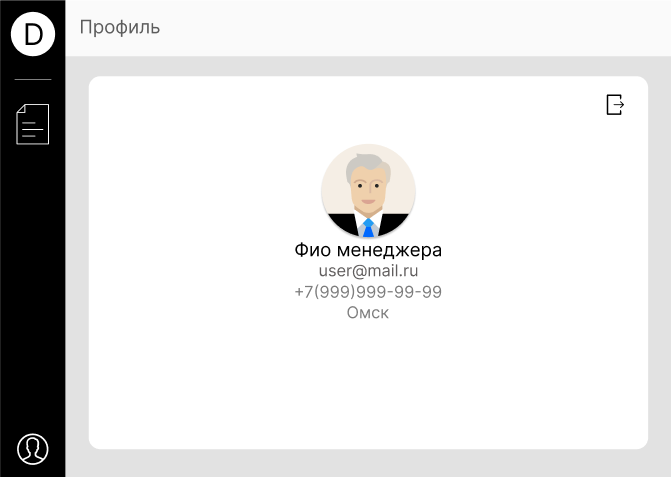


Рисунок 4 – Окно профиля

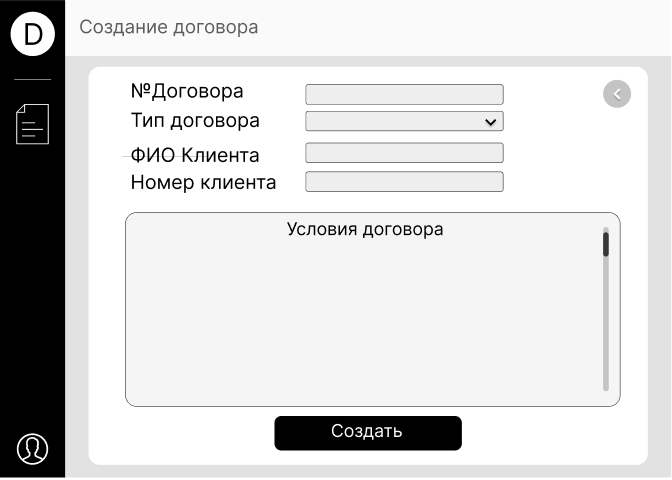


Рисунок 5 – Окно создания договора

Прототип приложения:

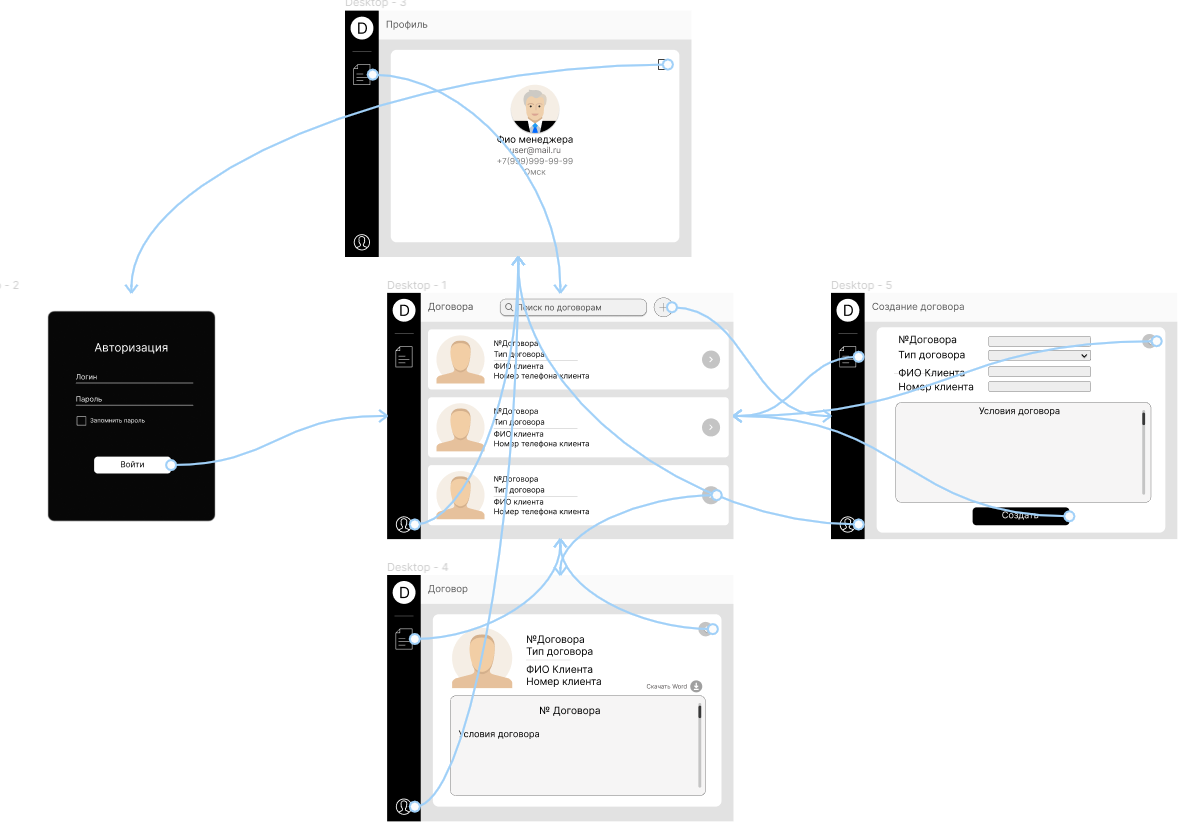


Рисунок 6 Прототип приложения

1. **Источники данных**
2. ГОСТ 28195-89 Оценка программных средств. Общие положения. – 01.09.2013
3. ГОСТ Р 59194-2020 Управление требованиями. Основные положения – 01.01.2021
4. ГОСТ Р ИСО ТО 12182-2002 Класификация программных средств – 01.07.2003
5. ГОСТ Р 56920-2016 Системная и программная инженерия. Тестирование ПО – 01.06.2017
6. ГОСТ Р 57100-2016 Системная и программная инженерия. Описание архитектуры – 01.09.2017
7. Фленов Михаил Библия C#; БХВ-Петербург - М., 2011.Гарнаев А. Самоучитель Visual Studio .NET 2003; БХВ-Петербург - М., 2003
8. Лотка Р. C# и CSLA .NET Framework. Разработка бизнес-объектов; Диалектика / Вильямс - М. Алекс Макки Введение в .NET 4.0 и Visual Studio 2010 для профессионалов; Диалектика / Вильямс - М., 2010.
9. Подбельский, В. В. Язык С#. Базовый курс / В.В. Подбельский. - М.: Финансы и статистика, Инфра-М, 2011. - 384 c.
10. Рихтер, Джеффри CLR via C#. Программирование на платформе Microsoft .NET Framework 4.0 на языке C# / Джеффри Рихтер. - М.: Питер, 2013. - 928 c.
11. Албахари, Джозеф C# 3.0. Справочник / Джозеф Албахари , Бен Албахари. - М.: БХВ-Петербург, 2012. - 944 c.
12. Альфред, В. Ахо Компиляторы. Принципы, технологии и инструментарий / Альфред В. Ахо и др. - М.: Вильямс, 2015. - **266** c.
13. https://metanit.com/sharp/entityframeworkcore/
14. <https://habr.com/ru/c#/>
15. <https://ru.wikipedia.org/wiki/C_Sharp>
16. <https://www.microsoft.com/ru-ru/windows-server> -