Министерство науки и высшего образования
Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования
**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра компьютерных систем в управлении и проектировании (КСУП)

**К ЗАЩИТЕ ДОПУСТИТЬ**
Заведующий кафедрой КСУП
д-р техн. наук, проф.
_____ Ю. А. Шурыгин
_____

# СИСТЕМА АВТОМАТИЗИРОВАННОГО ДОКУМЕНТООБОРОТА ДЛЯ ПРЕДПРИЯТИЯ В ОБЛАСТИ ЭКСПЕРТНОЙ ОЦЕНКИ ТЕПЛОЭНЕРГИТИЧЕСКИХ КОМПЛЕКСОВ

Бакалаврская работа (проект)
по направлению 09.03.01 — Информатика и вычислительная техника

Пояснительная записка
**КСУП.62.01.11 — 01 81 01**

Студент гр. 587-3
_____ А.Д. Андреянов
«___ » _____ 2021 г.

Руководитель
ст. преподаватель
каф ЭМИС ТУСУР
_____ И. Г. Афанасьева
«___ » _____ 2021 г.

Томск 2021

## Реферат

Бакалаврская работа (проект) содержит 40 с., 24 рис., 8 табл., 16 источников.

ДОКУМЕНТООБОРОТ, СИСТЕМА АВТОМАТИЗИРОВАННОГО ДОКУМЕНТООБОРОТА, АВТОМАТИЗАЦИЯ, DOCX.

Целью данной работы является исследование систем автоматизации документооборота и создание веб-приложения для реализации электронной системы документооборота.

В процессе работы изучена предметная область документооборота, рассмотрены аналоги программ, определены проблемы, поставлена задача для решения проблем, выбраны средства реализации, удовлетворяющие потребностям, создан проект предлагаемого решения, разработано Web-приложение в соответствии с заявленными требованиями.

Функциональные схемы разработаны в приложении draw.io, приложение написано в IDE PyCharm, выпускная квалификационная работа выполнена в текстовом редакторе Microsoft Word 2019.

**The abstract**

The bachelor's work (project) contains 40 pages, 24 fig., 8 tab., 16 sources.

DOCUMENT CIRCULATION, AUTOMATED DOCUMENT CIRCULATION SYSTEM, AUTOMATION, DOCX.

The purpose of this work is to study workflow automation systems and create a web application for the implementation of an electronic workflow system.

In the process of work, the subject area of workflow was studied, analogues of programs were considered, problems were identified, a task was set to solve problems, implementation means were selected that satisfy the needs, a draft of the proposed solution was created, a Web application was developed in accordance with the stated requirements.

The functional diagrams were developed in the draw.io application, the application was written in the PyCharm IDE, the final qualifying work was done in the Microsoft Word 2019 text editor.

Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра компьютерных систем в управлении и проектировании (КСУП)

УТВЕРЖДАЮ
Заведующий
кафедрой КСУП
д-р техн. наук, проф.
_____ Ю. А. Шурыгин
_____ 2021 г.

**ЗАДАНИЕ**

на бакалаврскую работу студенту Андреянову Алексею Дмитриевичу
группа 587-3 факультет вычислительных систем

1. Тема работы: Система автоматизированного документооборота для предприятия в области экспертной оценки теплоэнергетических комплексов

(утверждена приказом по вузу от _____ № _____ )

2. Срок сдачи студентом законченного проекта 05.07.2021

3. Назначение и область применения системы:

Данный программный продукт предназначен для хранения , просмотра, редактирования и создания документов в для предприятий, занимающихся экспертной оценкой теплоэнергетических комплексов

4. Требования к работе

Разработать программный продукт для повышения эффективности работы предприятия в области экспертной оценки теплоэнергетических комплексов

5. Перечень вопросов, подлежащих разработке

Разработать систему автоматизированного документооборота

_____
_____
_____
_____
_____

6. Перечень графического материала:
___Презентация_____
_____
_____
_____
_____

ЗАДАНИЕ СОГЛАСОВАНО:
Консультант по нормам и требованиям ЕСКД
_____

Руководитель ВКР
Афанасьева Инга Геннадьевна, ст. преподаватель кафедры ЭМИС ТУСУР
Ф.И.О. должность, место работы
«_____»_____ 2021 г. Подпись_____

Задание принято к исполнению
«_____»_____ 2021 г. Студент_____
подпись

**Оглавление**



Диск CD-R:
Пояснительная записка КСУП.62.01.11 — 01 81 01 на обороте
В формате MS Word 2019 обложки
Презентация в формате PowerPoint 2019
Графический материал:
Презентация в формате PowerPoint 2019……..……__слайдов
Твердая копия презентации…….__экземпляров

| | | | | | КСУП.62.01.11 — 01 81 01 | | | |
|---|---|---|---|---|---|---|---|---|
| Изм. | Лист | № докум. | Подпись | Дата | | | | |
| Разраб. | | А.Д. Андреянов | | | СИСТЕМА АВТОМАТИЗИРОВАННОГО ДОКУМЕНТООБОРОТА ДЛЯ ПРЕДПРИЯТИЯ В ОБЛАСТИ ЭКСПЕРТНОЙ ОЦЕНКИ ТЕПЛОЭНЕРГИТИЧЕСКИХ КОМПЛЕКСОВ | Лит. | Лист | Листов |
| Провер. | | И.Г. Афанасьева | | | | | | Общее число |
| | | | | | | ТУСУР, ФВС, каф.КСУП, гр.587-3 | | |
| Н. Контр. | | Хабибулина | | | | | | |
| Утверд. | | Шурыгин | | | | | | |

# 1 Введение

В настоящее время в различных организациях возрастает количество информационных потоков. В связи с этим возникает необходимость в автоматизации процессов, связанных с документооборотом.

Система автоматизированного документооборота является важным звеном в деятельности организации. Она повышает эффективность работы с документами, что непосредственно сказывается качество и скорость выполнения организацией своих функций.

Целью данной работы является исследование систем автоматизации документооборота и разработка Web-приложения для реализации системы документооборота.

В функционал веб приложения должно входить:

– Добавление новых данных;

– Редактирование данных;

– Удаление данных;

– Отслеживание работ;

– Составление новых стандартизированных документов в формате .docx.

# 2 Теоретическая часть

## 2.1 Описание предметной области

### 2.1.1 Документооборот

Документооборот — это система, включая в себя создание, обработку, прием, передачу, хранение, а также архивирование документов на предприятии [1].

Документооборот, в зависимости от сферы применения, можно подразделить на следующие виды:

– кадровый;

– управленческий;

– административно-хозяйственный;

– бухгалтерский.

– производственный.

Система документооборота имеет ряд преимуществ для предприятия:

– своевременный, эффективный и грамотный бухгалтерский, кадровый, управленческий и налоговый учет;

– оперативное взаимодействие с внешними агентами: вышестоящими организациями, дочерними подразделениями, государственными и контрольно-надзорными органами;

– эффективное управление предприятием;

– управление сотрудниками и контроль производственной деятельности, а также своевременная оплата труда.

Основные принципы документооборота для обеспечения бесперебойной работы системы и исключения утраты документов:

– грамотная маршрутизация – определенные схемы движения для каждого типа документов;

– непрерывность –равномерное распределение нагрузок, а также контролирование времени, отведенного на обработку документа;

– регулирование всех процедур исполнения документа для обеспечения

непрерывности и ритмичности;

– исключение дублирующийся операций – разработка оптимизированных маршрутов, которые исключают выполнение дублирующие операции.

Типы документов структуре документооборота:

– входящие – поступают из внешних каналов связи;

– исходящие – создаются внутри организации и отправляются внешним адресатам;

– внутренние – созданные внутри предприятия и служат для личного пользования;

Существуют три формы организации документооборота:

– централизованная – все документы хранятся в одном месте;

– децентрализованная – документы хранятся в структурных подразделениях, которые осуществляют их обработку, хранение и исполнение;

– смешанная – комбинация централизованной и децентрализованной форм.

### 2.1.2 Электронный документооборот

Система электронного документооборота – это автоматизированная система, сопровождающая процесс управления работой организации и обеспечивающая выполнение этой организациеи своих функций.

В настоящее время большинство компаний используют системы электронного документооборота. Он имеет ряд преимуществ в сравнении с бумажным:

– быстрый и удобный поиск документов;

– неограниченное время хранения информации без потери качества и меньше риски утраты информации;

– разграничение доступа к электронным документам;

– электронные системы обеспечивают строгий контроль за движением документов;

– экономия денежных средств компании.

## 2.2 Проблематика

Основная проблема поставленной темы заключается необходимости вести документооборот вручную. На предприятие поступают заявки, которые обрабатываются в ручном порядке. Сначала вручную составляется договор, смета и счета. По завершению работы так же составляются акты приемки. Эти процессы занимают большое количество времени, и вероятен фактор запутаться в документах или же допустить ошибки. Так же отслеживать работы, при большом количестве не удобно. Примерную схему работы компании можно увидеть на диаграмме, представленной на рисунке 2.1.

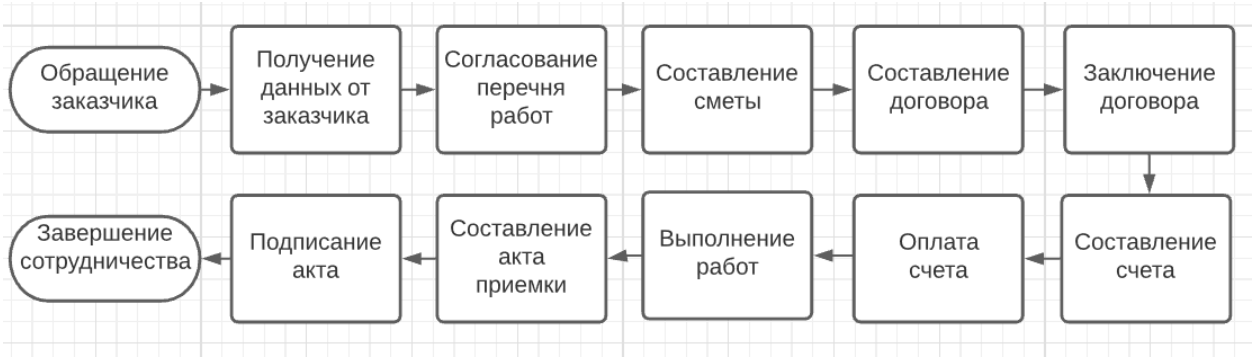



Рисунок 2.2.1 – Схема работы организации

Существуют программы, которые могут выполнят часть поставленных задач [8], например 1С:Документооборот [10]. Она является одной из лидеров в данной сфере. Преимущества программы:

– простой функционал;

– развитая партнерская сеть;

– интерфейс на разных платформах (десктопный интерфейс,web-клиент, приложения на Android и IOS);

– быстрый и удобный поиск;

– простая интеграция с другими продуктами 1С;

– простота настройки.

Недостатка программы:

– высокая стоимость (для небольших компаний);

– нет функции автоматического создания документов по образцу.

Еще одним популярным аналогом является система Контур.Диадок [9]. Преимущества программы:

– интеграция с 1С;

– распознавание текста;

– электронная подпись;

– интерфейс на разных платформах (Windows, Mac, web-клиент);

– поиск;

– сравнение документов;

– управление закупками.

Недостатка программы:

– высокая стоимость подписки;

– плохая техподдержка;

– нет функции автоматического создания документов по образцу.

Так же существует программа ELMA365 ECM [11] – быстрая СЭД с набором умных инструментов для внутреннего и внешнего документооборота. Преимущества программы:

– интеграция с 1С;

– распознавание текста;

– интерфейс на разных платформах (Windows, Mac, Linux, web-клиент);

– функции поиска;

– электронная подпись;

– относительно небольшая стоимость;

– сравнение документов;

– интеграция через API;

– управление закупками;

– бесшовная интеграция с Контур.Диадок.

Недостатка программы:

– нет функции автоматического создания документов по образцу.

Таким образом данные программы удовлетворяет лишь часть потребностей организации, но не имеют нужного функционала. Компания не крупная, поэтому стоимость будет тоже огромным минусом.

Учтя все нужные преимущества и недостатки, можно сделать вывод, что главная цель – это создать приложение, которое будет иметь простой и понятный функционал, удобный интерфейс, быстрый и удобный поиск, кроссплатформенность, возможность составление всех необходимых документов и возможность получить быстрый доступ к необходимому документу. Таким образом схема работы организации изменится в лучшую сторону (рисунок 2.2.1), большинство процессов будут автоматизированы, что сократит время и уменьшит вероятность ошибок.

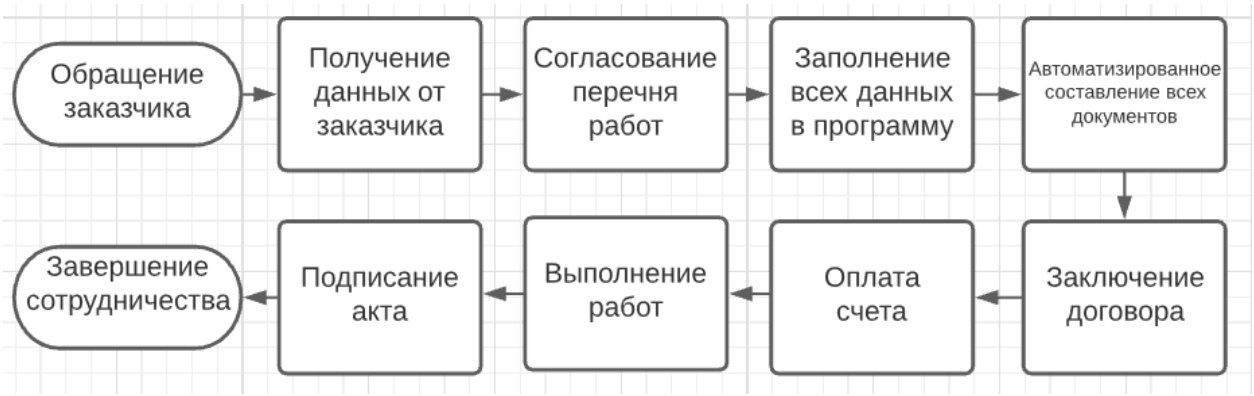



Рисунок 2.2.2 – Новая схема работы организации

## 2.3 Постановка задач

На основании пунктов 2.1 и 2.2 составим назначение приложения и сформируем его функционал. Приложение предназначено для оптимизации работы компании.

Основные требования:

– удобный и понятный интерфейс;

– возможность добавлять и удалять необходимую информацию;

– разграничение прав доступа;

– легкость в освоении программы;

– стабильную работоспособность;

– возможность хранить всю необходимую компании информацию в базе данных;

– возможность отслеживать информацию по работам;

– возможность создавать шаблонизированные документы в формате .docx;

– возможность отправлять документы на печать из приложения.

Приложение должно иметь интерфейс работы с данными (добавление, удаление и редактирование и интерфейс отслеживание работ, кнопки для создания документов в формате .docx, систему учетных записей, для разграничение прав доступа. Так же приложение должно стабильно работать, не иметь критических ошибок, иметь документацию для пользователей и разработчиков.

## 2.4 Выбор средств реализации

Для разработки приложения был выбран фреймворк разработки веб приложений Django использующий язык программирования Python. Так же в качестве можно было сделать десктопное приложение с помощью Windows Forms на языке C#, но выбор был сделан в пользу Python и Django, так как на нем проще написать приложение, и оно будет кроссплатформенным. Так же при необходимости можно загрузить его на сервер и пользоваться web-клиентом из любого места.

Средой разработки была выбрана IDE PyCharm. Она имеет очень удобный интерфейс и много полезных функций, в отличие от других редакторов кода, что упрощает разработку Web-приложения.

Для создания документов .docx будет использована библиотека docxtpl. Она имеет самую понятную документацию и является достаточно популярной, так что при возникновении трудностей проще будет их решить.

В качестве СУБД используем SQLite. Она используется по умолчанию в Django и является оптимальной, так как у нас будет не большой объем базы данных и обращений к ней так же будет не много.

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является объектно-ориентированным. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Язык является интерпретируемым из чего следует такой недостаток, как более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках [7].

Django — это высокоуровневый Python Web-фреймворк, позволяющий создавать безопасные и поддерживаемые Web-сайты в короткие сроки. Созданный опытными разработчиками, Django упрощает многие аспекты Web-разработки, позволяя сосредоточиться на написании Web-приложения без необходимости разрабатывать базовые компоненты. Он является бесплатным, имеет открытый исходный код, большое и активное сообщество, а также хорошую и понятную документацию [4,6,12].

Django помогает писать программное обеспечение, которое будет [5]:

– Полным. Django придерживается философии «Всё включено» и предоставляет почти всё, что разработчики могут захотеть сделать «из коробки». Поскольку всё, что вам нужно, является частью единого «продукта», всё это безупречно работает вместе, соответствует последовательным принципам проектирования и имеет обширную и актуальную документацию.

– Универсальным. Django может быть использован для создания практически любых типов Web-сайтов. Он может работать с любой клиентской средой и может пердавать информацию практически в любом формате. Также, при необходимости Django может быть расширен сторонними компонентами.

– Безопасным. Django включает в себя средства, позволяющие разработчикам избежать большинство распространённых ошибок безопасности, таких как SQL-инъекции, подделка межсайтовых запросов, межсайтовый

скриптинг и кликджекинг.  Также Django имеет безопасный способ управления учётными записями и паролями пользователе, избегая распространённые ошибки, связанные с хранением информации о сеансе в файлах cookie и с хранением пароля вместо хэша пароля [15].

– Масштабируемым. Django использует компонентную архитектуру, каждая её часть не зависит от других и может быть заменена или изменена при необходимости. Такое разделение на части означает, что Django может масштабироваться при увеличении трафика, при помощи добавления оборудования на любом уровне.

– Поддерживаемым. Код Django написан с использованием шаблонов и принципов проектирования, поощряющих создание поддерживаемого и повторно используемого кода. В частности, в нём используется принцип «Don't Repeat Yourself» (DRY, «не повторяйся»), поэтому отсутсвует ненужное дублирование, что приводит к сокращению объёма кода. Также Django способствует группировки связанных функциональных возможностей в повторно используемые модули – приложения (в соответствии с шаблоном Model View Controller (MVC)).

– Кроссплатформенным. Django написан на языке программирования Python, который поддерживается на многих известных платформах. Таким образом пользователь не привязан к конкретной серверной платформе и может запускать Web-приложение на различных версиях Windows, Linux или Mac OS. Также Django поддерживается большинством Web-хостингами, по средствам предоставления определённой инфраструктуры и документации для размещения сайтов, написанных при помощи Django.

PyCharm — это самая интеллектуальная Python IDE с полным набором средств для эффективной разработки на языке Python. Предоставляет средства для анализа кода, инструмент для запуска юнит-тестов, графический отладчик и поддерживает Web-разработку на фреймворке Django. PyCharm была разработана компанией JetBrains [3].

SQLite — компактная встраиваемая СУБД, написана на языке C, существует большое количество привязок к другим языкам программировании, поддерживает динамическое типизирование данных. Поддерживает чтения одновременно несколькими потоками или процессами, однако запись в базу можно осуществить лишь тогда, когда никаких другие запросы в данный момент не обслуживается. Поэтому данная база данных подходит для небольших проектов, управление в которых одновременно ведет один или же не большое количество пользователей [2,13].

## 3 Проект предлагаемого решения

Для работы с данными в проекте воспользуемся средствами, встроенными в фреймворк Django – моделями. Они определяют структуру хранимых данных: типы полей, максимальный размер, значения по умолчанию, параметры списка выбора, текст справки для документации, текст меток для форм. Реализуется с помощью классов в файле models.py.

Для начала разработаем структуру базы данных. Она будет содержать 8 таблиц: Договоры, Сметы, Счета, Акты, Исполнители, Заказчики, Работы и Сотрудники. Составим схему данных, содержащую связи между таблицами (рисунок 3.1).

Для того чтобы выстроить схему данных необходимо для каждой таблицы определить характеристики и свойства полей [14]. Они представлены в Таблицах 3.1 – 3.8.

Таблица 3.1 – Характеристики и свойства полей таблицы «Договоры»

| № | Наименование поля | Тип данных | Свойство поля |
|---|---|---|---|
| 1 | Код договора | Счётчик | PRIMARY KEY |
| 2 | Код заказчика | Счётчик | FOREIGN KEY |
| 3 | Код исполнителя | Счётчик | FOREIGN KEY |
| 4 | Наименование договора | Строка | |
| 5 | Описание договора | Строка | |
| 6 | Статус по оплате договора | Логический | |
| 7 | Дата заключения договора | Дата | |
| 8 | Дата окончания действия договора | Дата | |

Таблица 3.2 – Характеристики и свойства полей таблицы «Сметы»

| № | Наименование поля | Тип данных | Свойство поля |
|---|---|---|---|
| 1 | Код сметы | Счётчик | PRIMARY KEY |
| 2 | Код договора | Счётчик | FOREIGN KEY |
| 3 | Дата формирования сметы | Дата | |

Таблица 3.3 – Характеристики и свойства полей таблицы «Акты»

| № | Наименование поля | Тип данных | Свойство поля |
|---|---|---|---|
| 1 | Код акта | Счётчик | PRIMARY KEY |
| 2 | Код договора | Счётчик | FOREIGN KEY |
| 3 | Дата формирования акта | Дата | |

Таблица 3.4 – Характеристики и свойства полей таблицы «Счёта»

| № | Наименование поля | Тип данных | Свойство поля |
|---|---|---|---|
| 1 | Код счёта | Счётчик | PRIMARY KEY |
| 2 | Код договора | Счётчик | FOREIGN KEY |
| 3 | Дата формирования счёта | Дата | |
| 4 | Статус по оплате счёта | Логический | |

Таблица 3.5 – Характеристики и свойства полей таблицы «Работы»

| Наименование поля | Тип данных | Свойство поля |
|---|---|---|
| Код работы | Счётчик | PRIMARY KEY |
| Код договора | Счётчик | FOREIGN KEY |
| Вид работы | Строка | |
| Описание работы | Строка | |
| Дата начала проведения работы | Дата | |
| Дата окончания работы | Дата | |
| Статус выполнения работы | Логический | |

Таблица 3.6 – Характеристики и свойства полей таблицы «Сотрудники»

| № | Наименование поля | Тип данных | Свойство поля |
|---|---|---|---|
| 1 | Код сотрудника | Счётчик | PRIMARY KEY |
| 2 | ФИО сотрудника | Строка | |

Таблица 3.7 – Характеристики и свойства полей таблицы «Заказчики»

| № | Наименование поля | Тип данных | Свойство поля |
|---|---|---|---|
| 1 | Код заказчика | Счётчик | PRIMARY KEY |
| 2 | Наименование заказчика | Строка | |
| 3 | Адрес заказчика | Строка | |
| 4 | Почтовый индекс заказчика | Строка | |
| 5 | ИНН заказчика | Строка | |
| 6 | КПП заказчика | Строка | |
| 7 | ОГРН заказчика | Строка | |
| 8 | Расчётный счёт | Строка | |
| 9 | Корреспондентский счёт | Строка | |
| 10 | ФИО представителя заказчика | Строка | |

Таблица 3.8 – Характеристики и свойства полей таблицы «Исполнители»

| № | Наименование поля | Тип данных | Свойство поля |
|---|---|---|---|
| 1 | Код исполнителя | Счётчик | PRIMARY KEY |
| 2 | Наименование исполнителя | Строка | |
| 3 | Адрес исполнителя | Строка | |
| 4 | Почтовый индекс исполнителя | Строка | |
| 5 | ИНН исполнителя | Строка | |
| 6 | КПП исполнителя | Строка | |
| 7 | ОГРН исполнителя | Строка | |
| 8 | Расчётный счёт | Строка | |
| 9 | Корреспондентский счёт | Строка | |
| 10 | ФИО представителя исполнителя | Строка | |

Составим схему данных на основе представленных выше таблиц, содержащую связи между таблицами (рисунок 3.1).

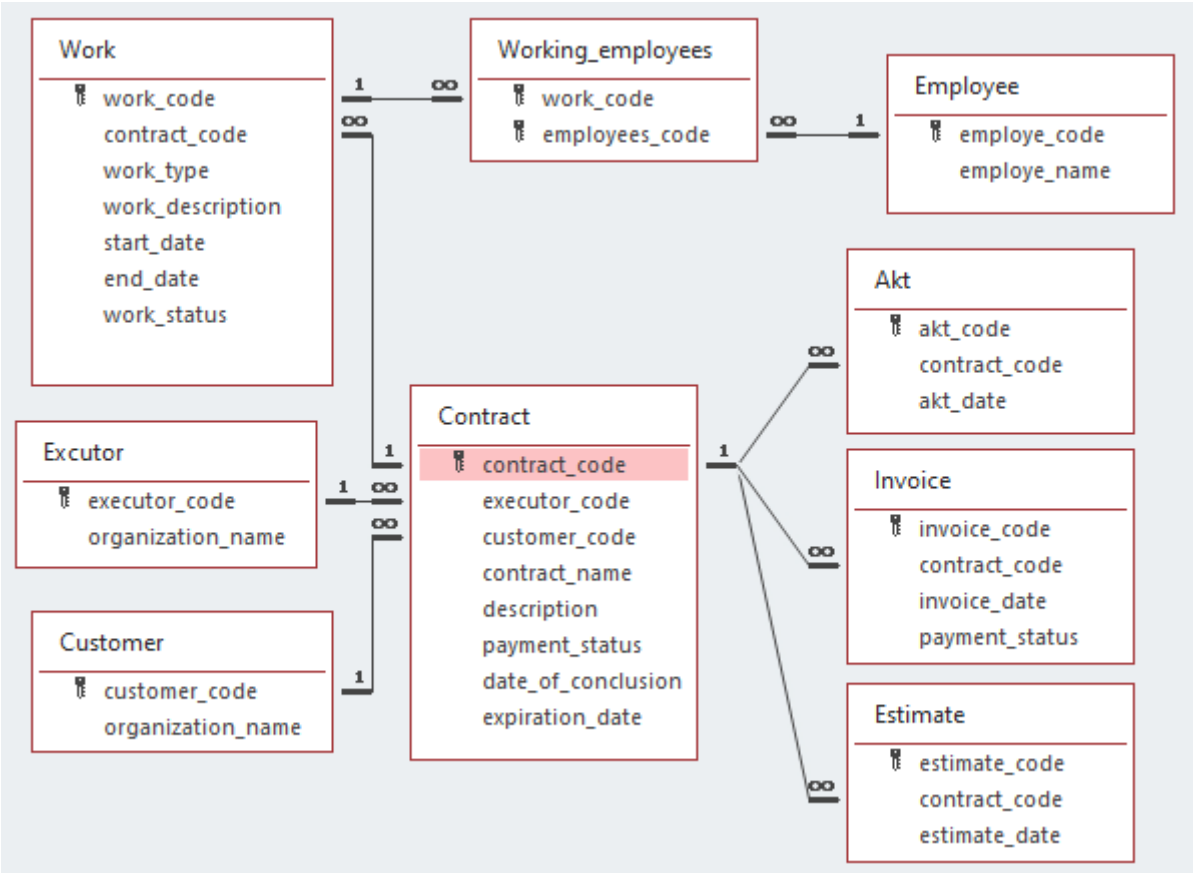



Рисунок 3.1 – Схема базы данных

Разработанная база данных соответствует представленной на рисунке 3.2 ER-модели (Entity-Relationship model).

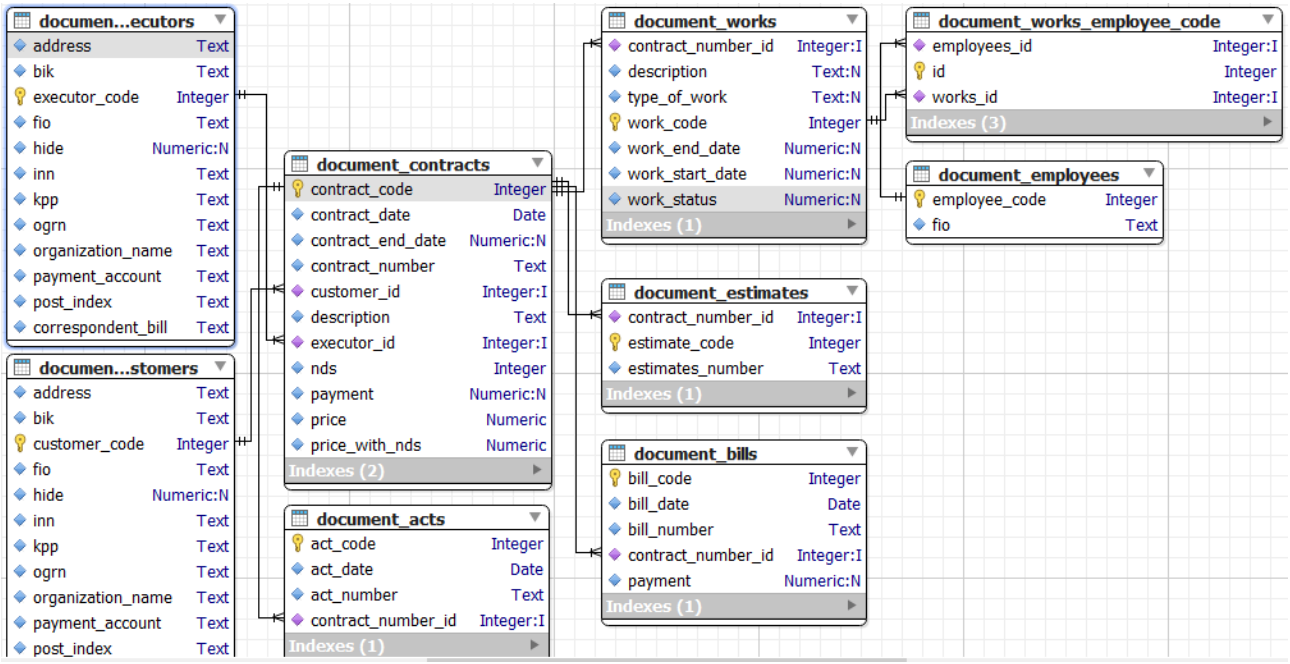



Рисунок 3.2– ER-модель базы данных

Все необходимые операции будут производиться на локальном сервере или на выделенном сервере, исходя из этого, можно выявить требования к системе:

– Процессор с количеством ядер выше 2 и частотой выше 2 ГГц;

– Оперативная память 4 Гб;

– Место на устройстве 300 МБ.

# 4 Разработка Web-приложения

Для начала необходимо установить все необходимое для разработки: язык программирования Python, среду разработки PyCharm, фреймворк Django, а также библиотеку для работы с Word документами docxtpl.

При создании проекта в PyCharm можно подключить виртуальное окружение venv для удобства переноса проекта на другие устройства и на сервер. Виртуальное окружение хранит в себе все установленные библиотеки, а также версию Python, которая была использована при разработке.

При установке фреймворка Django в папке с нашим проектом будут созданы необходимые папки и файлы. Структура файлов показана на рисунке 4.1.

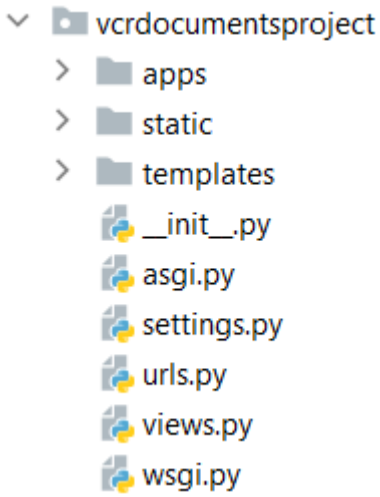



Рисунок 4.1 – Структура файлов и папок

В файле settings.py задаются настройки проекта, в файле urls.py ссылки, в файле views.py функции для обработки страниц. Папка templates и static создается вручную. В templates хранятся шаблоны html страниц, а в static хранятся статические файлы, такие как css стили, картинки, шрифты.

Для дальнейшей работы с Django необходимо создать приложение. Для этого необходимо написать команду в терминале:

*manage.py django-admin startproject document*

Приложение будет находиться в папке apps/document и будет иметь структуру, схожую с основным проетом (рисунок 4.3).

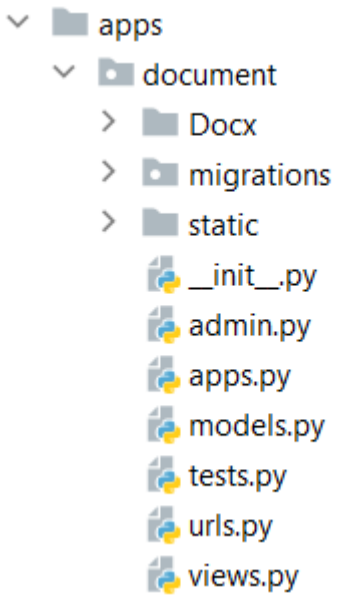



Рисунок 4.2 – Структура приложения document

В приложении появляются новые файлы: models.py – файл для создания моделей для базы данных, apps.py – файл для конфигурации приложения, test.py файл для создания тестов, admin.py – файл для конфигурации системы администрации, встроенной в Django. Так же вручную добавили папку Docx, для хранения шаблонов и преобразованных документов. Папка migrations появится после миграции моделей из файла models.py в базу данных [4,6].

Для проведения миграций необходимо описать все необходимые модели в файл models.py. Пример создания таблицы с договорами представлен на рисунке 4.3.

```
class Contracts(models.Model):
    contract_code = models.AutoField('Код договора', primary_key=True)
    contract_number = models.CharField('Номер договора', max_length = 20)
    customer = models.ForeignKey(Customers, on_delete=models.CASCADE)
    executor = models.ForeignKey(Executors, on_delete=models.CASCADE)
    description = models.TextField('Наименование')
    contract_date = models.DateField('Дата договора')
    contract_end_date = models.DateField('Дата окончания договора', null=True)
    price = models.DecimalField('Сумма',max_digits= 20, decimal_places=2)
    nds = models.IntegerField('НДС')
    price_with_nds = models.DecimalField('Сумма с НДС',max_digits= 20, decimal_places=2)
    payment = models.BooleanField('Оплата по договору', default=False, null=True)


    def __str__(self):
        return self.contract_number

    def was_pub(self):
        return self.date >= (timezone.now() - datetime.timedelta(days = 7))

    class Meta:
        verbose_name = "Договор"
        verbose_name_plural = "Договоры"
```

Рисунок 4.3 – Описание таблицы с договорами в файле models.py

После описания всех таблиц нашей БД необходимо внести изменения в саму базу данных. Для этого нужно ввести следующие команды в терминале:

*manage.py makemigrations*

*manage.py migrate*

После этого к нашему проекту подключена база данных, файл базы данных находится в корневой папке проекта. При последующем изменении моделей, для внесения изменений необходимо ввести аналогичные команды, после введения который будут внесены изменения в БД.

Для отображения данных на сайте необходимо создать шаблон базовой страницы. Создадим файл с названием base.html в папке templates. Для работы с html страницами в Django используется шаблонизатор jinja. С его помощью можно не писать для каждой страницы некоторые повторяющиеся элементы, такие как меню. Воспользуемся возможностями jinja и в файле base.html создадим элементы меню, а в шаблонах других страниц уже будет расширять файл base.html.

Для этого после элементов меню необходимо написать следующий код:

*{% block content %}*

*{% endblock %}*

Для расширения шаблона base.html необходимо будет написать следующий код в новом файле:

*{% extends 'base.html' %}*

*{% block content %}*

*{% endblock %}*

Таким образом на странице будет отображать контент с base.html и в контент, который мы разместили внутри блока block content в новом файле.

Создадим все необходимые шаблоны: для отображения списка договоров, для добавления договора, для отображения договора, для отображения списка заказчиков, для добавления заказчика, для отображения заказчика, для отображения списка исполнителей, для добавления исполнителя, для отображения исполнителя, для отображения списка сотрудников, для добавления сотрудника, для отображения сотрудника, для сохранения сотрудника, для отображения списка работ, для добавления работы, для отображения работы. Все эти шаблоны поместим в папку document. Структура папки с шаблонами представлена на рисунке 4.4.

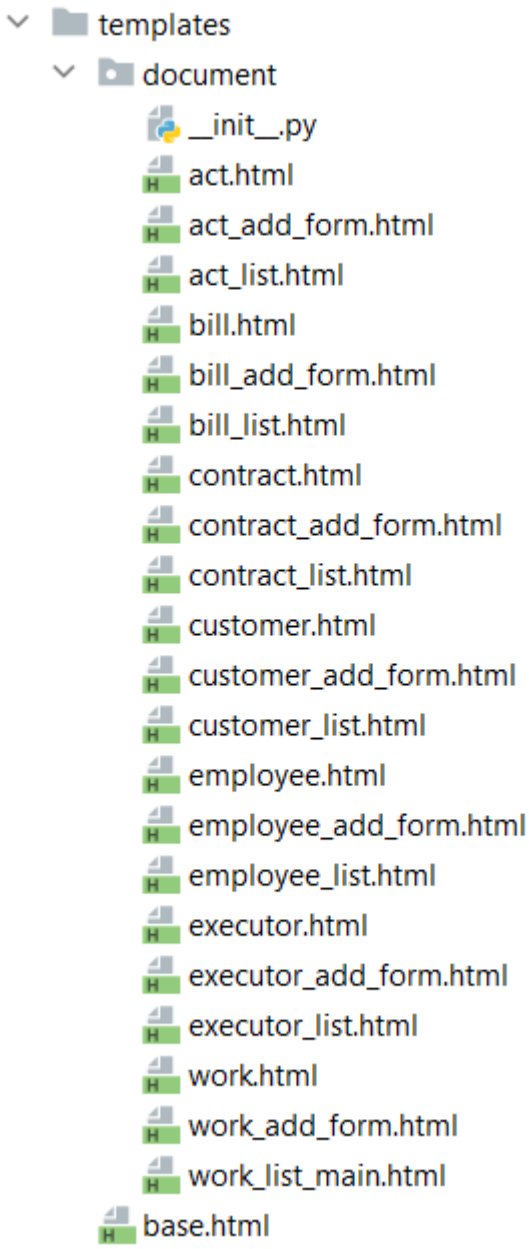



Рисунок 4.4 – Папка с шаблонами

Для обработки Html страниц необходимо написать функции в файле views.py в папке с нашим приложением. Функция должна иметь уникальное название, в качестве аргументов должна принимать обязательный аргумент request, так же может иметь дополнительные аргументы при необходимости. Возвращает функция так же аргумент request, путь до html шаблона в папке templates, а также при необходимости словарь аргументов, которые она передает. Функция отображения списка договоров имеет следующий вид:

```
def contract_list(request):
    contract_list = Contracts.objects.order_by('-contract_date')
    return render(request, 'document/contract_list.html', {'contract_list' : contract_list})
```

Функция получает из базы данных список договоров, отсортированный по дате договора, и рендерит страницу используя шаблон contract_list.html, передавая в него список этих договоров.

Таким образом у нас должны быть написаны функции для создания, сохранения, отображения, и отображения всех для договоров, сотрудников, исполнителей, заказчиков.

Так же необходимо настроить файл urls.py в папке нашего приложения, маршрутизации ссылок. Все страницы в нашем приложении будут начинаться на <url сайта>/contract/. Для обозначения ссылке необходимо в список urlpatterns добавить с path(), которые будет принимать 3 аргумента: '<путь>', функцию, которая будет обрабатывать ссылку (views.<название функции>), имя (name = 'имя'). На рисунке 4.5можно увидеть, как будет выглядеть файл urls.py.

```
app_name = 'contract'
urlpatterns = [
    path('', views.index, name = 'index'),
    path('contract_list/', views.contract_list, name = 'contract_list'),
    path('contract_list/search', views.contract_list_search, name='contract_list_search'),
    path('contract_list/add_form/', views.contract_add_form, name = 'contract_add_form'),
    path('contract_list/add_form/add', views.contract_add, name = 'contract_add'),
    path('contract_list_<str:filter>/', views.contract_list_filter, name='contract_list_filter'),
    path('contract_list/<str:contract_code>/', views.contract, name = 'contract'),
    path('contract_list/<str:contract_code>/print', views.contract_print, name = 'contract_print'),
    path('contract_list/<str:contract_code>/save/', views.contract_save, name = 'contract_save'),
    path('employee_list/', views.employee_list, name = 'employee_list'),
    path('employee_list/add_form/', views.employee_add_form, name = 'employee_add_form'),
    path('employee_list/add_form/add', views.employee_add, name = 'employee_add'),
    path('employee_list/<str:employee_code>/', views.employee, name = 'employee'),
    path('employee_list/<str:employee_code>/save/', views.employee_save, name = 'employee_save'),
    path('executor_list/', views.executor_list, name='executor_list'),
    path('executor_list/add_form/', views.executor_add_form, name='executor_add_form'),
    path('executor_list/add_form/add', views.executor_add, name='executor_add'),
    path('executor_list/<str:executor_code>/', views.executor, name='executor'),
    path('executor_list/<str:executor_code>/save/', views.executor_save, name='executor_save'),
    path('customer_list/', views.customer_list, name='customer_list'),
    path('customer_list/add_form/', views.customer_add_form, name='customer_add_form'),
    path('customer_list/add_form/add', views.customer_add, name='customer_add'),
    path('customer_list/<str:customer_code>/', views.customer, name='customer'),
    path('customer_list/<str:customer_code>/save/', views.customer_save, name='customer_save'),
    path('work_list_main/', views.work_list_main, name='work_list_main'),
    path('work_list/search', views.work_list_search, name='work_list_search'),
    path('work_list_main/<str:filter>', views.work_list_filter, name='work_list_filter'),
    path('work_list_main/save_default<str:work_code>/', views.work_list_save_default, name='work_list_save_default'),
```

Рисунок 4.5 – Файл urls.py

Разработка прототипа завершена, запустим локальный сервер, чтобы посмотреть на работу нашего web приложения. Для запуска необходимо написать команду:

*manage.py runserver*

Сервер запустится по адресу http://127.0.0.1:8000/.

Для добавления договора необходимо, чтобы был хотя-бы исполнитель и один заказчик. Форма для добавления исполнителя представлена на рисунке 4.6. Форма для добавления заказчика выглядит аналогично, за исключением заголовка.

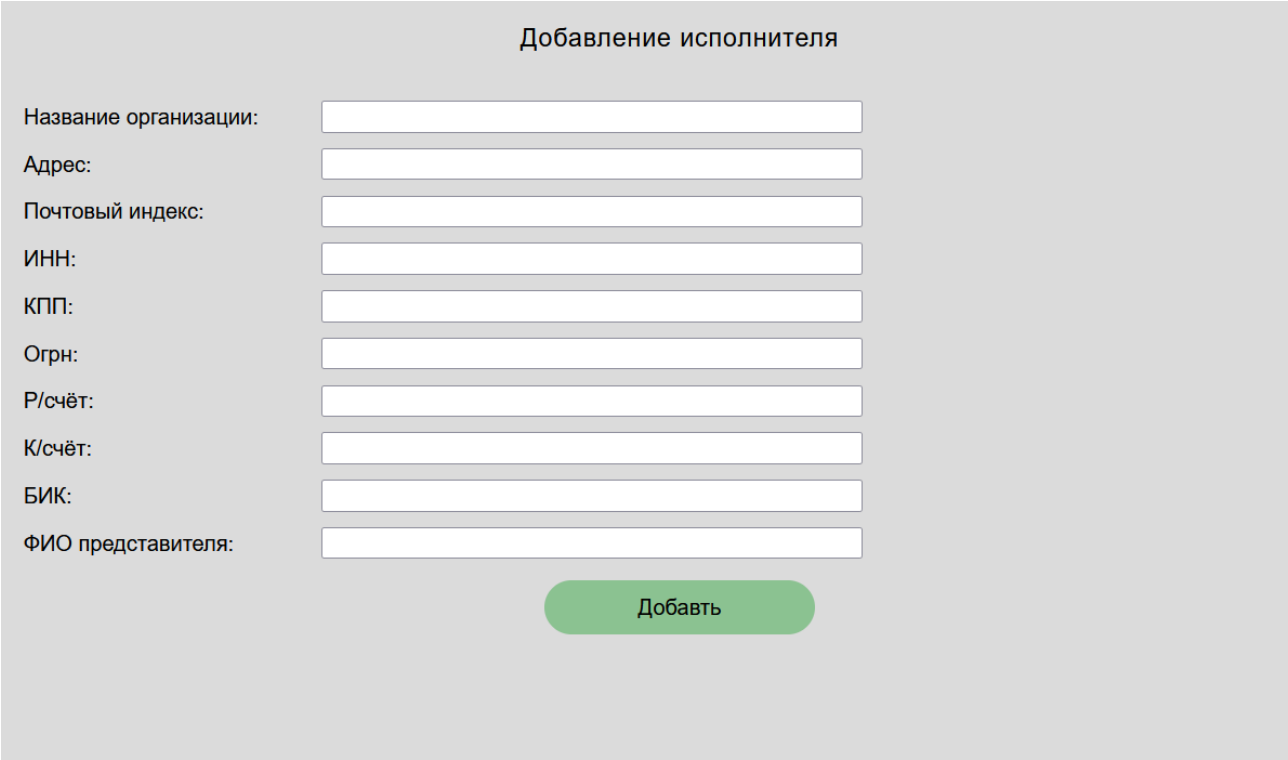



Рисунок 4.6 – Форма для добавления исполнителя

Заполним все данные. После нажатия на кнопку добавить, информация записывается в базу данных и перебрасывает на страницу с списком исполнителей (рисунок 4.7). При нажатии на исполнителя откроется страница для редактирования данных об исполнителе.

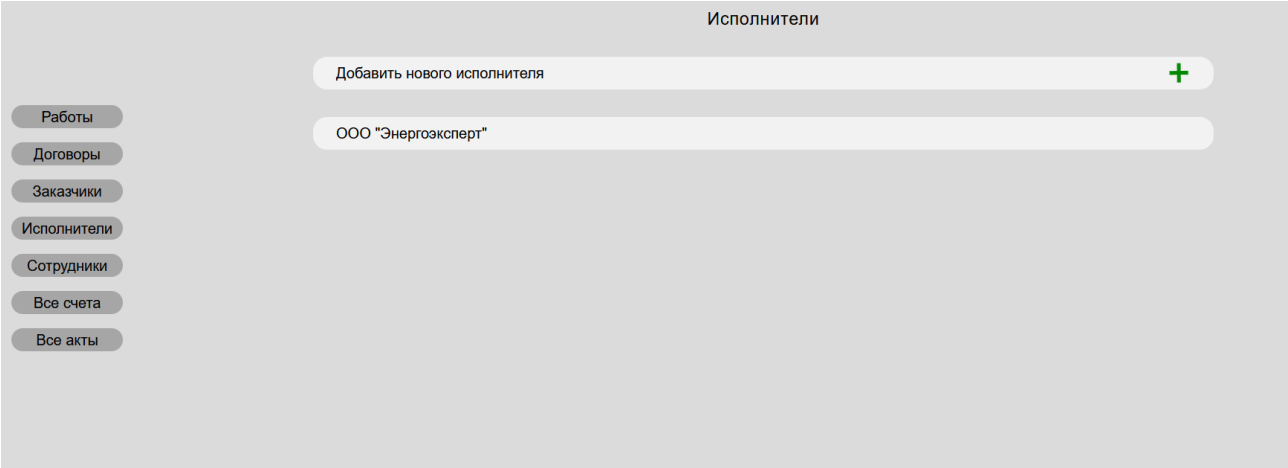



Рисунок 4.7 – Список исполнителей

Аналогично добавим заказчика.

Теперь можно добавить договор. Для этого перейдем на страницу со списком договоров и нажмем на кнопку Добавить новый договор. Откроется страница с добавлением договора (рисунок 4.8).

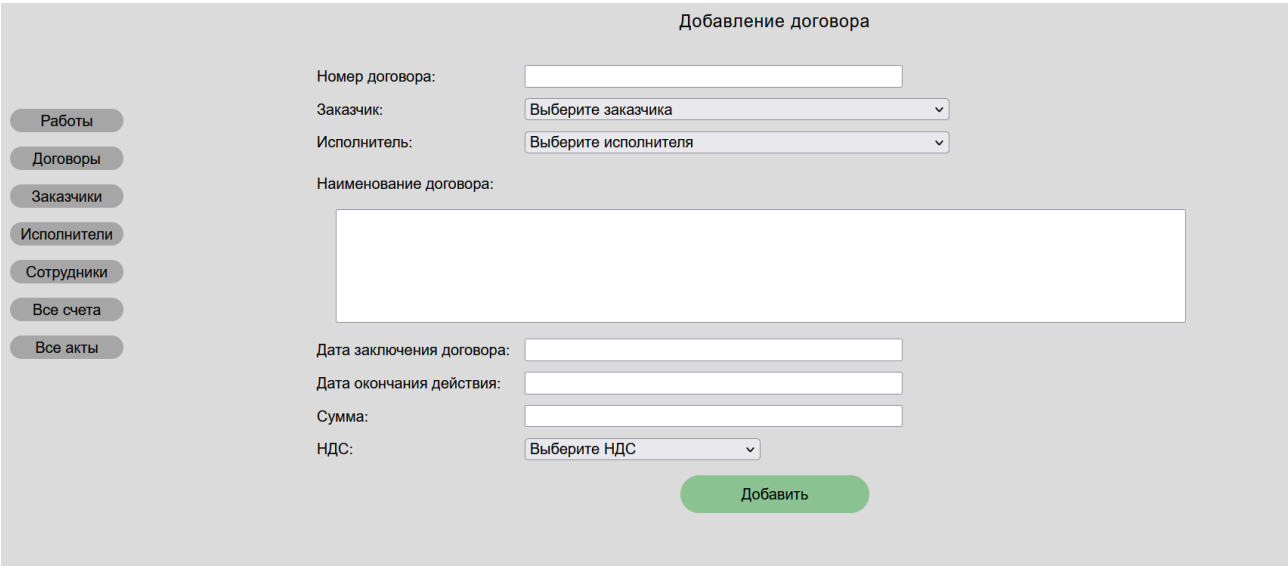



Рисунок 4.8 – Страница добавления договора

При нажатии на всплывающее меню с заказчиком, у нас появляется возможность выбрать заказчика, среди тех, которых мы добавили (рисунок 4.9). Аналогично можно выбрать исполнителя. В поле НДС у нас есть возможность выбрать НДС среди заранее указанных значений. После заполнения всех полей и нажатия кнопки сохранить добавить, данные записываются в базу данных и перебрасывает на страницу со списком договоров (рисунок 4.10). На данной

странице также реализован поиск договоров по номеру и фильтрация по актуальным договорам.

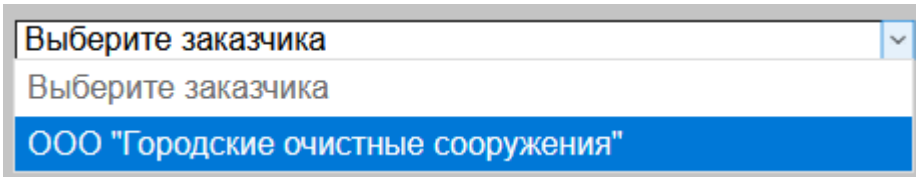



Рисунок 4.9 – Выбор заказчика

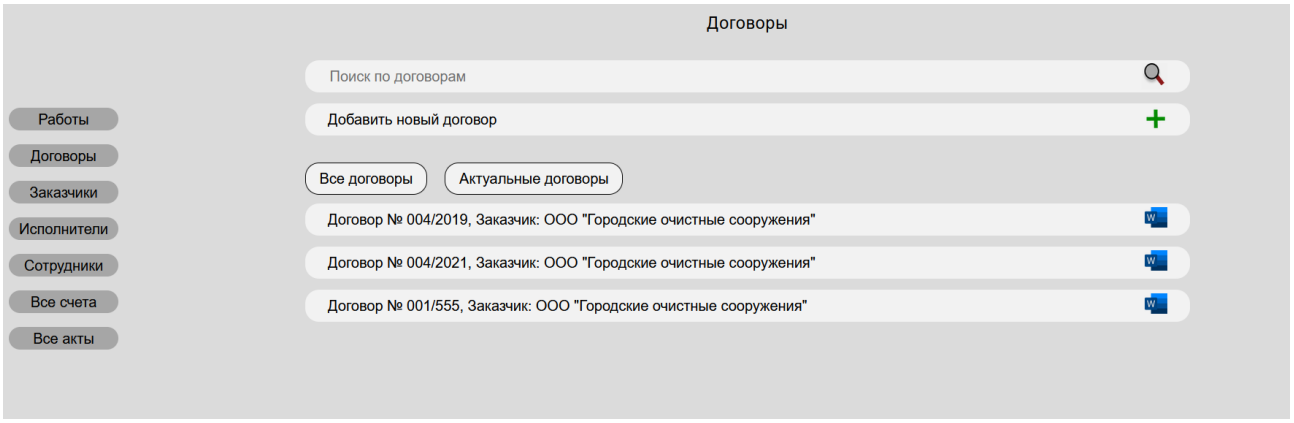



Рисунок 4.10 – Список договоров

Далее перейдем на вкладку сотрудники и нажмем добавить нового сотрудника, далее нажмем на кнопку Добавить. Откроется страница с списком сотрудников (рисунок 4.11). Добавим еще двух сотрудников.

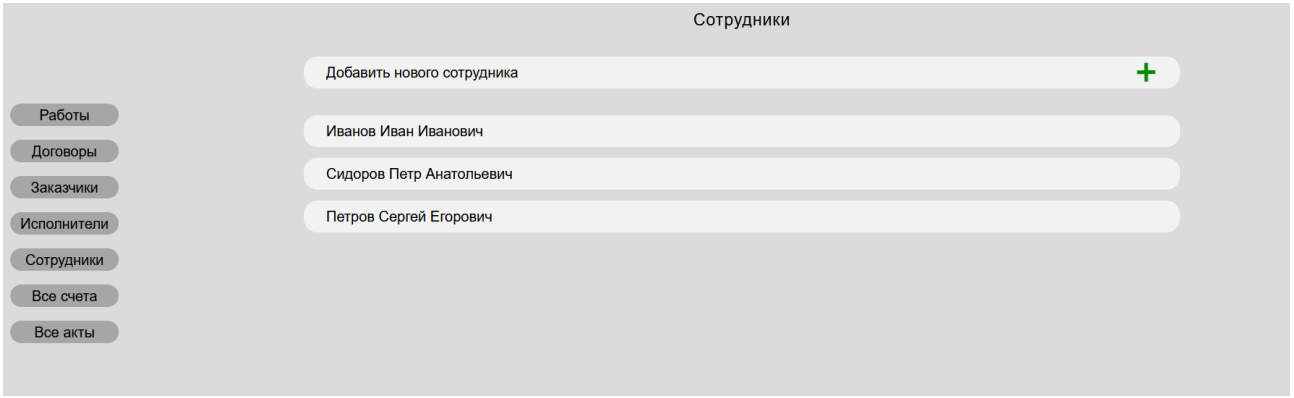



Рисунок 4.11 – Список сотрудников

Теперь можно добавить работу. На вкладке с списком работ, нажмем на кнопку добавить работу. Откроется страница с формой для добавления работы (рисунок 4.12). В Списке с сотрудниками можно выбрать сразу несколько сотрудников с помощью клика по сотруднику и нажатия кнопки ctrl.

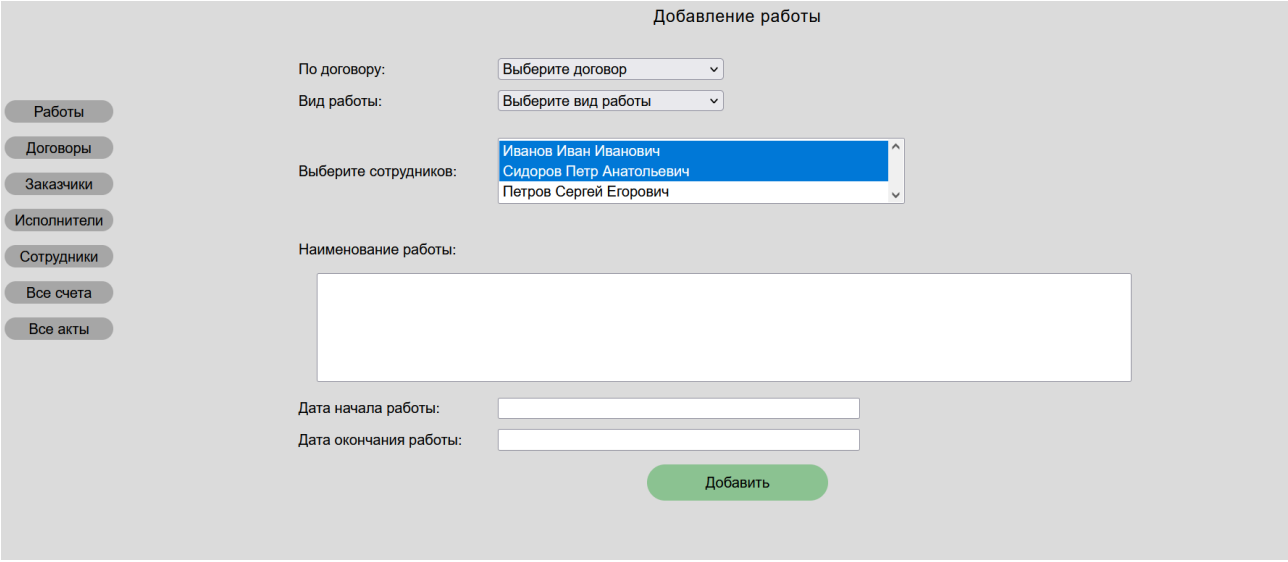



Рисунок 4.12 – Добавление работы

Заполним все поля и нажмем на кнопку добавить, откроется страница с списком работ (рисунок 4.13). На данной странице реализована фильтрация работ по актуальным работам, актуальным договорам, а также поиск работы по ее наименованию. Так же с этой страницы можно изменить состояние работы и статус оплаты договора, нажав на соответствующий чек бокс.

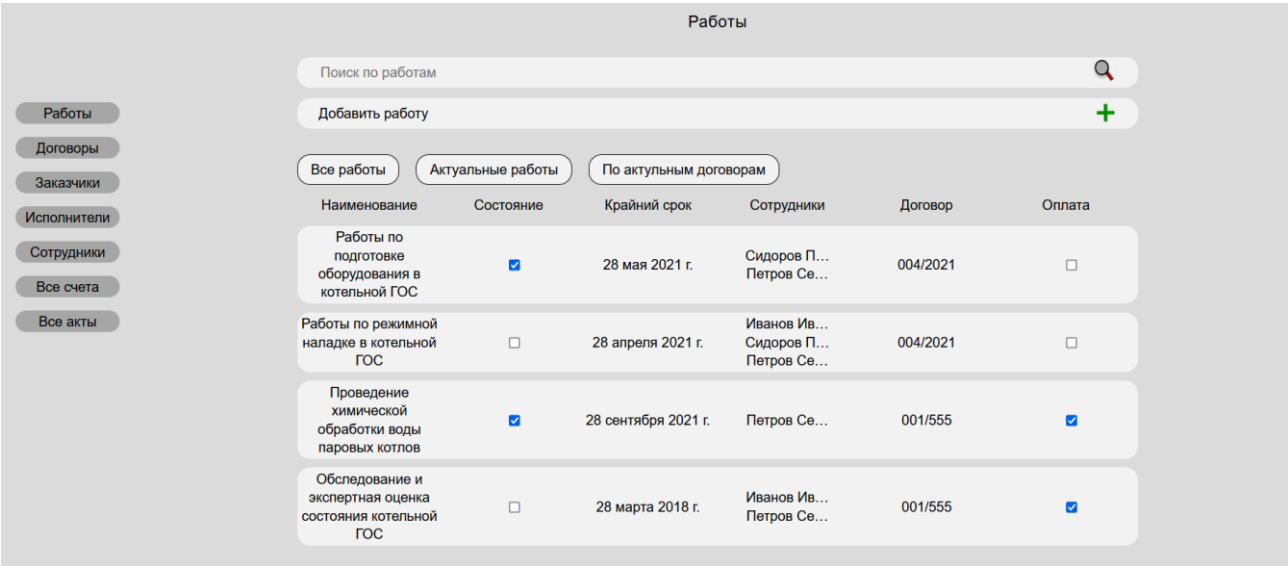



Рисунок 4.13 – Список работ

После добавление работы, она так же отображается в договоре и мы так же можем ее редактировать напрямую из договора (рисунок 4.14).

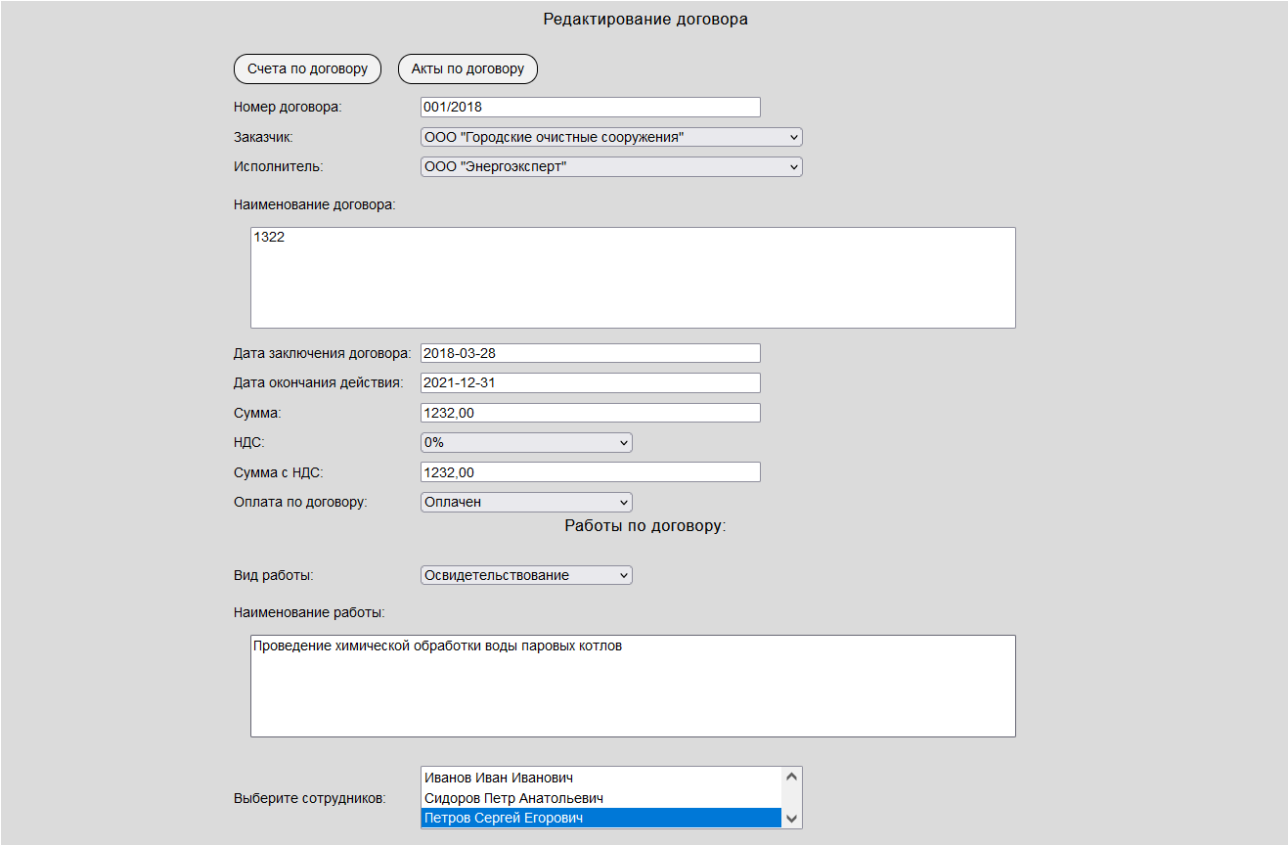



Рисунок 4.14 – Редактирование договора

Добавим страницы с счетами и актами. Функционал добавления нового счета и нового акта аналогичен представленным ранее. Добавим несколько счетов и актов. Страницы представлены на рисунке 4.15 и рисунке 4.16.

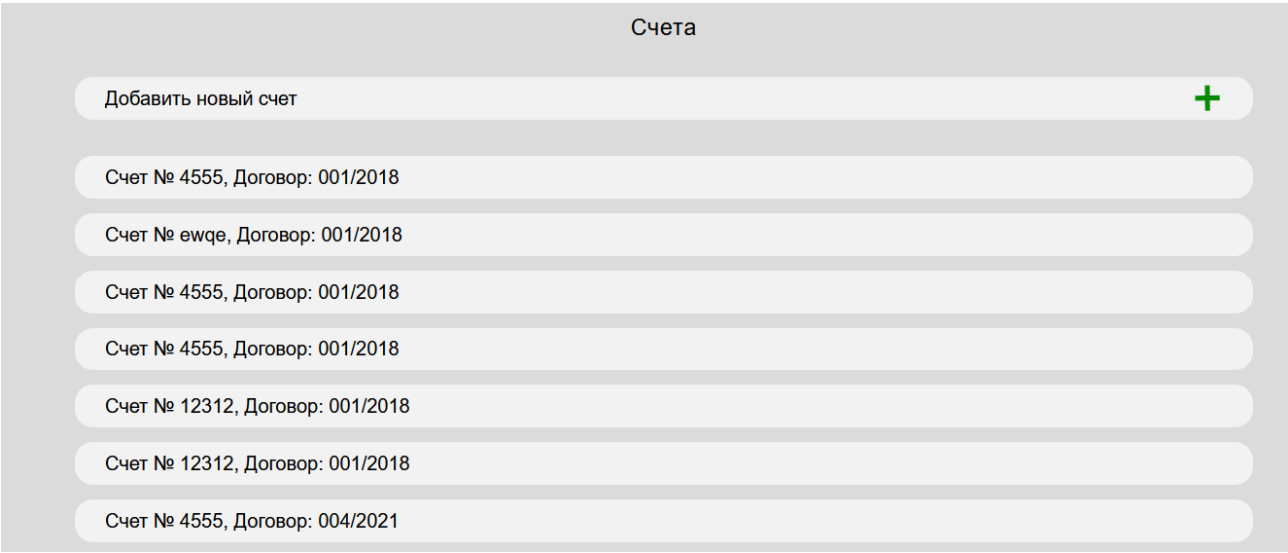



Рисунок 4.15 – Список счетов

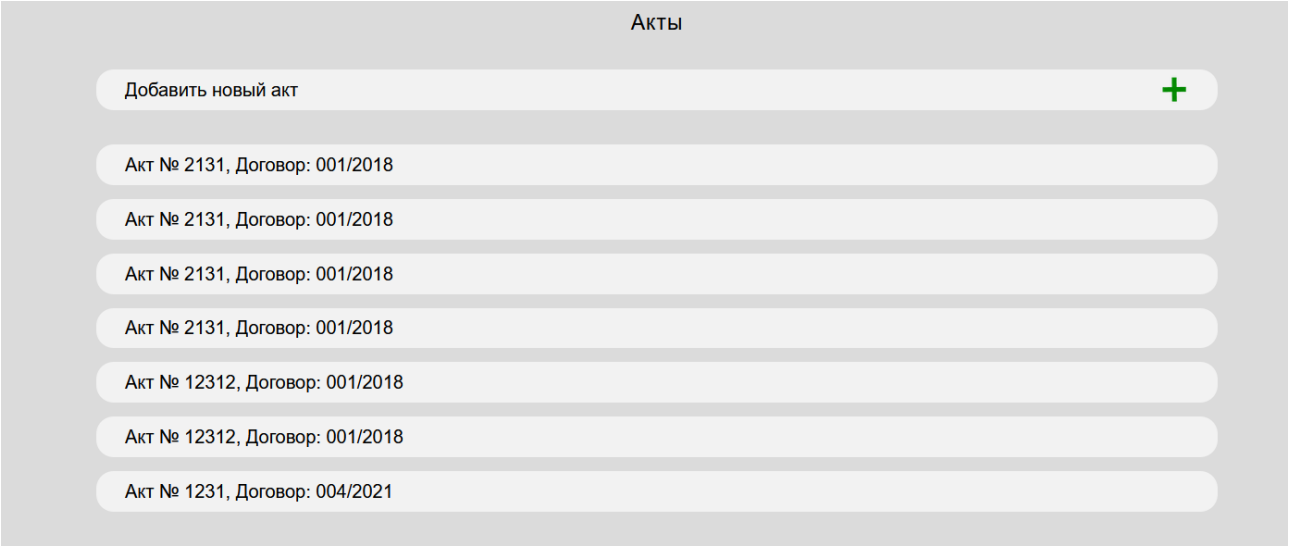



Рисунок 4.16 – Список актов

При переходе на страницу счетов или актов по договору через редактирование договора, откроются страница со счетами или актами по конкретному договору, а также можно при добавлении автоматически будет выбран этот договор. Счета по конкретному договору представлены на рисунке 4.17.


Рисунок 4.17 – Счета по конкретному договору

Функционал удаления данных реализован через панель администратора, для обеспечения сохранности данных.

Основной функционал Web-приложения реализован. Необходимо добавить функцию создания Договора в формате docx. Для этого создадим функцию в файле views.py. Для работы с библиотекой необходимо создать шаблон документа. В шаблоне вместо полей, которые у нас будут меняться впишем в

двойных квадратных скобках переменную, которую будет которая будет соответствовать соответствующему значению из базы данных по конкретному договору. Шаблон представлен на рисунке 4.18.

**Договор № {{contract_number}}**

г. Томск {{contract_date}}

{{customer}}, именуемое в дальнейшем как Заказчик, в лице генерального директора {{customer_fio}}, действующего на основании Устава, с одной стороны и {{executor}} с другой, именуемое в дальнейшем как «Исполнитель», в лице исполнительного директора {{executor_fio}}, действующего на основании доверенности № 18 от 15.01.2018 г. С другой стороны, именуемые по тексту «Стороны», заключили настоящий договор о нижеследующем:

1. **Предмет договора**

1.1. Заказчик получает, а Исполнитель обязуется выполнить следующие работы: «{{work_description}} {{customer}}».

1.2. Содержание выполняемых работ определяется приложенной сметой (Приложение №1 к настоящему Договору), являющейся неотъемлемой частью договора.

1.3. Сроки проведения работ: {{work_end_date}}.

2. **Стоимость работ, порядок расчёта**

2.1. Договорная цена определяется на основании сметы (Приложение №1) и составляет {{price}} руб. (девяносто четыре тысячи двести девяносто три рубля двадцать копеек, в том числе НДС {{nds}}% – {{nds_price}} руб. (четырнадцать тысяч триста восемьдесят три рубля семьдесят одна копейка).

2.2. Оплата работ производится Заказчиком путём перечисления денежных средств на расчётный счёт Исполнителя в течении 30 дней со дня подписания акта приёмки-сдачи выполненных работ. Оплата может быть произведена по соглашению Сторон иными способами, установленными действующим законодательством.

3. **Порядок сдачи и приёмки работ**

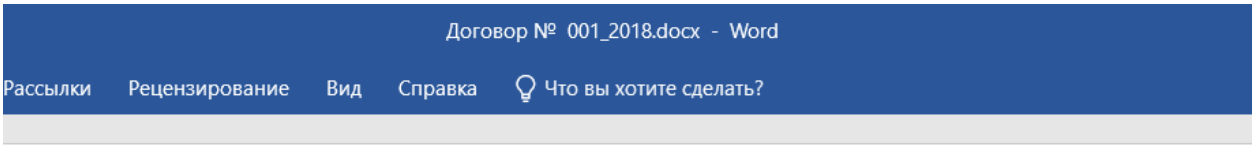
3.1. При завершении работ Исполнитель представляет Заказчику акт приёмки-сдачи работ и счёт-фактуру.

3.2. Заказчик в течении 5 рабочих дней со дня получения акта приёмки-сдачи работ и прочих документов, представляемых согласно настоящему Договору, обязан направить Исполнителю подписанный акт приёмки-сдачи выполненных работ, либо мотивированный отказ с указанием причин.

3.3. Передача оформленной в установленном порядке исполнительно-технической документации осуществляется с сопроводительными документами Исполнителя.

Рисунок 4.18 – Шаблон документа

Функцию обработки документа добавим в файл urls.py, а также вынесем кнопку, рядом с договором в списке договоров. При нажатии на кнопку будет генерироваться документ в формате docx, в названии будет иметь номер договора и содержать в себе отформатированный договор. Получившийся документ представлен на рисунке 4.19.




**Договор № 001/2018**

г. Томск 28 марта 2018 г.

ООО "Городские очистные сооружения", именуемое в дальнейшем как Заказчик, в лице генерального директора Воронова Олега Михайловича, действующего на основании Устава, с одной стороны и ООО "Энергоэксперт" с другой, именуемое в дальнейшем как «Исполнитель», в лице исполнительного директора Дубровина Алексея Валентиновича, действующего на основании доверенности № 18 от 15.01.2018 г. С другой стороны, именуемые по тексту «Стороны», заключили настоящий договор о нижеследующем:

**1. Предмет договора**

1.1. Заказчик получает, а Исполнитель обязуется выполнить следующие работы: «Работа 1 ООО "Городские очистные сооружения"».

1.2. Содержание выполняемых работ определяется приложенной сметой (Приложение №1 к настоящему Договору), являющейся неотъемлемой частью договора.

1.3. Сроки проведения работ: 2021-03-28.

**2. Стоимость работ, порядок расчёта**

2.1. Договорная цена определяется на основании сметы (Приложение №1) и составляет 15815.54 руб. (девяносто четыре тысячи двести девяносто три рубля двадцать копеек, в том числе НДС 18% – 2412.54 руб. (четырнадцать тысяч триста восемьдесят три рубля семьдесят одна копейка).

2.2. Оплата работ производится Заказчиком путём перечисления денежных средств на расчётный счёт Исполнителя в течении 30 дней со дня подписания акта приёмки-сдачи выполненных работ. Оплата может быть произведена по соглашению Сторон иными способами, установленными действующим законодательством.

**3. Порядок сдачи и приёмки работ**

3.1. При завершении работ Исполнитель представляет Заказчику акт приёмки-сдачи работ и счёт-фактуру.

3.2. Заказчик в течении 5 рабочих дней со дня получения акта приёмки-сдачи работ и прочих документов, представляемых согласно настоящему Договору, обязан направить Исполнителю подписанный акт приёмки-сдачи выполненных работ, либо мотивированный отказ с указанием причин.

Рисунок 4.19 – Созданный договор

Аналогичным образом реализован процесс создания сметы, счета и акта.

Диаграмма вариантов использования для Web приложения представлена на рисунке 4.20.

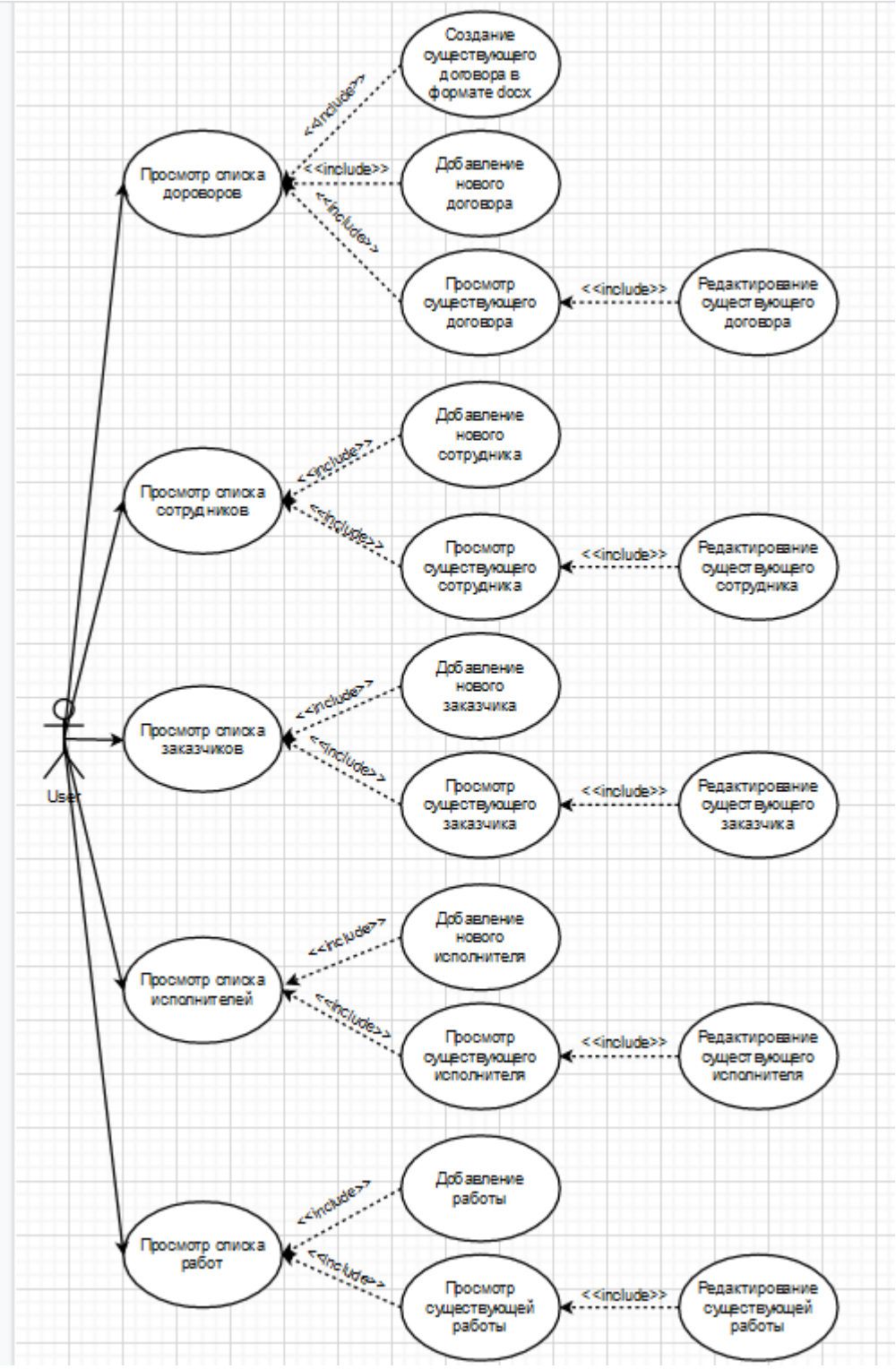



Рисунок 4.20 – Диаграмма вариантов использования

# 5 Заключение

В результате выполнения выпускной квалификационной работы:

– была изучена предметная область документооборота;

– рассмотрены аналоги программ по документообороту и выявлены недостатки;

– определены проблемы;

– поставлена задача для решения проблем;

– были выбраны средства реализации, удовлетворяющие потребностям для решения задачи;

– был создан проект предлагаемого решения, содержащий схему базы данных, требования к системе и описание взаимодействия с базой данных;

– было разработано Web-приложение в соответствии с заявленными требованиями.

Выпускная квалификационная работа выполнена в полном объеме в соответствии с техническим заданием.

## Сокращения, обозначения, термины и определения

1) БД – База данных.

2) СУБД – Система управления базами данных.

3) IDE (Integrated Development Environment) – Интегрированная среда разработки.

4) СЭД (Integrated Development Environment) – система электронного документооборота.

## Список использованных источников

1 Документооборот в организации. [Электронный ресурс]. – URL: https://www.sekretariat.ru/article/210898-organizatsiya-dokumentooborota-v-organizatsii-19-m4 (дата обращения: 04.06.2021)

2 SQLite – Национальная библиотека им. Н. Э. Баумана. [Электронный ресурс]. – URL: https://ru.bmstu.wiki/SQLite (дата обращения: 06.06.2021)

3 Возможности PyCharm. [Электронный ресурс]. – URL: https://www.jetbrains.com/ru-ru/pycharm/features/ (дата обращения: 04.05.2021)

4 Документация Django 1.9. [Электронный ресурс]. URL: https://djbook.ru/rel1.9/ref/databases.html (дата обращения: 04.05.2021)

5 Веб фреймворк Django (Python). [Электронный ресурс]. – URL: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django (дата обращения: 04.05.2021)

6 Django documentation. [Электронный ресурс]. – URL: https://docs.djangoproject.com/en/3.2/ (дата обращения: 04.05.2021)

7 Beginner's Guide to Python. [Электронный ресурс]. – URL: https://wiki.python.org/moin/BeginnersGuide (дата обращения: 05.05.2021)

8 Обзор и сравнения систем электронного документооборота. [Электронный ресурс]. – Режим доступа: http://ecm-obzor.blogspot.com/2015/02/1-1.html (дата обращения: 05.05.2021)

9 Контур.Диадок. [Электронный ресурс]. – URL: https://crmindex.ru/products/kontur_diadoc (дата обращения: 06.05.2021)

10 1С:Документооборот 8 [Электронный ресурс]. – URL: https://v8.1c.ru/doc8/ (дата обращения: 16.05.2021)

11 ELMA365 ECM. [Электронный ресурс]. – URL: https://crmindex.ru/products/elma365 (дата обращения: 06.05.2021)

12 Django. разработка веб-приложений на Python [Текст], Джеф Форсье, Пол Биссекс, Уэссли Чан – Пер.с.англ. — СПб: Символ-плюс, 2009 – 456 с.

13. Официальный сайт базы данных SQLite. [Электронный ресурс]. – URL: https://www.sqlite.org/index.html (дата обращения: 12.06.21).

14. Хабр - крупнейший ресурс для IT-специалистов. Основы правил проектирования базы данных [Электронный ресурс]. – URL: https://habr.com/ru/post/514364 (дата обращения: 03.06.21).

15. Безопасность в Django [Электронный ресурс]. – URL: https://docs.djangoproject.com/en/2.0/topics/security/ (дата обращения: 15.06.21).

16. Python-docx-template’s documentation [Электронный ресурс]. – URL: https://docxtpl.readthedocs.io/en/latest// (дата обращения: 23.06.21).