МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Кафедра «Прикладная математика и информатика»
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки, специальности)

Разработка социальных и экономических информационных систем
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Информационная система учета заказов и взаимодействия с заказчиками (на примере ООО «Атмосфера»)»

Обучающийся С.С. Моргунов
(И.О. Фамилия) (личная подпись)

Руководитель Н.Н. Казаченок
(ученая степень, звание, И.О. Фамилия)

Консультант И. Ю. Усатова
(ученая степень, звание, И.О. Фамилия)

Тольятти 2024

Аннотация

Выпускная квалификационная работа по теме: «Разработка информационной системы учёта заказов и взаимодействия с заказчиками. Внедрение этой системы позволит повысить эффективность взаимодействия с заказчиками и упростить учёт заказов. Система должна позволять обрабатывать, фиксировать, сохранять и отображать информацию в удобном графическом интерфейсе

Цель работы спроектировать, смоделировать и разработать систему, позволяющую вести учёт заказов и эффективно взаимодействовать с заказчиками.

Первая глава описание основной деятельности организации и составлена организационная структура. Разработаны и проанализированы модели бизнес-процессов «Как есть» и «Как должно быть». Провели анализ существующих разработок и поставили задачу на разработку информационной системы.

Во второй главе смоделированы диаграмма прецедентов, диаграмма деятельности в нотации UML, показывают, то, как пользователи будут взаимодействовать с системой. Спроектирована логическая модель базы данных в нотации IDEF1X.

Третья глава затрагивает реализацию всей системы. Для разработки информационной системы использовался высокоуровневый язык программирования Python, самый популярный язык программирования. Выбран веб фреймворк Django, со встроенной админ панелью и генератором шаблонов.

Выпускная квалификационная работа состоит из 52 страниц и содержит 29 рисунков, 3 таблицы, 22 источников.

Abstract

The topic of the graduation work is: "The development of an information system for accounting of orders and interaction with customers". The implementation of this system increases the efficiency of interaction with customers and simplifies the accounting of orders. The system should allow to process, record, save and display information in a convenient graphical interface.

The purpose of the work is to design and develop a system to keep track of orders and help communicate effectively with customers.

The graduation work consists of 52 pages and contains 29 figures, 3 tables, a list of 22 references.

The first chapter describes the main activities of the organization. The organizational structure is drawn up. Business process models "AS IT IS" and "HOW IT SHOULD BE" are developed and analyzed. The task to develop an information system is set.

In the second chapter the diagram of precedents, activity diagram in UML notation are modeled to show how users will interact with the system. The logical model of the database in IDEF1X notation is designed.

The third chapter deals with the implementation of the entire system. The high-level programming language Python, the most popular programming language, was used to develop the information system. Django web framework was chosen, with built-in admin panel and template generator.

As a result of the work, an information system for accounting of orders and interaction with customers was developed. During the development all the set tasks were fulfilled.

Оглавление

Введение

На текущий момент в современном мире имеется тренд на создание информационных систем, которые помогают эффективно взаимодействовать с заказчиками, клиентами. Данные системы способны оптимизировать процессы коммуникации между сотрудниками и заказчиками. На текущий момент на всех рынках огромная конкуренция и вперёд уходят, те организации, которые анализируют и улучшают свои бизнес-процессы путём внедрения современных информационных систем.

Главными условиями, определяющими программный продукт, как информационную систему, является способность фиксировать, запоминать, проводить операции с входящей информацией, предоставлять пользователю возможность работать с системой посредством графического отображение данных в удобном интерфейсе.

Следует разработать информационную систему для организации ООО «Атмосфера».

Объектом исследования является процесс учёта заказов и взаимодействия с заказчиками.

Улучшение качества коммуникации и автоматизации учёта подаваемых заказчиком заявок – предмет исследования.

Основная цель выпускной квалификационной работы, проектирование, моделирование и разработка системы, позволяющей вести учёт заказов и эффективно взаимодействовать с заказчиками.

Для достижения поставленной цели, следует разбить её на следующие задачи:

– анализ деятельности организации ООО «Атмосфера»;
– выявить возможные улучшения бизнес-процессов в существующей структуре;
– обосновать необходимость создания информационной системы;
– составить требования к аппаратно-программному обеспечению;

– обосновать и аргументировать выбор технологий для проектирования, моделирования и разработки системы.

– разработать программный продукт;

– провести испытания информационной системы к концу разработки;

В первой главе затрагивается деятельность организации ООО «Атмосфера», описание её организационной структуры, анализ существующих бизнес-процессов, на основе которых создаются модели в нотации IDEF0 «Как есть» и «Как должно быть», произведен анализ существующих разработок и на основе этих данных и моделей обосновывается необходимость внедрения информационной системы в компанию, ставится задача на разработку проекта.

Вторая глава описывает то, как пользователи будут взаимодействовать с системой, как она будет реагировать на действия пользователей. Здесь создаётся логическая модель базы данных, описывающая структуру сущностей и связей между ними. Также описывается, то, как система будет фиксировать, хранить, предоставлять данные поступающие от пользователей.

В третьей главе будет физическая реализация системы, выбор технологий для разработки программного продукта. В этой главе будет показано то, как поэтапно разрабатывалась система и то, как она будет графически выглядеть для пользователей. По окончанию разработки информационной системы проведём её тестирование, на основе данного тестирования можно выяснить, соответствует ли система разработанным требованиям и насколько качественно была выполнена работа по созданию информационной системы учёта заказов и взаимодействия с заказчиками.

Выпускная квалификационная работа состоит из 52 страниц и содержит 29 рисунков, 3 таблицы, 22 источников.

Глава 1 Анализ предприятия «Атмосфера»

1.1 Технико-экономическая характеристика информационной системы учета заказов и взаимодействия с заказчиками

«Атмосфера» инжиниринговая компания из г. Тольятти. Область её деятельности – внедрение энергоэффективных решений и технологий на предприятиях России.

Командой организации реализованы проекты:

- модернизация автоматизированных тепловых пунктов;
- модернизация автоматизированных систем вентиляции и кондиционирования воздуха производственных корпусов;
- реновация водозаборных скважин;
- модернизация и автоматизация водоочистных сооружений;
- модернизация водооборотных блоков;
- модернизация и автоматизация насосных станций;
- монтаж холодильных станций;
- модернизация компрессорных станций.

Компетенции организации подтверждены отзывами заказчиков, а также реальным результатом работ, которые могут продемонстрировать.

Результат внедрения проектов – снижение эксплуатационных расходов и экономия энергоресурсов на предприятиях, создание комфортного микроклимата.

Это команда профессионалов с опытом выполнения работ по модернизации тепловых узлов, вентиляционных промышленных систем, автоматизации и диспетчеризации энерго-производства. История команды начинается в 2002 году [6].

Специализируется на проектных, строительно-монтажных, пуско-наладочных работах в сфере инженерных промышленных систем. Сфера деятельности – от проекта до запуска в работу и получения результата от модернизации инженерных систем.

Предлагают уникальные автоматизированные системы, внедрение которых даст экономию энергоресурсов, снижение эксплуатационных расходов, получение нужных параметров воздуха в помещениях производства.

Программы для автоматизации и диспетчеризации создаются индивидуально для каждого предприятия, учитывая все особенности и пожелания Заказчика. Гарантируют качество оборудования и экономическую выгоду от реализации проектов для предприятия заказчика.

Ниже представлена организационная структура организации «Атмосфера» на рисунке 1.

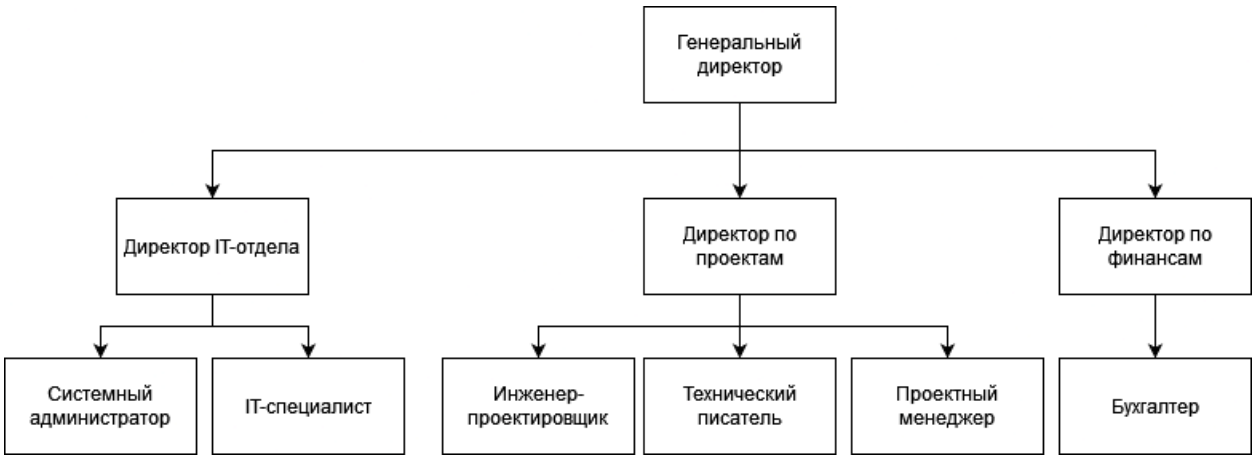


Рисунок 1 – Организационная структура организации

Во главе команды стоит генеральный директор, который руководит всеми бизнес-процессами компании, делегируя важные и ответственные задачи своим подчинённым:

– директор IT-отдела, отвечает за информационную обеспеченность и безопасность организации, внедрение новейших технологий и содержание имеющихся в соответствии с целями и задачами организации;
– системный администратор занимается обслуживанием и поддержкой информационной инфраструктуры. Устанавливает и настраивает лицензированное оборудование и программное обеспечение, следит за состоянием сети, сервера, проводит регулярное обслуживание всех устройств;
– it-специалист внедряет и автоматизирует процессы компании используя высокоуровневые языки программирования, применяя эффективные системы для экономии ресурсов организации;
– директор по проектам играет ключевую роль в построении всех бизнес-процессов, от качества управления зависит положение компании на рынке. Руководит разными проектами полученных от заказчиков, его основная задача принимать и оценивать качество выполненной работы, планировать и качественно распределять ресурсы выделенных на проект;
– инженер-проектировщик разрабатывает и проектирует технические решения, описывая их на чертежах. Выезжает в командировку на исследуемое предприятия, оценивая и анализируя потребности и пожелания заказчика, взаимодействует с другими инженерами согласовывает планируемое техническое решение;
– технический писатель описывает и создаёт документации к проектам, графически оформляет с соблюдением стандартов;
– проектный менеджер координирует действия команды, составляет планы, отвечает за коммуникацию, контролирует и ведёт отчётность по выполненным задачам. Повышает общую дисциплину сотрудников и качество исполнения обязанностей других;

– директор по финансам контролирует весь денежный поток компании, распределение денежных средств на проекты и нужды организации;
– бухгалтер ведёт учёт финансов и документооборота, выплачивает заработную плату сотрудникам и ведёт отчёты по совершённым сделкам организации;

Данная структура позволяет эффективно распределять ресурсы и нагрузку на каждого участника команды, позволяя своевременно сдавать проекты заказчикам. Перед организацией стоит главная задача, эффективно и качественно создавать, выполнять и сдавать проекты улучшающие технологии на предприятии, удовлетворять потребности заказчика.

1.2 Концептуальное моделирование информационной системы учета заказов и взаимодействия с заказчиками

1.2.1 Моделирование бизнес-процессов информационной системы учёта заказов и взаимодействия с заказчиками для постановки задачи автоматизированного варианта решения

Начнём с описания процессов от поступления заказа до конечной его сдачи. В организацию на электронную почту поступает запрос на сотрудничество от предприятия, предприятие кратко описывает в письме, что хочет увидеть и то, как себе представляет конечный результат.

Со стороны «Атмосферы» собираются требования, назначается встреча представителей на территории заказчика, для уточнения нюансов разработки и проектирования. На данную встречу выезжает в командировочную поездку директор по проектам и инженер проектировщик. На месте собирается опыт сотрудников предприятия заказчика, уточняются все требуемые размеры, чертежи, требования, пожелания, заключается договор на оказание услуг.

Представители организации возвращаются обратно в офис, где с полученной информацией формируют план по проектированию и разработке, ставят сроки выполнения и сдачи проекта, назначают ответственных за исполнения обязательств, после чего приступают к выполнению проекта. По окончанию проекта формируется список необходимых деталей, оборудования, запчастей.

Ответственные за проект выезжают, с готовым решением на предприятия заказчика, где внедряют новые технологии, следят за всеми этапами внедрения и сдачи в эксплуатации. Корректируют проект по ходу внедрения, если выявятся проблемы. Когда решение внедрили, составляется акт сдачи работы и подписываются договоры об успешном завершении проекта.

1.2.2 Разработка и анализ модели бизнес-процесса «как есть»

Для лучшего объяснения смоделируем бизнес-процессы в методологии IDEF0. Это очень простой и одновременно наглядный язык описания бизнес-процессов. С помощью этого стандарта возможна передача информации между разработчиками, менеджерами и пользователями. Стандарт очень тщательно разрабатывался, он удобен для проектирования, универсален.

Описание модели начинается с контекстной диаграммы, состоящей из одного блока, название этого блока показывает, какую основную функцию выполняет организация. В блок входят стрелки и выходят, информационные потоки разбиваются на различные виды:

– входящие: вводящие стрелки, которые ставят определенную задачу;
– выходящие: выводящие результат деятельности;
– контроль (сверху вниз): механизмы управления (положения, инструкции);
– механизмы (снизу вверх) – что используется для того, чтобы произвести необходимую работу [19];

На рисунке 2 представлена контекстная диаграмма, с которой начинается моделирование IDEF0.

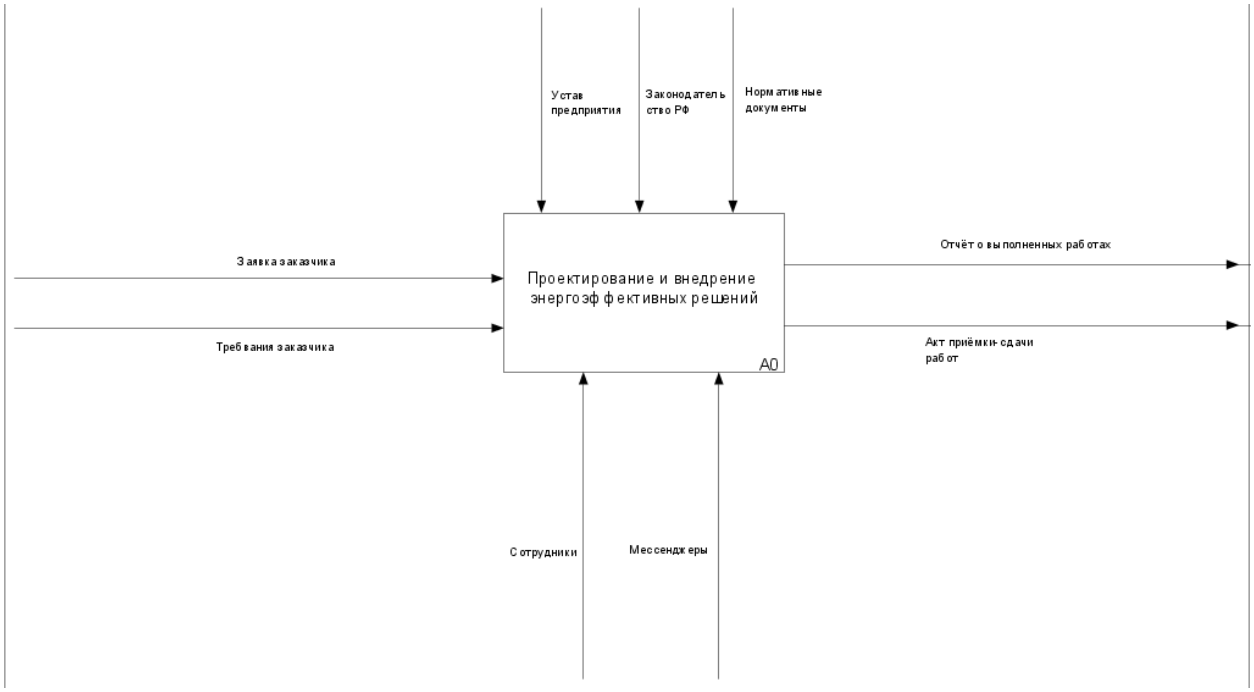


Рисунок 2 – Контекстная диаграмма A0

Основным процессом является проектирование и внедрение энергоэффективных решений. На вход данного процесса для его начала подаётся заявка заказчика и его требования необходимые для выполнения задач.

Организация руководствуется законодательством РФ, нормативными документами и уставом, для создания решений согласно всем ключевым стандартам.

Результат этого процесса, отчёт о проделанной работе и акт приёмки-сдачи работ, подписанными предприятием заказчика и компанией разрабатывающей проект.

Проектирование и внедрение энергоэффективных решений декомпозируется на 4 составляющих блока, представленных на рисунке 3. Отсюда видно, как активно используются мессенджеры для взаимодействия и обсуждений деталей с заказчиком.

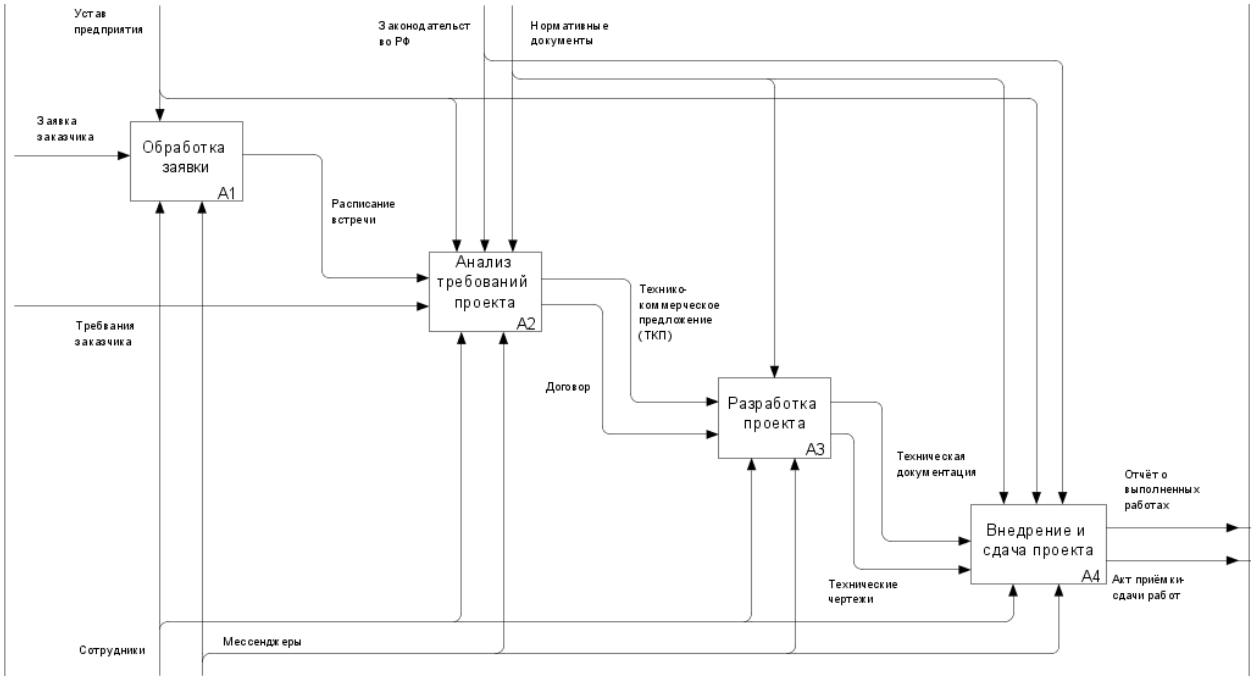


Рисунок 3 – Декомпозиция контекстной диаграммы A0

На данном этапе мы видим все основные процессы, происходящие в компании. Блок A1 процесса обработки заявки заказчика декомпозируем передав на вход заявку заказчика. Декомпозиция показана на рисунке 4.

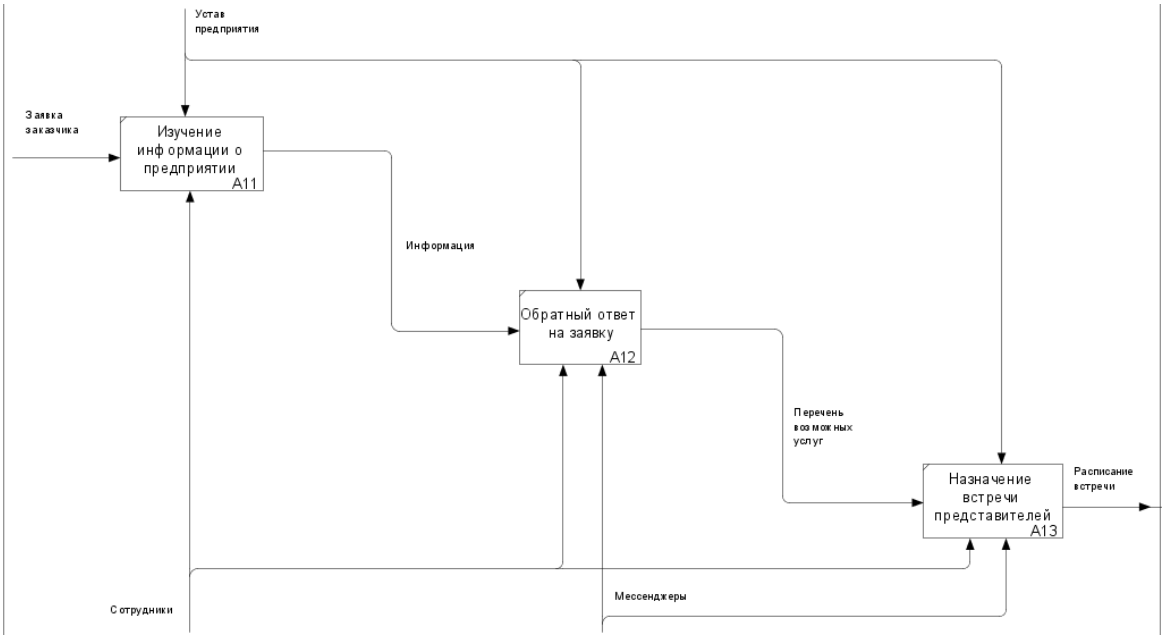


Рисунок 4 – Декомпозиция процесса обработки заявки A1

По итогу этого процесса назначается встреча представителей на выходе и осуществляется выезд специалиста на территорию заказчика.

Далее на рисунке 5 представлена декомпозиция анализа требований проекта.

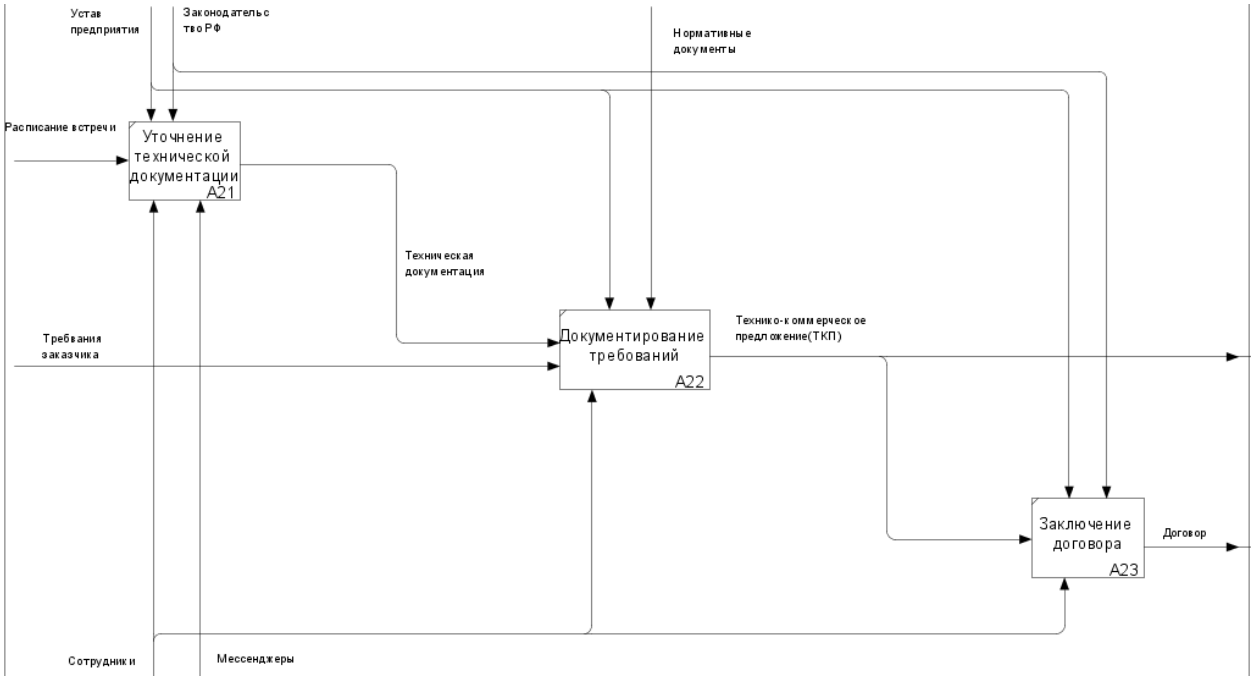


Рисунок 5 – Декомпозиция процесса анализа требований проекта A2

На этой декомпозиции описано как сотрудники собирают информацию о предприятии, находят существующие технические документации, после чего документируется требования и заключается договор на основании их.

1.2.3 Обоснование необходимости информационной системы для решения задач

Исходя из представленной модели IDEF0, можно сделать вывод о том, что можно улучшить средства коммуникации между заказчиком и исполнителем. На данный момент в организации используются мессенджеры для общения, обсуждения и уточнения информации у заказчика. У данного способа связи есть свои недостатки:

– зависимость от работы серверов мессенджера;

– мессенджер могут заблокировать;

– при потере доступа к своему профилю, теряются все переписки и данные;

– отправка файлов с ограничением по размеру;

– возможные угрозы безопасности и.т.д.;

В этом небольшом перечисление описаны самые очевидные проблемы, которые могут возникнуть при применении их в работе.

Во избежание данных недостатков можно разработать информационную систему, которая позволит пользователям делиться и отправлять техническую информацию без ограничений по размеру файлов и позволит не зависеть от серверов мессенджера.

Данная информационная система при этом должна быть надёжной и доступной, с удобным пользовательским интерфейсом.

Такая система позволит безболезненно переобучить сотрудников, так как по функционалу и интерфейсу, будет похожа на мессенджеры. Оптимизирует множество процессов и уменьшит трудозатраты на выполнения задач.

1.3 Анализ существующих разработок

В информационной сфере существует системы с похожим функционалом, их называют CRM (Customer Relationship Management) — это система управления взаимоотношениями с клиентами, которая помогает улучшить взаимодействие между компанией и клиентами [9].

Данные программные решения обладают большим, даже излишним функционалом, который будет избыточным для организации, что в следствии отразится на длительность обучения сотрудников в худшую сторону.

Также все готовые решения оплачивается в форме ежемесячной подписки, где потребуются большие средства, чтобы поддерживать такие системы.

Вот часть возможных решений существующих на рынке:

– битрикс24, социальный интранет. Содержит микроблоги, задачи, файлохранилище (с контролем версий), календарь, фотогалереи, мессенджер, экстранет, CRM, маркетинг, бизнес-процессы, учет рабочего времени, профайлы, отчеты;

– amoCRM, простая SaaS CRM система. Позволяет просто вести базу контактов и учет сделок (в привязке к контактам). Контакты и сделки можно помечать тэгами. На основании суммы и статуса сделок формируется воронка продаж. Есть API. Возможность добавлять поля в карточки сделок;

– evaCRM, CRM для малого и среднего бизнеса. Обеспечивает работу с клиентами, от приёма заявок через мессенджеры до выставления документов и товарного учёта. Автоматически захватывает потенциальных клиентов из коммуникационных каналов: Instagram, сайт, почта, WhatsApp, Telegram, телефония. Канбан продаж, карточки сделок, приём оплат, интеграция с банками, встроенный мультичат: возможность отправлять сообщения в разные мессенджеры прямо из карточки клиента;

– мегаплан, система для управления бизнесом в малой или средней компании любого профиля. Помогает повышать продажи, управлять сотрудниками и работать удаленно. В комплекте: CRM, выставление счетов, контроль сделок, менеджер задач, файловый сервер, внутренняя почта, форум, модуль для работы с персоналом;

Сравним описанные выше системы в таблице 1.

Таблица 1 – Анализ программных решений существующих разработок

Критерий	Битрикс24	amoCRM	EvaCRM	Мегаплан
Вид системы	облако/коробка	облако/коробка	облако/коробка	облако/коробка
Техническая поддержка	+	+	+	-
Мобильная версия	-	+	-	+
Гибкость настроек	-	+	-	+
Ежемесячная подписка	+	+	+	+

Исходя из небольшого сравнения, видно, что все CRM используют варианты, как облачные, так и коробочные. Также есть у многих мобильная версия, данный функционал излишен для нашего случая. Есть ежемесячные затраты на подписку и поддержку этих систем с учётом функциональных требований для организации.

1.4 Разработка модели бизнес-процесса «как должно быть»

На основе выявленных недостатков текущей реализации проектов ООО «Атмосфера», описание возможных функций нашей информационной системы и анализе существующих разработок, мы можем смоделировать концептуальную модель IDEF0 «как должно быть» [20].

На рисунке 6 представлена контекстная диаграмма «как должно быть» процесса проектирования и внедрения энергоэффективных решений.

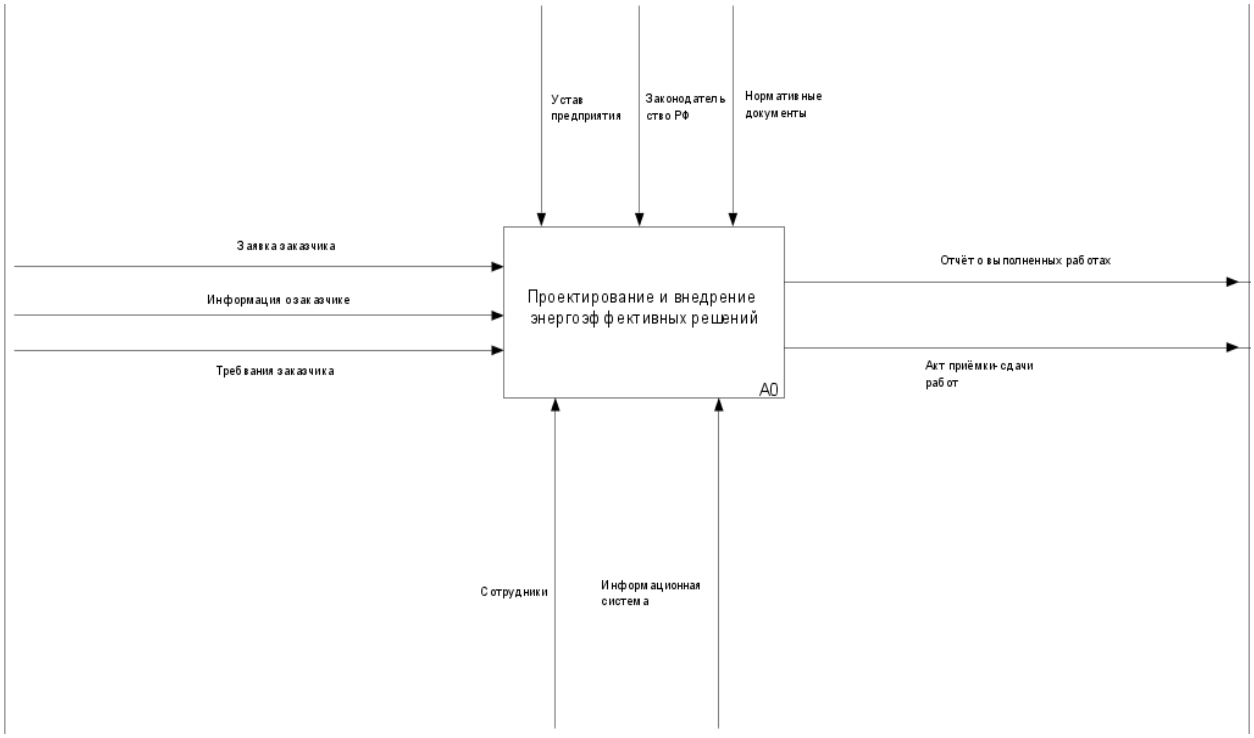


Рисунок 6 – Контекстная диаграмма A0

Относительно контекстной диаграммы «как есть», всё информационное взаимодействие с заказчиками перешло в информационную систему, также на вход теперь подаётся информация о заказчике, так как заказчик предоставил её системе.

Все улучшения после внедрения информационной системы будут оптимизировать время, потраченное на выполнение задач, то есть все основные бизнес-процессы, связанные с коммуникацией между сотрудниками и заказчиками.

Далее на рисунке 7 декомпозируем данный основной процесс.

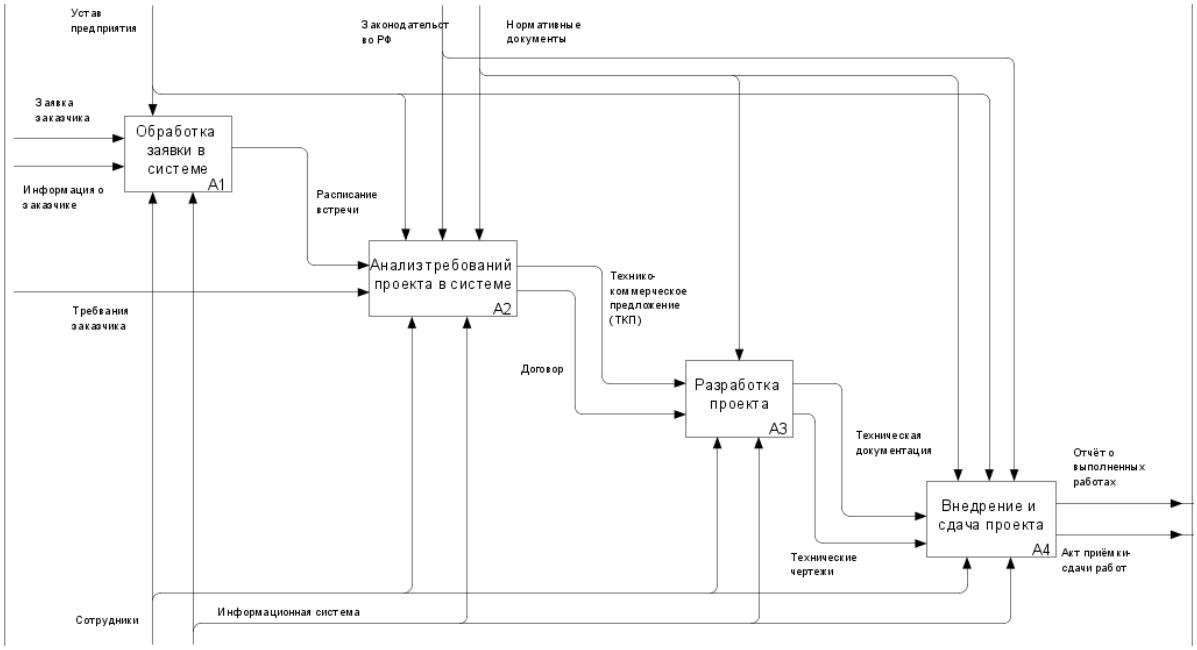


Рисунок 7 – Декомпозиция контекстной диаграммы A0

Мы видим, что часть процессов, теперь выполняются в системе. На рисунке 8 изображена декомпозиция процесса, где обрабатывается заявка.

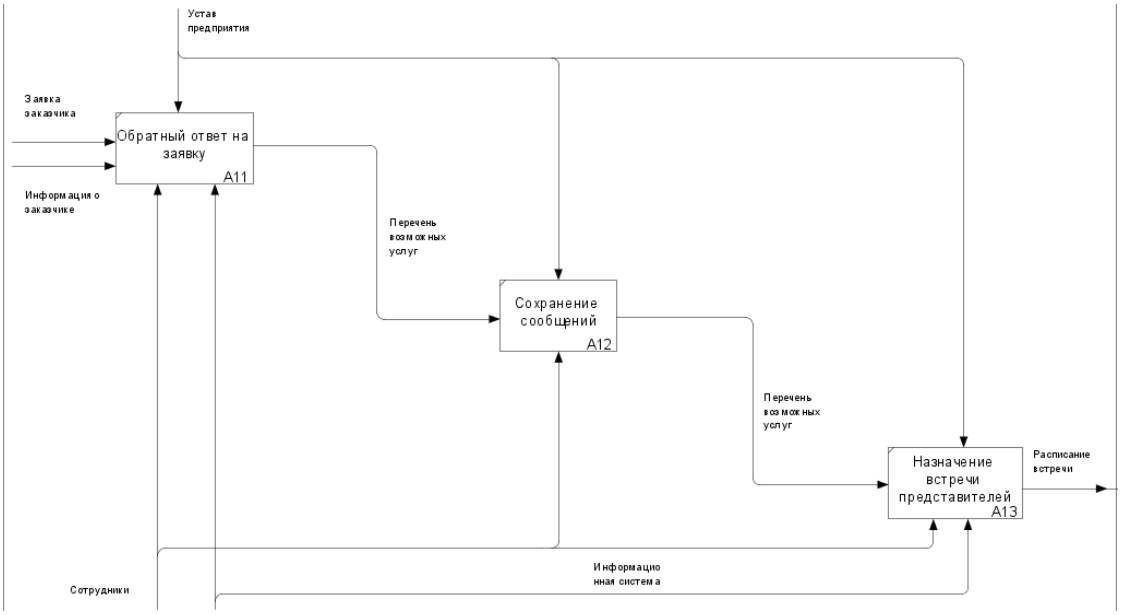


Рисунок 8 – Декомпозиция процесса A1 обработки заявки в системе

Добавился новый процесс сохранение сообщений, который выполняется только информационной системой. Принимая сообщения пользователей, сотрудник сохраняет их в базу данных.

Ниже на рисунке 9 представлена декомпозиция процесса А2 анализа требований.

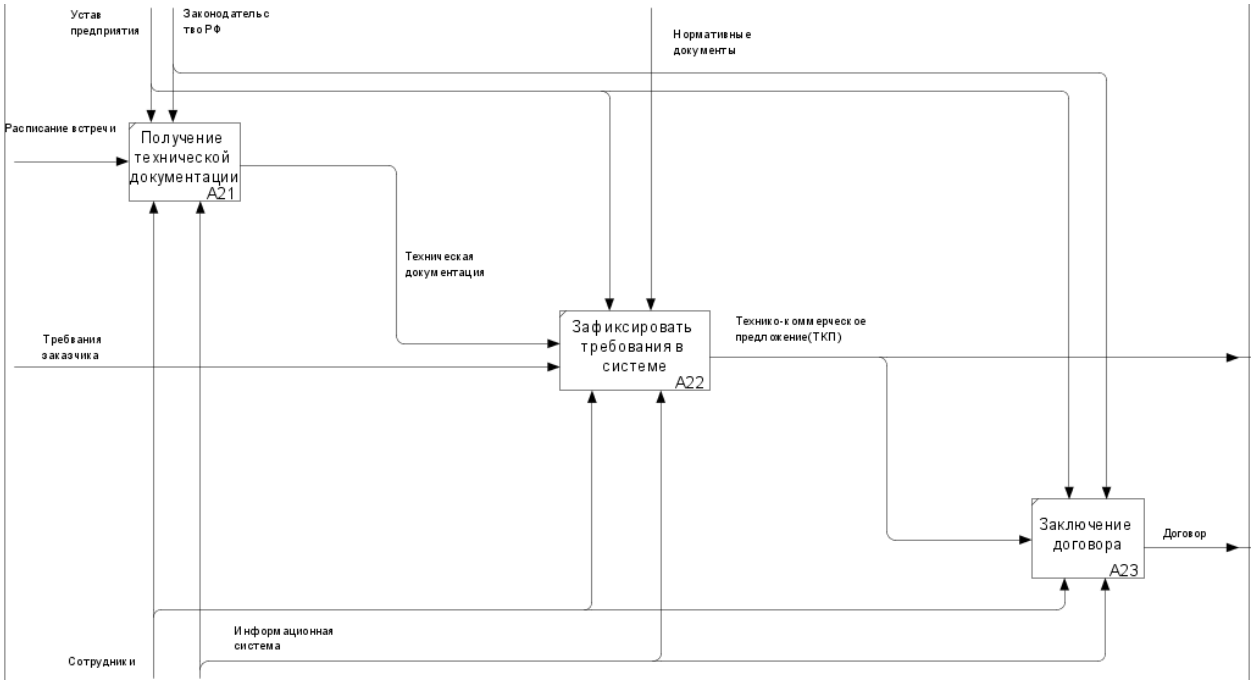


Рисунок 9 – Декомпозиция процесса А2 анализа требований проекта в системе

Все полученные сведения, вся техническая документация и требования также фиксируется в системе.

При анализе можно заметить, что главным изменением является добавление единого информационного ресурса, где есть возможность фиксировать техническую документацию и все возможные документы, полученные в ходе работы над проектами. На диаграммах можно увидеть все усовершенствования процессов после внедрения информационной системы.

1.5 Постановка задачи на разработку ИС взаимодействия с заказчиками

Основной целью разработки и создания нового решения, уменьшить влияние мессенджеров в работе и избежать проблемы связанные с данным способом связи. Улучшить эффективность работы менеджеров и инженеров проектировщиков.

Для проектировщиков улучшение будет связано с тем, что все нужные технические файлы будут хранится в одном месте распределённых по директориям и папкам. Уменьшить трудозатраты на разработку [20].

Информационная система должна будет обладать определённым набором функций:

– запись и сохранение файлов под определённый каталог базы знаний;

– загрузка файлов и документов на сервер;

– заказчик может написать менеджеру или инженеру проектировщику и.т.д.;

– исполнитель может написать заказчику;

– данные сообщений сохраняются в базе данных;

– сохранение дополнительной информации о предприятии;

Необходимо создать одновременный доступ к базе данных информационной системы для всех сотрудников организации ООО «Атмосфера» с разграничением прав доступа, как для пользователей самой организации, так и пользователей предприятий.

Выводы по главе 1

В первой главе была предоставлена организационная структура организации, на основе которой была описана вся деятельность организации в сфере инжиниринговых услуг и проанализированы основные бизнес-процессы.

Смоделирована и создана концептуальная модель «как есть», благодаря этой диаграмме были определены какие улучшения можно внести, чтобы улучшить эффективность и качество взаимодействия с заказчиками.

Проведён анализ существующих решений. Сравнили варианты и построили таблицу, выявили текущую стоимость ежемесячной подписки на использовании облачных или коробочных CRM. Провели постановку задачи на разработку проекта создания информационной системы взаимодействия с заказчиками.

На основе поставленных задачи и возможных улучшений была смоделирована IDEF диаграмма «как должно быть», на которой изобразили предлагаемые усовершенствования.

По окончанию данной главы можно сделать вывод о том, что организация ООО «Атмосфера» нуждается в совершенствовании и улучшении всех бизнес-процессов и внедрении информационной системы взаимодействия с заказчиками.

Глава 2 Логическое проектирование информационной системы

2.1 Выбор технологии логического моделирования информационной системы

При логическом моделировании используется такие методологии, как технологии объектно-ориентированного анализа и проектирования, основанных на стандартных нотациях языка UML, методология IDEF1X создана для моделирования сложных систем, позволяет отображать и анализировать модели деятельности широкого спектра сложных систем в различных разрезах. Очень часто IDEF1X используют для моделирования реляционных баз данных, также этот стандарт входит в семейство методологий IDEF [4]. На таблице 2 представлено небольшое сравнение двух методов проектирования.

Таблица 2 – Сравнение методологий логического моделирования

Критерий	UML	IDEF1X
Структурность	-	+
ООП	+	-
Область применения	больше подходит для проектирования ПО	больше подходит для проектирования БД

Из таблицы можно сделать выводы, что нотации UML и IDEF1X две совершенно разных и непохожих друг для друга нотаций. UML должна использоваться при создание сложных объектно-ориентированных систем, а IDEF1X так как использует концепцию «сущность-связь», лучше всего подойдёт для моделирования структур БД [2].

На основании данных утверждений выбор склоняется в сторону применения нотации UML в нашем случае.

2.2 Логическая модель информационной системы и ее описание

Диаграмма вариантов использования или как её называют по-другому, диаграмма прецедентов – это диаграмма в нотации UML, описывает функционал системы и разделяет на группы пользователей, которые могут взаимодействовать с этими функциями [11].

На рисунке 10 представлена диаграмма вариантов использования информационной системы взаимодействия с заказчиками.

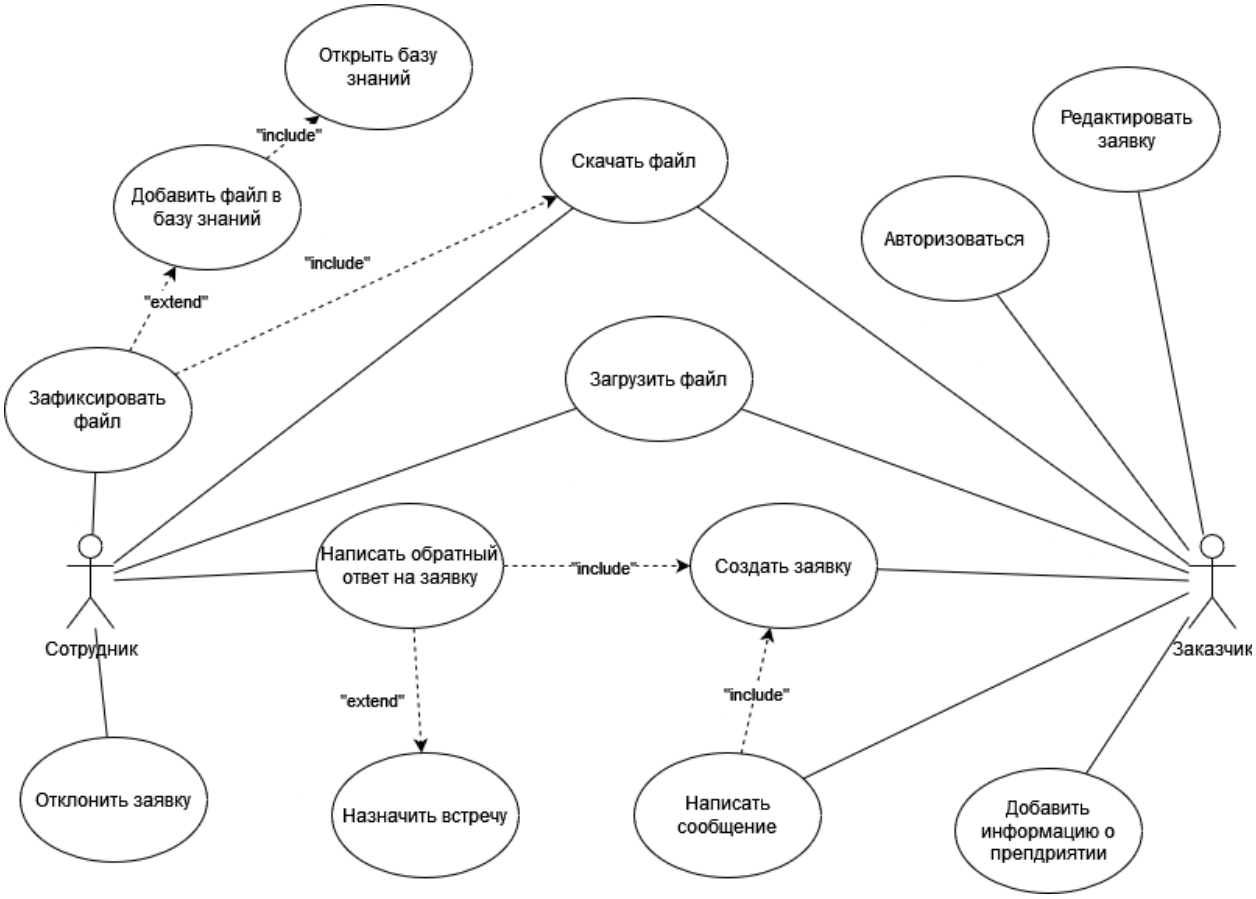


Рисунок 10 – Диаграмма вариантов использования

В этой схеме у нас есть два актёра, человечка, они обозначают сущности, которые могут пользоваться системой. Сущностями могут быть люди, технические устройства или даже другие системы. В нашем случае разбивается на сотрудника организации и заказчика [5].

Эллипсом называют прецедент – это функция системы, действие, которое может осуществить актёр. Прецеденты системы:

– написать сообщение;

– создать заявку;

– загрузить файл;

– добавить информацию о предприятии;

– авторизоваться;

– редактировать заявку;

– скачать файл;

– назначить встречу;

– написать обратный ответ на заявку;

– отклонить заявку;

– зафиксировать файл (сохранить файл в базе данных);

– добавить файл в базу знаний (структурированная страница со всеми файлами проекта);

– открыть базу знаний;

Прецеденты и сущности соединяются линиями. Сплошной линией (ассоциация) обозначают отношение между актёром и прецедентом, то с чем может взаимодействовать каждый присоединённый актёр [1].

Линия с пунктиром и надписью «include» является отношением включения, где для выполнения прецедента требуется обязательное выполнение вложенного другого. Отношение с надписью «extend» расширения, даёт возможность выполнить помимо основного прецедента, связанный вариант использования [11].

Диаграмма деятельности — под деятельностью понимается последовательное и параллельное выполнение действий, соединённых между собой потоками, которые идут от выходов одного узла к входам другого [21].

Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов [1].

Основные элементы диаграммы:

Закругленный прямоугольник — действия, которые выполняют участники системы, соединяются сплошной стрелкой с другими элементами.

Ромбы — решения. Узел условия предназначен для определения различных вариантов дальнейшего развития сценария. В точку ветвления входит ровно один переход, а выходит — два или более.

Чёрный круг — начало процесса. Начальный узел деятельности является узлом управления, в котором начинается поток.

Чёрный круг с обводкой — окончание процесса. Конечный узел деятельности является узлом управления, который останавливает все потоки данной диаграммы деятельности. На диаграмме может быть более одного конечного узла [5].

Для нашей системы очень важно обработать все входящие сообщения и зафиксировать их в базе данных. Также система должна проверять данные на валидность и, если данные некорректны, то вывести ошибку или уведомление о неправильном вводе.

Ниже, на рисунке 11 изображена диаграмма деятельности.

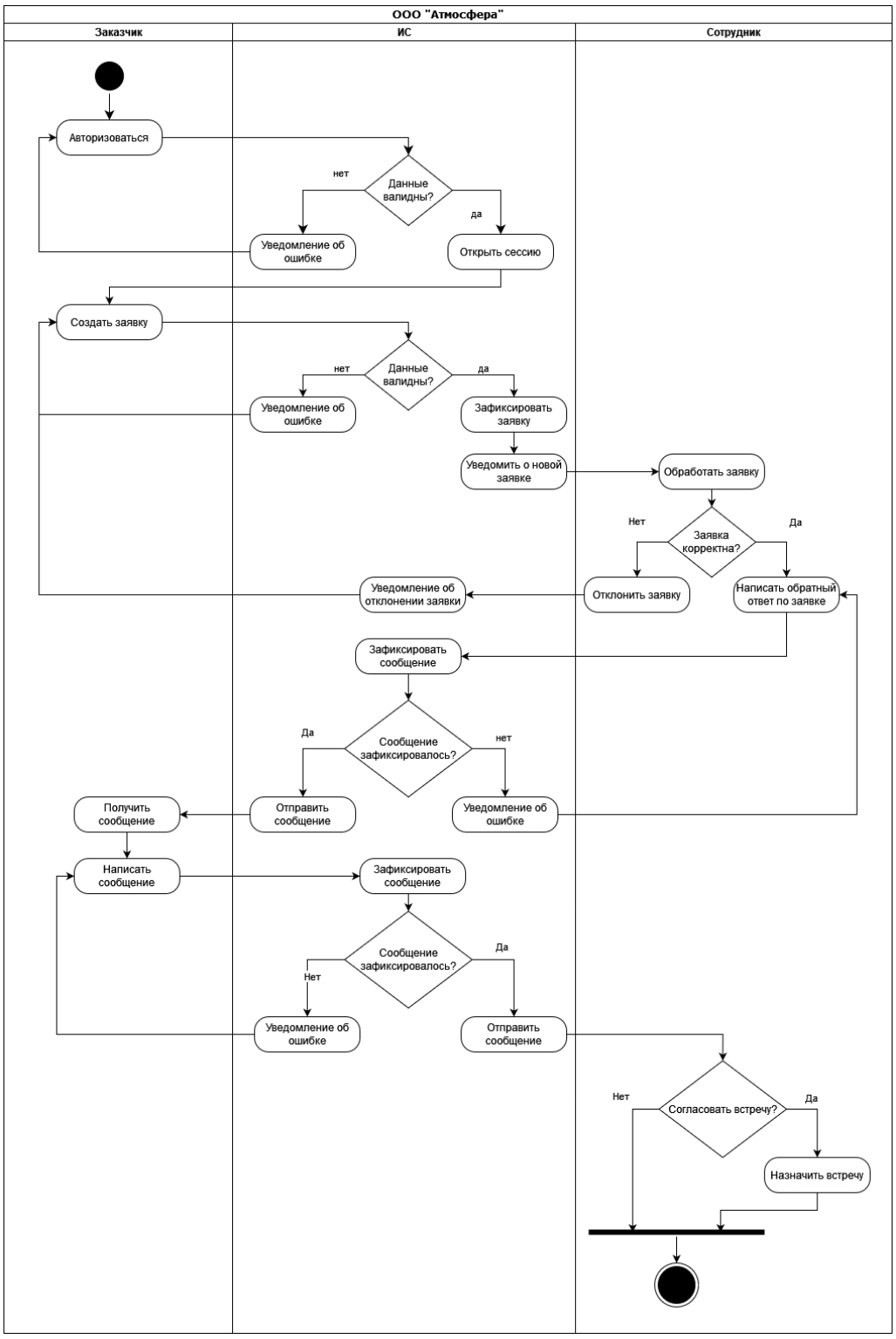


Рисунок 11 – Диаграмма деятельности

Диаграмма начинается с деятельности со стороны заказчика, где он авторизируется в системе, система проверяет данные и решает пускать пользователя дальше или уведомить его об ошибке.

Заказчик создаёт заявку, и данная заявка приходит сотруднику уведомлением от системы. В свою очередь сотрудник обрабатывает заявку и если заявка корректна, то пишет обратный ответ на данную заявку, после чего сообщение получает отправитель, где начинается диалог с заказчиком. Сообщение, отправленное пользователями фиксируются в системе. Итогом переговоров является назначение встречи между компаниями или не назначением её.

2.3 Проектирование базы данных информационной системы взаимодействия с заказчиками

Логическая модель данных – это модель, которая показывает существующие сущности в системе и их взаимосвязи [3].

При проектировании важно чётко продумать какими атрибутами будет обладать каждая сущность. Данная модель в дальнейшем поможет при создании базы данных для информационной системы.

Для разработки логической модели данных, мы используем методологию IDEF1X. Данная технология работает по принципу «сущность-связь».

Прежде чем перейти к созданию, можно заранее определить сущности для нашей модели:

- сотрудник,
- заказчик,
- заявка,
- предприятие,
- сообщение.

На рисунке 12 представлена логическая модель данных.

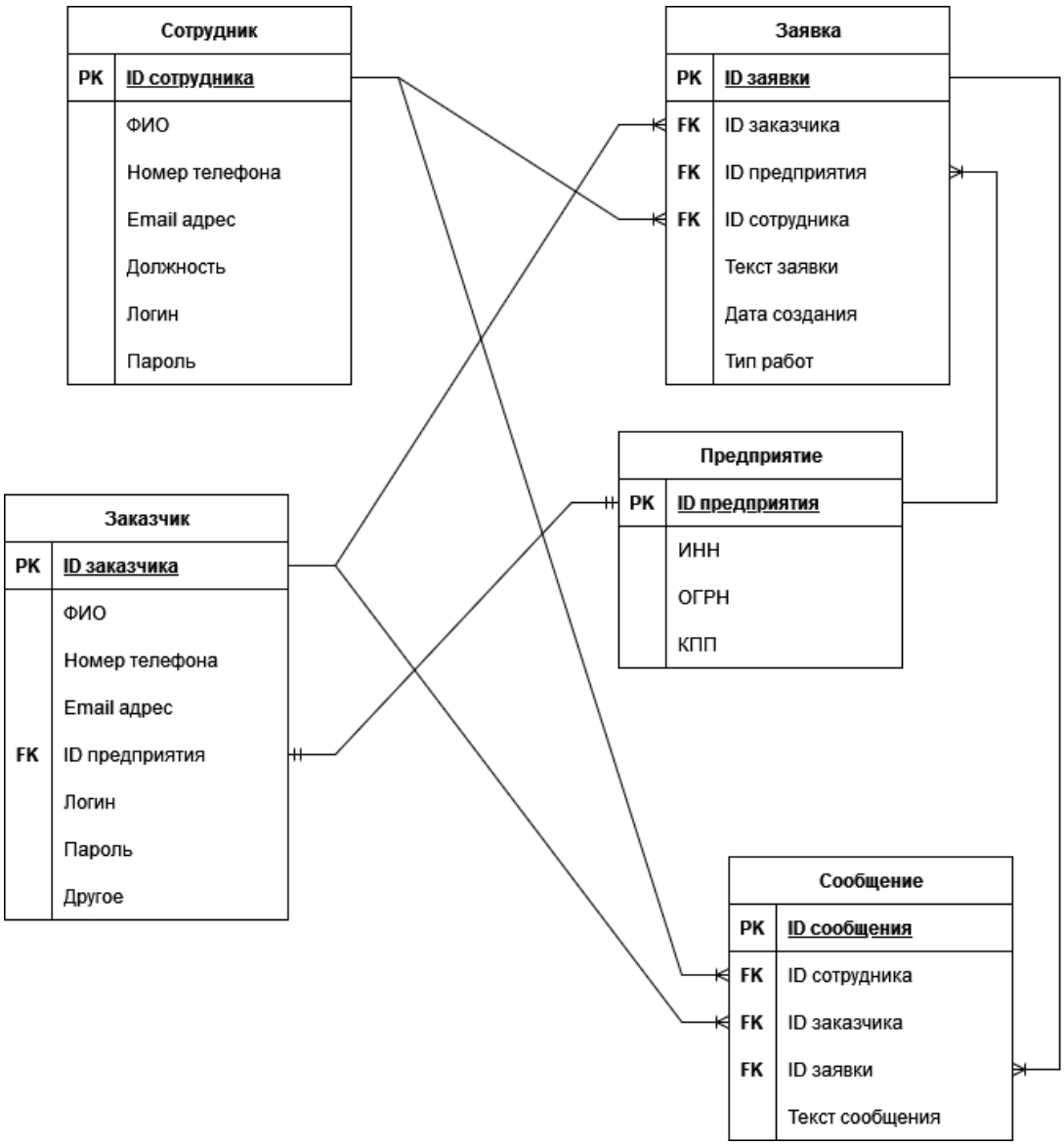


Рисунок 12 – Логическая модель базы данных

На показанной логической модели, видно, какие данные будут содержаться в системе. PK – primary key (первичный ключ), FK – foreign key (внешний ключ).

Сотрудник и заявка соединена связью один ко многим, то есть на одного сотрудника может приходиться много заявок, аналогично связаны заказчик и заявка, заказчик может создать сколько угодно заявок. Данные связи показывают, какими атрибутами связаны между собой сущности. Соединяются сущности от первичного ключа к внешнему и наоборот [3].

У сущности заказчика и предприятия связь один к одному, показывающая, что у одного заказчика, может быть только одна запись предприятия.

2.4 Требования к аппаратно-программному обеспечению информационной системы

Информационная система взаимодействия с заказчиками будет базироваться на клиент-серверной архитектуре. Вся логика, процессы и обработка данных будет держаться на сервере, клиент получает только необходимую информацию для отображения. Для создания и поддержки системы потребуется развёртывания сервера.

Системные требования сервера сильно зависят от того, какое максимальное количество пользователей может одновременно работать с системой. При развёртывании сервера следует использовать взаимно заменяемые компоненты для возможного будущего улучшения надёжности и безотказности работы системы вследствие дальнейшего масштабирования.

Системные требования для сервера с 32 битной архитектурой: операционная система семейства Linux, 8-ми ядерный процессор с тактовой частотой не меньше 2,5 ГГц. Свободное дисковое пространство размером в 1 Тб, тип накопителя SATA. Пропускная способность локальной сети 100 Мбит/сек. Оперативная память не меньше 8 Гб.

Системные требования для сервера с x86-64 архитектурой: операционная система семейства Linux, 8-ми ядерный процессор с тактовой частотой не меньше 2,5 ГГц. Свободное дисковое пространство размером в 1 Тб, тип накопителя SATA. Пропускная способность локальной сети 100 Мбит/сек. Оперативная память не меньше 16 Гб.

Также сервер должен быть оснащён портами для сетевого подключения, портами для подключения периферии и оснащён USB-портами для подключения внешних накопителей. На сервере должна быть реализована возможность удалённого подключения с разделением права доступа.

Для минимальных системных требований клиента достаточно наличие компьютера с стабильным интернет-соединением и с компонентами способные обеспечить быструю работоспособность браузера.

Выводы по главе 2

В этой главе мы создали логическую модель информационной системы в нотации UML, описали систему в таких диаграммах как: диаграмма вариантов использования и диаграмма деятельности.

Также была разработана логическая модель база данных в нотации IDEF1X, определили сущности, с которыми будет работать система и, то какие данные она будет хранить на сервере.

Были выявлены требования к аппаратно-программному обеспечению информационной системы.

С помощью концептуального и логического моделирования, видно, как система будет взаимодействовать с пользователями. Удалось достаточно подробно описать информационную систему всеми средствами моделирования.

Глава 3 Физическое проектирование информационной системы

3.1 Выбор технологии разработки программного обеспечения информационной системы

Выбор технологии должен соответствовать существующим требованиям и критериям, применяемых к информационной системе. Чтобы можно было удобно администрировать и управлять программным обеспечением, а также технологии, которые позволяют с лёгкостью развернуть систему на сервере.

Для разработки данной системы выбор пал на высокоуровневый язык программирования Python. На текущий момент данный язык находиться на первом месте по популярности. Из-за огромной аудитории существует множество обучающих и технических материалов, с помощью которых легко вникнуть в нюансы Python, что позволяет тратить меньше времени на разработку.

Популярными фреймворками веб-разработки этого языка являются Django, Flask и FastAPI [7].

Django - это фреймворк с открытым исходным кодом, который позволяет разработчикам создавать веб-приложения практически любого уровня. Он входит в число лучших фреймворков Python и заслуженно пользуется популярностью [13].

Ключевые особенности Django:

– наличие собственного ORM (объектно-реляционное отображение»);

– встроенный административный интерфейс;

– генерация шаблонов;

– библиотека работы с формами;

– система кэширования;

– система авторизации и аутентификации;

У фреймворка Flask небольшой размер исходной кодовой базы, поэтому его называют микро фреймворком. Но это не значит, что у него меньше возможностей, чем у того же Django. По умолчанию он включает в себя только обработчик запросов и генерация шаблонов, а простейшее приложение на Flask может состоять всего из нескольких строк. Разработчики этого фреймворка осознанно хотели сохранить ядро простым, но расширяемым.

Ключевые особенности Flask:

– встроенный сервер разработки и отладчик;

– встроенная поддержка модульного тестирования;

– совместимость с WSGI 1.0;

– множество расширений, предоставляемых сообществом.

Tornado – это расширяемый асинхронный веб-сервер и фреймворк. То есть при правильной настройке он может работать более с 10000 одновременных соединений. Это делает его отличным инструментом для создания приложений, требующих огромной производительности при работе с операциями ввода, вывода и поддержки огромного количества одновременных соединений.

Ключевые особенности Tornado:

– асинхронный режим работы;

– использование веб-сокетов;

– поддержка схем аутентификации и авторизации сторонних производителей.

Исходя из описаний трёх программных платформ, больше всего подходит для информационной системы взаимодействия с заказчиками Django Фреймворк, так как в нём содержится функционал администрирования и управления правами доступа к системе, благодаря которой, получиться легко подключить систему авторизации и регистрации пользователей, а также возможность упрощённо работать с базой данных с помощью моделей [18].

3.2 Разработка программного обеспечения информационной системы

При установке Django вместе с основными компонентами фреймворка, устанавливается сервер баз данных SQ Lite, особенностью которого является простота внедрения в среду разработки, быстродействие, высокая надёжность и имеющий все функции, требующиеся для работы с базами данных [17].

Для создания сущностей упомянутых ранее, создадим для них модели описывающие их [14].

На рисунке 13 представлена модель заявки.

```
class Request(models.Model):
    creator = models.ForeignKey(User, on_delete=models.CASCADE)
    text = models.TextField()
    created_at = models.DateTimeField(auto_now_add=True)
    WORK_TYPES = [
        ('Модернизация тепловых пунктов', 'Модернизация тепловых пунктов'),
        ('Монтаж центральных кондиционеров', 'Монтаж центральных кондиционеров'),
        ('Обслуживание и ремонт энергетического оборудования', 'Обслуживание и ремонт энергетического оборудования'),
    ]
    work_type = models.CharField(max_length=100, choices=WORK_TYPES)

    def __str__(self):
        return f'ID: {self.id}, Creator: {self.creator}, Created at: {self.created_at}'
```

Рисунок 13 – Модель сущности заявки

Атрибут тип работ указанный в сущности заявки, разделяется на список из пунктов:

– модернизация тепловых пунктов;

– монтаж центральных кондиционеров;

– обслуживание и ремонт энергетического оборудования.

Для того, чтобы данные зафиксировались в системе и базе данных, нужно создать функцию представления обрабатывающую запрос пользователя, позволяющая перейти на страницу шаблона, просмотра и создания заявки. Ниже на рисунке 14 показано представление для модели заявки [22].

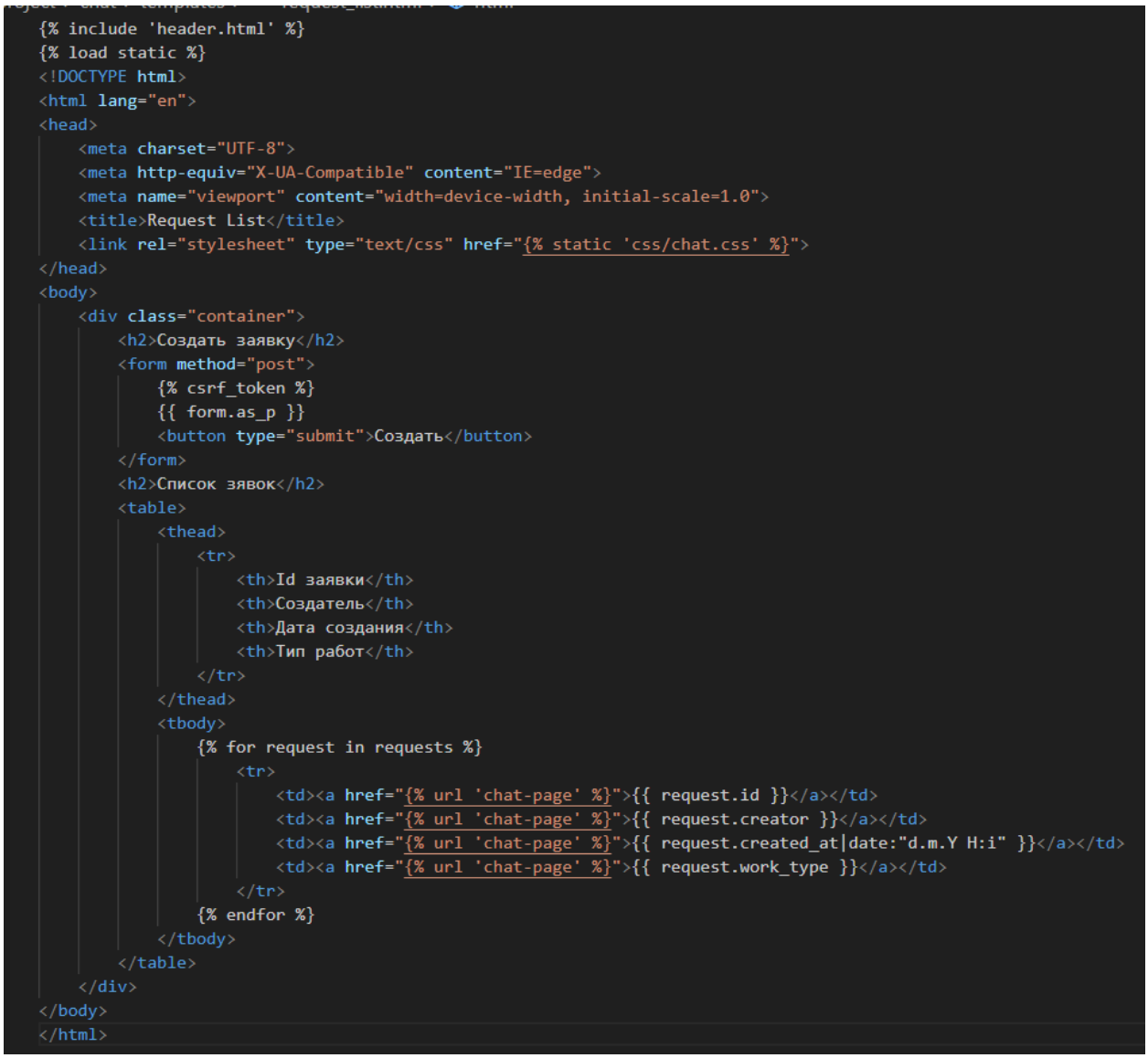
```python
def request_list(request):
    if request.method == 'POST':
        form = RequestForm(request.POST)
        if form.is_valid():
            request_obj = form.save(commit=False)
            request_obj.creator = request.user
            request_obj.save()
            return redirect('request')  # Перенаправляем на страницу с текущими заявками
    else:
        form = RequestForm()
    if request.user.has_perm('chat.view_request'):
        requests = Request.objects.all()
    else:
        requests = Request.objects.filter(creator=request.user)
    return render(request, 'request_list.html', {'requests': requests, 'form': form})
```

Рисунок 14 – Представление обработки запроса создания, просмотра заявки

С помощью данной функции проверяется какой тип запроса приходит к пользователю, если запрос Post, то это означает, что пользователь заполнил форму по созданию заявки нажал на кнопку, отправляющая Post запрос на сервер, после чего зафиксируются и сохранятся данные ведённые в форму в базу данных. Если поступит обычный запрос Get, сервер просто выведет пользователю все существующие заявки в базе данных.

Также имеется условие, что заказчик на странице заявок сможет увидеть список только своих предложений, а сотрудникам организации будет выведены все существующие заявки заказчиком и не будет доступно поле с созданием заявок. Неавторизированный пользователь не сможет попасть на страницу заявок и будет перенаправлен на страницу авторизации.

На рисунке 15 представлен код шаблона html.



```
{% include 'header.html' %}
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Request List</title>
    <link rel="stylesheet" type="text/css" href="{% static 'css/chat.css' %}">
</head>
<body>
    <div class="container">
        <h2>Создать заявку</h2>
        <form method="post">
            {% csrf_token %}
            {{ form.as_p }}
            <button type="submit">Создать</button>
        </form>
        <h2>Список заявок</h2>
        <table>
            <thead>
                <tr>
                    <th>Id заявки</th>
                    <th>Создатель</th>
                    <th>Дата создания</th>
                    <th>Тип работ</th>
                </tr>
            </thead>
            <tbody>
                {% for request in requests %}
                    <tr>
                        <td><a href="{% url 'chat-page' %}">{{ request.id }}</a></td>
                        <td><a href="{% url 'chat-page' %}">{{ request.creator }}</a></td>
                        <td><a href="{% url 'chat-page' %}">{{ request.created_at|date:"d.m.Y H:i" }}</a></td>
                        <td><a href="{% url 'chat-page' %}">{{ request.work_type }}</a></td>
                    </tr>
                {% endfor %}
            </tbody>
        </table>
    </div>
</body>
</html>
```

Рисунок 15 - Код html шаблона страницы заявки

Все данные получаемые в представлении попадают в шаблон, где через цикл выводятся в список заявок. В Django применяется определённый синтаксис с фигурными скобками, показывая, что мы используем переменные, подгруженные из представления [16].

Ниже показан скриншот графической части html шаблона для заказчика на рисунке 16.

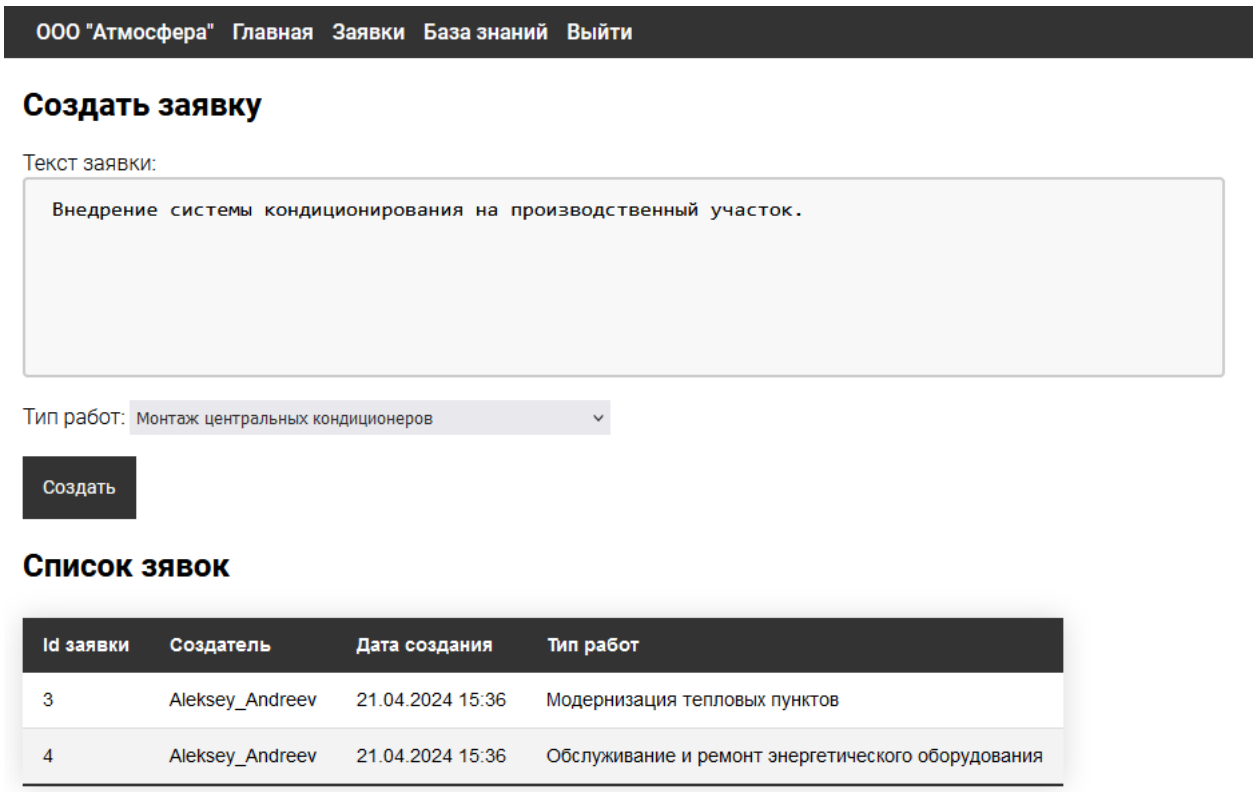


Рисунок 16 - Графическое отображение страницы заявок

На данной странице в минималистичном стиле выведена форма с полями текста заявки и список типов работ, которую можно заполнить и по кнопке создать заявку, которая отобразиться в списке заявок, также можно выбрать заявку нажать на строку и перейти к созданному чату на отдельной странице для созданной заявки.

Для страницы чата и сообщений требуется создать модель, чтобы сохранять историю диалога в базе данных [12]. На рисунке 17 показан фрагмент кода инициализации сообщений.

```python
class Message(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    content = models.TextField()
    timestamp = models.DateTimeField(auto_now_add=True)
    id_request = models.ForeignKey(Request, on_delete=models.CASCADE)

    def __str__(self):
        return f'{self.user.username} - {self.timestamp}'
```

Рисунок 17 – Модель сущности сообщений

Данная созданная модель фиксирует пользователя написавшего это сообщение, сохраняет содержание этого сообщения, которое выполнено в виде текстового поля, дата и время создания, фиксируется в момент отправки сообщения и id заявки, позволяющая определять к какой именно заявке относится диалог [8].

На рисунке 18 представлено два представления для страницы чата между пользователями.

```python
def chatPage(request, *args, **kwargs):
    if not request.user.is_authenticated:
        return redirect("login")
    context = {}
    messages = Message.objects.all()
    if request.method == 'POST':
        file = FileForm(request.POST, request.FILES)
        if file.is_valid():
            handle_uploaded_file(request.FILES['file'])
            return render(request, "chat/chatPage.html", {'messages': messages})
    else:
        file = FileForm()
    return render(request, "chat/chatPage.html", {'messages': messages})

def save_message(request):
    if request.method == 'POST':
        data = json.loads(request.body)
        message_content = data['message']
        user = request.user
        message = Message.objects.create(user=user, content=message_content)
        return JsonResponse({'message': 'Message saved successfully.'})
    else:
        return JsonResponse({'error': 'POST request required.'})
```

Рисунок 18 - Представления страницы чата

В данном случае на одну страницу используется две функции. Функция обработки основных запросов chatPage для прорисовки страницы ресурса. На странице происходит подключение по веб сокету, позволяющая в режиме реального времени обмениваться сообщениями между пользователями. Чтобы страница не перезагружалась, используется вторая функция save_messages для сохранения полученного сообщения. Чтобы обратиться к данному представлению save_messages в html шаблоне нужно подать Post запрос при отправке сообщения в обход подключения веб сокета [22].

Ниже на рисунке 19 показан фрагмент кода html шаблона, в котором используется JavaScript для подключения к веб сокету.

```
<script>
  const chatSocket = new WebSocket("ws://" + window.location.host + "/");
  chatSocket.onopen = function (e) {
    console.log("The connection was setup successfully !");
  };
  chatSocket.onclose = function (e) {
    console.log("Something unexpected happened !");
  };
  document.querySelector("#id_message_send_input").focus();
  document.querySelector("#id_message_send_input").onkeyup = function (e) {
    if (e.keyCode == 13) {
      document.querySelector("#id_message_send_button").click();
    }
  };
  document.querySelector("#id_message_send_button").onclick = function (e) {
    var messageInput = document.querySelector("#id_message_send_input").value;
    var currentTime = new Date().toISOString().slice(11, 19);
    fetch('/save-message/', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
        'X-CSRFToken': '{{ csrf_token }}'
      },
      body: JSON.stringify({
        message: messageInput,
        timestamp: currentTime
      })
      })
      .then(response => response.json())
      .then(data => console.log(data))
      .catch(error => console.error('Error:', error));
      var messageInput = document.querySelector(
        "#id_message_send_input"
      ).value;
      chatSocket.send(JSON.stringify({
        timestamp: currentTime,
        message: messageInput,
        username : "{{ request.user.username }}",
      }));
  };
```

Рисунок 19 - Html шаблон страницы чата

В части JavaScript кода создаётся новое подключение к веб сокету, прописан триггер на нажатие кнопки с определённым id, после которого с поля ввода сообщения, забирается написанный текст, через функцию fetch пробрасывается post запрос, ведущий к представлению save-message, для сохранения его в базе данных [12].

После сохранения, сообщение формируется в json формате и выводится в режиме реального времени без обновления страницы.

На рисунке 20 представлено графическое отображение html шаблона страницы чата.

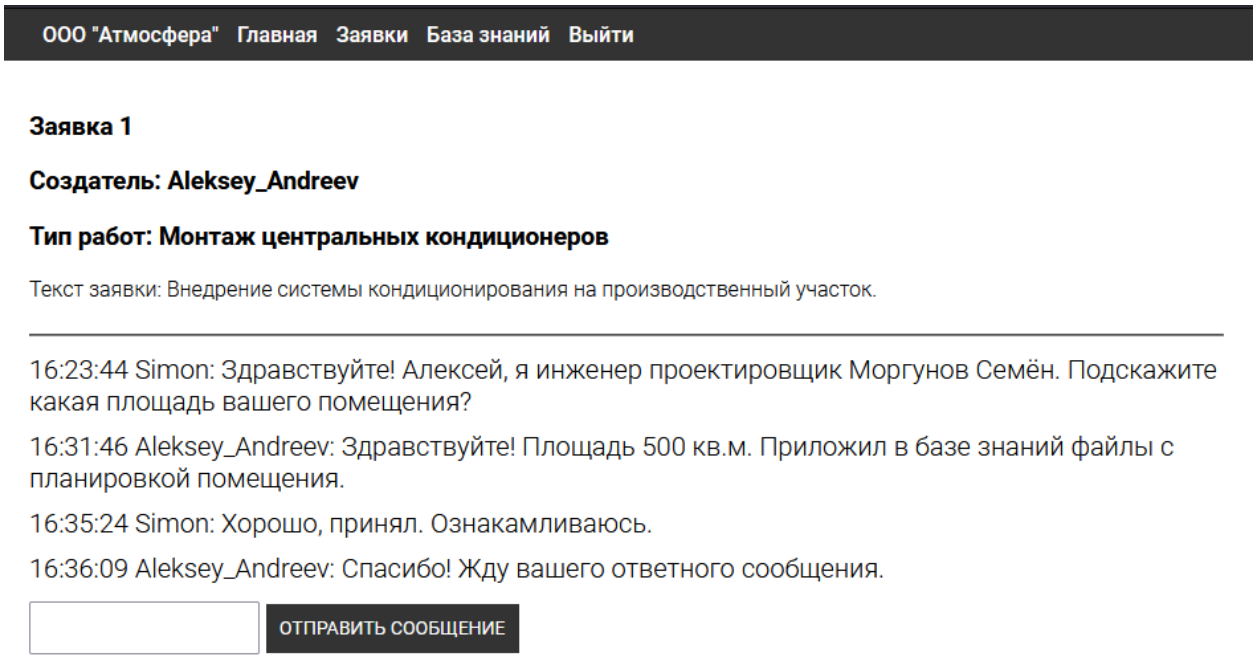


Рисунок 20 - Графическое отображение страницы чата

Каждое сообщение выводится друг за другом, с указанием времени, когда было написано это сообщение и кем. Сверху отображается информация о заявке, созданной ранее, есть поле ввода, в которое можно написать своё сообщение для отправки его другому пользователю.

Для файлов предусмотрена страница базы знаний, в которой, подгружаются и скачиваются все файлы. Для стороны заказчика, он может просматривать только свои файлы, своего предприятия. Сотрудники организации могут просматривать базу знаний разных заказчиков и скачивать и загружать файлы [16].

Модель для загрузки файлов ниже на рисунке 21.

```
class File(models.Model):
    name = models.CharField(verbose_name="Название файла", max_length=255)
    file = models.FileField(verbose_name="", upload_to='uploads/')
    uploaded_at = models.DateTimeField(auto_now_add=True)
```

Рисунок 21 - Модель загрузки файлов

В этой модели присутствует поле для выбора файла, создаться кнопка, которая позволит выбрать файл. Для данного поля указывается путь, по которому будет сохранятся файл на сервере. Также для каждого поля переименовано название для лучшего отображения на странице.

На рисунке 22 представлен код представления страницы базы знаний.

```
def file_list(request):
    if request.method == 'POST':
        form = FileUploadForm(request.POST, request.FILES)
        if form.is_valid():
            form.save()
            return redirect('file_list')
    else:
        form = FileUploadForm()
    files = File.objects.all()
    return render(request, 'files/file_list.html', {'files': files, 'form': form})
```

Рисунок 22 - Представление страницы базы знаний

Представление содержит обработку post запросов на загрузку файлов на сервер и создание метки этого файла в базе данных. Также в этом представлении создаётся переменная файлов, которая берёт всю информацию о существующих файлах в системе.

Ниже представлен фрагмент файла настроек проекта Django, на рисунке 23.

```
MEDIA_ROOT = f"{BASE_DIR}/uploads"
MEDIA_URL = 'uploads/'
```

Рисунок 23 – Путь, где находятся файлы

Для того чтобы, можно было загрузить файл, требуется указать пути сохранения файлов в конфигураторе проекта. Создание данных констант показывает компонентам Django, что по данному пути находятся медиа файлы. После этого можно сформировать ссылки на загрузку файлов.

На рисунке 24 представлено графическое отображение страницы базы знаний.

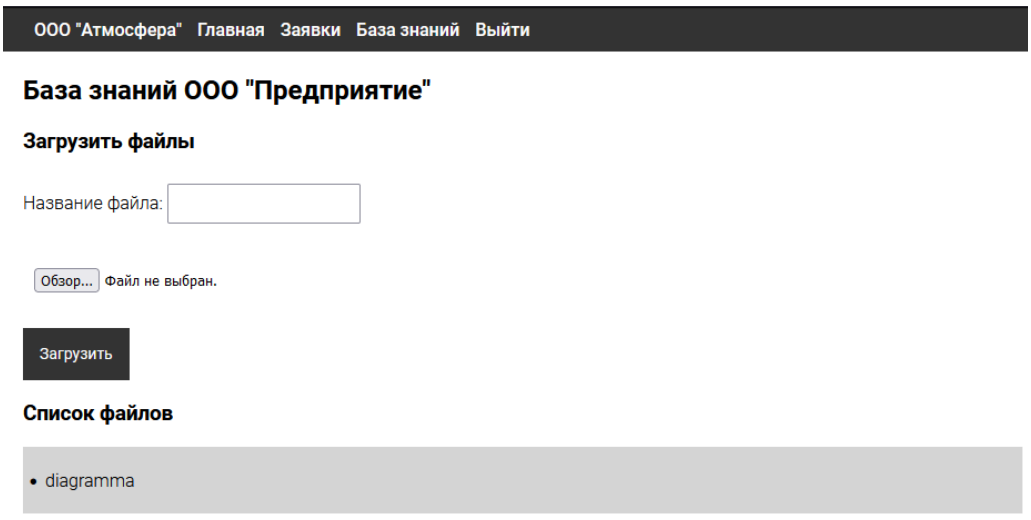


Рисунок 24 - Графическое отображение страницы базы знаний

На странице, сверху, для удобства добавлена форма загрузки файла. Чуть ниже представлен список файлов существующих для определённого предприятия. Нажав на название файла, формируется ссылка путь до файла, что позволяет пользователю скачать его.

3.3 Тестирование программного продукта

Во время приёма информационной системы следует провести испытание для следующих подсистем:

– подсистема авторизации пользователей;

– подсистема формирования заявки;

– подсистема диалогового чата;

– подсистема загрузки и отображения файлов.

Оценивается качество взаимодействия в процессе одновременного использования системы, смотрятся, как подсистемы реагируют между собой.

Целью тестирования является выявить ошибки и баги появившиеся в ходе разработки информационной системы. Следует ссылаться на следующие характеристики:

– полнота и качество реализации функций, указанных в техническом задании;

– работа пользователей в режиме реального времени;

– реакция система на ошибки пользователя;

– полнота действий доступных пользователю и их достаточность для функционирования системы;

– возможность работы пользователей без особой подготовки.

Для проведения тестирования и всевозможных испытаний воспользуемся методом черного ящика [10]. Испытания методом черного ящика продемонстрировано в таблице 3.

Таблица 3 – Проведения испытаний методом чёрного ящика

Действие	Результат
Авторизироваться в системе и открыть главную страницу	Открытая главная страница
Нажать на кнопку «Заявки» вверху в шапке информационной системы	Перенаправление на страницу заявок
Заполнить форму создания заявки и нажать на кнопку «Создать»	Обновление страницы и отображение заявки в списке
Нажать в списке на строку созданной заявки	Перенаправление на страницу чата прикреплённого к заявке
Написать в поле ввода сообщение и нажать на кнопку «Отправить сообщение»	Сообщение отобразиться в самом низу диалога с указанием даты и пользователя написавшего его
Нажать в шапке кнопку «База знаний»	Перенаправление на страницу базы знаний
Выбрать в списке организацию заказчика	Перенаправление на страницу базы знаний со списком файлов для организации заказчика
Ввести название файла, выбрать файл для загрузки и нажать на кнопку «Загрузить	Обновление страницы и отображение загруженного файла в списке файлов
Нажать в списке на загруженный файл	Перенаправление на скачивание файла и возращение на страницу базы знаний
Нажать в шапке на кнопку «Выйти»	Выход из системы, перенаправление на страницу авторизации

На рисунке 25 представлена страница авторизации в систему.

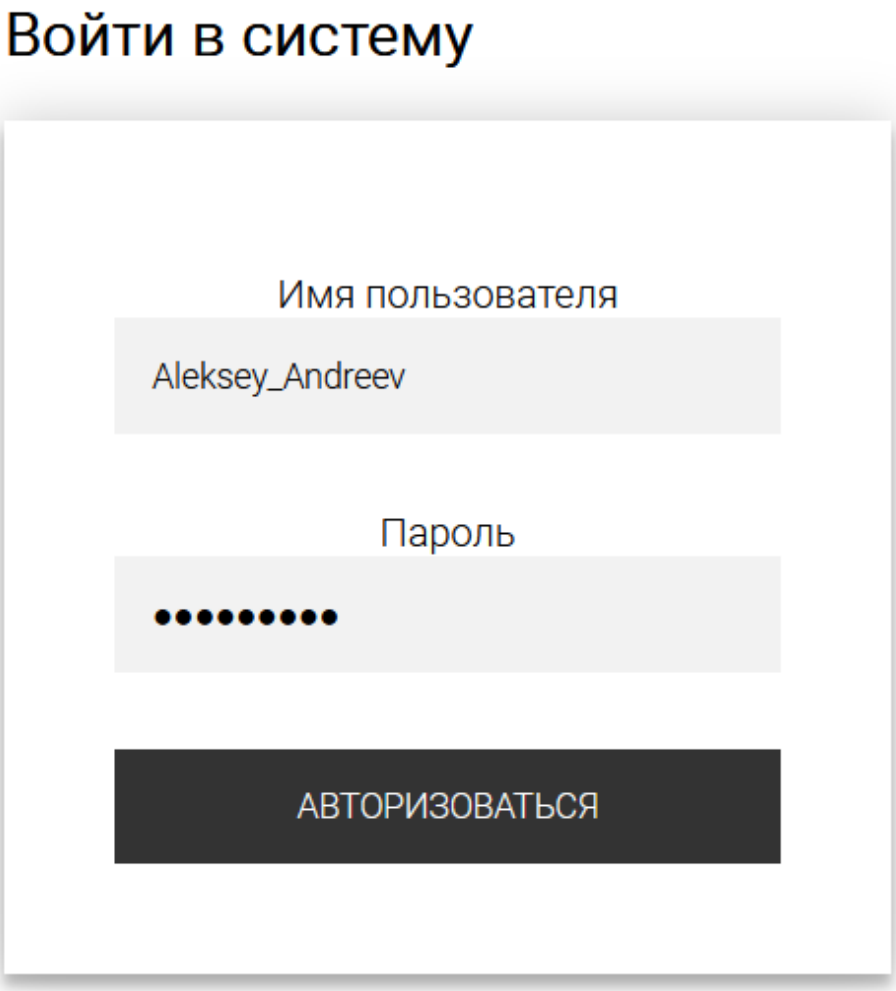


Рисунок 25 - Страница авторизации пользователя

При вводе неправильного пароля или имени пользователя, выведется сообщение об ошибке, что следует проверить корректность введённых данных [15].

По нажатию на кнопку авторизации автоматически перенаправляет на главную страницу.

Ниже на рисунке 26 представлена админ панель, в которой можно просматривать существующих авторизованных пользователей.

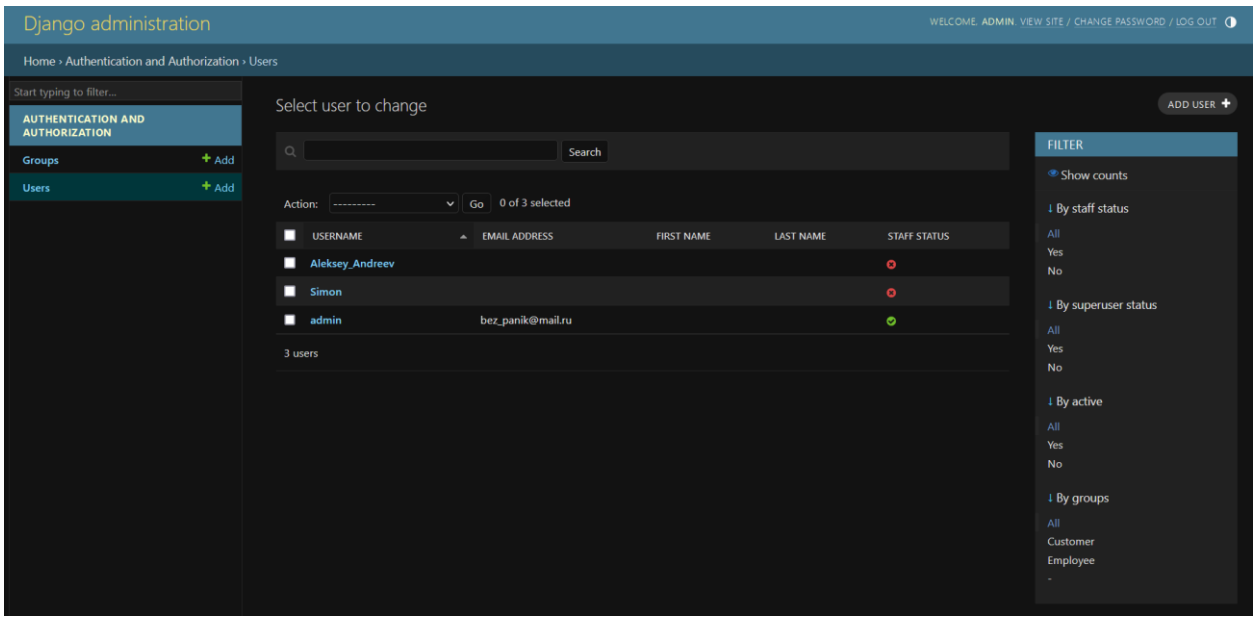


Рисунок 26 - Админ панель Django

На странице пользователей видны все пользователи системы, которые могут авторизоваться и пользоваться ей, также есть страница групп, на которой можно разделить права доступа к системе множеству пользователей [15].

При создании заявки и дальнейшего его отображения, заявка сохраняется в базе данных. На рисунке 27 показаны записи в базе данных в таблице заявок.

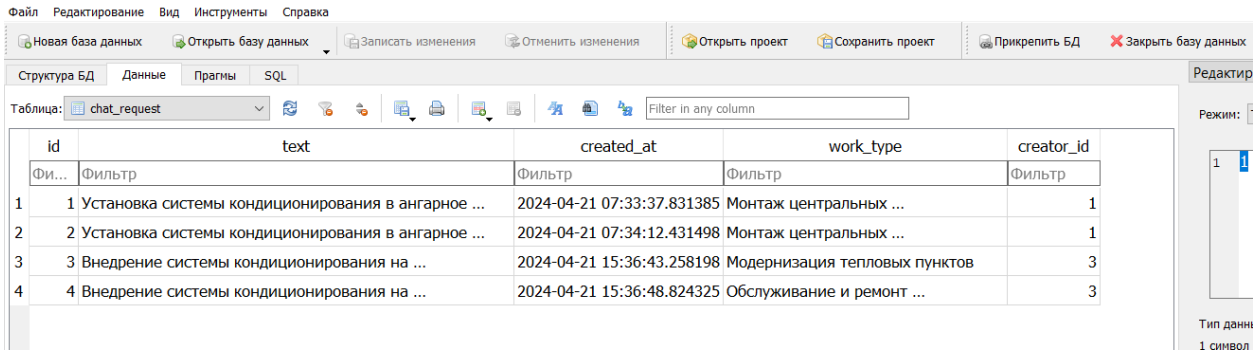


Рисунок 27 - Таблица заявок в базе данных SQLite

Для просмотра и управления базой данных используется бесплатное программного обеспечение с открытым исходным кодом «DB Browser for SQLite», позволяющая проверить зафиксированные данные для проведения тестирований [17].

Все созданные сообщения также сохраняются в базе данных, на рисунке 28 представлена таблица сохранённых сообщений.

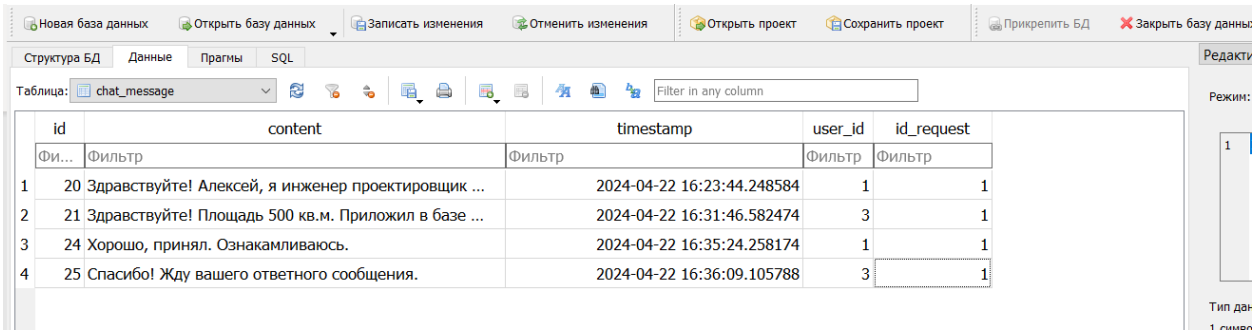


Рисунок 28 - Таблица сообщений в базе данных

В данной таблице записывается id сообщения, его содержание, время отправки, id пользователя написавшего сообщение и id заявки относящейся к этому сообщения для разделения чатов.

На рисунке 29 таблица файлов, загруженных в систему.

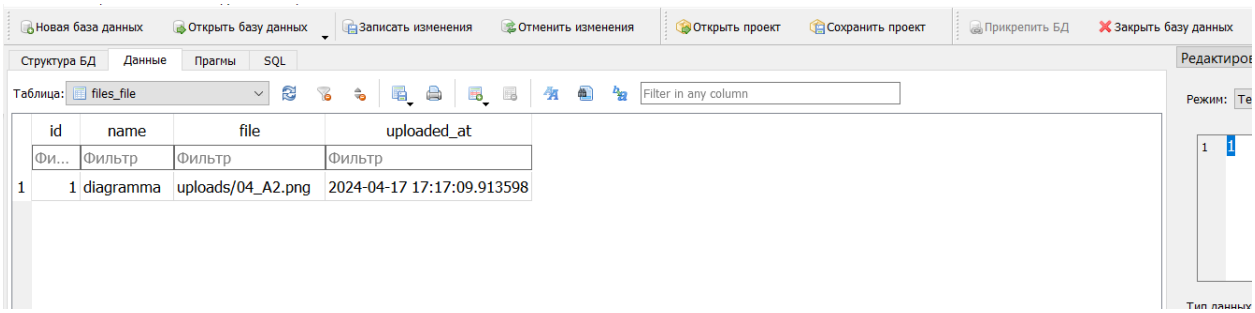


Рисунок 29 - Таблица файлов системы

Сохраняется не сам файл, а только его название, путь до файла и дата загрузки.

Выводы по главе 3

В этой главе показана физическая реализация информационной системы. Разработана система с использованием веб фреймворка Django на высокоуровневом языке программирования Python.

Поставленные задачи на разработку системы были выполнены в соответствии с требованиями, упомянутыми ранее. По ходу разработки были учтено удобство использования и создан минималистичный дизайн для комфортного взаимодействия с системой.

Также были проведены испытания, с помощью метода чёрного ящика. Выяснили, какой будет результат при выполнении действий. Во время проведения тестирования созданы пользователи заказчика и сотрудника, с помощью которых, проверили корректность разделения прав доступа и то, как система сохраняет все данные в базе данных.

Заключение

В результате выполнения выпускной квалификационной работы была разработана информационная система учёта заказов и взаимодействия с заказчиками. Организация ООО «Атмосфера» получила в своё распоряжение систему, соответствующая всем требованиям, сформулированными ранее. По ходу разработки были выполнены все поставленные задачи и была полностью достигнута поставленная цель.

Проанализировали существующие разработки помогающие взаимодействовать с заказчиками, называемыми CRM системами. Эти системы, также могут улучшить эффективность общения и коммуникации. Данные программные решение имеют ежемесячную подписку, обладают излишним функционалом для существующих задач организации. Реализованная информационная система учёта заказа и взаимодействия с заказчиками содержит в себе только нужный функционал, все данные будут храниться на территории организации, что обеспечит безопасность данных от третьих лиц.

По итогу можно сделать выводы о том, что система улучшает коммуникацию между сотрудниками и заказчиками. Обеспечивает бесперебойную загрузку и скачивание файлов без ограничений на странице базы знаний, позволяя загружать объёмные файлы чертежей и разного рода технических материалов. Внедрение информационной системы однозначно является правильным решением для организации на текущий момент времени в условиях огромной конкуренции и скорости обработки заявок от заказчиков.

Планируется улучшать и оптимизировать систему добавлением нового требуемого функционала за счёт расширения клиентской базы. При получении обратной связи от заказчиков улучшить графическое отображение страниц исходя из пожеланий клиентов.

Список используемой литературы и используемых источников

1. Буч Г., Рамбо Д., Джекобсон А. Язык UML Руководство пользователя — С-П.: Издательство «Питер», 2010 — 432 с.

2. Джейсон Мак-Колм Смит Элементарные шаблоны проектирования : Пер. с англ. — М. : ООО “И.Д. Вильямс”, 2013. — 304 с.

3. Мамедли Р.Э. М 22 Системы управления базами данных: Учебное пособие. – Нижневартовск: Издательство Нижневартовского государственного университета, 2021. – 214 с.

4. Мкртычев С.В. Методология построения проблемно-ориентированных систем управления операционной деятельностью страховой компании на основе объектно-структурного подхода : диссертация ... доктора технических наук : 05.13.10 / Мкртычев Сергей Вазгенович; [Место защиты: Том. гос. ун-т систем упр. и радиоэлектроники (ТУСУР) РАН]. - Тольятти, 2016. - 288 с. : ил. URL: https://search.rsl.ru/ru/record/01008801212 (дата обращения: 11.03.2024)

5. Розенберг Д., Скотт К. Применение объектного моделирования с использованием UML и анализ прецедентов.: Пер. с англ. М.: ДМК Пресс, 2002.

6. Atmosfera-energy.ru [Электронный ресурс]: Информация об организации ООО «Атмосфера», официальный сайт компании. URL: https://atmosfera-energy.ru/ (дата обращения: 12.03.2024).

7. Djangoproject.com [Электронный ресурс]: Официальный сайт веб фреймворка «Django». URL: https://www.djangoproject.com/ (дата обращения: 03.04.2024).

8. Geeksforgeeks.org [Электронный ресурс]: Realtime chat app using Django. URL: https://www.geeksforgeeks.org/realtime-chat-app-using-django/ (дата обращения: 08.04.2024).

9. Gendalf.ru [Электронный ресурс]: что такое «CRM-система» и чем она полезна бизнесу? URL: https://gendalf.ru/news/marketing/chto-takoe-crm-sistema/ (дата обращения: 24.03.2024).

10. Habr [Электронный ресурс]: Тестирование методом черного ящика. URL: https://habr.com/ru/articles/462837/ (дата обращения: 19.04.2024).

11. Habr [Электронный ресурс]: Использование диаграммы вариантов использования UML при проектировании программного обеспечения. URL: https://habr.com/ru/articles/566218/ (дата обращения: 20.03.2024).

12. Inexture.com [Электронный ресурс]: Creating a Chat Application with Django Channels. URL: https://www.inexture.com/chat-app-django-channels/ (дата обращения: 08.04.2024).

13. Metanit [Электронный ресурс]: Руководство по веб-фреймворку Django. URL: https://metanit.com/python/django/ (дата обращения: 02.04.2024).

14. Mozilla.org [Электронный ресурс]: Django Tutorial Part 9: Working with forms. URL: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Forms (дата обращения: 03.04.2024).

15. Proglib [Электронный ресурс]: Django с нуля. Часть 2: регистрация, авторизация, ограничение доступа. URL: https://proglib.io/p/django-s-nulya-chast-2-registraciya-avtorizaciya-ogranichenie-dostupa-2022-06-08 (дата обращения: 07.04.2024).

16. Python.org [Электронный ресурс]: Python 3.12.3 documentation. URL: https://docs.python.org/3/ (дата обращения: 06.04.2024).

17. SQLite.org [Электронный ресурс]: SQLite documentation. URL: https://www.sqlite.org/docs.html (дата обращения: 15.04.2024).

18. Tproger [Электронный ресурс]: 10 лучших фреймворков для веб-разработки на Python. URL: https://tproger.ru/articles/10-luchshih-frejmvorkov-dlja-veb-razrabotki-na-python (дата обращения: 02.04.2024).

19. Trinion.org [Электронный ресурс]: IDEF0. Знакомство с нотацией и пример использования. URL: https://trinion.org/blog/idef0-znakomstvo-s-notaciey-i-primer-ispolzovaniya (дата обращения: 23.03.2024).

20. Unicraft.org [Электронный ресурс]: Описание бизнес-процессов: из хаоса в порядок. URL: https://www.unicraft.org/blog/11661/business-process/ (дата обращения: 13.03.2024).

21. Wikipedia [Электронный ресурс]: Диаграмма деятельности. URL: https://ru.wikipedia.org/wiki/Диаграмма_деятельности (дата обращения: 27.03.2024).

22. "Understand Django: An Exploration of the Django Web Framework" by Matt Layman