Министерство образования и науки РТ
ГАПОУ «Альметьевский политехнический техникум»

Специальность 09.02.07
Информационные системы (по отраслям)

ПРОЕКТ

Проектирование и разработка информационной системы взаимодействия дилеров и отдела продаж на крупном нефтяном предприятии

Студент А.Л. Каримов
Руководитель Е.А. Куприянова

2021

Содержание

Введение

В настоящее время все компании в своей деятельности используют разные программные средства, которые позволяют автоматизировать практически все процессы в организации, а также упрощают взаимодействие клиентов предприятия. При этом программные средства являются сложными для персонала и для клиентов, которые требуют соответствующего обучения и знаний для их использования.

Привлечение и удержание дилеров — одно из важнейших направлений работы производственно-торгового предприятия. Эффективная дилерская сеть — это стабильный доход компании и серьезное преимущество в глазах потребителей.

Однако, часто бывает, что увеличение количества дилеров влечет за собой не только дополнительную прибыль, но и увеличение штата сотрудников отдела продаж.

В настоящее время в некоторых нефтяных предприятиях, имеется ряд проблем:

1. Вся работа с дилерами ведется по телефону и электронной почте. Заказ оформляется письмом на электронный адрес, обновления по нему обсуждаются в личном общении. Общение и оформление заказа, который отнимает много времени у сотрудников.
2. Нет актуальной информации о наличии товара на складе. Информация об остатках поступает менеджерам поздно и не обновляется в течение дня. При формировании заказа нельзя точно сказать, доступно ли нужное количество товара.
3. Анализ объемов продаж дилерам, остатки на их складах, планирование производства происходит в ручном виде аналитиком предприятия на основе догадок и предположений. Все данные собираются в один файл в формате Excel, из него вручную создаются отчеты.
4. Руководители отдела не получают точной информации о работе сотрудников своего отдела и не могут точно планировать их загрузку, ставить планы продаж для предприятия.

Это и определяет актуальность выбора темы проекта

На основании этого, целью проекта является:

1. Повышение эффективности принятия оперативных управленческих решений в части компании за счет предоставления консолидированной информации руководству компании.
2. Возможность получения более подробной информации о деятельности партнеров компании: структуре и объеме продаж всех товаров, особенностям работы, каналам сбыта, торговых представителях.
3. Контроль работы отделов и сотрудников компании, возможность видеть оперативную и объективную информацию по всем этапам работы с партнерами.
4. Создание конкурентного преимущества компании при работе с партнерами за счет удобного инструмента ведения бизнес деятельности.
5. Увеличение эффективности работы сотрудников компании.

Для этого необходимо решить следующие задачи:

a. Проанализировать предметную область
b. Реализовать управление проектом
c. Определить объекты предметной области
d. Выбрать методы и средства разработки
e. Выбрать и обосновать методы и средства разработки базы данных
f. Организовать хранение данных
g. Реализовать профессиональные возможности информационной системы
h. Протестировать информационные системы
i. Определить модели эксплуатации программного продукта
j. Реализовать информационную безопасность
k. Спроектировать развитие программного продукта
l. Разработать руководство пользователя
m. Разработать руководство администратора

n. Разработать руководство программиста

Объектом проектирования является нефтяная отрасль.

Предметом - является автоматизация и взаимодействие отдела продаж с дилерами, а также контроль учета продукции на нефтяном предприятии.

1. АНАЛИТИЧЕСКАЯ ЧАСТЬ

1.1 Анализ предметной области

«Татнефть» - одна из крупнейших российских нефтяных компаний, в составе которой динамично развиваются нефтегазодобыча, нефтепереработка, нефтегазохимия, шинный комплекс, сеть АЗС, электроэнергетика, разработка и производство оборудования для нефтегазовой отрасли и блок сервисных структур. «Татнефть» также участвует в капитале компаний финансового сектора. «Татнефть» является одной из крупнейших российских публичных компаний с рыночной капитализацией более 28 млрд. долларов США на конец 2019 года.

Миссия Компании:

Обеспечение поступательного развития Компании как одного из крупнейших вертикально-интегрированных российских производителей нефти и газа, продуктов нефтегазопереработки и нефтехимии на основе эффективного управления активами акционеров, рационального использования природных ресурсов и корпоративной социальной ответственности.

В целях увеличения акционерной стоимости Компании за счет дальнейшего повышения конкурентоспособности, сохранения лидерских позиций в отрасли и обеспечения устойчивого роста реализуется Стратегия Группы «Татнефть» до 2030 года. В рамках Стратегии предусмотрены увеличение добычи нефти до 38,4 млн тонн при 100-процентном восполнении запасов, расширение производства высококачественных нефтепродуктов, продолжение диверсификации бизнеса, обеспечение роста эффективности инвестиций.

Финансовый потенциал ПАО «Татнефть» позволяет сегодня осуществлять крупные инвестиционные проекты как за счет собственных, так и заемных средств, обеспечивать высокую дивидендную доходность акционерам, сохраняя при этом на высоком уровне финансовую устойчивость и ликвидность.

В области нефтедобычи существенным ресурсным активом ПАО «Татнефть» и перспективным объектом наращивания добычи являются значительные запасы сверхвязкой нефти (СВН). Опыт и компетенции специалистов Компании

позволяют рентабельно разрабатывать месторождения СВН с использованием собственных инновационных технологий и внедрения IT-решений.

Одним из важнейших проектов Компании является эксплуатация и расширение в Нижнекамске Комплекса нефтеперерабатывающих и нефтехимических заводов «ТАНЕКО» (Комплекс НП и НХЗ).

Компания реализует стратегическую программу развития сети автозаправочных комплексов Компании. В настоящее время в составе Группы «Татнефть» функционируют более 795 АЗС. Рознично-сбытовая сеть Компании обеспечивается высококачественным экологичным топливом собственного производства и предлагает своим клиентам постоянно расширяемый ассортимент сопутствующих товаров и услуг.

Достижения в производственной, природоохранной, социальной деятельности, высокий уровень корпоративного управления, открытость и прозрачность Компании высоко оцениваются акционерами, деловыми партнерами и инвестиционным сообществом в целом.

Корпоративная социальная ответственность ПАО «Татнефть» направлена на создание эффективных и безопасных рабочих мест, минимизацию воздействия на окружающую среду и поддержание благоприятной среды в регионах деятельности Компании, социальную поддержку работников и членов их семей.

Среди многообразия поисков путей развития рынка, средств производства, новых направлений деятельности коммерческо-посреднических организаций и предприятий вызывают значительный интерес научные исследования и практические новации, объединяемые понятием логистики.

В течение последних лет бурно развиваются основанные на информатике новые логистические технологии. Информационные системы занимают в этих технологиях центральное положение. Предприятие является открытой системой, которая материальным и информационным потоками связана с поставщиками, потребителями, экспедиторами и транспортными организациями. При этом возникают трудности преодоления мест стыка между информационными

системами предприятия и других организаций. В местах стыка материальный или информационный поток переходить через границы правомочия и ответственности отдельных подразделений предприятия или через границы самостоятельных организаций. Обеспечение плавного проделывания мест стыка является программные обеспечения. На вопрос что произойдет с предприятием если данная информационная система не будет внедрена:

1. Не будет эффективности принятия оперативных управленческих решений в части компании за счет предоставления консолидированной информации руководству компании.
2. Не будет более подробной информации о деятельности партнеров компании: структуре и объеме продаж всех товаров, особенностям работы, каналам сбыта, торговых представителях.
3. Контроль работы отделов и сотрудников компании, не будет возможности видеть оперативную и объективную информацию по всем этапам работы с партнерами.
4. На предприятии не увеличится объем продаж, но и сохранить позиции на местном рынке будет сложно.

Данная разработанная информационная система будет похожа на те же самые аналогичные программы такие как:

Программа «1С» — это продукт фирмы «1С», предназначенный для автоматизации предприятий. Все программные решения разработчика созданы на базе единой технологической платформы и функционирует по общим принципам.

CRM-система — это прикладное программное обеспечение для организаций, предназначенное для автоматизации стратегий взаимодействия с заказчиками (клиентами), в частности, для повышения уровня продаж, оптимизации маркетинга и улучшения обслуживания клиентов путем сохранения информации о клиентах и истории взаимоотношений с ними, установления и улучшения бизнес-процессов и последующего анализа результатов.

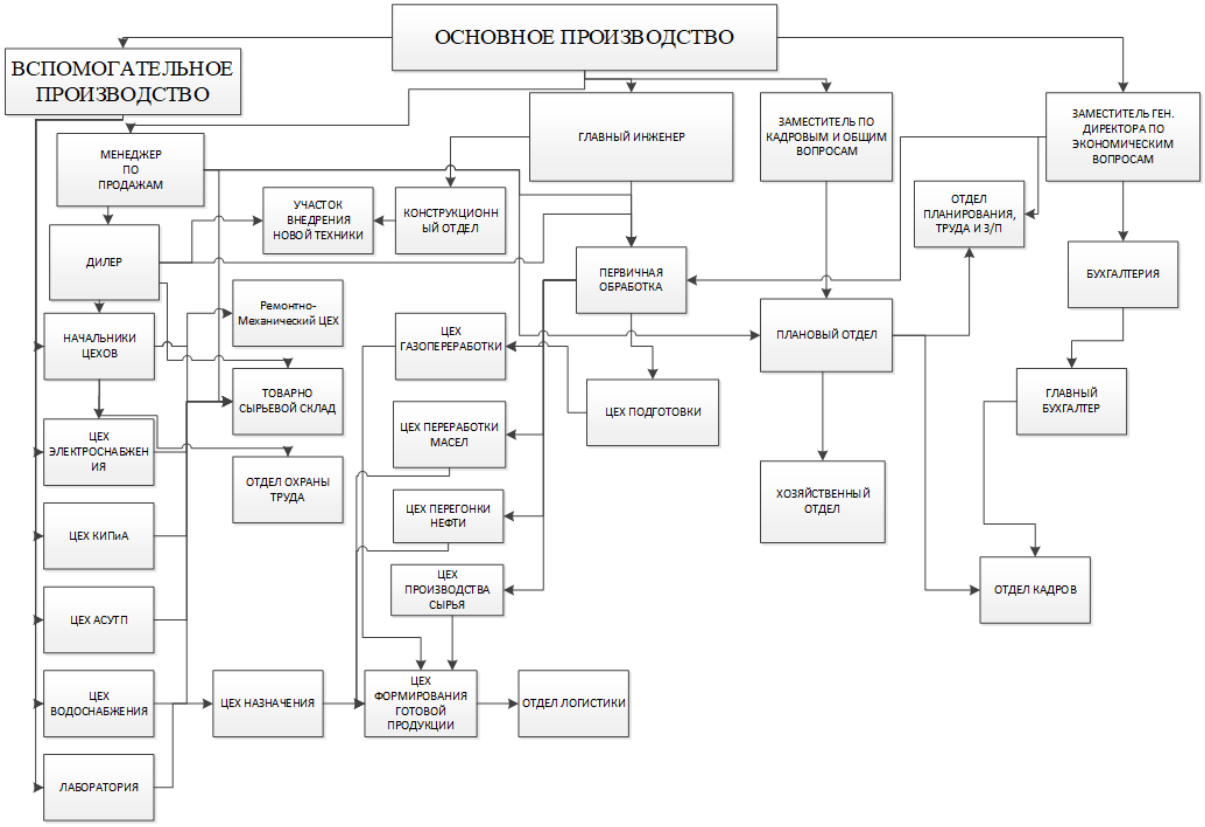


Рисунок 1.1.1 Организационная структура предприятия

Таблица 1.1 Характеристика внутренних процессов предприятия

Процесс	**Инициатор процесса**	**Используемые объекты**	**Под процессы**	**Период существования**
Заявка на покупку товара	Менеджер по продажам	Электронная форма	1. Заполнение формы для заказчика 2. Отправка формы дилеру 3. Уточнение заполненных форм	3-5 дней
Принятие заявки	Дилер	Электронная форма	1. Проверка формы заявки 2. Просмотр складов 3. Уточнение заявки заказчика	1-3 дней

Продолжение таблицы 1.1

			4. Просмотр наличие товара на складах	
Изменение заявки	Дилер	Электронная форма	1. Формирование и редактирование заявки 2. Осуществление звонка заказчику для уточнения информации	1-2 дней
Оформление счета на оплату	Бухгалтер	Электронный документооборот	1. Оформление расчета стоимости товара 2. Оформление счет-фактуры заказчика 3. Отправка счет-фактуры заказчику	2 дней
Планирования продаж	Менеджер по продажам	Электронный документооборот	1. Создание целей для организации 2. Формирование отчёта для работников	2-3 дней
Формирование задач организации	Менеджер по продажам	Электронный документооборот	1. Создание задач для работников 2. Отправка задач работника	1 дней

Продолжение таблицы 1.1

			3. Редактирование задач	
Формирование оплат труда работников	Бухгалтер	Электронный документооборот	1. Расчёт часов работников 2. Расчёт количество продаж работников	30 дней
Отправка товара заказчику	Отдел логистики	Логистический документооборот	1. Принятие путевого листа 2. Отправка удобным заказчиком логистической компанией 3. Проверка оплаты счёт-фактуры	3-4 дней
Добавление и просмотр отчётности работников	Менеджер по продажам	Электронный документооборот	1. Установление сроков для отчёта 2. Просмотр отчётов 3. Просмотр планирования 4. Редактирование отчёта 5. Удаление отчёта	3-4 дней
Просмотр складов и закупки для складов	Дилер	Ресурсный документооборот	1. Просмотр складов 2. Редактирование товаров 3. Удаление товаров	3 дней

Продолжение таблицы 1.1

			4. Заказ товара на выдачу 5. Формирование счёт-фактуры товара	

Таблица 1.2 Характеристика внешних воздействий на предметную область

Характеристика воздействия	**Инициатор**	**Задействованные объекты**	**Задействованные процессы**	**Период воздействия**
Поступление нового заказа	Заказчик	Менеджер Бухгалтерия Дилер	Оформить заказ. Распечатать накладную. Заключить договор. Собрать заказ. Отправить в логистическую компанию.	3-5 дней
Поступление отчётности работника	Работники организации	Директор Менеджер Бухгалтер	Распечатать отчёт. Рассчитать количество продаж Удалить отчёт	2-4 дней
Планирование задач организации	Менеджер по продажам	Работники организации Склад Бухгалтерия Дилер Директор	Оформить задачу Смена статуса задачи Редактирование задач Удаление задач Установление сроков задач	2 дней

Продолжение таблицы 1.2

Поступление нового товара на склад	Отдел склада	Дилер Бухгалтерия Менеджер	Собрать заказ Удалить заказ Заключить договор Распечатать накладную Проверить оплату заказа	3-4 дней
Оплата заказа	Заказчик	Бухгалтер	Распечатать счёт-фактуру Заключить договор Проверить наличие оплаты заказа Отмена заказа	2 дней

Таблица 1.3 Список документов, регламентирующих предметную область

Наименование документа	**Статус документа**	**Задействованные объекты**	**Задействованные процессы**
Устав организации	Внутренний	Все сотрудники организации	Все процессы организации.
Положение об оплате труда	Внутренний	Все сотрудники организации	Начисление заработной платы. Подписание договора. Оформление расчетных листов.

Продолжение таблицы 1.3

Положение об охране труда	Внутренний	Все сотрудники организации	Техника безопасности. Организация рабочих мест. Составление отчетности по охране труда. Контроль мероприятия по предупреждению несчастных случаев.
Трудовой договор	Межрегиональный	Вся организация	Ознакомление прав и обязанностей сторон. Условие труда. Ответственность сторон.
Должностная инструкция	Внутренний	Отделы организации	Обеспечение четкого разграничения обязанностей между работниками. Организация .взаимодействия сотрудников при выполнении сложных работ
Лицензирование	Региональный	Вся организация	Осуществление и безопасность выпускаемой продукции для потребителя. Соблюдение норм закона и отсутствие нарушений со стороны организации

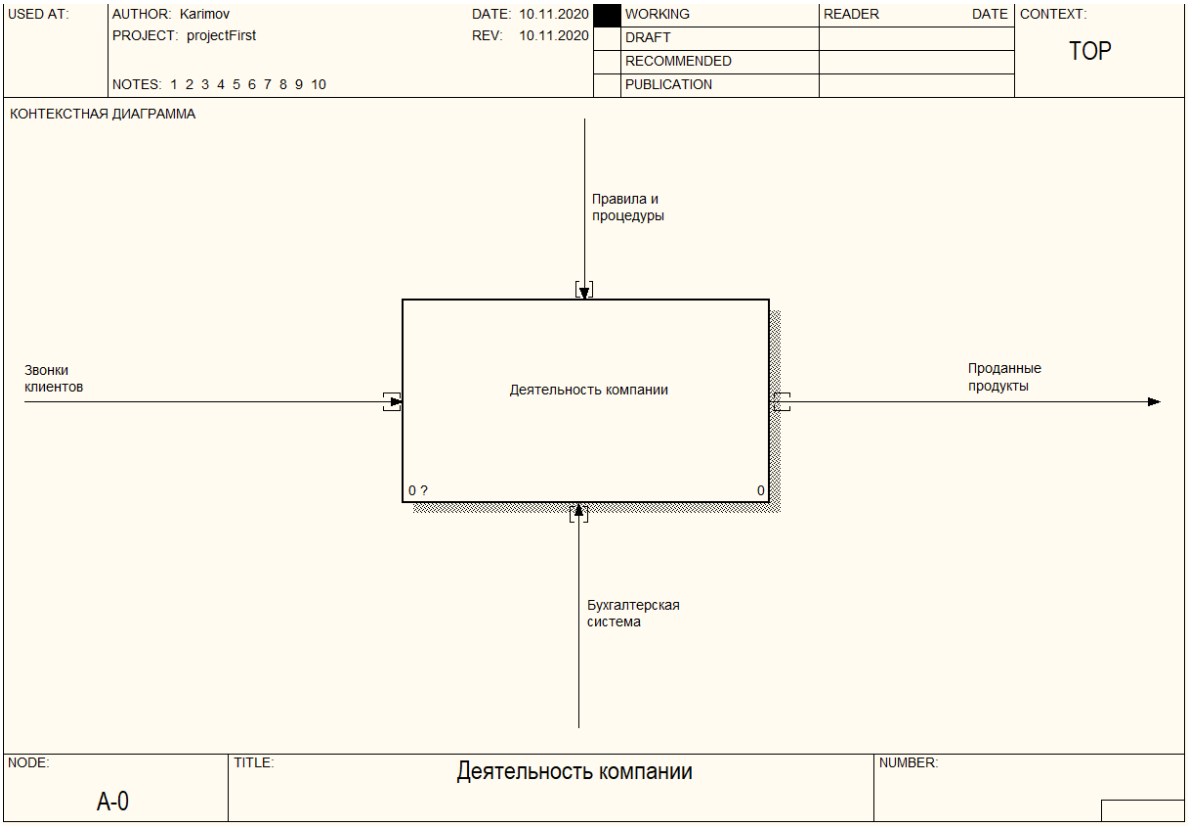


Рисунок 1.1.2 Контекстная диаграмма предприятия

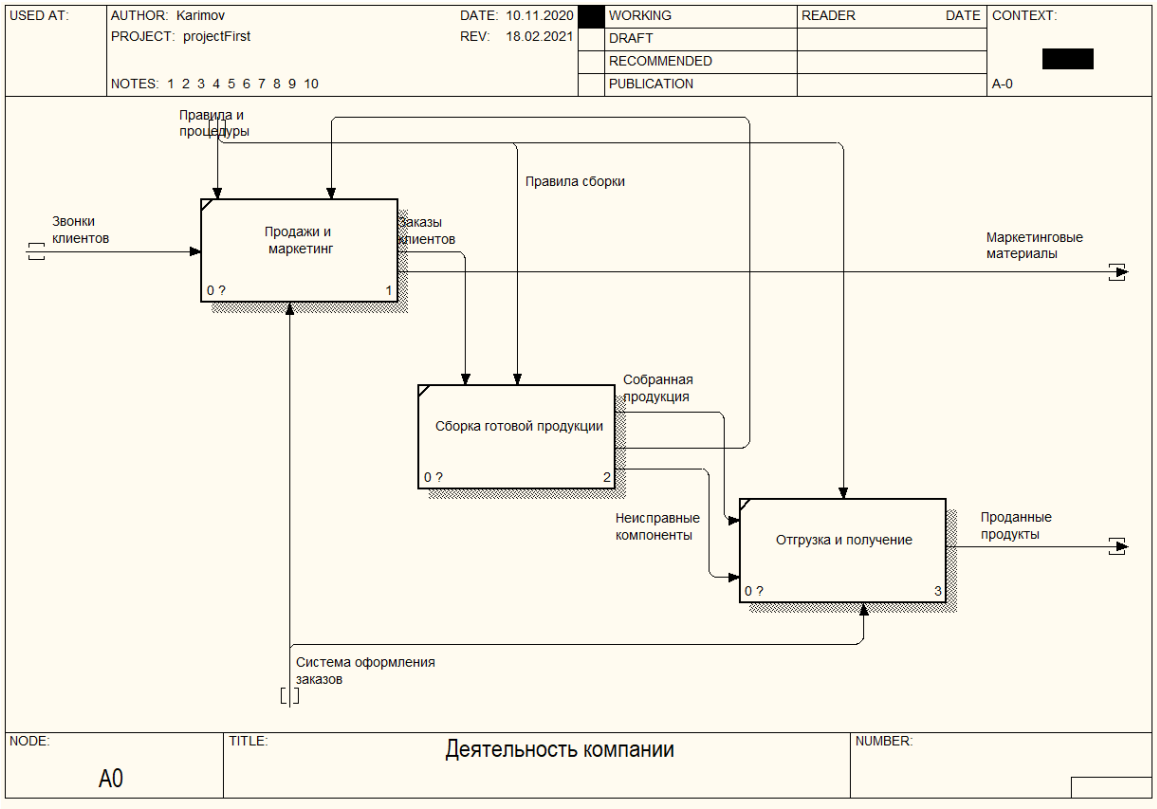


Рисунок 1.1.3 Диаграмма декомпозиции

Таблица 1.5 используемые программные обеспечения предприятием

Наименование ПО	1С: предприятие	Веб-портал	Word	Excel	Visio	Windows 10
Производитель	Фирма 1С	Россия	Microsoft	Microsoft	Microsoft	Microsoft
Возможность мобильной версии	-	+	-	-	-	-
Личный кабинет	+	+	-	-	-	-

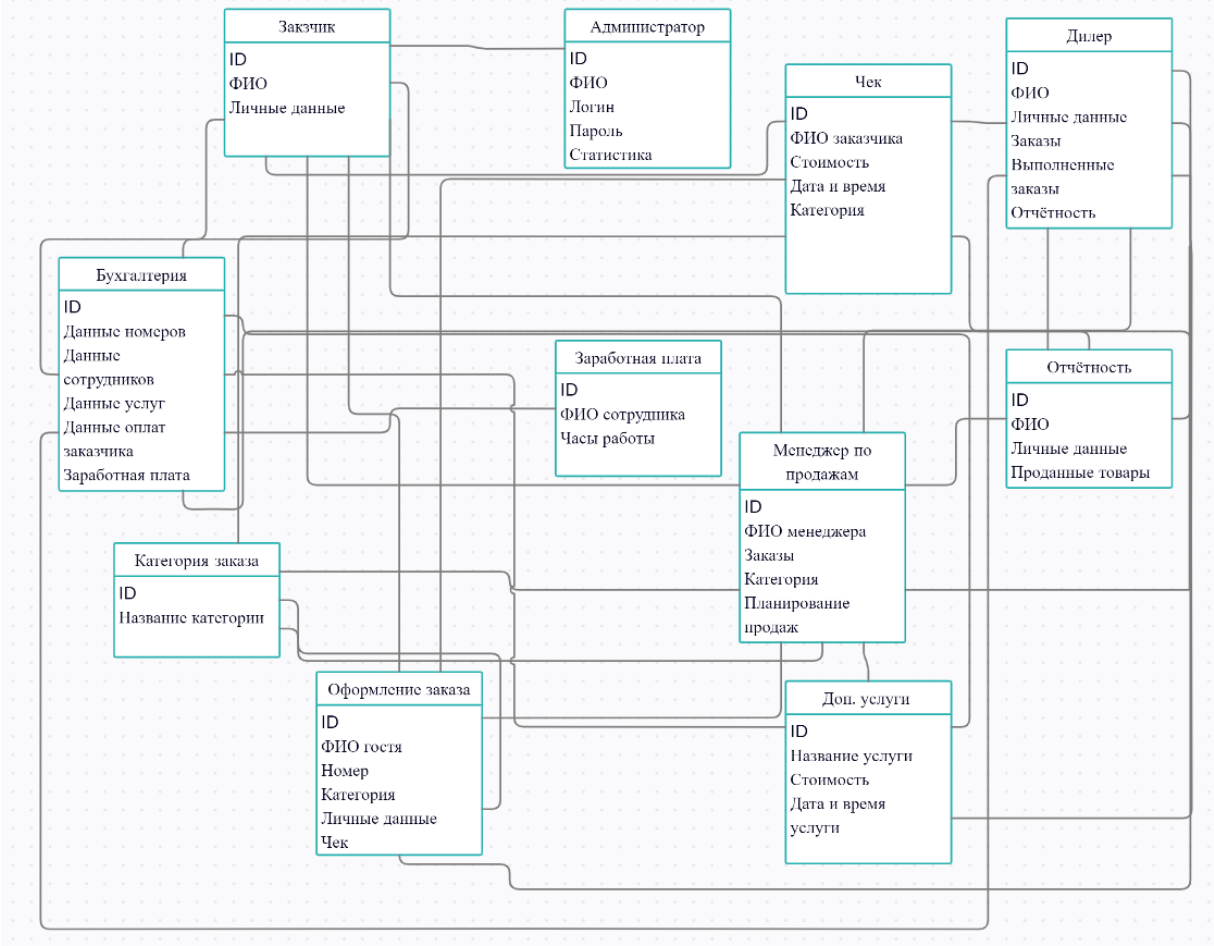


Рисунок 1.1.4 ERD диаграмма

1.2 Управление проектом

Продуктивность — показатель, который руководители и успешные менеджеры проектов постоянно стремятся повышать. На то есть веская причина: сегодня сложно вести успешный бизнес из-за высокой конкуренции.

Однако, количество отработанных часов не всегда пропорционально хорошим результатам. Стрессы и переработки, наоборот, часто ведут к выгоранию сотрудников и, как результат, понижению эффективности. Во избежание таких последствий очень важно максимально грамотно выстроить рабочие процессы с момента их планирования.

А для того чтобы повысить продуктивность, компании должны оставаться в тонусе, изучать рынок, следить за трендами и внедрять нововведения.

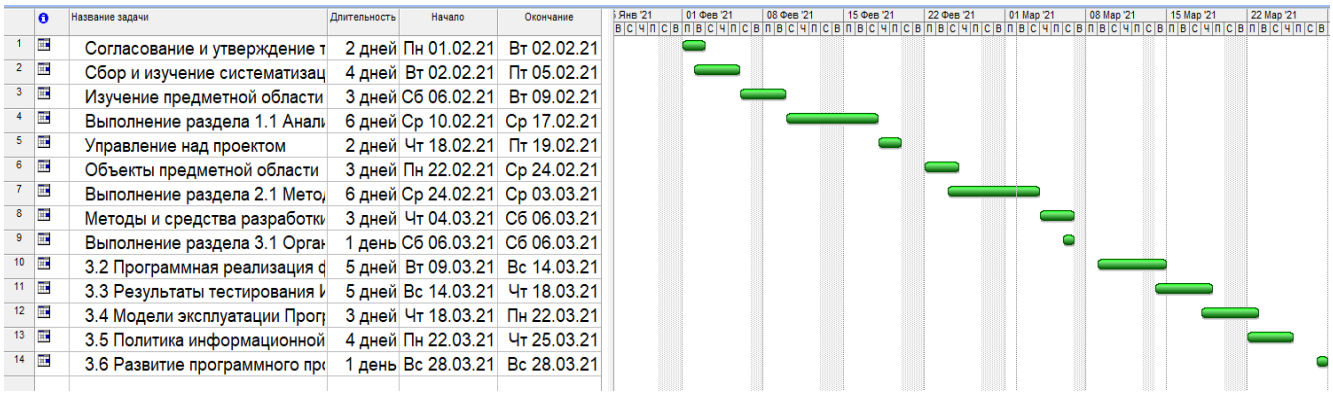
Существует множество базовых инструментов для улучшения работы команды и достижения целей организации. Один из них — диаграмма Ганта.

Диаграмма Ганта — это инструмент, позволяющий визуализировать и управлять проектами, структурировать их выполнение и виде

Диаграмма Ганта представляет собой горизонтальные полосы, расположенные между двумя осями:

- **Вертикальная**. Это список задач.
- **Горизонтальная**. Это временная шкала проекта.

Каждая полоса обозначает проект, задачу или подзадачу, которые нужно выполнить в определенный срок. График построен в хронологическом порядке, что помогает отслеживать дедлайны и последовательность выполнения задач.



	Название задачи	Длительность	Начало	Окончание
1	Согласование и утверждение т	2 дней	Пн 01.02.21	Вт 02.02.21
2	Сбор и изучение систематизац	4 дней	Вт 02.02.21	Пт 05.02.21
3	Изучение предметной области	3 дней	Сб 06.02.21	Вт 09.02.21
4	Выполнение раздела 1.1 Анали	6 дней	Ср 10.02.21	Ср 17.02.21
5	Управление над проектом	2 дней	Чт 18.02.21	Пт 19.02.21
6	Объекты предметной области	3 дней	Пн 22.02.21	Ср 24.02.21
7	Выполнение раздела 2.1 Мето	6 дней	Ср 24.02.21	Ср 03.03.21
8	Методы и средства разработки	3 дней	Чт 04.03.21	Сб 06.03.21
9	Выполнение раздела 3.1 Орган	1 день	Сб 06.03.21	Сб 06.03.21
10	3.2 Программная реализация д	5 дней	Вт 09.03.21	Вс 14.03.21
11	3.3 Результаты тестирования И	5 дней	Вс 14.03.21	Чт 18.03.21
12	3.4 Модели эксплуатации Прогр	3 дней	Чт 18.03.21	Пн 22.03.21
13	3.5 Политика информационной	4 дней	Пн 22.03.21	Чт 25.03.21
14	3.6 Развитие программного пр	1 день	Вс 28.03.21	Вс 28.03.21

Рисунок 1.2.1 Диаграмма Ганта

1.3 Объекты предметной области

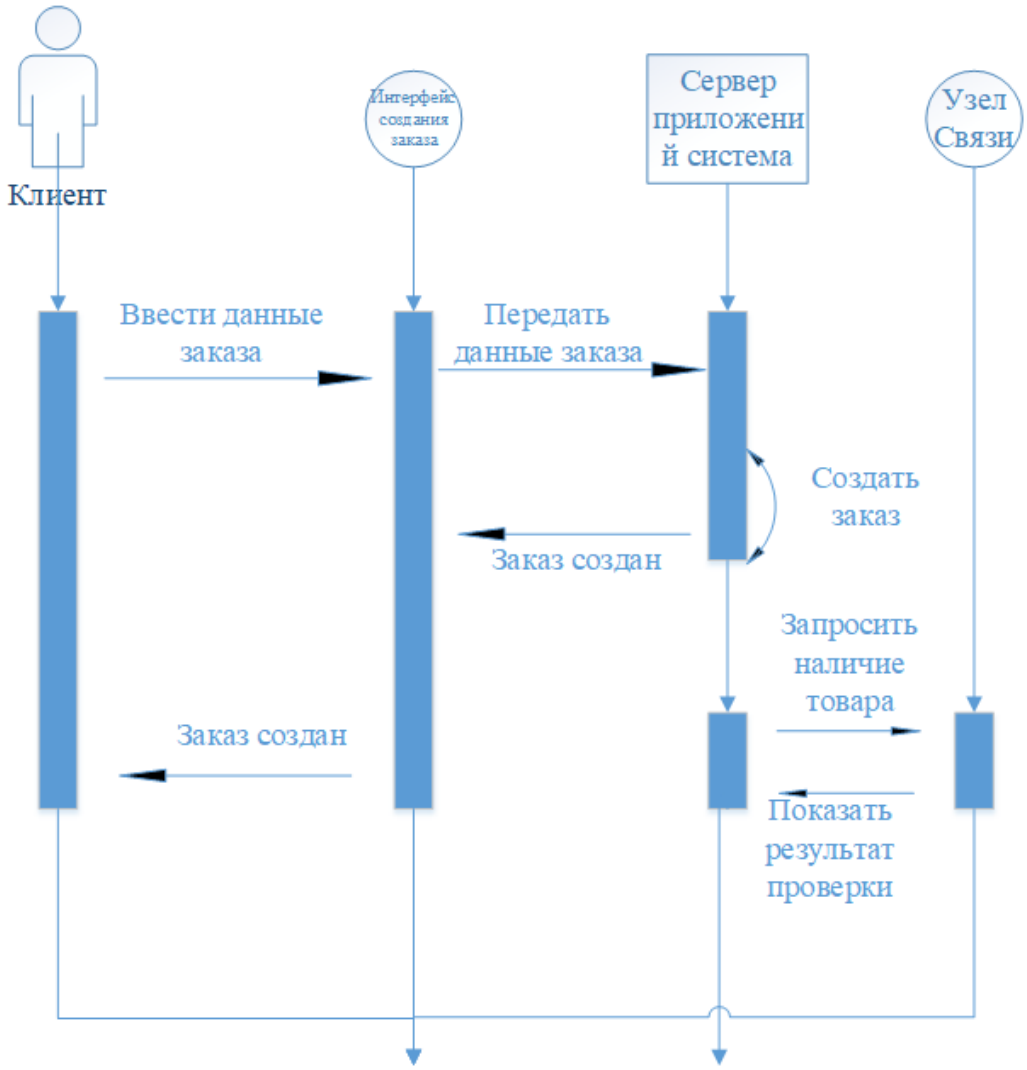


Рисунок 1.3.1 Диаграмма последовательности заказчика

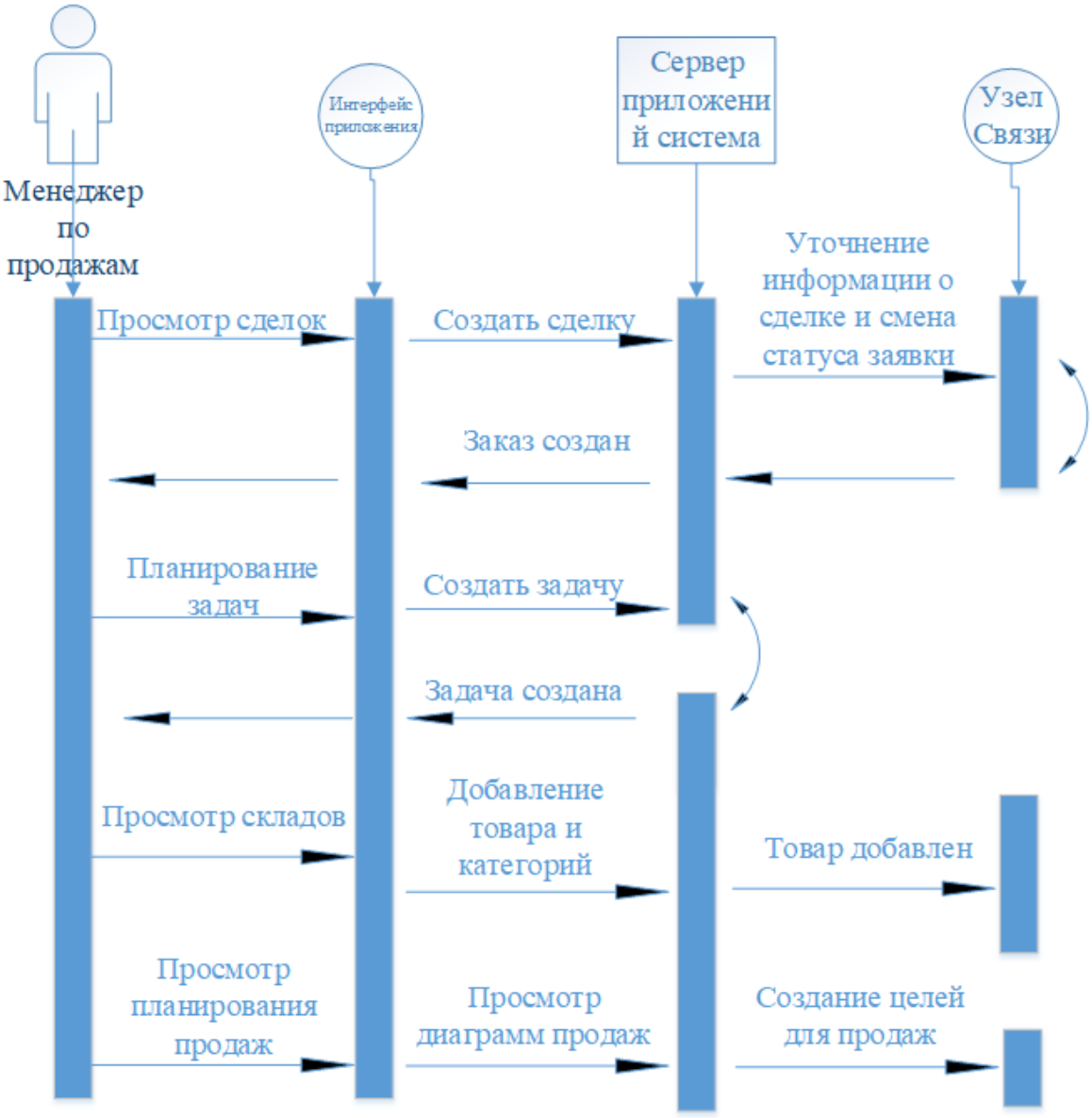


Рисунок 1.3.2 Диаграмма последовательности менеджера по продажам

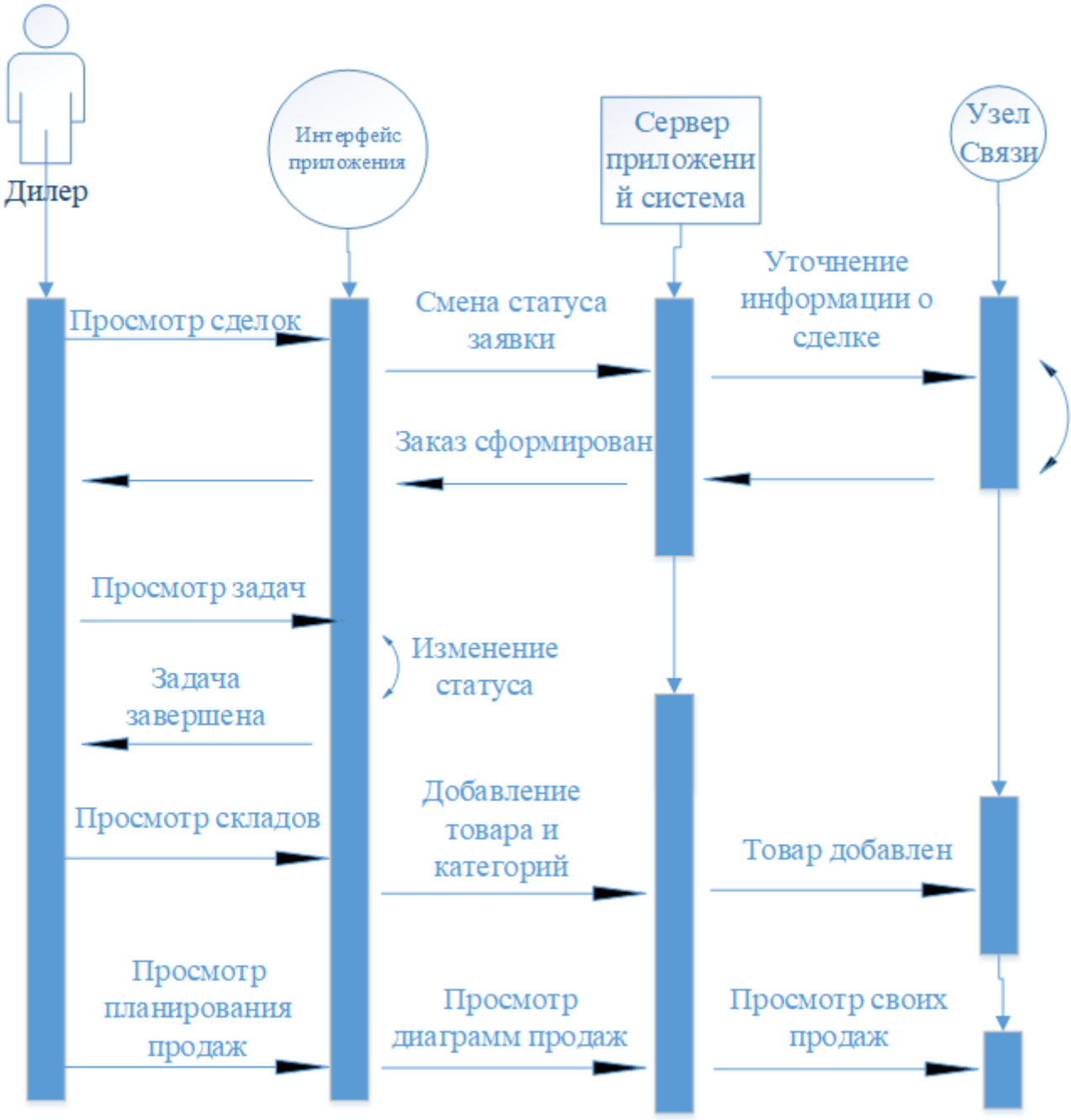


Рисунок 1.3.3 Диаграмма последовательности дилера

2. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

2.1 Методы и средства разработки

Выбирая между мобильным приложением и веб-приложением, выбор пал в сторону последнего, обуславливается данный выбор таким преимуществом, как предоставляют бизнесу и клиентам огромный спектр преимуществ, по сравнению с обычными сайтами. Решив запустить свой проект веб приложения, (в отличие от десктопного) можно сделать его доступным для массовой аудитории.

Веб приложения гораздо проще обновлять, чем их настольные аналоги. Чтобы обновить веб-приложение, необходимо внести изменения только в одном месте: на сервере, на котором обрабатывается информация. Обновление программного обеспечения для компьютера является гораздо более сложной задачей: необходимо разработать, протестировать и, наконец, распространить обновление среди всех конечных пользователей.

По сравнению с обычными веб-сайтами, веб-приложение с большей вероятностью получает лучшие результаты на основе приема повторных посетителей. Логика проста: оно более интерактивно, чем сайт, поэтому пользователи с большей вероятностью вернутся, чтобы использовать его во второй раз.

Преимущества веб-приложения:

1. Установка веб-приложений дешевле и намного проще.
2. Обновление веб-приложений дешевле и намного проще
3. Веб-приложения более универсальны и практичны для конечного пользователя.
4. Веб-приложения облегчают организацию хранения данных.

Для разработки программного средства используются следующие программные средства:

HTML «язык гипертекстовой разметки» — стандартизированный язык разметки веб-страниц во Всемирной паутине. Код HTML интерпретируется браузерами; полученная в результате интерпретации страница отображается на экране монитора компьютера или мобильного устройства.

CSS - это каскадные таблицы стилей. Язык, который отвечает за описание внешнего вида HTML-документа.

UML - унифицированный язык моделирования язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

BPwin - это программный продукт, разработанный компанией ltd. Logic Works. Он предназначен для поддержки процесса создания информационных систем. Относится к категории CASE средств верхнего уровня.

JavaScript - язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией спецификации ECMAScript.

Vue - это прогрессивный фреймворк для создания пользовательских интерфейсов. В отличие от фреймворков-монолитов, Vue создан пригодным для постепенного внедрения. Его ядро в первую очередь решает задачи уровня представления (view), что упрощает интеграцию с другими библиотеками и существующими проектами. С другой стороны, Vue полностью подходит и для создания сложных одностраничных приложений (SPA, Single-Page Applications).

Visio — векторный графический редактор, редактор диаграмм и блок-схем для Windows.

Firebase - это облачная база данных, которая позволяет пользователям хранить и получать сохраненную информацию, хранит текстовые данные в JSON

формате и предоставляет удобные методы для чтения, обновления и извлечения данных.

2.2 Методы и средства разработки базы данных

В настоящее время на рынке программного обеспечения используются такие средства как для создания, управления, редактирования баз данных, как:

1. Oracle RDBMS (она же Oracle Database) на первом месте среди базы данных. Система популярна у разработчиков, проста в использовании, у нее понятная документация, поддержка длинных наименований, JSON, улучшенный тег списка и Oracle Cloud.

Достоинства

- Самые свежие инновации и впечатляющий функционал уже внедрены в этом продукте, поскольку компания Oracle стремится держать планку даже на фоне других разработчиков базы данных.
- Базы данных от Оракул является крайне надёжной, фактически это эталон надёжности среди подобных систем.

Недостатки

- Стоимость Oracle может оказаться непомерно высокой, особенно для небольших организаций.
- Система может потребовать значительных ресурсов уже сразу после установки, поэтому возможно потребуется модернизировать оборудование для внедрения Oracle.

2. MySQL

MySQL - одна из самых популярных баз данных для веб-приложений. Фактически, является стандартом de facto для веб-серверов, которые работают под управлением операционной системы Linx. MySQL - это бесплатный пакет программ, однако новые версии выходят постоянно, расширяя функционал и улучшая безопасность. Существуют специальные платные версии, предназначенные для коммерческого использования. В бесплатной версии наибольший упор делается на скорость и надежность, а не на

полноту функционала, который может стать и достоинством, и недостатком - в зависимости от области внедрения.

Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB. Продукт распространяется как под GNU General Public License, так и под собственной коммерческой лицензией. Помимо этого, разработчики создают функциональность по заказу лицензионных пользователей. Именно благодаря такому заказу почти в самых ранних версиях появился механизм репликации.

Эта база данных позволяет выбирать различные движки для системы хранения, которые позволяют менять функционал инструмента и выполнять обработку данных, хранящихся в различных типах таблиц. Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Более того, база данных MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых типов таблиц. Благодаря открытой архитектуре и GPL-лицензированию, в база данных MySQL постоянно появляются новые типы таблиц. Она также имеет простой в использовании интерфейс, и пакетные команды, которые позволяют удобно обрабатывать огромные объемы данных. Система невероятно надежна и не стремится подчинить себе все доступные аппаратные ресурсы.

Достоинства

- Распространяется бесплатно
- Прекрасно документирована
- Предлагает много функций, даже в бесплатной версии
- Пакет MySQL включен в стандартные репозитории наиболее распространённых дистрибутивов операционной системы Linux, что позволяет устанавливать её элементарно

- Поддерживает набор пользовательских интерфейсов
- Может работать с другими базами данных, включая DB2 и Oracle.

Недостатки

- Придётся потратить много времени и усилий, чтобы заставить MySQL выполнять несложные задачи, хотя другие системы делают это автоматически, например, создавать инкрементные резервные копии.
- Отсутствует встроенная поддержка XML или OLAP.
- Для бесплатной версии доступна только платная поддержка.

3. MongoDB

Еще одна бесплатная база данных, которая имеет коммерческую версию - MongoDB, она предназначена для приложений, которые используют как структурированные, так и неструктурированные данные. Ядро является очень гибким и работает при подключении базы данных к приложениям через драйверы MongoDB. Существует широкий выбор доступных драйверов, поэтому легко найти драйвер, который будет работать с требуемым языком программирования.

Поскольку изначально система MongoDB не была разработана для обработки моделей реляционных данных (хотя может это выполнять), могут возникнуть проблемы производительности, если вы попытаетесь использовать её таким образом. Однако, движок предназначен для обработки различных данных, которые нельзя отнести к реляционным, и может хорошо справляться там, где другие движки работают медленно или бессильны.

MongoDB 3.2 - это последняя версия, и она имеет новую подключаемую систему движков хранения. Документы могут быть проверены в процессе обновления или выполнения вставок, а функции текстового поиска были улучшены. Новая способность частичного индексирования может привести к более высокой производительности, уменьшая размер индексов.

Достоинства

- Скорость и простота в использовании
- Движок поддерживает json и другие традиционные документы NoSQL.

- Данные любой структуры могут быть сохранены/прочитаны быстро и легко.

Недостатки

- SQL не используется в качестве языка запросов.
- Инструменты для перевода SQL-запросов в MongoDB доступны, но их следует рассматривать именно как дополнение.
- Программа установки может занять много времени.

4. PostgreSQL

PostgreSQL является одним из нескольких бесплатных популярных вариантов базы данных, часто используется для ведения баз данных веб-сайтов. Это была одна из первых разработанных систем управления базами данных, поэтому в настоящее время она хорошо развита, и позволяет пользователям управлять как структурированными, так и неструктурированными данными. Может быть использован на большинстве основных платформ, включая Linux. Прекрасно справляется с задачами импорта информации из других типов баз данных с помощью собственного инструментария.

Движок БД может быть размещен в ряде сред, в том числе виртуальных, физических и облачных. Самая свежая версия, PostgreSQL 9.5, предлагает обработку больших объемов данных и увеличение числа одновременно работающих пользователей. Безопасность была улучшена благодаря поддержке DBMS_SESSION.

Достоинства

- Является масштабируемым и способен обрабатывать терабайты данных.
- Поддерживает формат json.
- Существует множество предопределенных функций.
- Доступен ряд интерфейсов.

Недостатки

- Документация туманна, поэтому, возможно, ответы на некоторые вопросы придется искать в интернете.
- Конфигурация может смутить неподготовленного пользователя.
- Скорость работы может падать во время проведения пакетных операций или выполнения запросов чтения.

5. DB2

Созданная компанией IBM, DB2 представляет собой базы данных, которая имеет возможности NoSQL, и может читать JSON и XML-файлы. Ввиду того, что система разрабатывалась для серверов компании IBM модельного ряда iSeries, логично, что система работает на Windows, Linux и Unix.

Диалект языка SQL, используемый в DB2 за редкими исключениями строго декларативен, система снабжена многофазовым оптимизатором, строящим по этим декларативным конструкциям план выполнения запроса. В диалекте SQL DB2 отсутствуют подсказки оптимизатору, мало развит (а долгое время вообще отсутствовал) язык хранимых процедур, и, таким образом, всё направлено на поддержание декларативного стиля написания запросов. Язык SQL DB2 при этом является вычислительно полным, то есть потенциально позволяет в декларативной форме определять любые вычислимые соответствия между исходными данными и результатом. Это достигается в том числе за счёт использования табличных выражений, рекурсии и других развитых механизмов манипулирования данными.

Оптимизатор DB2 широко использует статистику распределения данных в таблицах (если процесс её сбора был выполнен администратором базы данных), поэтому один и тот же запрос на языке SQL может быть оттранслирован в совершенно различные планы выполнения в зависимости от статистических характеристик данных, которые он обрабатывает.

В рамках концепции повышения уровня интеграции средств безопасности в компьютерной системе, DB2 не имеет собственных средств аутентификации пользователей, интегрируясь со средствами операционной системы или

специализированными серверами безопасности. В рамках DB2 осуществляется только авторизация пользователей, аутентифицированных системой.

DB2 является единственной реляционной базы данных общего назначения, имеющей реализации на аппаратно-программном уровне (система IBM i; также в оборудовании мэйнфреймов IBM System z реализуются средства поддержки DB2).

Современные версии DB2 обеспечивают расширенную поддержку использования данных в формате XML, в том числе операции с отдельными элементами документов XML.

Текущая версия DB2 - это LUW 11.1, которая предлагает разнообразные улучшения и доработки. Одно из них, ускорение Blu , которое предназначено, для того чтобы сделать эту базу данных быстрее. Пропуск данных предназначен для повышения быстродействия системы с большим количеством данных, чем может она может вместить в себя. Последняя версия DB2 также обеспечивает усовершенствованные функции аварийного восстановления, совместимости и аналитики.

Достоинства

- Blu Acceleration позволяет грамотно задействовать ресурсы для объёмных баз данных.
- Может быть размещена в облачном хранилище, на физическом сервере, или же и там, и там одновременно.
- Несколько задач могут выполняться одновременно с помощью планировщика задач.
- Коды ошибок и коды завершения позволяют легко отследить, какие задания выполняются или выполнились с помощью планировщика задач.

Недостатки

- Цена за пределами бюджета многих физических лиц и небольших организаций.

- Сторонние приложения или дополнительное программное обеспечение требуется, для того чтобы заставить функционировать кластеры или несколько вторичных узлов.
- Базовая поддержка доступна только в течение трех лет; после этого, вы должны заплатить за это.

Таблица 2.2.1 Характеристика систем управления базами данных

Наименование	Oracle	MySQL	MongoDB	PostgreSQL	Firebase
Разработчик	Американская корпорация	Корпорация Oracle	Компания NASDAQ с 2017 года	Сообщество PostgreSQL	Американская компания.
Стоимость	Бесплатный доступ к использованию	Использование платное в зависимости от объема памяти	Бесплатный доступ к использованию	Бесплатный доступ к использованию	Бесплатный доступ к использованию
Оперативность	Функционирование системы на большинстве платформ	Гибкая и несложная в использовании	Удобна в использовании, используют когда нужна масштабируемая база данных	Поддерживает сложные структуры и широкий спектр встроенных и определяемых пользователем типов данных	Изменяется и синхронизируется в реальном времени и хранит данные в JSON. Имеет аутентификацию, и все данные защищаются через соединение SSL.

Администрирование базами данных предусматривает выполнение функций, направленных на обеспечение надежного и эффективного функционирования системы баз данных, адекватности содержания базы данных информационным

потребностям пользователей, отображения в базе данных актуального состояния предметной области.

Нормализация базы данных - это процесс структурирования базы данных, обычно реляционной базы данных, в соответствии с серией так называемых нормальных форм для уменьшения избыточности данных и улучшения целостности данных. Впервые он был предложен Эдгаром Ф. Коддом как часть его реляционной модели. Нормализация влечет за собой организацию столбцов (атрибутов) и таблиц (отношений) базы данных, чтобы гарантировать, что их зависимости должным образом подкрепляются ограничениями целостности базы данных. Это достигается путем применения некоторых формальных правил либо путем синтеза (создание нового проекта базы данных), либо декомпозиции (улучшения существующего проекта базы данных).

Основная цель базы данных - предоставить возможность оперативного поиска необходимой информации. База данных должна удовлетворять информационные потребности и содержать в себе информацию.

Администратор базы данных – это специалист, который отвечает за разработку требований к различным базам данных организации. Он занимается проектированием, эффективным использованием, поддержанием целостности и сопровождением работы хранилища. Администратор управляет записями учетного типа и организует систему защиты от несанкционированного использования находящихся в базе сведений.

Основными задачами администратора по базам данных является обеспечение бесперебойной работы всего оборудования, находящегося в организации (сетей, серверов и прочей электроники). Деятельность специалиста включает выполнение определенных алгоритмов для переработки и распределения всего объема информации на предприятии (ее обслуживание и диспетчеризацию), что позволит непрерывно извлекать и пользоваться при необходимости нужными сведениями.

Операции над данными – переводят базу данных из одного допустимого состояния в другое.

Основные операции:

- Идентификация данного и нахождение его позиции в БД;
- Выборка (чтения) данных;
- Добавление новых данных (запись);
- Удаление данных;
- Модификация данных.

3. ПРОЕКТНАЯ ЧАСТЬ

3.1 Организация хранения данных

Таблица 1 catalog необходима для хранения товаров на складе

Имя	Назначение	Тип данных
Id	Идентифицированный номер товара	Number
category	Отображает категорию товара	Text
cost	Цена товара за 1 условную единицу	Number
key	Идентифицированный ключ товара на складе	Number
num	Кол-во товара в наличии	Number
photoName	Фотография товара	img
title	Название товара	Text
type	Единица измерения товара в миллилитрах, литрах, тоннах	Text

Таблица 2 users необходима для хранения работников

Имя	Назначение	Тип данных
Id	Идентифицированный номер работника	Number
email	Отображает логин и электронную почту работника	Text
fio	ФИО работника	Text
img	Изображение работника	img
phonehome	Личный номер телефона работника	Number

Продолжение таблицы 2 users

phonework	Рабочий телефон работника	Number
position	Должность работника	Text

Таблица 3 CompetedDeals необходима для хранения выполненных сделок

Имя	Назначение	Тип данных
Id	Идентифицированный номер сделки	Number
address	Отображает адрес заказчика	Text
customer	Заказчик	Text
date	Дата создания сделки	Date
key	Идентифицированный ключ сделки	Number
num	Номер заказа	Number
profit	Прибыль сделки	Number
status	Статус выполненной сделки	Text

Таблица 4 Subject необходима для предмета договора заказчиками

Имя	Назначение	Тип данных
Id	Идентифицированный номер закрытой сделки дилером	Number
key	Идентифицированный ключ сделки	Number
num	Количество выбранного товара	Number
sum	Сумма количества товара	Number
Title	Названия предметов договора	Text
type	Тип сделки	Text

Таблица 5 Transactions необходима для хранения созданных сделок менеджерами

Имя	Назначение	Тип данных
Id	Идентифицированный номер сделки	Number
address	Отображает адрес заказчика	Text
customer	Заказчик	Text
date	Дата создания сделки	Date
key	Идентифицированный ключ сделки	Number
num	Номер заказа	Number
status	Выбор статус заказа	Text
title	Название сделки	Text
type	Тип сделки	Text

Таблица 6 Task необходима для хранения созданных задач менеджерами

Имя	Назначение	Тип данных
Id	Идентифицированный номер задачи	Number
contact	Отображает контакты для связи с заказчиком	Text
date	Дата задачи	Date
description	Описание для выполнения задачи	Text
title	Название задачи	Text

Таблица 7 Reports необходима для хранения закрытых отчётов, поданных дилерами

Имя	Назначение	Тип данных
Id	Идентифицированный номер отчёта	Number
date	Дата создания сделки	Date
key	Идентифицированный ключ сделки	Number
profit	Прибыль сделки	Number
Title	Название закрытого отчёта	Text
Type	Тип выполненного отчёта (неделя, месяц, год)	Text
worker	Уникальный номер дилера	Number

Таблица 8 planing необходима для хранения планирования продаж

Имя	Назначение	Тип данных
Id	Идентифицированный номер планирования	Number
Type	Тип планирования (Год, месяц, неделя)	Text
Target	Сумма цели на продажи	Number

3.2 Программная реализация функциональных возможностей

После входа в веб-приложение проходим авторизацию

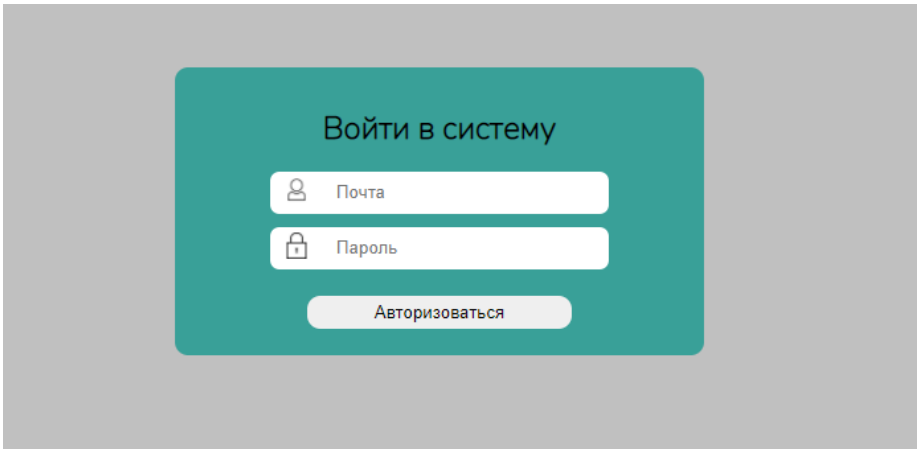


Рисунок 3.2.1 Окно авторизации работника

```
<template>
<layout-login>
<div class="">
<Loader v-if="loading"/>
<div class="auth" v-else>
<h2>Войти в систему</h2>
<form @submit.prevent="login">
<div>
<img src="../assets/user.png" alt="1" />
<input type="email" placeholder="Почта"  required v-model.trim="email"/>
</div>
<div>
<img src="../assets/pass.png" alt="2" />
<input type="text" placeholder="Пароль" required v-model.trim="password"/>
</div>
<button type="submit">Авторизоваться</button>
<br>
<small class="error" :class="{'display':error}">Неверный логин или пароль</small>
</form>
</div>
</div>
</layout-login>
</template>
<script>
import LayoutLogin from "../layouts/LayoutLogin.vue"
import Loader from '../components/Loader'
export default {
name: "Login",
```

```
data: ()=>({
email: '',
password: '',
error: false,
loading: false,
}),
components: {Loader, LayoutLogin},
mounted() {
if(this.$store.getters.isLogin){
this.$router.push('/')
}
},
methods:{
async login(){
this.loading = true
let formData = {
email: this.email,
password: this.password
}
try {
await this.$store.dispatch("login", formData)
await this.$router.push("/")
} catch (e) {
this.error = true
}
this.email = ''
this.password = ''
this.loading = false
},
```

```
}
};
</script>
actions: {
async login({dispatch, commit}, {email, password}) {
try {
await firebase.auth().signInWithEmailAndPassword(email, password)
commit('setUser', firebase.auth().currentUser)
await dispatch('fetchUserData')
} catch (e) {
throw e
}
},
async fetchUserData({commit, getters}) {
try {
const uid = getters.getUid
if (uid === null) {
throw 'error'
}
const userData = (
await firebase
.database()
.ref(`/users/${uid}`)
.once('value')
).val()
commit('setUserData', userData)
return userData
} catch (e) {
throw e
```

}

},

После того как прошли процедуру авторизации менеджера, имеются следующие возможности такие как: создать сделку, добавить задачу, посмотреть сделки, изменить статус сделки, добавлять продукцию на склад, просмотр сформированных отчётов дилерами, распечатать отчёт сделки, установить цель планирования продаж.

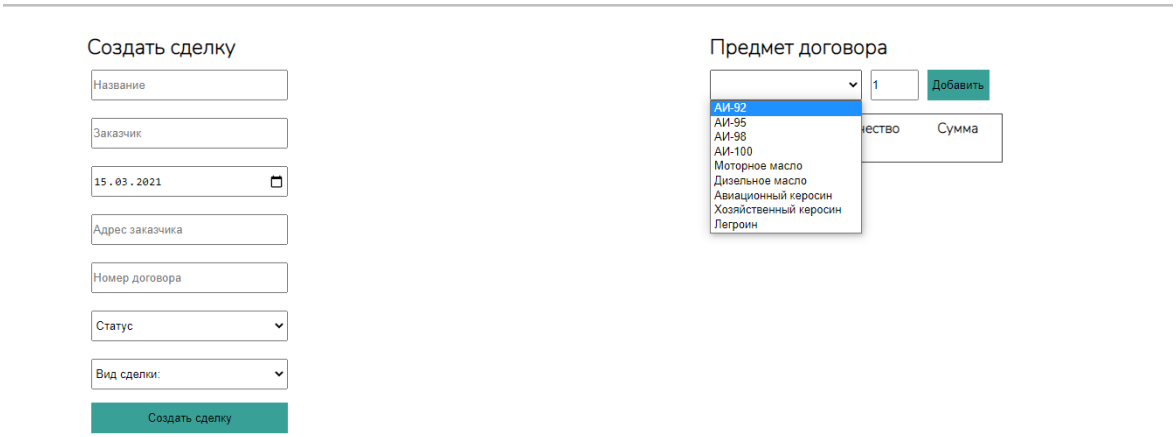


Рисунок 3.2.2 Окно создания сделки менеджером

```
<template>
 <main-layout>
<Loader v-if="loading"/>
<div class="addtransaction" v-else>
<div>
<form @submit.prevent="submitHandler">
<h2>Создать сделку</h2>
<input
placeholder="Название"
required
type="text"
v-model="title"
/><br/>
```

```
<input
placeholder="Заказчик"
required
type="text"
v-model="customer"
/><br/>
<input id="date" required type="date" v-model="date"/><br/>
<input
placeholder="Адрес заказчика"
required
type="text"
v-model="address"
/><br/>
<input
placeholder="Номер договора"
required
type="text"
v-model="num"
/><br/>
<select required v-model="status">
<option disabled selected value="">Статус</option>
<option value="1">В обработке</option>
<option value="2">В ожидании</option>
<option value="3">Выполнен</option>
<option value="4">Отклонен</option>
</select
><br/>
<select required v-model="type">
<option disabled selected value="">Вид сделки:</option>
```

```
<option value="оптовая">Оптовая</option>
<option value="розничная">Розничная</option>
</select
><br/>
<button type="submit">Создать сделку</button>
</form>
</div>
<Subjects :catalog="catalog" :subjs="subjects" @subjects="subject"/>
</div>
</main-layout>
</template>
<script>
import MainLayout from '../layouts/LayoutMain'
import Loader from '../components/Loader'
import Subjects from '../components/transaction/Subjects'
export default {
name: 'AddTransaction',
components: {Subjects, Loader, MainLayout},
data() {
return {
title: '',
date: '',
address: '',
status: '',
type: '',
customer: '',
num: '',
subjects: [],
catalog: [],
```

```
loading: true,
}
},
async mounted() {
await this.$store.dispatch('fetchCatalog')
this.catalog = this.$store.getters.getCatalog
this.date = this.getDate
this.loading = false
},
methods: {
async submitHandler() {
this.loading = true
const data = {
title: this.title,
date: this.date,
address: this.address,
status: this.status,
type: this.type,
customer: this.customer,
num: this.num,
subjects: this.subjects,
}
if(this.subjects.length){
try {
await this.$store.dispatch('createTransaction', data)
for (const subj of this.subjects) {
const num = +this.catalog[subj.itemKey].num - +subj.num
await this.$store.dispatch('changeCatalogNum', {key: subj.itemKey, num })
}
```

```
this.$toast.success('Сделка создана')
} catch (e) {
this.$toast.error(e.message)
console.error(e)
} finally {
this.title = this.address = this.status = this.type = this.customer = this.num = ''
this.date = this.getDate
this.subjects = []
this.loading = false
}
}
},
subject(subjects) {
this.subjects = subjects
}
},
computed: {
getDate() {
let dd = (new Date).getDate()
if (dd < 10) dd = '0' + dd
let mm = (new Date).getMonth() + 1
if (mm < 10) mm = '0' + mm
let yy = (new Date).getFullYear()
if (yy < 10) yy = '0' + yy

return yy + '-' + mm + '-' + dd
}
}
}
```

</script>

Создание задачи менеджером

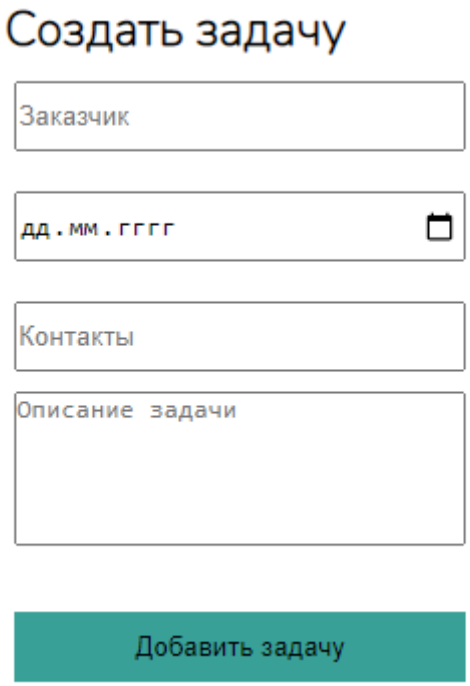


Рисунок 3.2.3 Окно создания задачи менеджером

<template>

<main-layout>

<Loader v-if="loading" />

<div class="addtask" v-else>

<form @submit.prevent="submitHandler">

<h2>Создать задачу</h2>

<input

placeholder="Заказчик"

required

type="text"

v-model="title"

/>

<input id="date" required type="date" v-model="date" />

<input

```
placeholder="Контакты"
required
type="text"
v-model="contact"
/><br />
<textarea
name=""
id=""
cols="30"
rows="5"
placeholder="Описание задачи"
v-model="description"
></textarea
><br />
<br />
<button type="submit">Добавить задачу</button>
</form>
</div>
  </main-layout>
</template>
<script>
import MainLayout from "../layouts/LayoutMain";
import Loader from "../components/Loader";
export default {
name: "AddTask",
components: { Loader, MainLayout },
data: () => ({
title: "",
date: "",
```

```
contact: "",
description: "",
loading: false,
}),
methods: {
async submitHandler() {
this.loading = true;
const data = {
title: this.title,
date: this.date,
contact: this.contact,
description: this.description,
};
try {
await this.$store.dispatch("createTask", data);
this.title = this.date = this.contact = this.description = "";
this.$toast.success("Сохранено");
} catch (e) {
this.$toast.error(e.message);
console.error(e);
} finally {
this.loading = false;
}
},
},
};
</script>
```

Смена статуса на сделке и отправка на сервер к завершённым сделкам

Рисунок 3.2.4 Смена статуса сделки и отправка на сервер менеджером

```
<template>
<layout-main>
<Loader v-if="loading"/>
<div class="deal" v-else>
<Dealitem  @statusChange="statusChange" :deal="deal"/>
<Dealpred  :subjects="deal.subjects"/>
</div>
</layout-main>
</template>
<script>
import LayoutMain from '../layouts/LayoutMain.vue'
import Dealitem from '../components/deal/Dealitem.vue'
import Dealpred from '../components/deal/Dealpred.vue'
import Loader from '../components/Loader'
export default {
name: 'Deal',
data(){
return {
loading: true,
key: this.$route.params.key,
deal: null
```

```
}
},
components: {Loader, LayoutMain, Dealitem, Dealpred},
methods:{
async statusChange(deal){
try{
await this.$store.dispatch('changeDealStatus', deal)
if(+deal.status === 3){
let profit = 0
this.deal.subjects.forEach(subj => profit += subj.sum)
await this.$store.dispatch('completeDeal', {...deal, profit, weekday: (new Date()).toLocaleString('ru', { weekday:'long'}),})
}
this.$toast.success('Сохранено')
} catch (e) {
this.$toast.error(e.message)
}
}
},
async mounted(){
await this.$store.dispatch('fetchDeal', this.key)
this.deal = this.$store.getters.getDeal
this.loading =  false
},
}
</script>
```

Добавление продукции на склад, и отправка на сервер.

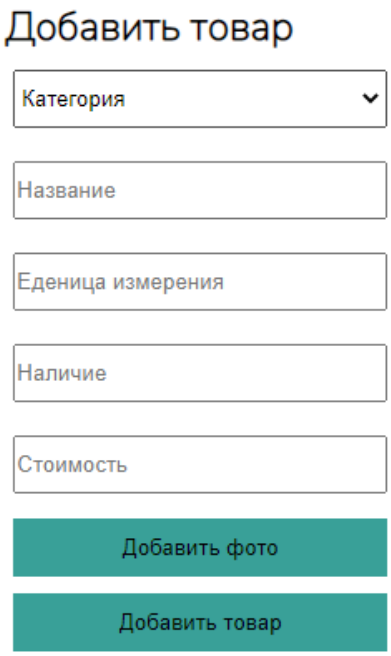


Рисунок 3.2.5 Окно добавление продукции на склад и отправка на сервер менеджером

```
<template>
<main-layout>
<Loader v-if="loading"/>
<div class="addItem" v-else>
<form @submit.prevent="submitHandler">
<h2>Добавить товар</h2>
<select required v-model="category">
<option disabled selected value="null">Категория</option>
<option value="Бензин">Бензин</option>
 <option value="Нефтяные масла">Нефтяные масла</option>
<option value="Керосин">Керосин</option>
<option value="Лигроин">Лигроин</option>
</select><br/>
<input
placeholder="Название"
required
```

```
type="text"
v-model="title"
/><br/>
<input
placeholder="Еденица измерения"
required
type="text"
v-model="type"
/><br/>
<input
placeholder="Наличие"
required
type="number"
v-model="num"
/><br/>
<input
placeholder="Стоимость"
required
type="number"
v-model="cost"
/><br/>
<input @change="loadPhoto" accept=".jpg,.png,.bmp,.jpeg" ref="file" required style="display:none" type="file"/>
<button @click="addPhoto">Добавить фото</button>
<button type="submit">Добавить товар</button>
<br/>
</form>
<img ref="img"/>
</div>
```

```
</main-layout>
</template>
<script>
import MainLayout from '../layouts/LayoutMain'
import Loader from '../components/Loader'
export default {
name: 'AddItem',
components: {Loader, MainLayout},
data: () => ({
category: null,
title: '',
num: '',
cost: '',
loading: false,
}),
methods: {
async submitHandler() {
this.loading = true
const data = {
category: this.category,
title: this.title,
num: this.num,
cost: this.cost,
type: this.type,
photoName: this.$refs.file.files[0].name,
}
try {
await this.$store.dispatch('saveProductImg', this.$refs.file.files[0])
await this.$store.dispatch('createProduct', data)
```

```
this.category = 1
this.title = this.num = this.cost = this.type = ''
this.$toast.success('Сохранено')
} catch (e) {
this.$toast.error(e.message)
console.error(e)
} finally {
this.loading = false
}
},
addPhoto() {
this.$refs.file.click()
},
loadPhoto(e) {
const reader = new FileReader()
reader.onload = (ev) => {
this.$refs.img.src = ev.target.result
}
reader.readAsDataURL(e.srcElement.files[0])
}
},
}
</script>
```

3.3 Результаты тестирования информационной системы

Тестирование программного обеспечения – это:

– процесс исследования ПО с целью получения информации о качестве продукта;

– процесс проверки соответствия заявленных к продукту требований и реально реализованной функциональности, осуществляемый путем наблюдения за его работой в искусственно созданных ситуациях и на ограниченном наборе тестов, выбранных определенным образом;

– оценка системы с тем, чтобы найти различия между тем, какой система должна быть и какой она есть.

В широком смысле, тестирование – это одна из техник контроля качества (Quality Control), которая включает планирование, составление тестов, непосредственно выполнение тестирования и анализ полученных результатов.

Важно понимать, что тестирование ПО включает не только собственно проведение тестов, но и многие другие действия, связанные с процессом обеспечения качества:

– анализ и планирование;

– разработку тестовых сценариев;

– оценку критериев окончания тестирования;

– написание отчетов;

– рецензирование документации (в том числе и исходного кода);

– проведение статического анализа.

Тестирование необходимо потому, что все мы совершаем ошибки. Некоторые из них могут быть незначительными, в то время как другие – иметь самые разрушительные последствия. Все, что производится человеком, может содержать ошибки (так уж мы, люди, устроены). Именно поэтому любой продукт нуждается в проверке – тестировании, прежде чем его можно будет эффективно и безопасно использовать. Для проверки корректной работы методов созданного программного продукта создано и реализовано 6 юнит-тестов.

```
PASS tests/unit/example.spec.js
  AddItem.vue
    √ item added successful (2ms)
  AddTask.vue
    √ task added successful
  Login.vue
    √ login
  Profile.vue
    √ renders profile
  Planing.vue
    √ fetch plan
  Storage.vue
    √ fetch storage

Test Suites: 1 passed, 1 total
Tests:       6 passed, 6 total
Snapshots:   0 total
Time:        3.195s
Ran all test suites.
```

Рисунок 3.3.1 Результат юнит-тестов программного продукта

3.4 Модель процесса эксплуатации программного продукта

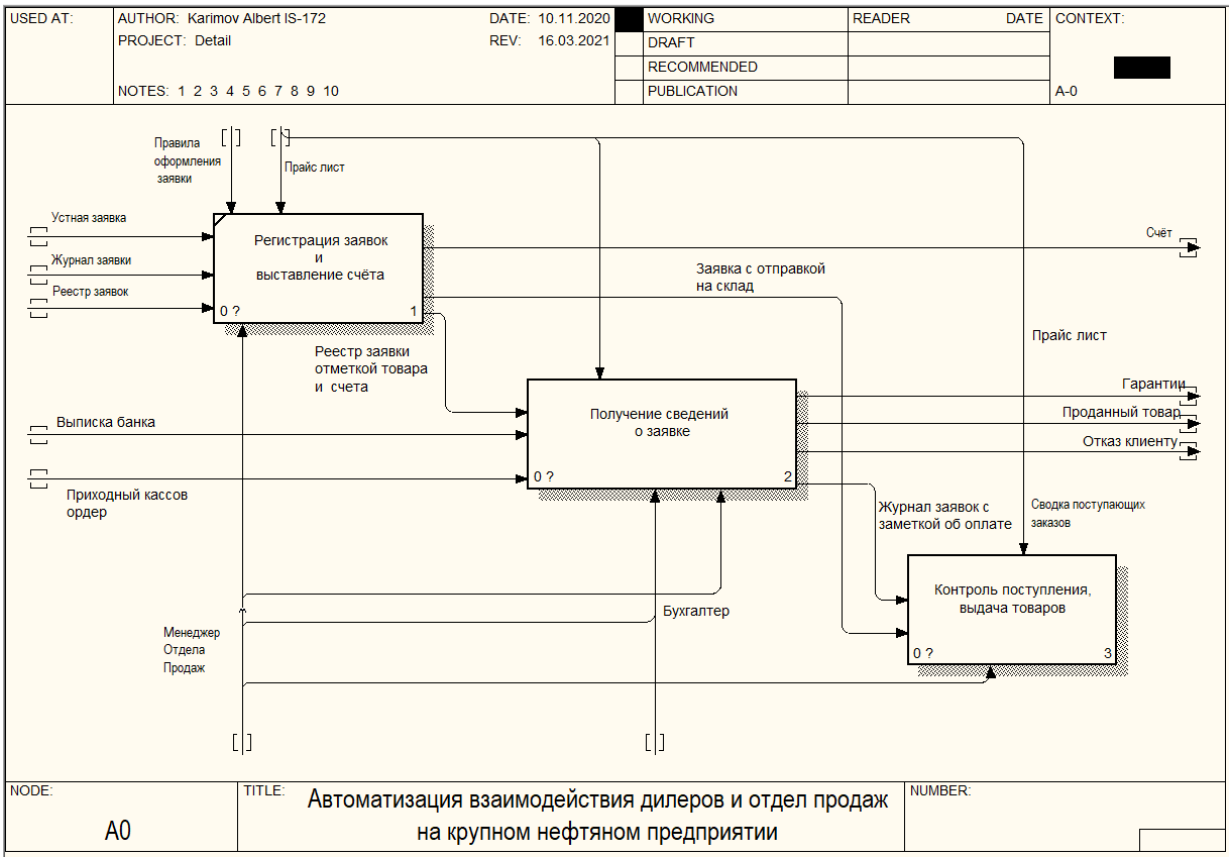


Рисунок 3.4.1 Диаграмма IDEF0 взаимодействия информационной системы с другими объектами

3.5 Политика информационной безопасности

Этот раздел показывает совокупность правил, процедур, практических методов и принципов, для обеспечения конфиденциальности и сохранности информации.

Виды информационной безопасности:

1. Системы обнаружения и предотвращения вторжений
2. Защита корпоративной сети
3. Защита корпоративной электронной почты
4. Антивирусная защита
5. Защита от DDoS
6. Защита от утечки информации
7. Защита персональных данных

Персональные данные или личностные данные — сведения, относящиеся к прямо или косвенно определённому или определяемому физическому лицу (субъекту персональных данных), которые могут быть предоставлены другим лицам

Разграничение доступа заключается в том, чтобы каждому зарегистрированному пользователю предоставить возможности беспрепятственного доступа к информации в пределах его полномочий и исключить возможности превышения этих полномочий.

Таблица 3.5.1 Характеристика доступа к информации

Роль	Характеристика
Admin	Все права и имеет доступ ко всем пользователям (Добавлять пользователей, менять должность работникам, просмотр веб-приложения)
SysAdmin	Выполнение действий на сервере (менять базу данных, корректировать таблицы базы данных, добавлять работников, удалять работников)
Менеджер отдела продаж	Имеет доступ к сделкам, делам сотрудников, участвует в документообороте (Создавать сделки, менять статус сделки,

	проверять отчёты сотрудников, устанавливать планирование продаж, добавлять задачи для сотрудников, просмотр завершенных сделок, просматривать продукцию на складе, добавлять продукцию на склад и менять её количество)
Дилер	Ведет личные дела со сделками (Просматривать сделки, менять статус сделок, просматривать продукцию на складе, добавлять продукцию на склад, отправлять отчёт, просматривать задачи, просматривать планирование продаж)

Авторизация — предоставление определённому лицу или группе лиц прав на выполнение определённых действий; а также процесс проверки данных прав при попытке выполнения этих действий.

Регистрация — запись, фиксация фактов или явлений с целью учёта и придания им статуса официально признанных актов (регистрация рождения или брака); внесение в список, в книгу учёта.

Аутентификация — процедура проверки подлинности, например, проверка подлинности пользователя путём сравнения введённого им пароля с паролем, сохранённым в базе данных пользовательских логинов; подтверждение подлинности электронного письма путём проверки цифровой подписи письма по открытому ключу отправителя;

Пример авторизации веб-приложения:

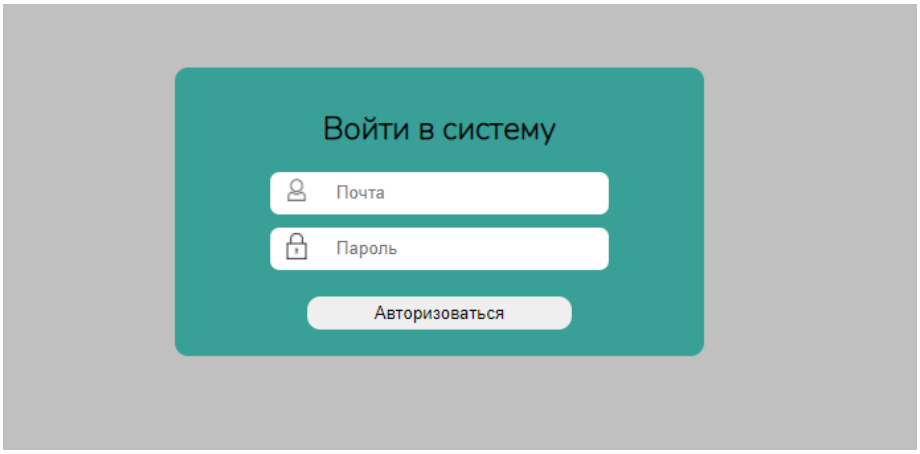


Рисунок 3.5.1 Окно авторизации работника

<template>

<layout-login>

```
<div class="">
<Loader v-if="loading"/>
<div class="auth" v-else>
<h2>Войти в систему</h2>
<form @submit.prevent="login">
<div>
<img src="../assets/user.png" alt="1" />
<input type="email" placeholder="Почта"  required v-model.trim="email"/>
</div>
<div>
<img src="../assets/pass.png" alt="2" />
<input type="text" placeholder="Пароль" required v-model.trim="password"/>
</div>
<button type="submit">Авторизоваться</button>
<br>
<small class="error" :class="{'display':error}">Неверный логин или пароль</small>
</form>
</div>
</div>
</layout-login>
</template>
<script>
import LayoutLogin from "../layouts/LayoutLogin.vue"
import Loader from '../components/Loader'
export default {
name: "Login",
data: ()=>({
email: '',
```

```
password: '',
error: false,
loading: false,
}),
components: {Loader, LayoutLogin},
mounted() {
if(this.$store.getters.isLogin){
this.$router.push('/')
}
},
methods:{
async login(){
this.loading = true
let formData = {
email: this.email,
password: this.password
}
try {
await this.$store.dispatch("login", formData)
await this.$router.push("/")
} catch (e) {
this.error = true
}
this.email = ''
this.password = ''
this.loading = false
},
}
};
```

```
</script>
actions: {
async login({dispatch, commit}, {email, password}) {
try {
await firebase.auth().signInWithEmailAndPassword(email, password)
commit('setUser', firebase.auth().currentUser)
await dispatch('fetchUserData')
} catch (e) {
throw e
}
},
async fetchUserData({commit, getters}) {
try {
const uid = getters.getUid
if (uid === null) {
throw 'error'
}
const userData = (
await firebase
.database()
.ref(`/users/${uid}`)
.once('value')
).val()
commit('setUserData', userData)
return userData
} catch (e) {
throw e
}
},
```

3.6 Развитие программного продукта

Диаграмма вариантов прецедентов – это тип поведенческой диаграммы UML, который часто используется для анализа различных систем. Они позволяют визуализировать различные типы ролей в системе и то, как эти роли взаимодействуют с системой. Это руководство по диаграмме вариантов использования охватывает следующие темы и поможет вам лучше создавать сценарии использования.

Как уже упоминалось ранее, диаграммы прецедентов используются для сбора требований к использованию системы. В зависимости от ваших требований можно использовать эти данные различными способами. Ниже приведены несколько способов их использования.

- Идентификация функций и как с ними взаимодействуют роли – основное назначение диаграмм сценариев использования.
- Для представления системы на высоком уровне – особенно полезно при представление руководителям или заинтересованным сторонам. Можно выделить роли, которые взаимодействуют с системой, и функциональные возможности, предоставляемые системой, не углубляясь во внутреннюю работу системы.
- Идентификация внутренних и внешних факторов – это может показаться простым, но в больших сложных проектах система может быть идентифицирована как внешняя роль в другом случае использования.

Разработанный программный продукт имеет основную структуру, поэтому подразумевает изменение как на стороне пользователя, так и на стороне администратора. Помимо этого, могут быть добавлены новые модули, которые позволят автоматизировать разные процессы для организации, уменьшая не только на отделы, но и на сотрудников. Первоочередные изменения отображены на диаграмме прецедентов, которые будут изменяться и добавляться новые автоматизированные модули в систему.

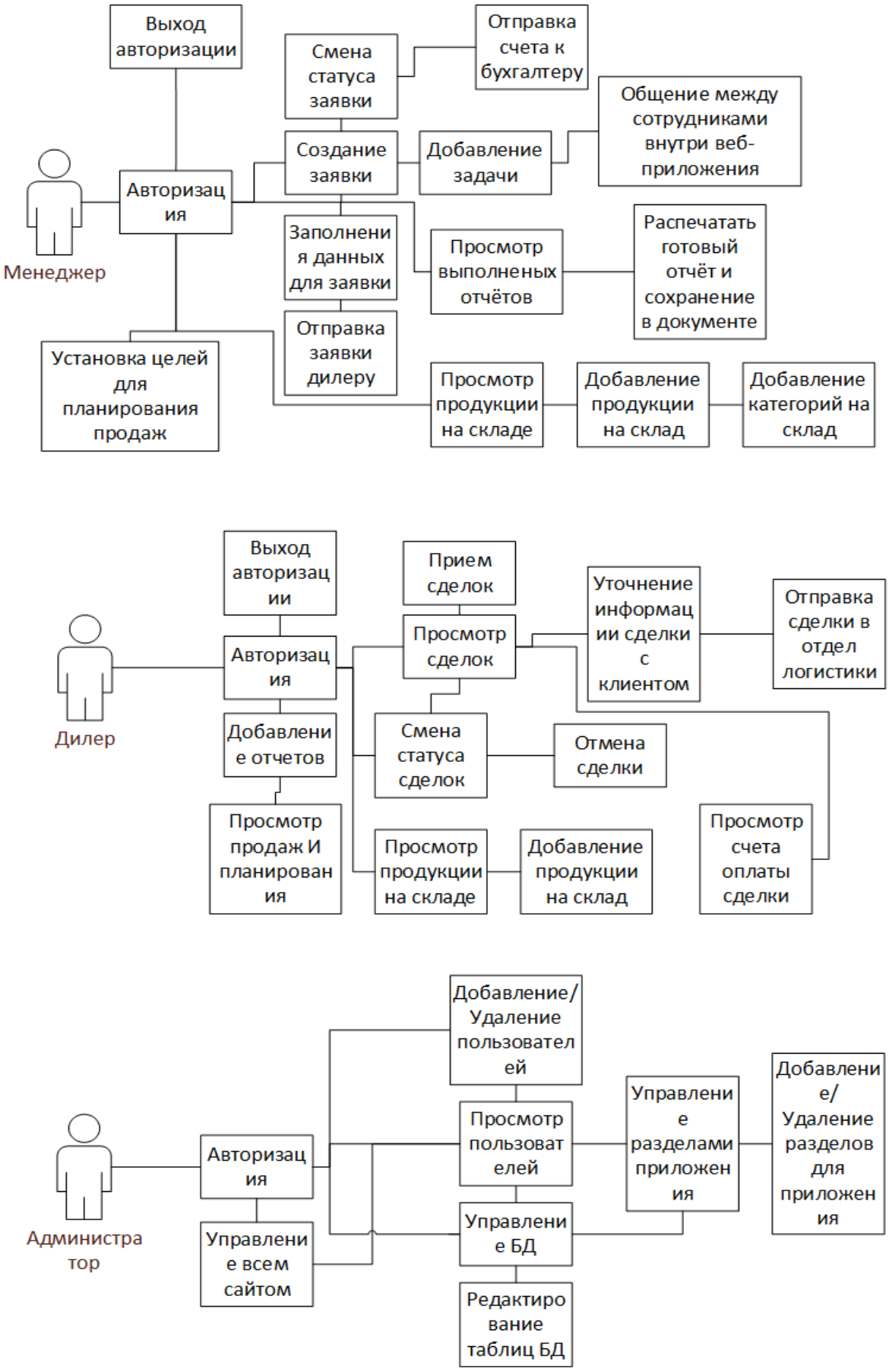


Рисунок 3.6.1 Диаграмма прецедентов

Согласно диаграмме прецедентов, возможные изменения могут быть следующими:

1. Добавление бухгалтерии в веб-приложение
2. Добавление окна для отправки в отдел логистики
3. Реализовать общение между сотрудниками внутри приложения
4. Добавление категорий на склад с продукцией
5. Просмотр счета по оплате в сделках
6. Доработка функциональности администратора
7. Добавление серверного рендеринга с использованием Vue SSR

Заключение

На основании полученных результатов были рассмотрены синтетический и аналитический анализ. Привлечение и удержание дилеров — одно из важнейших направлений работы производственно-торгового предприятия. Эффективная дилерская сеть — это стабильный доход компании и серьезное преимущество в глазах потребителей. Подводя итог проделанной работы, выполнены следующие цели:

1. Повышена эффективность принятия оперативных управленческих решений в части компании за счет предоставления консолидированной информации руководству компании.
2. Возможность получения более подробной информации о деятельности партнеров компании: структуре и объеме продаж всех товаров, особенностям работы, каналам сбыта, торговых представителях.
3. Контроль работы отделов и сотрудников компании, возможность видеть оперативную и объективную информацию по всем этапам работы с партнерами.
4. Создание конкурентного преимущества компании при работе с партнерами за счет удобного инструмента ведения бизнес деятельности.
5. Увеличение эффективности работы сотрудников компании.

В ходе проектирования были выполнены следующие задачи:

1. Разработана автоматизированная информационная система контролирующая отдел продаж и дилеров.
2. Разработан личный профиль работников
3. Разработана личная рабочая область для работников
4. Разработан существующий склад с продукцией
5. Разработано окно подачи отчёта
6. Разработано окно для просмотра планирования продаж
7. Разработано просмотр отчётов для менеджера по продажам
8. Разработано окно у менеджера для создания сделок

9. Разработано окно у менеджера для добавления задач
10. Разработано окно у менеджера для добавления продукции на склад
11. Разработано окно для установки целей планирования продаж на период продаж
12. Разработано окно для просмотра документооборота готовых продаж, выполненных дилерами

Таким образом можно сделать вывод, что разработанный продукт позволит предприятию не только увеличить объем продаж, но и сохранить позиции на местном рынке, что обеспечит стабильное развитие предприятия.

Литература

Учебная

1. HTML5 и CSS3. Разработка сайтов для любых браузеров и устройств. - СПб.: Питер, 2016. - 304 с.: ил.

2. HTML5, CSS3 и JavaScript. Исчерпывающее руководство/Дженнифер Роббинс; [пер. с англ. М. А. Райтман]. - 4-е издание. - М.: Эксмо, 2016. - 528 с.

3. HTML5. Разработка приложений для мобильных устройств. - СПб.: Питер, 2015. - 480 с.: ил.

4. Бардзелл, Джеффри MacromediaDreamweaver MX 2004 с ASP, ColdFusion и PHP. Из первых рук (+ CD-ROM) / Джеффри Бардзелл. - М.: Эком, 2017. - 560 с.

5. Бенкен, Е. PHP, MySQL, XML. Программирование для Интернета / Е. Бенкен. - М.: БХВ-Петербург, 2016. - 352 с.

6. Информационные системы предприятия: учеб. пособие / А.О. Варфоломеева, А.В. Коряковский, В.П. Романов. — 2-е изд., перераб. и доп. — М. : ИНФРА-М, 2019. — 330 с. — (Среднее профессиональное образование).

Интернет-ресурсы

7. Официальная документация по Node.js [Интернет ресурс] - URL: https://nodejs.org/en/docs/

8. Официальная документация по NPM [Интернет ресурс] - URL: https://docs.npmjs.com/

9. Официальная документация по Vue [Интернет ресурс] - URL: https://ru.vuejs.org/v2/guide/

10. Официальная документация по Vue cli [Интернет ресурс] - URL: https://cli.vuejs.org/ru/guide/

11. Официальная документация по Vue cli [Интернет ресурс] - URL: https://babeljs.io/docs/en/

12. Официальная документация по Vue router [Интернет ресурс] - URL: https://router.vuejs.org/ru/

13. Официальная документация по Vuex [Интернет ресурс] - URL: https://vuex.vuejs.org/ru/

14. Официальная документация по Web Pack [Интернет ресурс] - URL: https://webpack.js.org/

15. Официальные стандарты JSON [Интернет ресурс] - URL: https://www.json.org

16. Сайт веб-документации MDN [Интернет ресурс] - URL: https://developer.mozilla.org/ru/

17. Сайт веб-документации SCSS [Интернет ресурс] - URL: https://habr.com/ru/post/140612/

Приложение 1

Руководство пользователя

Целью работы является разработка веб-приложения «Автоматизация взаимодействия дилеров и отдел продаж в нефтяных предприятиях».

Для разработки программного обеспечения была собрана вся необходимая информация, тщательно изучены предметная область предприятия и инвентарная база.

Реализация данного веб-приложения включает в себя задачи:

- Проанализировать предметную область
- Реализовать управление проектом
- Определить объекты предметной области
- Выбрать методы и средства разработки
- Выбрать и обосновать методы и средства разработки базы данных
- Организовать хранение данных
- Реализовать профессиональные возможности информационной системы
- Протестировать информационные системы
- Определить модели эксплуатации программного продукта
- Реализовать информационную безопасность
- Спроектировать развитие программного продукта

Для работы с веб-приложением желательно использовать следующие характеристики:

ПК с операционной системой, Windows 7 и выше.

Браузер Google Chrome последней версии или Mozilla Firefox.

Подключение к корпоративной сети через Wi-Fi или Ethernet со скоростью не менее 2 мб/с.

Рекомендуемые требования

Для работы с веб-приложением рекомендуется использовать следующие характеристики:

ПК с операционной системой, Windows 7 и выше.

Браузер Google Chrome последней версии или Mozilla Firefox.

Подключение к корпоративной сети через Wi-Fi или Ethernet со скоростью не менее 5 мб/с.

Установка и запуск Web – приложения

- Установка приложения

Для работы с веб-приложением выполнения каких-либо установок не требуется, веб-приложение уже размещено на хостинге.

- Запуск приложения

Запустите браузер (Google Chrome или Yandex Browser)

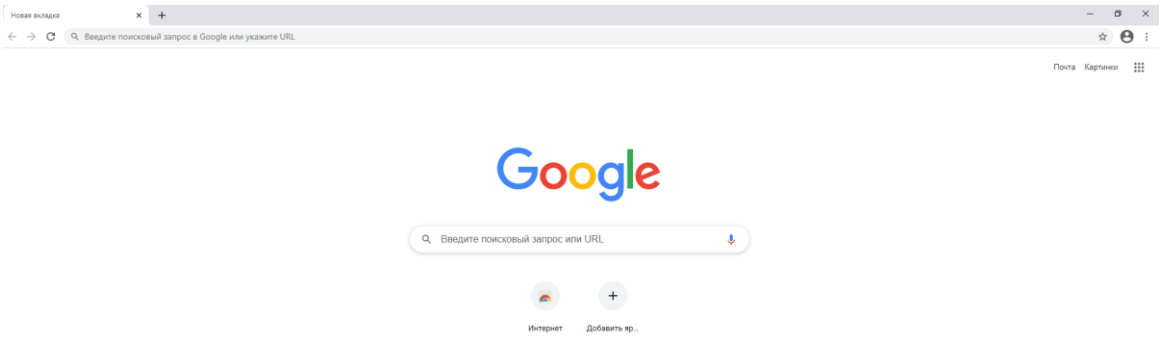


Рисунок 1 – «Запуск браузера»

Запуск веб приложения

Зайдем в веб-приложение по веб адресу: https://kpdp-db010.web.app/

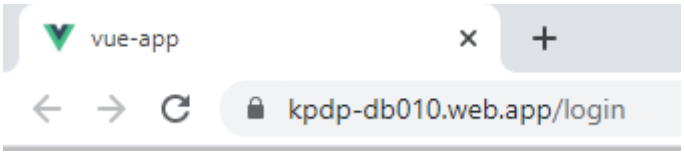


Рисунок 2 – «Веб-приложение»

Работа с веб-приложением

Для работы в веб-приложении необходимо авторизоваться (рис 2):

Введите логин

Введите пароль

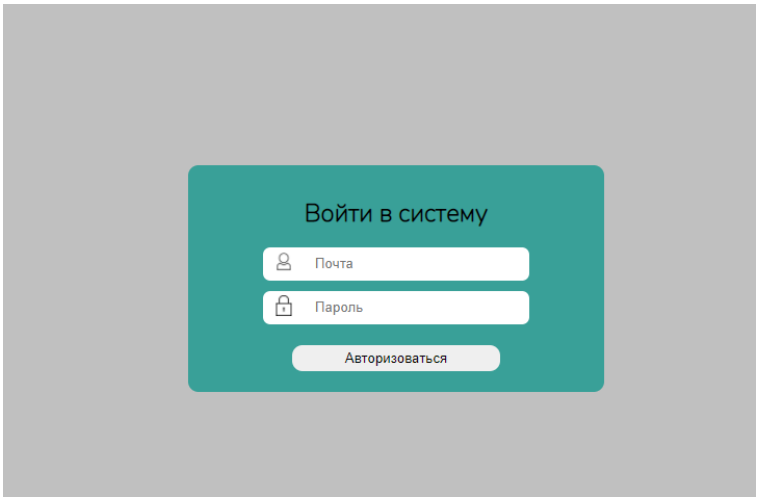


Рисунок 3 – «Авторизация»

Личный профиль»

Зайдем под пользователем “Дилер” и попадем в личный профиль пользователя (рис 3).

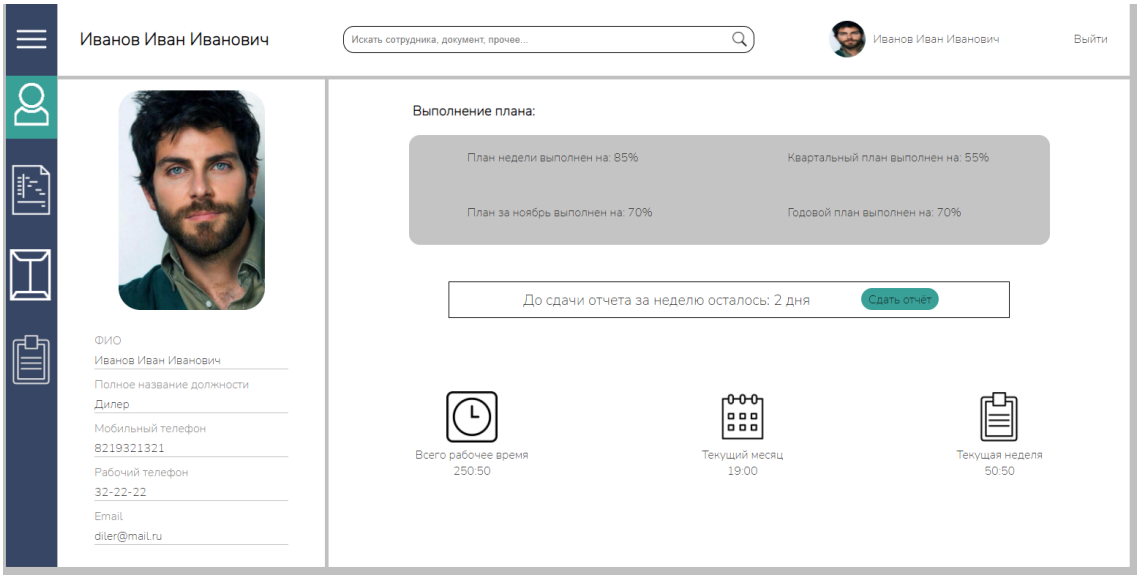


Рисунок 4 «Личный профиль»

Рабочая область дилера

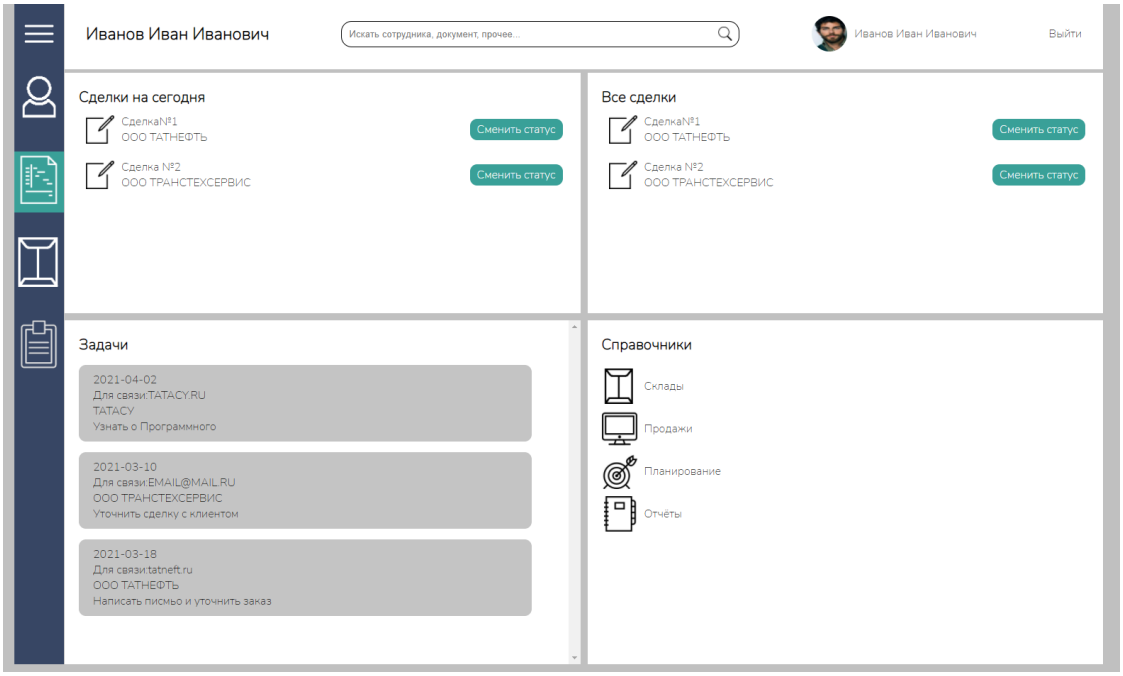


Рисунок 5– «Рабочая область»

Все активные сделки

Для открытия сделки следует нажать на ячейку сделки, которую вы хотите открыть как показано на рисунке 5.

Рисунок 6 – «Активная сделка»

Далее перешли на сделку, мы можем изменить статус заказа, можем посмотреть всю необходимую информацию о сделке, как показано на рисунке 6.

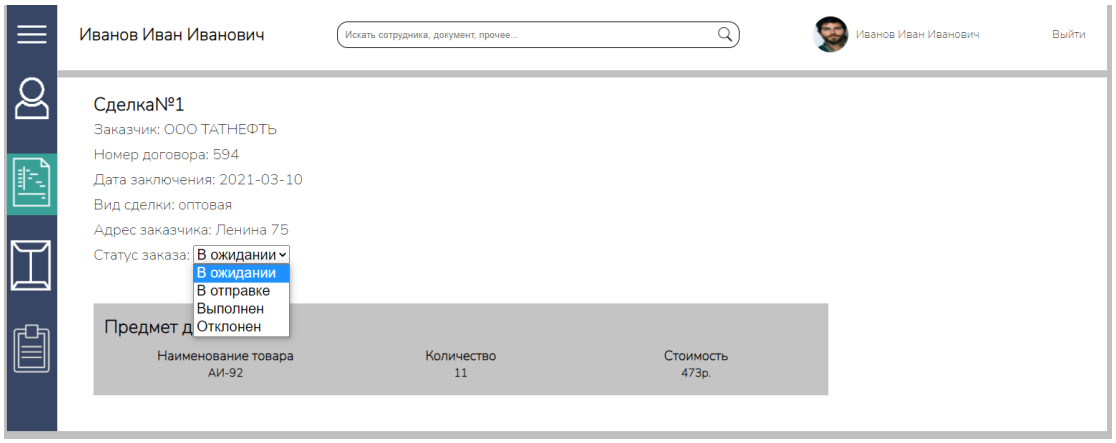


Рисунок 7 – «Информация о сделке»

Работа с панелью менеджера

В первую очередь для начала работы с панелью администратора следует авторизоваться как менеджер. Далее мы видим рабочую область менеджера, мы можем создать сделку, добавить задачу, посмотреть сделку, сменить статус. (рис 7).

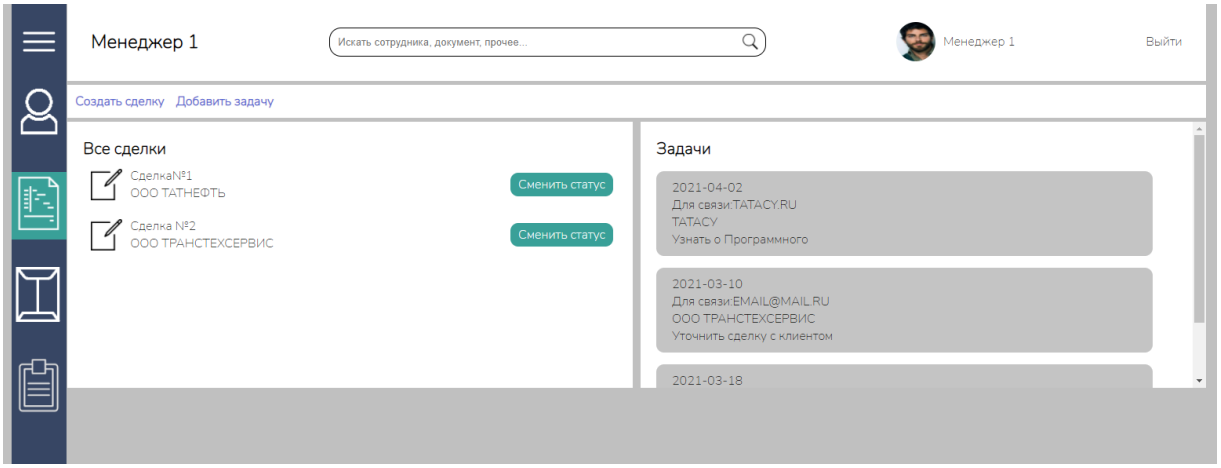


Рисунок 8 – «Рабочая область менеджера»

Панель создания сделки и задач

В панели создания сделки нажав на кнопку «Создать сделку» появляются формы для создания сделки справа в предмете договора, где мы можем выбрать товары со склада.

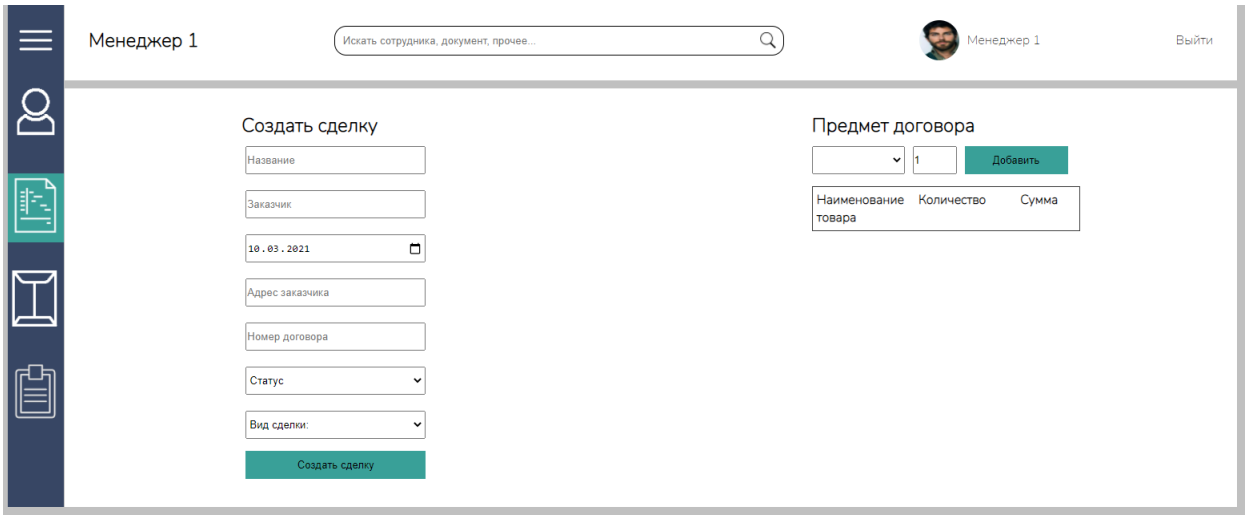


Рисунок 9 – «Формы для создания сделок»

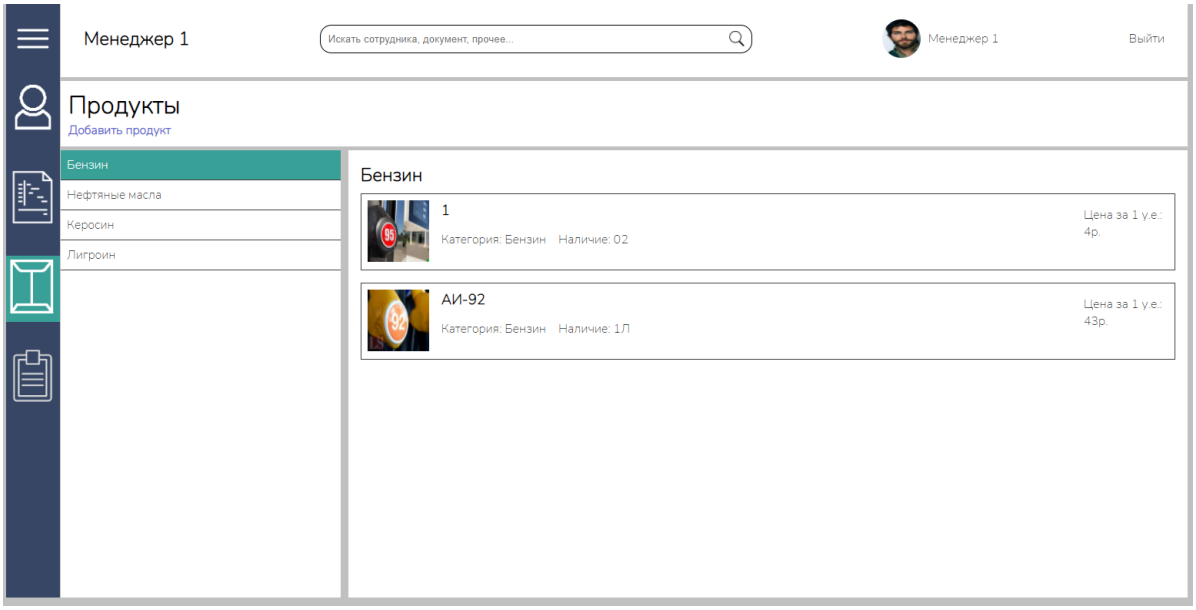


Рисунок 10 – «Товары, выбранные со склада»

В панели создания задачи, нажав на кнопку «Добавить задачу», появляются формы для создания задач.

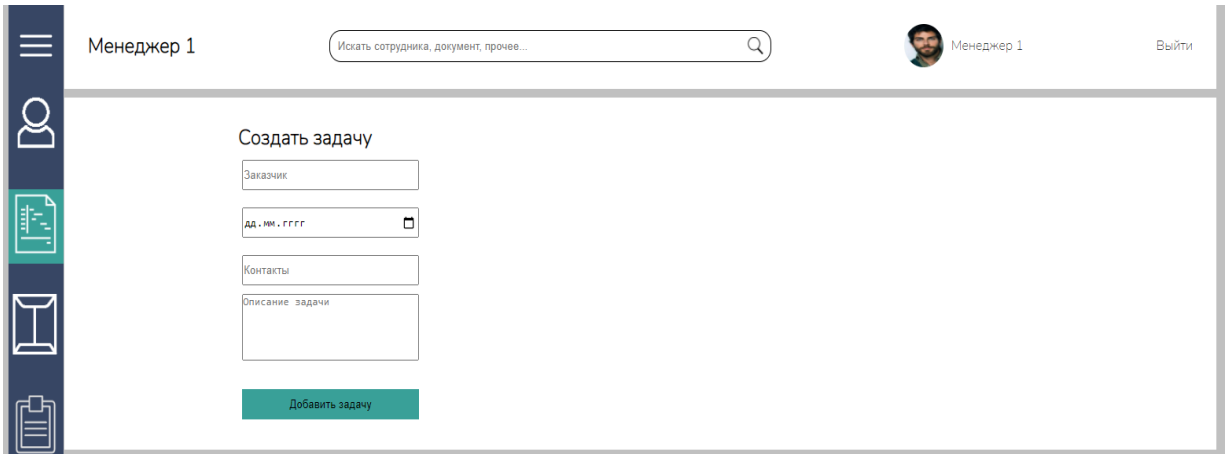


Рисунок 11 – «Создание задачи»

Рабочее окно склада с продукцией

В панели склада, мы можем фильтровать и добавлять новые продукты.

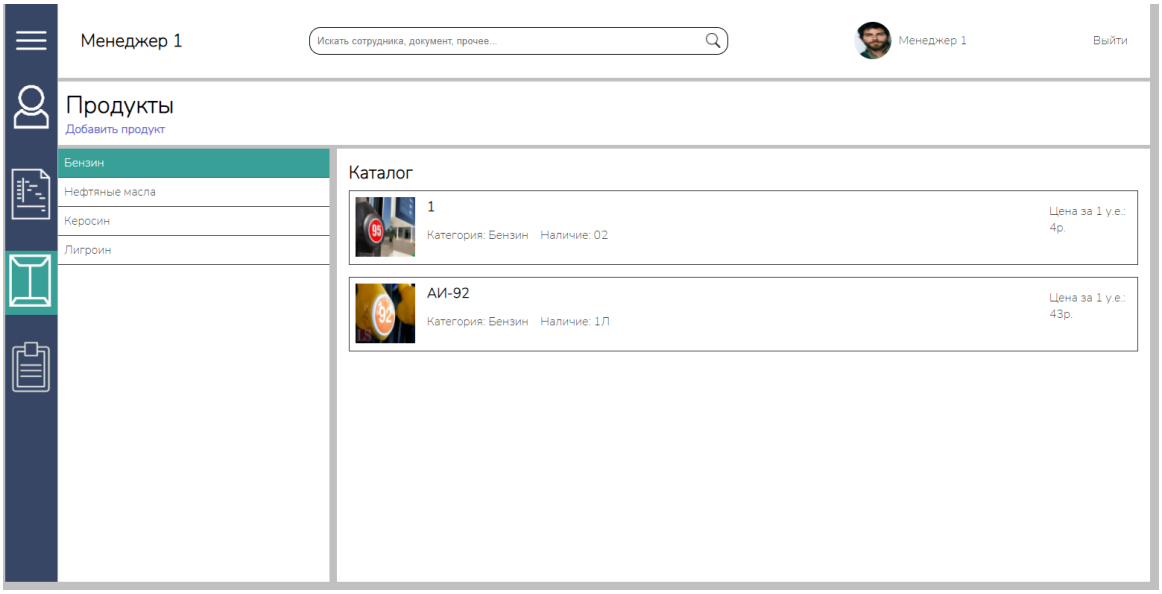


Рисунок 12 – «Окно склада с продукцией»

При нажатии на кнопку “Добавить продукт” открываются формы для добавления продукции (рис 13).

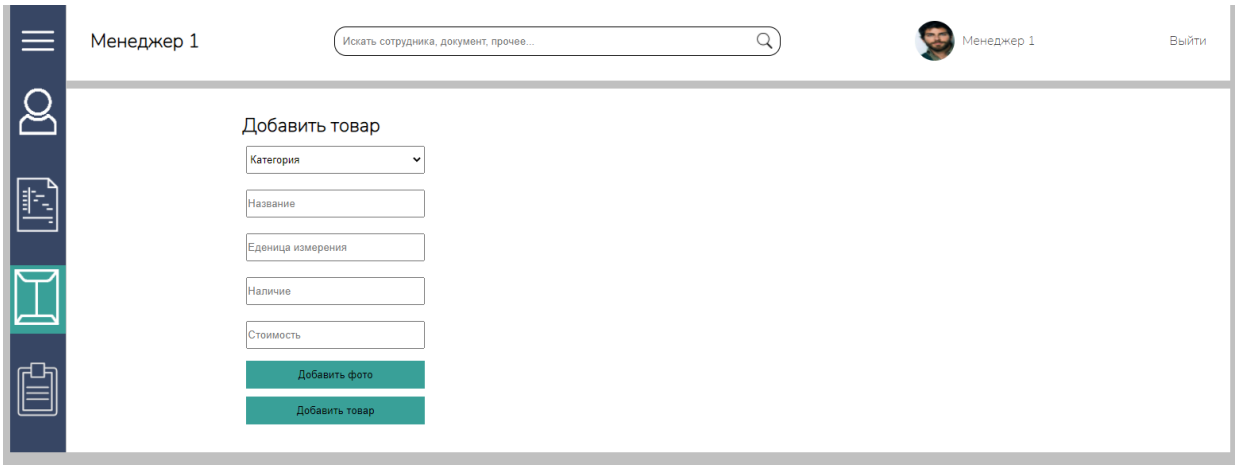


Рисунок 13 – «Формы для добавления товара на склад»

Рабочая область для подачи и просмотра отчётности

В панели отчётности для всех сотрудников, мы видим другую подачу отчёта, она формирует отчёт по закрытым сделкам, и автоматически добавляет в отчёт, подсчитывает прибыль, автоматически отправляет менеджеру на проверку.

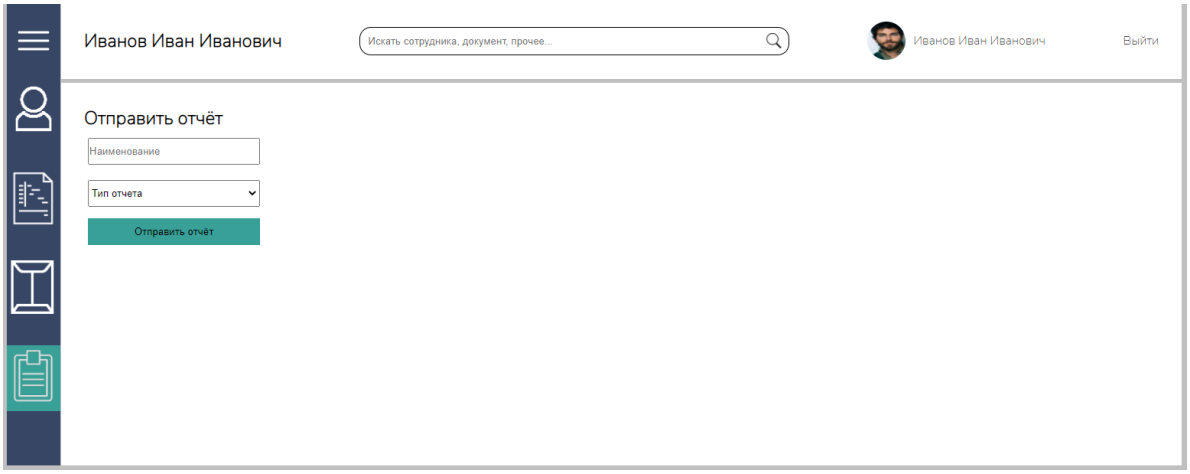


Рисунок – 14 «Подача отчёта у сотрудников»

Далее зайдем в панель отчётности у менеджера, видим поданные отчёты сотрудниками.

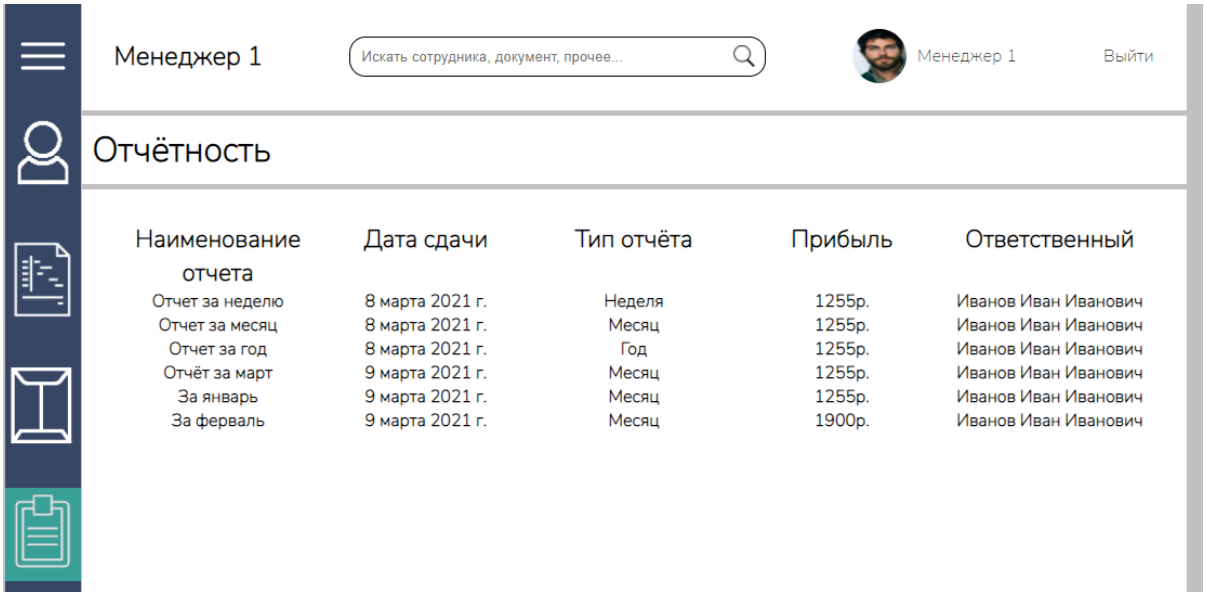


Рисунок 15 – «Отправленные отчёты у менеджера»

Нажав на конкретный отчёт откроется, окно где будет отчёт сотрудника, мы можем переходить по всем его закрытым сделкам. А также смотреть информацию, распечатать информацию о сделке.

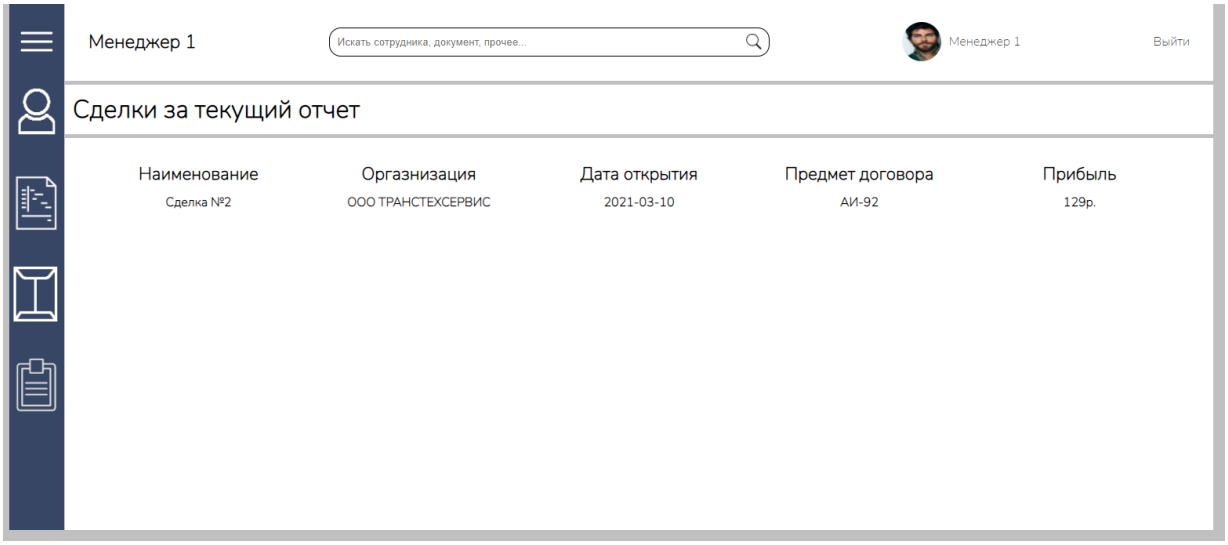


Рисунок 16– «Закрытые сделки работника»

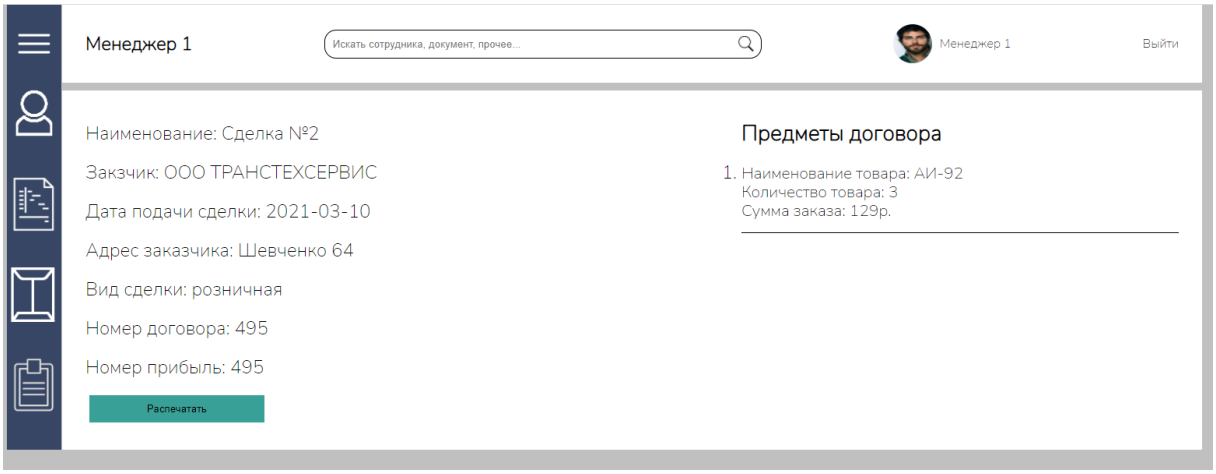


Рисунок 17– «Определенная закрытая сделка работника»

При нажатии на кнопку “Распечатать” открывается отдельное окно с видом распечатки документа

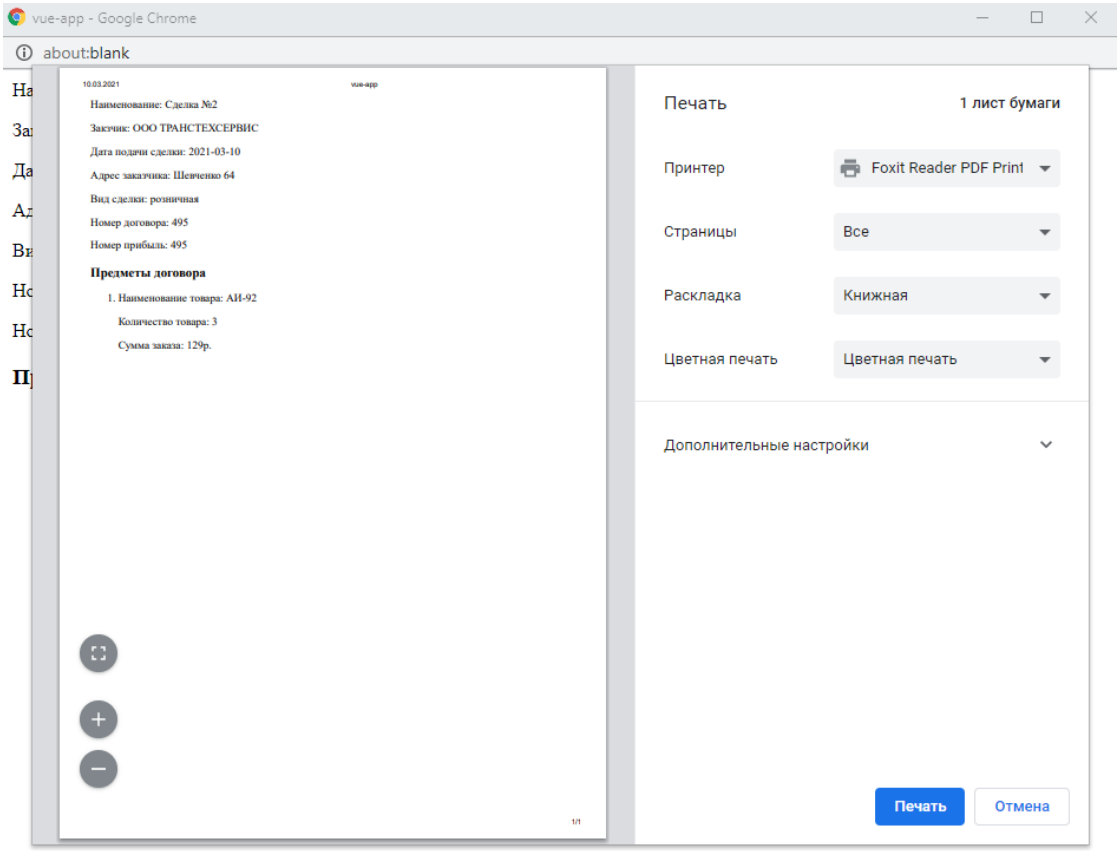


Рисунок 18– «Окно печати документа»

Входные – выходные данные

Сервер данного приложения написан на Node.js, основной библиотекой разработки интерфейса является Фреймворк Vue.js.

Используемые языки:

- Javascript
- Typescript
- CSS
- HTML
- Vue.js

База данных FIREBASE.

Архитектура приложения основана на JS и Vue.

Выходные данные: ответ запроса из базы данных и генерация изображений по этим данным.

Ссылка на веб-приложение: https://kpdp-db010.web.app/

Для доступа к правам Админа: логин- admin, пароль- admin123

Для доступа к правам Дилера: логин- diler@mail.ru, пароль- diler123

Для доступа к правам Менеджера: логин- man@mail.ru, пароль- 123123

Приложение 2

Руководство администратора

Целью администратора является администрирование внутри веб-приложения. Его обязанности и задачи состоят в том, чтобы администратор мог управлять, добавлять, сотрудников для предприятия, а также следить за состоянием веб-приложения.

Для работы с веб-приложением желательно использовать следующие характеристики:

ПК с операционной системой, Windows 7 и выше.

Браузер Google Chrome последней версии или Mozilla Firefox.

Подключение к корпоративной сети через Wi-Fi или Ethernet со скоростью не менее 2 мб/с.

Рекомендуемые требования

Для работы с веб-приложением рекомендуется использовать следующие характеристики:

ПК с операционной системой, Windows 7 и выше.

Браузер Google Chrome последней версии или Mozilla Firefox.

Подключение к корпоративной сети через Wi-Fi или Ethernet со скоростью не менее 5 мб/с.

Установка и запуск Web – приложения

- Установка приложения

Для работы с веб-приложением выполнения каких-либо установок не требуется, веб-приложение уже размещено на хостинге.

- Запуск приложения

Запустите браузер (Google Chrome или Yandex Browser)

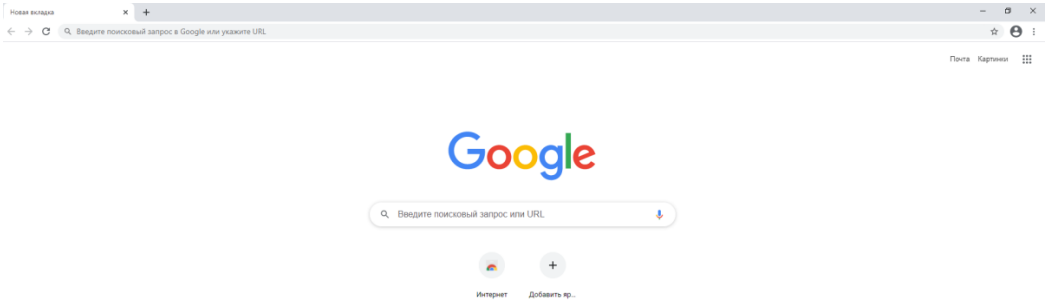


Рисунок 1 – «Запуск браузера»

Запуск веб приложения

Зайдем в веб-приложение по веб адресу: https://kpdp-db010.web.app/

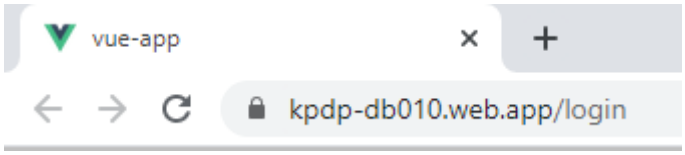


Рисунок 2 – «Веб-приложение»

Работа с веб-приложением

Для работы в веб-приложении необходимо авторизоваться (рис 2):

Авторизуемся под администратором

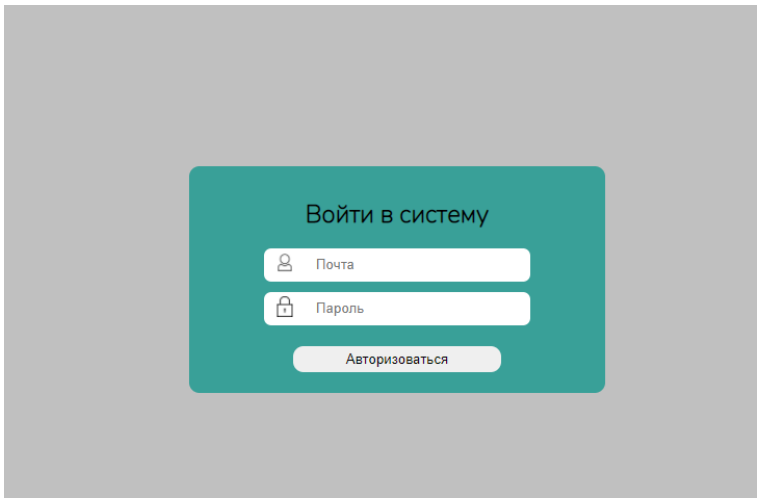


Рисунок 3 – «Авторизация»

Попадаем на профиль администратора, затем зайдем на вторую вкладку в админ панель, и введем пароль от админа панели (рис 4)

Рисунок 4 – «Админ панель для администратора»

Переходим в меню админ панели добавление работника на предприятие, можем добавить работника на предприятие, тем самым отправляем запрос на базу данных веб-приложения. (рис 5)

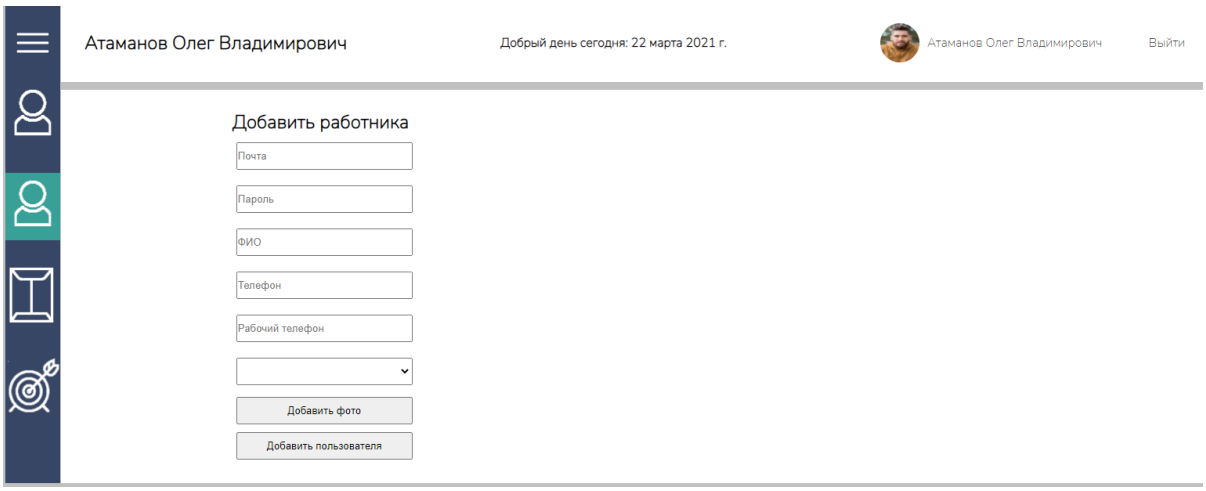


Рисунок 5 – «Меню добавление работника в админ панели»

Рабочее окно склада с продукцией

В панели склада, мы можем фильтровать и добавлять новые продукты.

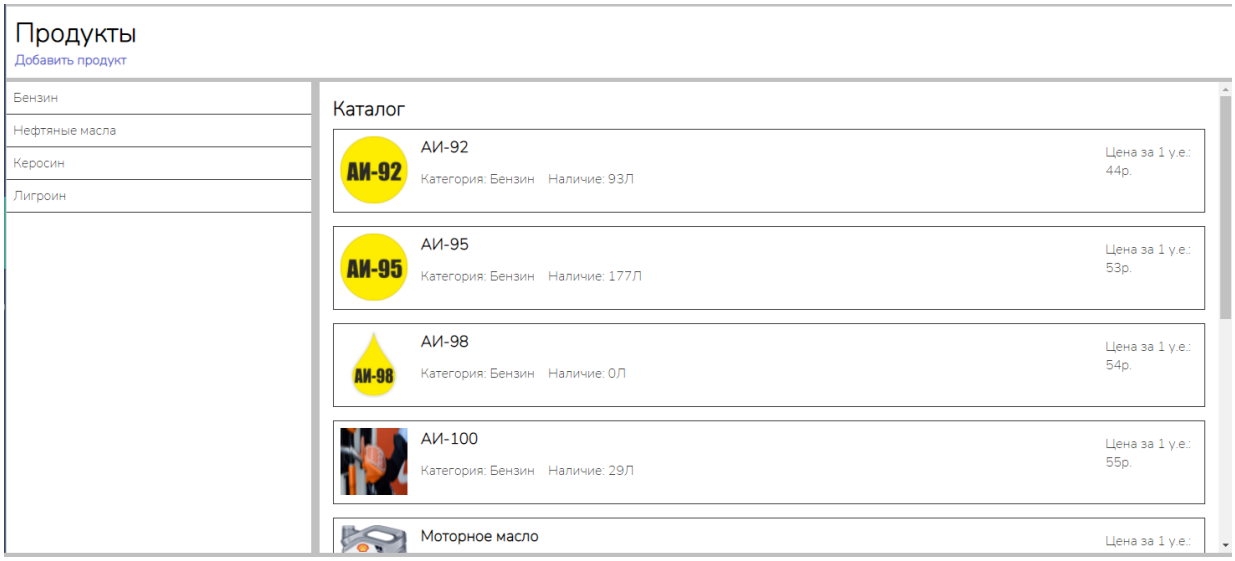


Рисунок 6 – «Окно склада с продукцией»

При нажатии на кнопку “Добавить продукт” открываются формы для добавления продукции (рис 7).

Рисунок 7 – «Формы для добавления товара на склад»

Входные – выходные данные

Сервер данного приложения написан на Node.js, основной библиотекой разработки интерфейса является Фреймворк Vue.js.

Используемые языки:

1. Javascript
2. Typescript
3. CSS
4. HTML
5. Vue.js

6. База данных FIREBASE.

Архитектура приложения основана на JS и Vue.

Выходные данные: ответ запроса из базы данных и добавление пользователей по этим данным.

Ссылка на веб-приложение: https://kpdp-db010.web.app/

Для доступа к правам Админа: логин- admin, пароль - admin123

Для доступа к правам Админ Панели: пароль - admin

Для доступа к правам Дилера: логин - diler@mail.ru, пароль - diler123

Для доступа к правам Менеджера: логин - man@mail.ru, пароль - 123123

Приложение 3

Электронный носитель с программой и пояснительной запиской