МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования

Кафедра /департамент /центр Прикладная математика и информатика
(наименование кафедры/департамента/центра полностью)
09.03.03 Прикладная информатика
(код и наименование направления подготовки, специальности)
Разработка программного обеспечения
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка программного обеспечения информационной системы управления продажами»

Обучающийся В. А. Новиков _____
(Инициалы Фамилия) (личная подпись)
Руководитель к.т.н., Н.В. Хрипунов
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2024

Аннотация

Название представленной выпускной квалификационной работы: Разработка программного обеспечения информационной системы управления продажами.

Объект выпускной квалификационной работы: система управления продажами компании.

Предмет исследования: автоматизация информационных процессов для оптимизации работы с заказами и улучшения клиентского сервиса.

Структурно работа представлена введением, четырьмя разделами, заключением и списком использованных источников.

В первой главе представлен анализ объекта, описание предметной области, а также определены ключевые сущности и метрики бизнес-процесса.

Во второй главе проведён анализ существующих решений для автоматизации обработки заказов, обоснован выбор инструментов разработки и сформулированы требования к программной системе.

В третьей главе описаны архитектура программного продукта и проектирование взаимодействия модуля «умный поиск» с другими компонентами системы.

В четвертой главе представлена реализация алгоритмов бизнес-логики, структура данных, пользовательский интерфейс, а также приведены результаты тестирования системы и её экономическое обоснование.

Пояснительная записка содержит 44 листа печатного текста, 9 иллюстраций, 2 страницы кода. Она состоит из введения, аннотации, 4 главы, заключения и списка использованных источников, включающего 21 наименование.

Оглавление

Введение

Современный бизнес во многом зависит от скорости и точности процессов, особенно когда речь идет о продажах и обслуживании клиентов. Чтобы оставаться конкурентоспособным и эффективно удовлетворять запросы покупателей, компании всё чаще обращаются к автоматизации своих операций. В условиях растущей конкуренции и увеличения объемов работы ключевыми факторами успеха становятся скорость, точность и минимизация ручного труда. Автоматизация процессов позволяет достичь этих целей, оптимизируя рабочие процессы и сокращая влияние человеческого фактора .

ВКР посвящена разработке программы под названием "умный поиск", которая позволит компании "ВсеИнструменты.ру" автоматизировать процесс обработки клиентских заказов. Программа должна будет распознавать запросы клиентов, содержащие информацию о товарах и их количестве, и автоматически добавлять эти данные в систему оформления заказов. Это решение позволит сократить время на обработку заявок, уменьшить количество ошибок и повысить общую эффективность бизнес-процессов.

Целью данной выпускной квалификационной работы является создание автоматизированной системы, которая будет интегрирована в процесс управления заказами компании. Эта программа будет основана на простом пользовательском интерфейсе и использовании методов распознавания текста для автоматизации обработки заказов.

Для достижения цели работы необходимо выполнить следующие задачи:

- проанализировать текущие процессы обработки заказов в компании "ВсеИнструменты.ру";
- изучить существующие решения для автоматизации обработки клиентских запросов;

- спроектировать базу данных для хранения информации о товарах и заказах;
- разработать интерфейс программы "умный поиск", который будет удобен для использования сотрудниками компании;
- тестировать и внедрить программу в рабочую среду компании.

Важную роль в проекте играет моделирование бизнес-процессов. Этот метод позволяет визуализировать все этапы работы, начиная от получения клиентских запросов до окончательного оформления заказа. Моделирование также помогает понять, как автоматизация может изменить текущий рабочий процесс и какие улучшения она принесёт.

Результатом данной ВКР станет программа, которая облегчит работу сотрудников компании "ВсеИнструменты.ру", ускорив процесс оформления заказов и повысив их точность. Программа "умный поиск" позволит компании сократить затраты времени на рутинные операции и улучшить качество обслуживания клиентов.

Глава 1 Анализ объекта

1.1 Описание предметной области

«ВсеИнструменты.ру» занимается продажей строительных и хозяйственных товаров от поставщиков к конечному потребителю. В сферу интересов и задач магазина входят:

- закупка товаров у поставщиков;
- продажа товаров клиентам;
- автоматизация процесса обработки заказов;
- обеспечение прозрачной отчетности по продажам и складу.

В организационной структуре компании несколько отделов: отдел продаж, отдел закупок, складской отдел и отдел логистики. Также в компании есть IT-отдел [7], который занимается поддержкой и развитием информационных систем предприятия. Процесс управления продажами включает в себя взаимодействие между менеджерами по продажам, менеджерами по закупкам, складом и отделом логистики.

Основной целью разрабатываемого программного обеспечения является автоматизация процесса добавления товаров в заказы клиентов рисунок 1, что должно сократить участие человека в рутинных операциях [5].

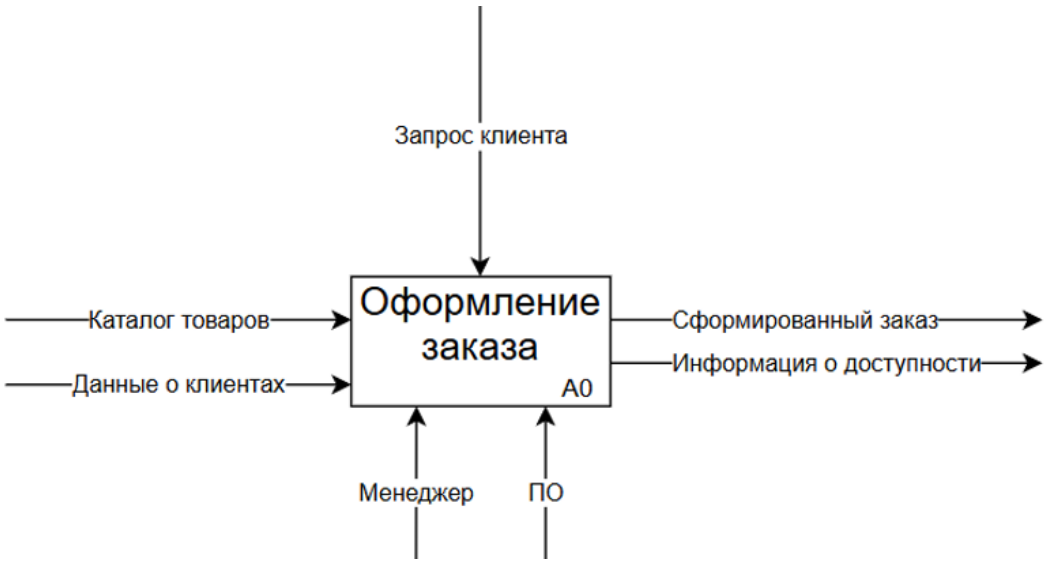


Рисунок 1 – Контекстная диаграмма

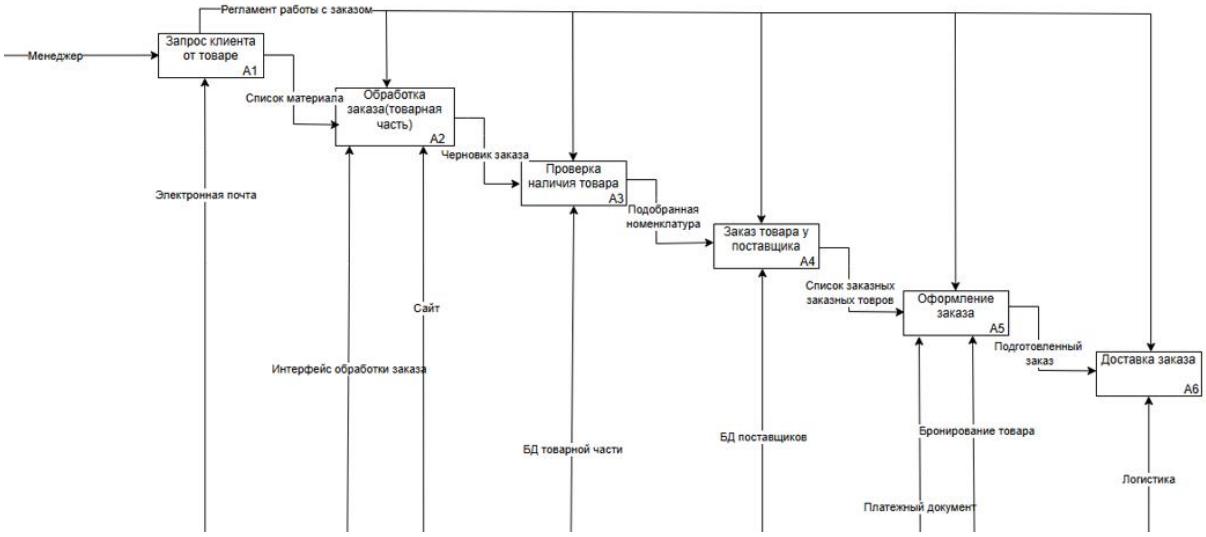


Рисунок 2 – Диаграмма бизнес-процесса заказа idef0 «Как есть»

Описание схемы деятельности предприятия можно выделить следующие ключевые сущности предметной области [15]:

- поставщики — компании, предоставляющие товар для дальнейшей продажи;
- товары — ассортимент товаров, находящихся в продаже и на складе;

- склад — место хранения товаров, где ведется учет наличия и движение товаров;
- продажи — данные о реализованных товарах, выполненных заказах и их статусах;
- сотрудники — менеджеры по продажам, закупкам, сотрудники склада и логистики, которые участвуют в процессе обработки заказов;
- логистика — отдел, занимающийся доставкой товаров конечному покупателю.

Описание процесса заказа, рисунок:

- клиент оставляет запрос менеджеру по телефону, почте или на сайте;
- менеджер создает заказ в срм системе и консультирует клиента по заказу, добавляет нужное количество товара, ищет по номенклатуре нужную позицию, добавляет в заказ;
- заказ попадает на участок сборки товара на складе, если товар имеется в наличии, в случае если товара нет, заказ попадает к менеджеру по закупкам, а затем поступает на склад;
- склад передает заказ отделу логистики, которые, в свою очередь, передают заказ курьеру, который везет заказ к месту выдачи;
- клиент получает заказ [21].

1.2 Описание основных метрик бизнес-процесса

Для эффективного управления процессом обработки заказов и его оптимизации в компании ООО "ВсеИнструменты.ру" мы выделяем несколько ключевых метрик, которые помогают отслеживать производительность и качество выполнения заказа. Основные показатели, на которые мы ориентируемся, включают в себя:

- время обработки заказа менеджером;
- ошибки при выполнении заказа;

- процент неправильно обработанных заказов;
- время сборки заказа на складе;
- время доставки заказа;

Время обработки заказа менеджером — это время, которое уходит на доставку товара клиенту, начиная с момента передачи заказа в отдел логистики и до его получения клиентом. Чем быстрее заказ будет доставлен, тем выше удовлетворенность клиента.

Ошибки при выполнении заказа отражают, сколько времени уходит у менеджера на создание заказа в CRM-системе, начиная с момента получения запроса от клиента до момента передачи заказа на склад. Данная метрика является критически важной, так как напрямую влияет на общую скорость выполнения заказа и удовлетворенность клиента. Быстрая обработка заказов позволяет ускорить процесс доставки и избежать задержек на ранних этапах.

Процент неправильно обработанных заказов учитывает количество ошибок, которые возникают на разных этапах выполнения заказа, включая ввод неверной информации, неправильное указание товара или количества. Ошибки на этом этапе могут привести к неверной сборке заказа на складе, что отрицательно сказывается на времени доставки и на репутации компании.

Время сборки заказа на складе демонстрирует долю заказов, которые были неправильно обработаны по различным причинам — будь то ошибки менеджера при внесении информации, неверная сборка на складе или ошибки в логистике. Высокий процент неправильно обработанных заказов ведет к возвратам, перерасходу ресурсов и снижению доверия клиентов.

Время доставки заказа отражает время, которое уходит на нахождение и подготовку товаров для конкретного заказа на складе. Длительная сборка может замедлить доставку и увеличить время обработки заказа в целом.

Таким образом, можно сделать вывод, что результатом работы стало понимание необходимости оптимизации существующих процессов и внедрения решений, которые позволят ускорить обработку заказов и улучшить качество обслуживания клиентов. Это станет базой для проектирования и разработки программного обеспечения, которое отвечает потребностям компании и повышает её конкурентоспособность.

Глава 2 Постановка задачи определение требований к программной системе

2.1 Постановка задачи

Одними из основных метрик в этом бизнес-процессе являются:

- время обработки заказа менеджером;
- ошибки при выполнении заказа;
- процент неправильно обработанных заказов.

Данные метрики находятся вначале нашего бизнес-процесса, тем самым они влияют на весь остальной бизнес-процесс.

Основные метрики для решения:

- время обработки заказа менеджером;
- ошибки при выполнении заказа;
- процент неправильно обработанных заказов.

Задача: сократить время обработки заказа за счет автоматизации процесса поиска товаров. Программа "Умный поиск" должна автоматически извлекать товар из базы [2] по запросу клиента (например, через ключевые слова или описание) и добавлять его в заказ с указанным количеством. Это позволит менеджеру быстрее оформлять заказы и сократить задержки при их обработке.

При ручной обработке заказов допускаются ошибки, такие как неверное количество товара, неправильный выбор позиции, или некорректные данные о наличии товара на складе. Эти ошибки приводят к задержкам, возвратам, и, как следствие, к недовольству клиентов.

Задача: минимизировать количество ошибок при обработке заказов. Программа должна автоматически проверять наличие товара на складе, корректно добавлять его в заказ, и предупреждать менеджера о проблемах (например, если товар отсутствует или его недостаточно). Это позволит снизить количество ошибок, связанных с человеческим фактором, и

повысить точность выполнения заказов.

Процент неправильно обработанных заказов негативно сказывается на работе компании. Проблемы, связанные с неверными позициями в заказах или неправильным количеством, требуют дополнительных ресурсов на исправление, что создает неудобства как для компании, так и для клиентов.

Задача: снизить процент неправильно обработанных заказов за счет внедрения автоматизации. Программа "Умный поиск" должна исключить ручной ввод информации, автоматически подбирая нужные позиции и проверяя наличие товара на складе. Это позволит улучшить качество обработки заказов и уменьшить количество допущенных ошибок.

На данный момент, менеджеры по продажам тратят значительное количество времени на поиск нужного товара в базе, его добавление в заказ и проверку наличия на складе. Это не только снижает общую эффективность работы, но и увеличивает время, которое клиент ждет оформления заказа.

Программа "Умный поиск": решение проблем.

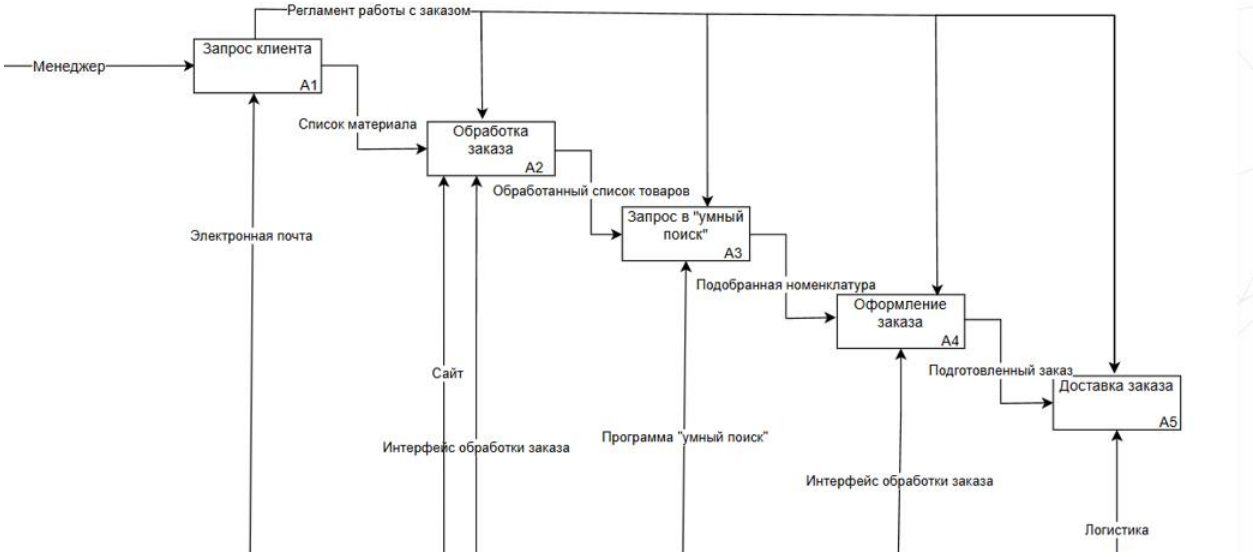


Рисунок 3 – Диаграмма бизнес-процесса заказа idef0 «Как должно быть»

Основной целью внедрения программы "Умный поиск" является улучшение работы менеджеров по продажам, которые находятся на первом этапе взаимодействия с клиентом и формируют заказ, рисунок 3. Программа обеспечит:

- сокращение времени обработки заказов за счет автоматического поиска товаров и добавления их в систему;
- уменьшение количества ошибок за счет автоматической проверки наличия товара на складе и корректного добавления позиций в заказ;
- снижение процента неправильно обработанных заказов, что повысит точность выполнения заказов и удовлетворенность клиентов.

2.2 Анализ существующих решений

На рынке уже существует ряд продуктов, для решения наших задач. Вот некоторые из них:

- Zoho CRM;
- HubSpot;
- Content AI.

Zoho CRM предлагает функцию интеллектуального поиска, которая значительно облегчает пользователям поиск нужной информации, включая записи, задачи и контакты. Система использует продвинутые алгоритмы, которые анализируют не только введенные пользователем запросы, но и учитывают частоту поиска, а также предыдущие действия пользователя. Это позволяет формировать более точные результаты, адаптированные под потребности конкретного пользователя, и значительно ускоряет процесс работы с системой, обеспечивая более высокий уровень продуктивности.

HubSpot оснащен мощным поисковым движком, который предоставляет пользователям возможность находить информацию по разнообразным критериям, включая ключевые слова, теги и метаданные. Эта система обладает высокой степенью гибкости и может быстро

обрабатывать запросы, обеспечивая релевантные результаты. Кроме того, HubSpot предлагает рекомендации на основе активности пользователей, анализируя их предыдущие поисковые запросы и взаимодействия с системой. Это позволяет значительно улучшить пользовательский опыт и повышает эффективность работы с платформой, так как пользователи могут быстрее находить нужные им данные.

Content AI использует современные алгоритмы машинного обучения и обработки естественного языка для автоматизации процессов создания, анализа и управления контентом. Эти технологии позволяют системе генерировать текст и другие формы контента с учетом контекста и предпочтений целевой аудитории, что существенно улучшает качество производимого контента. Content AI также может анализировать данные о производительности контента, предоставляя рекомендации по его оптимизации и персонализации. Это обеспечивает более глубокое взаимодействие с пользователями и помогает компаниям создавать контент, который отвечает потребностям их аудитории, повышая вовлеченность и удовлетворенность клиентов.

Минусы интеграции в систему:

- избыточная функциональность - многие системы предлагают набор функций, который часто оказывается избыточным для конкретных нужд бизнеса. Это может усложнить интерфейс и затруднить обучение пользователей, что в итоге приводит к снижению общей эффективности;
- высокие затраты на содержание - сопровождение и поддержка существующих решений могут требовать значительных финансовых ресурсов. Эти затраты включают регулярное обновление программного обеспечения, техническую поддержку и возможные расходы на обучение персонала;
- высокие затраты на интеграцию - процесс интеграции существующих систем с новыми решениями может быть сложным и дорогим. Часто

требуется время и ресурсы для настройки интерфейсов, обеспечения совместимости и тестирования системы, что увеличивает общую стоимость проекта;

- дополнительные затраты на приобретение продукта - закупка программного обеспечения может подразумевать не только прямые расходы на лицензионные соглашения, но и дополнительные затраты на лицензии для пользователей, обучение и возможные консультации, что в итоге значительно увеличивает общие инвестиции;
- потенциальная угроза безопасности - использование устаревших или плохо защищенных систем может подвергать компанию рискам утечки данных и других киберугроз. Уязвимости в безопасности могут привести к финансовым потерям и ущербу репутации, что делает защиту информации критически важной задачей.

2.3 Обоснование

В результате детального анализа недостатков существующих систем можно сделать вывод, что разработка программы умного поиска собственными силами является наиболее целесообразным и эффективным решением. Это позволит не только избежать проблем, связанных с избыточной функциональностью, но и оптимизировать расходы на содержание системы [19].

Создание собственного решения даст возможность сосредоточиться исключительно на тех функциях, которые действительно необходимы для бизнеса. Это позволит избежать перегрузки интерфейса и упростить обучение пользователей, что, в свою очередь, повысит общую продуктивность команды. Программа станет адаптирована под специфические потребности компании, обеспечивая интуитивно понятный и удобный интерфейс.

Кроме того, разработка собственного программного обеспечения поможет сократить высокие затраты на содержание существующих

решений. Будет полный контроль над функционалом и появится возможность вносить изменения по мере необходимости, что исключит дополнительные расходы на сторонние услуги и техническую поддержку. Это не только сократит затраты, но и упростит процесс обновления системы, что сделает её более актуальной и эффективной [18].

Также стоит отметить, что создание системы с нуля позволит избежать сложных и дорогих процессов интеграции. Возможно заранее спроектировать архитектуру так, чтобы она идеально соответствовала текущим и будущим требованиям, что существенно упростит взаимодействие с другими системами и повысит их совместимость.

Безопасность является критически важным аспектом. Создавая собственное решение, будет возможным изначально заложить необходимые меры по защите данных, что значительно снизит риски утечек и других киберугроз. Это поможет не только защитить компанию от финансовых потерь, но и сохранить её репутацию на высоком уровне.

Таким образом, можно сделать вывод, что разработка программы умного поиска собственными силами не только отвечает специфическим требованиям, но и способствует оптимизации затрат, повышению безопасности и улучшению общей эффективности бизнес-процессов. Это стратегически верный шаг для компании [20].

Глава 3 Проектирование программной системы

3.1 Разработка архитектуры программного продукта

Имеющиеся модули системы работают по микро-сервисной архитектуре, модуль умный поиск, должен так же быть спроектирован с учетом этого.

В рамках этой архитектуры, уже есть такие модули [1]:

- CRM;
- реляционная база данных с товарной частью ;
- модуль обработки заказов с реляционной базой данных «Postgresql»;
- интерфейс для работы с заказами (аналог 1С);
- список названий товаров на основе «Elasticsearch».

Микро сервисная архитектура. Имеющиеся модули системы работают по микро сервисной архитектуре, где каждый сервис выполняет определенную функцию и взаимодействует с другими через четко определенные интерфейсы [6]. Это обеспечивает гибкость, легкость масштабирования и упрощает поддержку. Для модуля "умный поиск" важно соблюдать принципы микро-сервисной архитектуры (рисунок 4), чтобы он также был легко масштабируемым и мог взаимодействовать с другими модулями, не нарушая общую архитектуру [4].

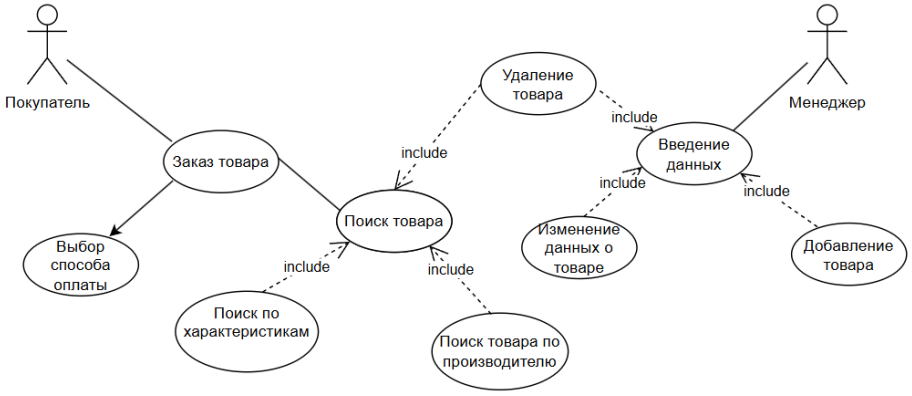


Рисунок 4 - Диаграмма прецедентов заказа интернет-магазина

Основные модули системы:

- CRM - Управляет данными о клиентах и заказах, позволяет менеджерам видеть и редактировать информацию о клиентах и текущих заказах;
- модуль товарной части с реляционной базой данных - хранит информацию о товарах, их характеристиках и наличии [12];
- модуль обработки заказов с реляционной базой данных на PostgreSQL - обрабатывает заказы, связывает информацию о заказе с данными о клиенте и товаре;
- интерфейс для работы с заказами (аналог 1С) - служит основным инструментом для менеджеров при создании и корректировке заказов;
- список названий товаров на основе Elasticsearch - быстрое индексирование и поиск по каталогу товаров, что необходимо для быстрого отклика на поисковые запросы.

Модуль "умный поиск" должен взаимодействовать с несколькими из этих компонентов для выполнения своей основной задачи - автоматическое добавление товара в заказ на основе клиентского запроса [16].

"Умный поиск" будет взаимодействовать с другими модулями:

- со списком названий товаров — для нечеткого поиска необходимого товара;
- с модулем товарной части — для получения информации о наличии товара, и его характеристиках и цене;
- с модулем обработки заказов — для автоматического добавления товара и количества в заказ, который обрабатывается на складе и логистикой;
- с CRM — для создания новых заказов и автоматического заполнения информации о клиентах;
- с интерфейсом для работы с заказами – для получения запроса на «умный поиск».

3.2 Описание взаимодействие модуля «Умный поиск» с другими модулями системы

Инициализация поиска товара в интерфейсе для работы с заказами. В интерфейсе для работы с заказами менеджер видит кнопку для открытия «поп-ап» окна модуля «Умный поиск». Нажатие на эту кнопку вызывает открытие окна, в котором находятся два элемента:

- Text Field — поле для ввода, куда менеджер может ввести название товара;
- Button — кнопка, с помощью которой менеджер отправляет запрос на поиск товара.

Отправка запроса на поиск товара в «Умный поиск». Когда менеджер вводит название товара и количество (в штуках или упаковках) и нажимает на кнопку, интерфейс для работы с заказами отправляет POST-запрос в модуль «Умный поиск». POST-запрос содержит текст с названием товара и указанием количества штук или упаковок. Запрос отправляется на определенный API-эндпоинт модуля «Умный поиск» для дальнейшей обработки.

Обработка запроса от интерфейса для работы с заказами.

Модуль «Умный поиск» получает POST-запрос от интерфейса. На этом этапе происходит:

- выделение сущностей из запроса: модуль анализирует тело запроса, извлекая из него название товара и запрашиваемое количество;
- подготовка данных для передачи в другие модули системы.

Взаимодействие со списком названий товаров для поиска товара. Модуль «Умный поиск» отправляет POST-запрос с названием товара в модуль со списком названий товаров. Запрос на поиск отправляется с целью выполнения нечеткого поиска для определения подходящего товара, даже если название введено с незначительными ошибками или сокращениями. Запрос анализируется, и возвращается информация о наиболее релевантных совпадениях;

Обработка ответа от модуля со списком названий товаров. В ответе от модуля со списком названий товаров приходит сущность найденного товара. Ответ включает:

- полное название товара;
- идентификатор товара для дальнейших запросов;
- уточненное название, если был выполнен нечеткий поиск.

Взаимодействие с модулем товарной части для проверки наличия. После получения точного наименования и идентификатора товара, модуль «Умный поиск» отправляет POST-запрос в модуль товарной части. Запрос включает название и требуемое количество товара (в штуках или упаковках). Цель запроса — уточнить фактическое наличие на складе, чтобы подтвердить возможность выполнения заказа в нужном количестве.

Обработка ответа от модуля товарной части. В ответе от модуля товарной части приходит информация о наличии товара. Уточняется фактическое количество товара на складе. Если запрошенное количество превышает наличие, модуль товарной части указывает максимальное доступное количество. Передача данных в интерфейс для подтверждения менеджером. Модуль «Умный поиск» формирует ответ на

основе данных, полученных от модулей списка названий товаров и товарной части.

Ответ содержит:

- название товара и подтвержденное количество;
- дополнительную информацию (например, максимальное доступное количество, если запрошенное превышает наличие).

Ответ возвращается в интерфейс для работы с заказами, где менеджер может подтвердить заказ или скорректировать его параметры.

Взаимодействие с модулем обработки заказов для создания заказа. После подтверждения менеджером, интерфейс для работы с заказами отправляет окончательный POST-запрос в модуль обработки заказов. Этот запрос включает идентификатор товара и утвержденное количество. Модуль обработки заказов принимает данные и создает новую запись о заказе в CRM-системе, которая отслеживает статусы и продвижение заказа на следующих этапах, включая сборку и доставку.

Применение микро сервисной архитектуры позволяет гибко масштабировать отдельные модули, например, "умный поиск" можно будет увеличить по мощностям независимо от других компонентов. За счет автоматизации поиска и добавления товаров экономится время менеджеров и снижается количество ошибок при оформлении заказов, что ведет к сокращению расходов на исправление ошибок и повышению удовлетворенности клиентов.

Таким образом, можно сделать вывод, что разработанный процесс автоматизирует рутинные операции, снижая затраты времени менеджеров и уменьшая количество ошибок при оформлении заказов. Внедрение системы способствует повышению операционной эффективности компании, сокращению затрат на исправление ошибок и повышению удовлетворенности клиентов за счёт более быстрого и точного выполнения заказов.

Глава 4 Реализация программной системы

4.1 Разработка схем данных

Ключевые схемы данных, разработанные для обеспечения корректной работы системы. На этапе проектирования схем были определены структуры, которые позволяют обрабатывать запросы пользователей и формировать ответы системы [17]. Основой разработки стало использование библиотеки Pydantic [12], которая обеспечивает строгую типизацию данных и валидацию входных параметров.

На рисунке 5 представлена схема запроса из интерфейса умного поиска, реализованная через класс ParseIn [14]. Она принимает основной текст запроса от пользователя, который далее используется для поиска подходящих товаров. На рисунке 7 представлена схема ответа от модуля товарной части, реализованная с помощью класса ProductDetailsResponse. Она включает данные о названии товара, количестве на складе, статусе его наличия (например, "в наличии", "под заказ" или "отсутствует") и актуальной цене за единицу товара.

```
2
3
4   class ParseIn(BaseModel):
5       """Входная схема для основного запроса."""
6
7       text: str = Field(description="Основной текст запроса")
8
```

Рисунок 5 - Схема запроса из интерфейса умного поиска

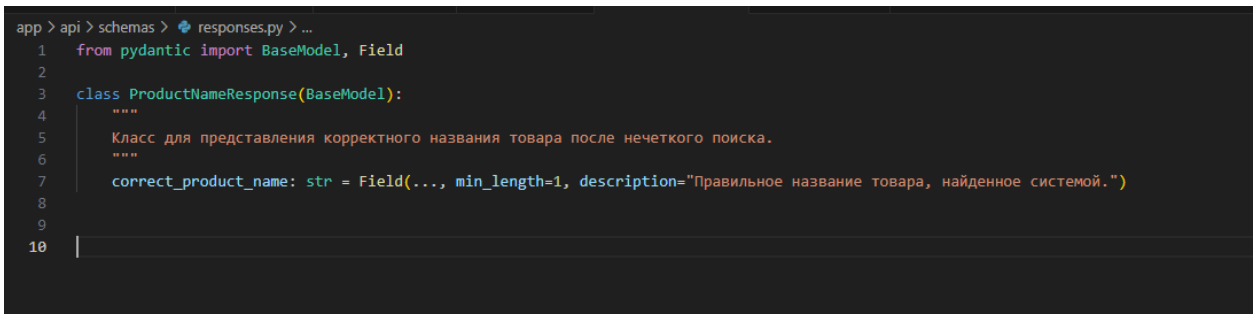


Рисунок 6 - Схема ответа от модуля с нечетким поиском

На рисунке 6 показана схема ответа от модуля с нечетким поиском, описанная через класс ProductNameResponse. Эта схема возвращает корректное название товара после анализа и обработки пользовательского запроса [13].

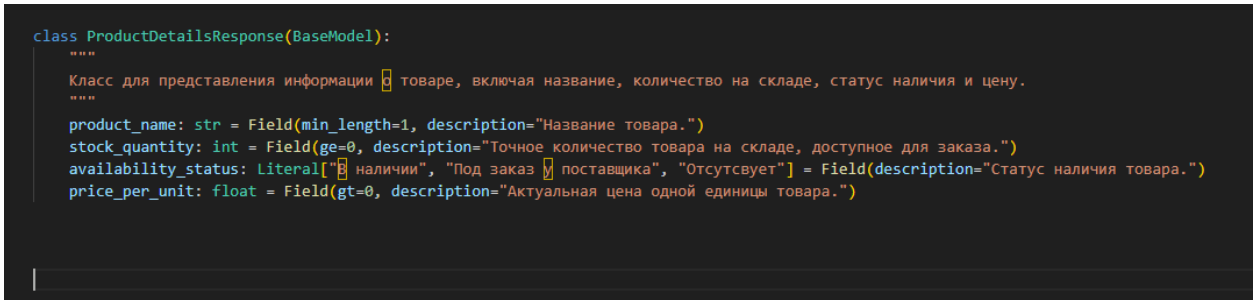


Рисунок 7 - Схема ответа от модуля товарной части

На рисунке 7 представлена схема ответа от модуля товарной части, реализованная с помощью класса ProductDetailsResponse. Она включает данные о названии товара, количестве на складе, статусе его наличия (например, "в наличии", "под заказ" или "отсутствует") и актуальной цене за единицу товара.

4.2 Разработка реализации бизнес-логики

4.2.1 Бизнес-логика основного запроса

Для обработки запроса, введенного менеджером в "умный поиск", используется алгоритм, который анализирует текст запроса, выделяет из него название товара и определяет количество. Данный алгоритм учитывает случаи, когда количество единиц товара может быть не указано в запросе. Если количество отсутствует, по умолчанию устанавливается значение "1".

Детальный процесс обработки запроса менеджера:

- паттерн для поиска названия товара и количества: алгоритм использует шаблон регулярного выражения, который выделяет две части текста: название товара и количество. Первая часть шаблона ориентирована на извлечение текста, содержащего название товара, и охватывает буквы русского и латинского алфавитов, а также пробелы. Вторая часть шаблона нацелена на определение количества, указанного в числовом виде, и поддерживает такие единицы измерения, как «штук», «упаковок», «штуки», «упаковки». Для обработки случаев, когда количество не указано, эта часть шаблона задана как необязательная.
- выделение названия товара: при анализе запроса алгоритм первым делом извлекает название товара. Это название, если оно присутствует, очищается от лишних пробелов в начале и конце и сохраняется как ключевой элемент, необходимый для последующего поиска товара в базе данных. Такой подход обеспечивает точное и корректное извлечение даже при случайных пробелах в начале или конце названия.
- определение количества: после выделения названия товара алгоритм проверяет наличие числового значения, указывающего количество товара. Если количество указано, оно преобразуется в целое число и используется в дальнейшем запросе. В случае если

количество отсутствует (например, запрос «шурупы» без указания числового значения), алгоритм автоматически назначает количество равное единице. Это позволяет корректно обрабатывать запросы, не требуя от менеджера точного указания количества, что удобно для ускоренной обработки.

- формирование результата обработки: после успешного извлечения информации о товаре и количестве алгоритм формирует структурированный результат в виде словаря, который содержит ключи «product_name» для названия товара и «quantity» для количества. Эти данные могут быть затем переданы в другие модули системы для поиска и добавления товара в заказ.

4.2.2 Запрос к списку названий товаров с нечетким поиском

Основные этапы выполнения запроса к списку названий товаров с нечетким поиском

Формирование списка названий товаров. В базе данных заранее создан и поддерживается список всех названий товаров в ассортименте компании. Этот список формируется с учетом возможных ошибок при вводе, транслитерации и фонетических сходств, что позволяет системе находить максимально релевантные товары. При формировании списка используются алгоритмы, такие как:

- фонетическое индексирование для учета фонетических сходств;
- транслитерация для адаптации названий, которые могут быть введены на латинице;
- метрики схожести строк для учета опечаток и близких по написанию слов.

Получение запроса. В запросе приходит текст с названием товара. Поиск по списку названий товаров. С помощью нечеткого поиска система анализирует список названий товаров и сравнивает его с введенным

запросом. На этом этапе учитываются:

- фонетическое сходство — если название товара звучит аналогично запросу, система считает его релевантным;
- транслитерация — если товар введен на латинице, система преобразует его в кириллицу (или наоборот), чтобы обеспечить максимальное соответствие;
- схожесть строк — используется для поиска товаров, название которых отличается от запроса на 1-2 символа (например, "шурупаверт" вместо "шуруповерт").

Сопоставление результатов и фильтрация. Результаты поиска сортируются по степени релевантности, при этом сначала отображаются товары с наибольшим уровнем совпадения. Если запрос был введен с ошибкой, но система находит товар с высокой вероятностью совпадения, он выводится в результатах первым. Если степень совпадения низкая, товар может быть исключен из выдачи, чтобы избежать нерелевантных предложений.

Возвращение результата поиска. После завершения поиска:

- если товар найден, система возвращает соответствующую информацию (название товара, идентификатор и, при необходимости, другие параметры) в модуль умного поиска;
- если ни один товар не удовлетворяет условиям запроса, система возвращает None, указывая на отсутствие соответствий. В этом случае менеджеру предлагается уточнить запрос или проверить корректность ввода.

4.2.3 Запрос к модулю товарной части

После успешного завершения нечеткого поиска и получения точного названия товара от модуля списка названий товаров, следующий этап включает запрос к модулю товарной части. Этот модуль управляет данными о товарных позициях, хранящимися в реляционной базе данных, и

предоставляет информацию о наличии, количестве, цене и возможности заказа у поставщика. Это помогает системе точно и оперативно формировать заказ.

Основные этапы реализации запроса к модулю товарной части:

- получение результата от модуля списка названий товаров - модуль «умного поиска» завершает поиск в списке названий товаров и получает наиболее релевантное название товара;
- формирование запроса в модуль товарной части - название товара, полученное на предыдущем этапе, передается в модуль товарной части. Запрос формируется в виде API-запроса, обычно через POST-запрос, содержащий название товара — точное название для поиска;
- поиск в базе данных модуля товарной части - модуль товарной части, используя реляционную базу данных (например, PostgreSQL), обрабатывает запрос и выполняет поиск товара по его названию.

В базе данных товар имеет такие атрибуты как:

- название товара (уникальное поле);
- количество — текущее количество товара на складе или под заказ;
- цена — актуальная цена товара для расчета стоимости заказа;
- статус наличия — указывает, доступен ли товар для немедленного заказа или требуется закупка у поставщика;
- формирование ответа и возврат данных в модуль «умного поиска» Модуль товарной части формирует ответ с нужной информацией, включая;
- название товара — для подтверждения идентификации;
- количество на складе — точное количество, доступное для заказа;
- статус наличия — сведения о наличии на складе или доступности у поставщика;
- цена — актуальная цена одной единицы товара;

- передача данных для оформления заказа.

Полученные данные от модуля товарной части возвращаются в модуль «умного поиска»;

4.2.4 Анализ полученных данных из модуля товарной части

На предыдущем этапе взаимодействия с модулем товарной части система «умного поиска» получила полные данные по запрошенному товару, включая название товара, количество на складе, статус наличия (в наличии или требуется заказ у поставщика) и цену. Эти данные должны быть проанализированы для последующей передачи в интерфейс менеджера, чтобы обеспечить корректное и точное представление информации о товаре для дальнейшей работы с заказом.

Основные этапы анализа данных, полученных из модуля товарной части:

Анализ доступного количества товара. Первый этап анализа данных включает проверку доступного количества товара. Если запрашиваемое количество больше фактического количества на складе, система:

- определяет, доступно ли меньшее количество, чтобы предложить менеджеру альтернативный объем, если полное выполнение запроса невозможно;
- обрабатывает статус наличия, чтобы выяснить, может ли товар быть заказан у поставщика, если на складе он отсутствует в нужном объеме.

Проверка статуса наличия. Данные о статусе наличия товара дают представление о доступности на складе или необходимости заказа у поставщика. Система анализирует этот статус:

- если статус наличия показывает, что товар «в наличии», система может без задержек предложить его к добавлению в заказ;
- если товар «доступен к заказу у поставщика», система должна отобразить эту информацию в интерфейсе, чтобы менеджер мог

уточнить сроки и условия доставки для клиента.

Анализ ценовой информации. Система проверяет, указана ли цена товара, и отображает ее в интерфейсе. При наличии различных прайсов для оптовых и розничных клиентов цена может быть скорректирована на этапе анализа для выбора наиболее подходящего варианта. Цена товара будет отображена с учетом текущих условий, таких как скидки или специальные предложения, если они применимы для данного клиента. Формирование ответа для интерфейса менеджера. После анализа данных система формирует ответ, содержащий всю необходимую информацию для отображения в интерфейсе менеджера:

- название товара — для идентификации в заказе;
- количество на складе — если доступное количество меньше запрашиваемого, система указывает доступный объем;
- статус наличия — «в наличии» или «доступен к заказу у поставщика».
- цена — актуальная стоимость единицы товара.

Эти данные передаются в интерфейс менеджера в виде структурированного JSON-ответа, чтобы менеджер мог быстро оценить информацию и при необходимости скорректировать заказ. Возвращение данных в интерфейс менеджера. После завершения анализа данных сформированный ответ возвращается в интерфейс менеджера. Интерфейс отображает все полученные сведения, включая название товара, доступное количество, статус наличия и цену. Пример ответа в интерфейсе менеджера:

Допустим, система запрашивала информацию о товаре «Шуруповерт Makita» и получила следующие данные от модуля товарной части:

- название: «Шуруповерт Makita»;
- доступное количество: 10 штук (запрашивалось 15);
- статус наличия: «Доступен к заказу у поставщика»;
- цена: 2500 руб. за штуку.

Система возвращает эти данные в интерфейс менеджера, где

отображается:

- название товара: Шуруповерт Makita;
- доступное количество: 10 (информация о возможном ожидании дополнительного количества);
- статус наличия: доступен к заказу у поставщика;
- цена: 2500 руб.

Менеджер может на основе этих данных принять решение — подтвердить доступное количество или предложить клиенту возможность ожидания для получения полного объема. Преимущества анализа данных перед возвращением их в интерфейс.

Такой детальный анализ данных перед их передачей в интерфейс менеджера обеспечивает точность и полноту информации, что позволяет снизить количество ошибок при оформлении заказов и улучшить взаимодействие с клиентами [7].

4.2.5 Добавление товара в заказ

После завершения анализа и подтверждения данных о товаре (цена, количество и статус наличия), интерфейс «умного поиска» отправляет финальный запрос с этими параметрами в модуль обработки заказов. Этот модуль отвечает за создание и управление заказами [10]. На этом этапе товар добавляется в заказ, и система фиксирует все данные для последующих этапов обработки, включая сборку и доставку. После чего окно программы закрывается.

4.3 Реализация компонентов приложения.

4.3.1 Функция парсинга названия.

Регулярные выражения:

pattern = r"([а-яА-ЯёЁa-zA-Z\s]+)\s(\d+)\s*(штук|упаковок|штуки|упаковки)?". Шаблон регулярного

выражения продемонстрирован в коде, и он поддерживает как русские, так и латинские буквы, что делает его универсальным для разных языков. Также использование символов \s и ? позволяет справляться с пробелами и необязательными частями строки, что добавляет гибкости к тому, как текст может быть введен пользователем. Группировка и извлечение данных:

```
product_name = match.group(1).strip()  # Название товара
quantity = int(match.group(2)) if match.group(2) else 1  # Количество товара (1, если не указано)
```

Код использует метод group() для извлечения конкретных частей совпадения. Это делает его легким для понимания и поддержания. Применение strip() для названия товара также обеспечивает очистку от лишних пробелов, что важно для корректной обработки данных.

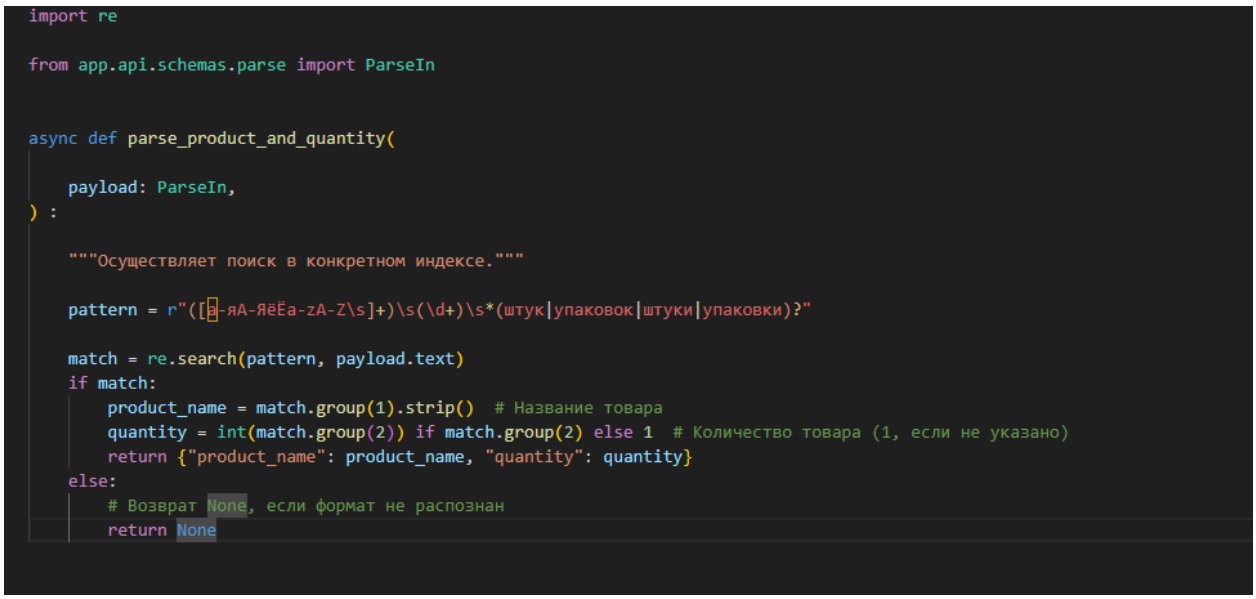
Умное управление количеством:

```
quantity = int(match.group(2)) if match.group(2) else 1  # Количество товара (1, если не указано)
```

Здесь реализовано простое и эффективное управление количеством товара. Если пользователь не указывает количество, функция по умолчанию устанавливает его равным 1. Это логичное решение, поскольку многие товары могут быть заказываемы по одному, если не указано иное.

Обработка ошибок:

```
else:
    return None
```



```python
import re

from app.api.schemas.parse import ParseIn


async def parse_product_and_quantity(

    payload: ParseIn,
) :

    """Осуществляет поиск в конкретном индексе."""

    pattern = r"([а-яА-ЯёЁa-zA-Z\s]+)\s(\d+)\s*(штук|упаковок|штуки|упаковки)?"

    match = re.search(pattern, payload.text)
    if match:
        product_name = match.group(1).strip()  # Название товара
        quantity = int(match.group(2)) if match.group(2) else 1  # Количество товара (1, если не указано)
        return {"product_name": product_name, "quantity": quantity}
    else:
        # Возврат None, если формат не распознан
        return None
```

Рисунок 8 – Функция парсинга названия товара и количества штук

Код включает простую логику обработки ошибок: если введенный текст не соответствует ожидаемому формату, функция возвращает None. Это упрощает дальнейшую обработку и позволяет вызывающему коду определить, была ли выполнена операция успешно или нет.

Чистота и ясность кода:

- функция имеет четкий и понятный интерфейс, принимая только один аргумент payload, что делает её удобной для использования в других частях кода;
- докстринг в начале функции кратко объясняет её назначение, что полезно для других разработчиков, которые могут работать с этим кодом.

Эти моменты делают код эффективным и удобным для дальнейшей модификации, расширения и поддержки.

4.3.2 Реализация интерфейса

Реализация интерфейса

Для реализации интерфейса используется HTML, CSS и JavaScript с библиотекой React. Основные элементы интерфейса включают. Заголовок с

названием "Умный поиск товаров заказа". Номер заказа, который отображается рядом с заголовком. Кнопки для очистки, копирования ненайденных товаров и добавления единиц. Таблицу для отображения товаров, найденных по запросу. Кнопку "Показать" для отправки запроса или обновления данных.

Ниже приведен пример кода, реализующий интерфейс с использованием React.

```
import ReactDOM from 'react-dom';

const App = () => {
  const [products, setProducts] = useState([]);
  const orderNumber = "№2411-100100-94008";
  const clearProducts = () => {
    setProducts([]);
  };

  const copyNotFound = () => {
    const notFoundItems = "Товары, которые не были найдены"; //
    navigator.clipboard.writeText(notFoundItems).then(() => {
      alert("Ненайденные товары скопированы в буфер обмена.");
    });
  };

  const addQuantity = () => {
    alert("Добавление шт. для выбранных товаров");
  };

  const showProducts = () => {
    const productData = [
      { name: 'Пример товара 1' },
```

```
            { name: 'Пример товара 2' }
        ];
        setProducts(productData);
    };

    return (
        <div className="container">
            <h1>Умный поиск товаров заказа <span className="order-number">{orderNumber}</span></h1>

            <div className="toolbar">
                <button onClick={clearProducts}>Очистить</button>
                <button onClick={copyNotFound}>Копировать ненайденные</button>
                <button onClick={addQuantity}>Добавить шт.</button>
                <button onClick={showProducts} id="showBtn">Показать</button>
            </div>

            <table id="productTable">
                <thead>
                    <tr>
                        <th>Товар</th>
                    </tr>
                </thead>
                <tbody>
                    {products.map((product, index) => (
                        <tr key={index}>
                            <td>{product.name}</td>
                        </tr>
```

```
                ))}
              </tbody>
            </table>
          </div>
        );
      };

      ReactDOM.render(<App />, document.getElementById('root'));
```

Описание кода. HTML: Основная структура страницы с подключением CSS и скрипта React. CSS: Стили для выравнивания, цветовой гаммы, размера текста и других визуальных элементов. React (JavaScript):

- функция clearProducts очищает таблицу товаров, устанавливая пустой массив products.
- функция copyNotFound копирует текст ненайденных товаров в буфер обмена.
- функция addQuantity имитирует добавление количества единиц для выбранных товаров.
- функция showProducts заполняет таблицу примерами товаров, чтобы продемонстрировать работу интерфейса.

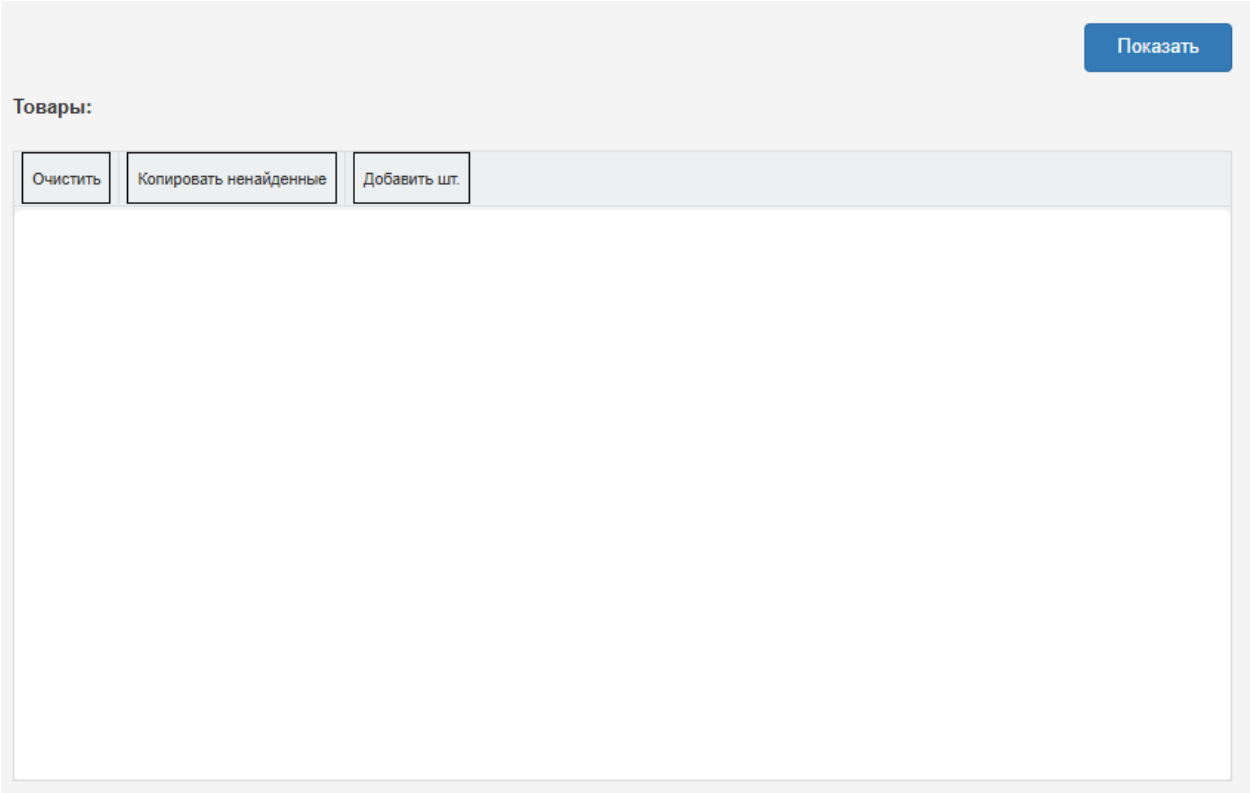


Рисунок 9 – Интерфейс «Умного поиска»

Этот интерфейс предоставляет базовый функционал для управления товарными позициями в заказе и реализован с использованием React, что позволяет легко обновлять и манипулировать состоянием приложения в реальном времени.

4.4 Тестирование приложения

Тестирование является важным этапом разработки, так как оно позволяет выявить ошибки, улучшить производительность и повысить качество конечного продукта [9]. В данной главе были рассмотрены методы тестирования, применённые для проверки корректности работы приложения.

4.4.1 Ручное тестирование

На этапе ручного тестирования я проверил основные функции приложения, выполняя действия, аналогичные тем, что будут выполнять конечные пользователи. Этот процесс включал проверку корректности работы всех модулей, таких как "умный поиск", запрос к товарной базе и модуль обработки заказов. Во время тестирования я ввожу различные запросы, чтобы проверить, правильно ли приложение находит товар даже при наличии опечаток и ошибок в названии, и корректно ли оно обрабатывает ситуацию, если товар отсутствует. Также я тестирую функции добавления товара в заказ, проверки наличия на складе и расчёта цены.

Для проведения ручного тестирования был составлен, чек-лист [11], в котором описаны основные функции и действия, подлежащие проверке. Проверки выполнялись по шагам, после каждого из которых я фиксировал результаты и, при необходимости, вносил исправления в код.

4.4.2 Автоматическое тестирование

В процессе разработки были созданы несколько автоматических тестов с использованием библиотеки pytest, чтобы проверить функциональные части приложения. Автоматическое тестирование включает в себя тесты для модуля "умного поиска", в том числе проверки на нечеткий поиск, правильность извлечения количества товара из текста, а также на корректность обработки отсутствия товара [3].

Были созданы тесты, проверяющие:

- Поиск товара с опечаткой и транслитерацией.
- Обработку запроса, в котором указано количество товаров.
- Правильность вывода, если товар не найден.

Автоматические тесты позволили убедиться, что приложение корректно обрабатывает различные варианты входных данных и возвращает ожидаемый результат. Также эти тесты обеспечивают

возможность быстрого повторного тестирования после внесения изменений в код.

Таким образом, можно сделать вывод, что модуль «Умный поиск» обеспечивает автоматизацию процесса обработки заказов, включая нечеткий поиск товаров, проверку их наличия и формирование структурированных данных для создания заказа. Реализация бизнес-логики и интерфейса повышает точность, снижает ошибки и сокращает время работы сотрудников, что улучшает эффективность всей системы управления продажами.

4.4.3 Ручное тестирование интерфейса

Интерфейс приложения также подвергался ручному тестированию. Я проверил, как выглядит интерфейс, как пользователи взаимодействуют с полями ввода, кнопками и таблицей товаров, отображающейся в интерфейсе.

В рамках ручного тестирования интерфейса я оценивал следующие аспекты:

Удобство расположения элементов управления (кнопки «Очистить», «Копировать ненайденные», «Добавить шт.»). Корректность отображения данных в таблице товаров.

Работа кнопок, которые позволяют удалять введенные данные, копировать ненайденные товары и добавлять их в список.

Проверка корректности перехода в другие части приложения и закрытия окна после завершения добавления товара в заказ.

На основе тестирования было проверено, что интерфейс интуитивно понятен и не вызывает трудностей в использовании, а все функциональные элементы работают так, как задумано.

Заключение

В ходе выполнения ВКР была разработана программа «умный поиск» для компании «ВсеИнструменты.ру», направленная на автоматизацию и оптимизацию процесса управления заказами. Основная задача заключалась в создании автоматизированной системы, которая облегчает работу менеджеров и позволяет ускорить процесс обработки запросов от клиентов, особенно в сегменте b2b. Благодаря внедрению данной системы компания получает более структурированный и эффективный процесс обработки заказов, что минимизирует влияние человеческого фактора на результат.

Работа включает несколько этапов разработки:

- анализ и моделирование бизнес-процессов компании;
- проектирование модуля «умный поиск»;
- реализация интерфейса, разработка функциональных и интеграционных модулей для взаимодействия с другими подсистемами компании.

Важным преимуществом стало использование микро сервисной архитектуры, что позволило легко интегрировать модуль «умного поиска» с существующими системами: CRM, складским и логистическим модулями, а также базой данных с товарной частью. Программа автоматизирует обработку запросов клиентов, анализирует название товара, количество и статус наличия на складе, после чего автоматически добавляет найденные позиции в заказ, освобождая менеджеров от рутинной работы и снижая риск ошибок.

Поставленные задачи, включая анализ организационной структуры, разработку архитектуры и интерфейса, а также моделирование взаимодействия модулей системы, были успешно выполнены. Программа была протестирована с использованием ручного и автоматического тестирования, что позволило убедиться в ее корректной работе и соответствии требованиям.

Результаты работы продемонстрировали, что внедрение «умного поиска» способствует значительной экономии времени на обработку заказов, повышает точность данных и качество обслуживания клиентов. Благодаря возможности расширения и адаптации под изменения потребностей предприятия, данное программное обеспечение также является перспективным для дальнейшего развития и интеграции с новыми системами и сервисами компании.

Список используемой литературы и используемых источников

1. Базы данных. В 2х кн. Кн. 2. Распределенные и удаленные базы данных: Учебник / В.П. Агальцов. М.: ИД ФОРУМ: НИЦ ИнфраМ, 2013. 272 с.: ил.; 60х90 1/16. (Высшее образование). (переплет) ISBN 9785819903940 Режим доступа: http:// znanium.com/catalog/product/372740 (дата обращения: 11.09.2024).
2. Боггс, У. UML и Rational Rose / У. Боггс, М. Боггс. - М.: Издательство «ЛОРИ», 2001. - 582 с. (дата обращения: 11.09.2024).
3. Бугорский, В.Н. Сетевая экономика и проектирование информационных систем: учеб.пос. / В.Н. Бугорский, Р.В. Соколов. СПб.: Питер, 2007. 320 с. (дата обращения: 11.09.2024).
4. Буч, Г. Язык UML. Руководство пользователя / Г. Буч, Дж. Рамбо, А. Якобсон. - СПб.: Питер, 2004. - 432 с. (дата обращения: 11.09.2024).
5. Вагнер, Билл C# Эффективное программирование / Билл Вагнер. - М.: ЛОРИ, 2017. - 320 с. (дата обращения: 11.09.2024).
6. Вендров, А. М. Проектирование программного обеспечения экономических информационных систем. / А. М. Вендров. – Москва: Финансы и статистика, 2020. – 203 с. (дата обращения: 11.09.2024).
7. Верников Г. Основные методологии обследования организации. Стандарт IDEF0 http://www. consulting.ru/main/ mgmt/texts/m7/079_idef.shtml (дата обращения: 11.09.2024).
8. Гниденко И.Г. Информационные технологии в бизнесе: Учебное пособие / И.Г. Гниденко, С.А. Соколовская. – Москва: Вектор, 2005. – 160 с. (дата обращения: 11.09.2024).
9. Диаграмма вариантов использования (UseCase diagram) // Flexberry PLATFORM Documentation [Электронный ресурс]. – URL: https://flexberry.github.io/ru/fd_use-case-diagram.html (дата обращения: 11.09.2024).

10. Клир Д.Системология: автоматизация решения системных задач / Пер. с англ. М.: Радио и связь, 1990. 544с. (дата обращения: 11.09.2024).

11. Куликов С. С. Тестирование программного обеспечения. Базовый курс / С.С Куликов. – Минск: Четыре четверти, 2020. – 310 с. (дата обращения: 11.09.2024).

12. Лутц М. Изучаем Python, том 1, 5-е издание / М. Лутц. – СПб: Диалектика, 2019. – 832 с. – ISBN 978-5-907144-52-1 (дата обращения: 11.09.2024).

13. Малыхина, М. Базы данных: основы, проектирование, использование / М. Малыхина. - М.: БХВ-Петербург, 2016. - 512 с. (дата обращения: 11.09.2024).

14. Плаксин М. А. Тестирование и отладка программ для профессионалов будущих и настоящих / М. А. Плаксин. – Москва: БИНОМ. Лаборатория знаний, 2013. – 167 с. (дата обращения: 11.09.2024).

15. Постолит, А. В. Python, Django и PyCharm для начинающих / А. В. Постолит. – Москва: БХВ, 2021. – 465 с. – ISBN 978-5-9775-6779-4 (дата обращения: 11.09.2024).

16. Рамбо Дж. UML 2.0. Объектно-ориентированное моделирование и разработка, 2-е изд. / Дж. Рамбо, М. Блаха. – СПб: Питер, 2021. – 544 с. – ISBN 978-5-4461-9428-5 (дата обращения: 11.09.2024).

17. Саммерфилд , М. Программирование на Python 3. Подробное руководство / , М. Саммерфилд. – Москва: Символ-Плюс, 2009. – 11 с (дата обращения: 11.09.2024).

18. Серверные языки программирования [Электронный ресурс]. – URL: https://web-creator.ru/articles/server_side_languages (дата обращения: 11.09.2024).

19. Уткин В.Б. Информационные системы и технологии в экономике / В.Б. Уткин. – Москва: ЮНИТИ, 2003. – 334 с. (дата обращения: 11.09.2024).

20. Pydantic Documentation for version: v2.9.2. Pydantic is the most widely used data validation library for Python [Электронный ресурс]. URL: https://docs.pydantic.dev/latest/ (дата обращения: 11.09.2024).

21. UML Diagrams Tutorial. Learning Guides: сайт. – URL: https://www.visual-paradigm.com/guide/uml-unified-modeling-language (дата обращения: 11.09.2024).